



V2V EDTECH LLP

Online Coaching at an Affordable Price.

OUR SERVICES:

- Diploma in All Branches, All Subjects
- Degree in All Branches, All Subjects
- BSCIT / CS
- Professional Courses



+91 93260 50669



v2vedtech.com



V2V EdTech LLP



v2vedtech



WINTER – 2023 EXAMINATION
Model Answer – Only for the Use of RAC Assessors

Subject Name: Java Programming

Subject Code: 22412

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.
- 8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		Attempt any <u>FIVE</u> of the following:	10 M
	a)	Enlist any two logical operators and two bitwise operators.	2 M
	A 1 Ans	<p>Logical Operators:</p> <ol style="list-style-type: none"> 1. AND Operator (&&) – if(a && b) [if true execute else don't] 2. OR Operator () – if(a b) [if one of them is true to execute else don't] 3. NOT Operator (!) – !(a<b) [returns false if a is smaller than b] <p>Bitwise Operator:</p> <ol style="list-style-type: none"> 1. Bitwise OR () 2. Bitwise AND (&) 3. Bitwise XOR (^) 4. Bitwise Complement (~) 5. Bitwise left shift(<<) 6. Bitwise right shift(>>) 	<p>List any two Logical operator : 2 marks</p> <p>List any two Bitwise operator : 2 marks</p>



	b)	Define constructor.	2 M
	Ans	A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.	Correct/suitable definition- 1 M Syntax or



		For Example: <pre> class Test { Test() { // constructor body } } </pre>	Example- 1 M								
	c)	Write down the syntax of array declaration, initialization.	2 M								
Ans	<p>The syntax of declaring an array in Java is given below.</p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> datatype [] arrayName; </div> <p>Here, the datatype is the type of element that will be stored in the array, square bracket[] is for the size of the array, and arrayName is the name of the array.</p> <p>The syntax of initializing an array is given below.</p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> datatype [] arrayName = new datatype [size]; </div>		1 M- array declaration and 1 M-array initialization								
	d)	List out different ways to access package from another package.	2 M								
Ans	<p>There are three ways to access the package from outside the package.</p> <ul style="list-style-type: none"> import package.*; import package.classname; fully qualified name 		Any 2 correct ways – 2 M								
	e)	Differentiate between starting thread with run() method and start() method.	2 M								
Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">start()</th> <th style="width: 50%; text-align: center;">run()</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">Creates a new thread and the run() method is executed on the newly created thread.</td> <td style="padding: 5px;">No new thread is created and the run() method is executed on the calling thread itself.</td> </tr> <tr> <td style="padding: 5px;">Can't be invoked more than one time</td> <td style="padding: 5px;">Multiple invocation is possible</td> </tr> <tr> <td style="padding: 5px;">Defined in java.lang.Thread class.</td> <td style="padding: 5px;">Defined in java.lang.Runnable interface and must be overridden in the implementing class.</td> </tr> </tbody> </table>		start()	run()	Creates a new thread and the run() method is executed on the newly created thread.	No new thread is created and the run() method is executed on the calling thread itself.	Can't be invoked more than one time	Multiple invocation is possible	Defined in java.lang.Thread class.	Defined in java.lang.Runnable interface and must be overridden in the implementing class.	Any 2 valid points- 2 M
start()	run()										
Creates a new thread and the run() method is executed on the newly created thread.	No new thread is created and the run() method is executed on the calling thread itself.										
Can't be invoked more than one time	Multiple invocation is possible										
Defined in java.lang.Thread class.	Defined in java.lang.Runnable interface and must be overridden in the implementing class.										



	<p>It starts thread to begin execution, JVM calls run method of this thread.</p> <p>Syntax: public void start()</p> <p>A new thread will be created and it is responsible to complete the job.</p>	<p>It is used to perform operations by thread.</p> <p>Syntax: public void run()</p> <p>No new thread will be created and main thread will be responsible to complete the job.</p>	
f)	State the classes that can an applet extend.		2 M
Ans	<ul style="list-style-type: none"> • Graphics • Font • Color 		Any 2 classes-2 M
g)	Give syntax to open a file using InputStream class.		2 M
Ans	<p>Attach a file to a FileInputStream as this will enable us to read data from the file as shown below as follows:</p> <p style="text-align: center;">FileInputStream input = new FileInputStream("input.txt");</p> <p>Now in order to read data from the file, we should read data from the FileInputStream as shown below:</p> <p style="text-align: center;">ch=fileInputStream.read();</p>		Correct syntax-2 M
2.	Attempt any <u>THREE</u> of the following:		12 M
a)	Write a program lo display ASCII value of a number 9.		4 M
Ans	<pre>public class asciivalue { public static void main(String args[]) { // Character whose ASCII is to be computed char ch = '9'; // Creating a new variable of type int and assigning the character value. int ascii = ch; // Printing the ASCII value of above character System.out.println("The ASCII value of " + ch+ " is: " + ascii); } } </pre> <p>Output:</p> <p>The ASCII value of 9 is: 57</p>		For any correct program: 4m



	b) Write a program to sort the elements of an array in ascending order.	4 M
Ans	<pre>class arraysort { public static void main(String args[]) { int a[]={85,95,78,45,12,56,78,19}; int i=0; int j=0; int temp=0; int l=a.length; for(i=0;i<l;i++) { //apply bubble sort for(j=(i+1);j<l;j++) { if(a[i]>a[j]) { temp=a[i]; a[i]=a[j]; a[j]=temp; } } } System.out.println("Ascending order of numbers:"); for(i=0;i<l;i++) System.out.println(""+a[i]); } }</pre> <p>Output: Ascending order of numbers: 12 19 45 56 78 78 85 95</p>	For any correct logic and program: 4m
	c) Define Thread. Draw life cycle of Thread.	4 M

<p>Ans</p>	<p>Thread is a smallest unit of executable code or a single task is also called as thread. Each tread has its own local variable, program counter and lifetime. A thread is similar to program that has a single flow of control.</p> <div style="text-align: center;"> <pre> graph TD Newborn[Newborn] -- start() --> Running([Running]) Running -- yield() --> Runnable([Runnable]) Runnable --> Running Runnable -- stop() --> Dead[Dead] Runnable -- suspend() --> Blocked[Blocked] Blocked -- resume() --> Runnable Blocked -- notify() --> Runnable Blocked -- stop() --> Dead Newborn -- stop() --> Dead </pre> </div> <p style="text-align: center;">Fig: Life cycle of Thread</p>	<p>Definition of thread- 2 M</p> <p>Diagram: 2M</p>
<p>d)</p>	<p>Write a program to read a file and then count number of words.</p>	<p>4 M</p>
<p>Ans</p>	<pre> // Java program to count the no. of words in a file import java.io.*; public class Test11 { public static void main(String[] args) throws IOException { File file = new File("C:\\Program Files\\Java\\jdk1.7.0_80\\bin\\a.txt"); FileInputStream fileInputStream = new FileInputStream(file); InputStreamReader inputStreamReader = new InputStreamReader(fileInputStream); BufferedReader bufferedReader = new BufferedReader(inputStreamReader); String line; int wordCount = 0; int paraCount = 0; while ((line = bufferedReader.readLine()) != null) { if (line.equals("")) { </pre>	<p>2 M - correct variable and object creation</p> <p>2 M - valid logic to count words from file.</p>



	<pre>paraCount += 1; } else { String words[] = line.split("\\s+"); wordCount += words.length; } } System.out.println("Total word count = "+ wordCount); } }</pre> <p>C:\Program Files\Java\jdk1.7.0_80\bin>javac Test11.java C:\Program Files\Java\jdk1.7.0_80\bin>java Test11 Total word count = 8</p>	
3.	Attempt any <u>THREE</u> of the following:	12 M
a)	Write a program which displays functioning of ATM machine, (Hint: Withdraw, Deposit, Check Balance and Exit)	4 M
Ans	<pre>import java.util.Scanner; public class ATM_Transaction { public static void main(String args[]) { int balance = 5000, withdraw, deposit; Scanner s = new Scanner(System.in); while(true) { System.out.println("Automated Teller Machine"); System.out.println("Choose 1 for Withdraw"); System.out.println("Choose 2 for Deposit"); System.out.println("Choose 3 for Check Balance"); System.out.println("Choose 4 for EXIT"); System.out.print("Choose the operation you want to perform:"); int n = s.nextInt(); switch(n) {</pre>	4 M for correct program Or any other relevant logic should be considered



	<pre>case 1: System.out.print("Enter money to be withdrawn:"); withdraw = s.nextInt(); if(balance >= withdraw) { balance = balance - withdraw; System.out.println("Please collect your money"); } else { System.out.println("Insufficient Balance"); } System.out.println(""); break; case 2: System.out.print("Enter money to be deposited:"); deposit = s.nextInt(); balance = balance + deposit; System.out.println("Your Money has been successfully deposed"); System.out.println(""); break; case 3: System.out.println("Balance : "+balance); System.out.println(""); break; case 4: System.exit(0); } } }</pre>	
b)	Differentiate between method overloading and method overriding.	4 M



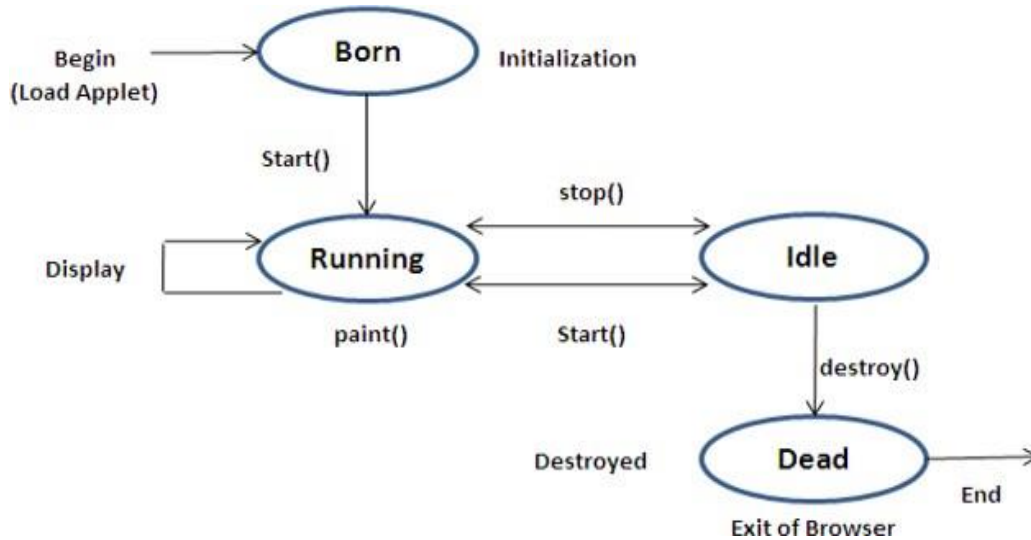
Ans	Method Overloading	Method Overriding	4 M for any four correct point
	Method overloading is a compile-time polymorphism.	Method overriding is a run-time polymorphism.	
	Method overloading helps to increase the readability of the program.	Method overriding is used to grant the specific implementation of the method which is already provided by its parent class or superclass.	
	It occurs within the class.	It is performed in two classes with inheritance relationships.	
	Method overloading may or may not require inheritance.	Method overriding always needs inheritance.	
	In method overloading, methods must have the same name and different signatures.	In method overriding, methods must have the same name and same signature.	
	In method overloading, the return type can or can not be the same, but we just have to change the parameter.	In method overriding, the return type must be the same or co-variant.	
	Static binding is being used for overloaded methods.	Dynamic binding is being used for overriding methods.	
	Poor Performance due to compile time polymorphism.	It gives better performance. The reason behind this is that the binding of overridden methods is being done at runtime.	
	Private and final methods can be overloaded.	Private and final methods can't be overridden.	
	The argument list should be different while doing method overloading.	The argument list should be the same in method overriding.	
c)	Explain applet life cycle in detail.	4 M	



Ans

The applet life cycle can be defined as the process of how the object is created, started, stopped, and destroyed during the entire execution of its application. It basically has five core methods namely `init()`, `start()`, `stop()`, `paint()` and `destroy()`. These methods are invoked by the browser to execute.

2 M for diagram
and 2 M for
explanation



1. Initialization State (The `init()` method):

- The life cycle of an Applet is begin on that time when the applet is first loaded into the browser and called the `init()` method.
- The `init()` method is called only one time in the life cycle on an Applet. The `init()` method is basically called to read the “PARAM” tag in the html file.
- The `init ()` method retrieve the passed parameter through the “PARAM” tag of html file using `getParameter()` method All the initialization such as initialization of variables and the objects like image, sound file are loaded in the `init ()` method.
- After the initialization of the `init()` method user can interact with the Applet and mostly applet contains the `init()` method.

• **Syntax:**

```
public void init()
{
---
---
}
```

2. Running State (The `start()` method):

- The `start` method of an Applet is called after the initialization method `init()`. This method may be called multiples time when the Applet needs to be started or restarted.



- For Example if the user wants to return to the Applet, in this situation the start() method of an Applet will be called by the web browser and the user will be back on the applet. In the start method user can interact within the applet.
- Syntax:-

```
public void start()
{
.....
.....
}
```

3. Idle (The Stop() method):

- An applet becomes idle when it is stopped from running. The stop() method stops the applet and makes it invisible.
- Stopping occurs automatically when we leave the page containing the currently running applet. We can also do so by calling the stop() method explicitly.
- The stop() method can be called multiple times in the life cycle of applet like the start () method or should be called at least one time.
- For example the stop() method is called by the web browser on that time When the user leaves one applet to go another applet and the start() method is called on that time when the user wants to go back into the first program or Applet.
- **Syntax:-**

```
public void stop()
{
.....
.....
}
```

4. Dead State (The destroy() method):

- The destroy() method is called to terminate an Applet. an Applet is said to be dead when it is removed from memory.
- This occurs automatically by invoking the destroy() method when we quit the browser. It is useful for clean-up actions, such as releasing memory after the applet is removed, killing off threads and closing network/database connections.
- Thus this method releases all the resources that were initialized during an applet's initialization.
- **Syntax:-**

```
public void destroy()
{
.....
.....
}
```



	<p><u>5. Display State (The paint() method):</u></p> <ul style="list-style-type: none">• The paint() method is used for applet display on the screen. The display includes text, images, graphics and background.• This happens immediately after the applet enters into the running state. Almost every applet will have a paint() method and can be called several times during an applet's life cycle.• The paint() method is called whenever a window is required to paint or repaint the applet.• Syntax:- public void paint(Graphics g) { }	
d)	<p>Differentiate between Byte Stream Class and Character Stream Class. (Any four points)</p>	4 M



	Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: left;">Byte Stream Class</th> <th style="width: 50%; text-align: left;">Character Stream Class</th> </tr> </thead> <tbody> <tr> <td>Byte streams access the file byte by byte (8 bits).</td> <td>A character stream will read a file character by character (16 bits).</td> </tr> <tr> <td>Byte stream classes are classified into: <ol style="list-style-type: none"> 1. Input Stream Classes 2. Output Stream Classes </td> <td>Character stream classes are classified into: <ol style="list-style-type: none"> 1. Reader class 2. Writer class </td> </tr> <tr> <td>InputStream/OutputStream class is byte-oriented.</td> <td>The Reader/Writer class is character-oriented.</td> </tr> <tr> <td>The methods for byte streams generally work with byte data type.</td> <td>The methods for character streams generally accept parameters of data type <i>char</i> parameters.</td> </tr> <tr> <td>Byte-stream classes end with the suffix InputStream and OutputStream.</td> <td>Character-stream classes end with the suffix Reader or Writer.</td> </tr> <tr> <td>It is possible to translate character stream into byte stream with OutputStreamWriter.</td> <td>It is possible to translate byte stream into a character stream with InputStreamReader.</td> </tr> <tr> <td>Byte streams specifically used for reading and writing data in byte format.</td> <td>The advantage of character streams, that is make it easy to write programs, which is not dependent upon a specific character encoding.</td> </tr> <tr> <td>No conversion needed.</td> <td>Character streams convert the underlying data bytes to Unicode, which is a costly operation.</td> </tr> <tr> <td>InputStream and OutputStream are used for reading or writing binary data.</td> <td>Reader and Writer uses Unicode, hence they can be internationalized. Hence in some cases they are more efficient than byte streams.</td> </tr> </tbody> </table>	Byte Stream Class	Character Stream Class	Byte streams access the file byte by byte (8 bits).	A character stream will read a file character by character (16 bits).	Byte stream classes are classified into: <ol style="list-style-type: none"> 1. Input Stream Classes 2. Output Stream Classes 	Character stream classes are classified into: <ol style="list-style-type: none"> 1. Reader class 2. Writer class 	InputStream/OutputStream class is byte-oriented.	The Reader/Writer class is character-oriented.	The methods for byte streams generally work with byte data type.	The methods for character streams generally accept parameters of data type <i>char</i> parameters.	Byte-stream classes end with the suffix InputStream and OutputStream.	Character-stream classes end with the suffix Reader or Writer.	It is possible to translate character stream into byte stream with OutputStreamWriter.	It is possible to translate byte stream into a character stream with InputStreamReader.	Byte streams specifically used for reading and writing data in byte format.	The advantage of character streams, that is make it easy to write programs, which is not dependent upon a specific character encoding.	No conversion needed.	Character streams convert the underlying data bytes to Unicode, which is a costly operation.	InputStream and OutputStream are used for reading or writing binary data.	Reader and Writer uses Unicode, hence they can be internationalized. Hence in some cases they are more efficient than byte streams.	4 M for any correct 4 point
Byte Stream Class	Character Stream Class																						
Byte streams access the file byte by byte (8 bits).	A character stream will read a file character by character (16 bits).																						
Byte stream classes are classified into: <ol style="list-style-type: none"> 1. Input Stream Classes 2. Output Stream Classes 	Character stream classes are classified into: <ol style="list-style-type: none"> 1. Reader class 2. Writer class 																						
InputStream/OutputStream class is byte-oriented.	The Reader/Writer class is character-oriented.																						
The methods for byte streams generally work with byte data type.	The methods for character streams generally accept parameters of data type <i>char</i> parameters.																						
Byte-stream classes end with the suffix InputStream and OutputStream.	Character-stream classes end with the suffix Reader or Writer.																						
It is possible to translate character stream into byte stream with OutputStreamWriter.	It is possible to translate byte stream into a character stream with InputStreamReader.																						
Byte streams specifically used for reading and writing data in byte format.	The advantage of character streams, that is make it easy to write programs, which is not dependent upon a specific character encoding.																						
No conversion needed.	Character streams convert the underlying data bytes to Unicode, which is a costly operation.																						
InputStream and OutputStream are used for reading or writing binary data.	Reader and Writer uses Unicode, hence they can be internationalized. Hence in some cases they are more efficient than byte streams.																						
4.		Attempt any <u>THREE</u> of the following:	12 M																				
	a)	Explain implicit and explicit type conversion with example in detail.	4 M																				
Ans		<p style="text-align: center;"><u>Widening (Implicit)</u></p> <ul style="list-style-type: none"> • The process of assigning a smaller type to a larger one is known as widening or implicit. <p style="text-align: center;">Byte → short → int → long → float → double</p>	2 M for Implicit with example And 2 M for Explicit with example																				



	<p>For e.g. class widening</p> <pre>{ public static void main(String arg[]) { int i=100; long l=i; float f=l; System.out.println("Int value is"+i); System.out.println("Long value is"+l); System.out.println("Float value is"+f); } }</pre>	
	<p><u>Narrowing (Explicit)</u></p> <ul style="list-style-type: none">• The process of assigning a larger type into a smaller one is called narrowing.• Casting into a smaller type may result in loss of data. <p>double → long → int → short → byte</p> <p>For e.g. class narrowing</p> <pre>{ Public static void main(String[]) { Double d=100.04; Long l=(long) d; Int i=(int) l; System.out.println("Int value is"+i); System.out.println("Long value is"+l); System.out.println("Float value is" } }</pre>	
b)	Write a program to show the use of copy constructor.	4 M
Ans	<pre>class student { int id; String name; student(int i, String n) { id=i; name=n; } }</pre>	4 M for any suitable correct program



	<pre>student (student s)//copy constructor { id=s.id; name=s.name; } void display() { System.out.println(id+" "+name) } public static void main(String args[]) student s1=new student(111, "ABC"); s1.display(); student s2= new student(s1); s2.display(); } }</pre>	
c)	Write a program to show the Hierarchical inheritance.	4 M
Ans	<pre>import java.io.*; abstract class shape { float dim1,dim2; void getdata() { DataInputStream d=new DataInputStream(System.in); try { System.out.println("Enter the value of Dimension1: "); dim1=Float.parseFloat(d.readLine()); System.out.println("Enter the value of Dimension2: "); dim2=Float.parseFloat(d.readLine()); } catch(Exception e) { System.out.println("General Error"+e); } } void disp() { System.out.println("Dimension1= "+dim1); System.out.println("Dimension2= "+dim2); } abstract void area(); } class rectangle extends shape</pre>	4 M for correct program (Any relevant example can be consider)



```
{
double area1;
void getd()
{
super.getdata();
}
void area()
{
area1=dim1*dim2;
System.out.println("The Area of Rectangle is: "+area1);
}
}
class triangle extends shape
{
double area1;
void getd()
{
super.getdata();
}
void area()
{
area1=(0.5*dim1*dim2);
System.out.println("The Area of Triangle is: "+area1);
}
}
class methodoverl
{
public static void main(String args[])
{
rectangle r=new rectangle();
System.out.println("For Rectangle");
r.getd();
r.disp();
r.area();
triangle t=new triangle();
t.getd();
t.disp();
t.area();
}
}

OR

class A
{
```



	<pre>public void methodA() { System.out.println("method of Class A"); } } class B extends A { public void methodB() { System.out.println("method of Class B"); } } class C extends A { public void methodC() { System.out.println("method of Class C"); } } class D extends A { public void methodD() { System.out.println("method of Class D"); } } class JavaExample { public static void main(String args[]) { B obj1 = new B(); C obj2 = new C(); D obj3 = new D(); //All classes can access the method of class A obj1.methodA(); obj2.methodA(); obj3.methodA(); } }</pre>	
d)	Explain any four font methods with example.	4 M
Ans	Font is a class that belongs to the java.awt package. Following are the methods of Font class:	2 M for any four- font method with



description and 2
M for example

Methods	Description
String getFamily()	Returns the name of the font family to which the invoking font belongs.
static Font getFont(String property)	Returns the font associated with the system property specified by <i>property</i> . null is returned if <i>property</i> does not exist.
String getFontName()	Returns the face name of the invoking font.
String getName()	Returns the logical name of the invoking font.
int getSize()	Returns the size, in points, of the invoking font.
int getStyle()	Returns the style values of the invoking font.
int hashCode()	Returns the hash code associated with the invoking object.
boolean isBold()	Returns true if the font includes the BOLD style value. Otherwise, false is returned.
boolean isItalic()	Returns true if the font includes the ITALIC style value. Otherwise, false is returned.
boolean isPlain()	Returns true if the font includes the PLAIN style value. Otherwise, false is returned.

Example:

```
import java.awt.*;
import java.applet.*;
public class Shapes extends Applet
{
    Font f,f1;
    String s,msg;
    String fname;
    String ffamily;
    int size;
    int style;
    public void init()
    {
        f= new Font("times new roman",Font.ITALIC,20);
        setFont(f);
        msg="is interesting";
        s="java programming";
        fname=f.getFontName();
        ffamily=f.getFamily();
        size=f.getSize();
        style=f.getStyle();
    }
}
```



	<pre>String f1=f.getName(); } public void paint(Graphics g) { g.drawString("font name"+fname,60,44); g.drawString("font family"+ffamily,60,77); g.drawString("font size "+size,60,99); g.drawString("fontstyle "+style,60,150); g.drawString("fontname "+f1,60,190); } } /*<applet code=Shapes.class height=300 width=300></applet>*/</pre>	
e)	Write a program to append content of one file into another file.	4 M
Ans	<pre>import java.io.*; class copyf { public static void main(String args[]) throws IOException { BufferedReader in=null; BufferedWriter out=null; try { in=new BufferedReader(new FileReader("input.txt")); out=new BufferedWriter(new FileWriter("output.txt")); int c; while((c=in.read())!=-1) { out.write(c); } System.out.println("File copied successfully"); } finally { if(in!=null) { in.close(); } if(out!=null) { out.close(); } } }</pre>	4 M for correct program



		} } }																											
5.		Attempt any <u>TWO</u> of the following:	12 M																										
	a)	Explain vector with the help of example. Explain any 3 methods of vector class.	6 M																										
Ans		<ul style="list-style-type: none"> • Vector is a data structure that is used to store a collection of elements. Elements can be of all primitive types like int, float, Object, etc. Vectors are dynamic in nature and accordingly, grow or shrink as per the requirement. • Vector Class in Java is found in the java.util package. • Vector class is a child class of the AbstractList class and implements the List interface. Therefore, we can use all the methods of the List interface. • Vectors are known to give ConcurrentModificationException when accessed concurrently at the time of modification. • When a Vector is created, it has a certain capacity to store elements that can be defined initially. This capacity is dynamic in nature and can be increased or decreased. • By definition, Vectors are synchronized, which implies that at a time, only one thread is able to access the code while other threads have to wait. <p>Vectors are created like arrays. It has three constructor methods</p> <pre>Vector list = new Vector(); //declaring vector without size Vector list = new Vector(3); //declaring vector with size Vector list = new Vector(5,2); //create vector with initial size and whenever it need to grows, it grows by value specified by increment capacity</pre> <p>Methods of Vector class:</p> <table border="1"> <thead> <tr> <th>Method Name</th> <th>Task performed</th> </tr> </thead> <tbody> <tr> <td>list.firstElement()</td> <td>It returns the first element of the vector.</td> </tr> <tr> <td>list.lastElement()</td> <td>It returns last element of the vector</td> </tr> <tr> <td>list.addElement(item)</td> <td>Adds the item specified to the list at the end.</td> </tr> <tr> <td>list.elementAt(n)</td> <td>Gives the name of the object at nth position</td> </tr> <tr> <td>list.size()</td> <td>Gives the number of objects present in vector</td> </tr> <tr> <td>List.capacity()</td> <td>This method returns the current capacity of the vector.</td> </tr> <tr> <td>list.removeElement(item)</td> <td>Removes the specified item from the list.</td> </tr> <tr> <td>list.removeElementAt(n)</td> <td>Removes the item stored in the nth position of the list.</td> </tr> <tr> <td>list.removeAllElements()</td> <td>Removes all the elements in the list.</td> </tr> <tr> <td>list.insertElementAt(item, n)</td> <td>Inserts the item at nth position.</td> </tr> <tr> <td>List.contains(object element)</td> <td>This method checks whether the specified element is present in the Vector. If the element is been found it returns true else false.</td> </tr> <tr> <td>list.copyInto(array)</td> <td>Copies all items from list of array.</td> </tr> </tbody> </table> <p>Example: import java.util.*;</p>	Method Name	Task performed	list.firstElement()	It returns the first element of the vector.	list.lastElement()	It returns last element of the vector	list.addElement(item)	Adds the item specified to the list at the end.	list.elementAt(n)	Gives the name of the object at nth position	list.size()	Gives the number of objects present in vector	List.capacity()	This method returns the current capacity of the vector.	list.removeElement(item)	Removes the specified item from the list.	list.removeElementAt(n)	Removes the item stored in the nth position of the list.	list.removeAllElements()	Removes all the elements in the list.	list.insertElementAt(item, n)	Inserts the item at nth position.	List.contains(object element)	This method checks whether the specified element is present in the Vector. If the element is been found it returns true else false.	list.copyInto(array)	Copies all items from list of array.	<p>Correct explanation-2 M</p> <p>List of constructors and methods of vector class-2 M</p> <p>Example – 2 M</p>
Method Name	Task performed																												
list.firstElement()	It returns the first element of the vector.																												
list.lastElement()	It returns last element of the vector																												
list.addElement(item)	Adds the item specified to the list at the end.																												
list.elementAt(n)	Gives the name of the object at nth position																												
list.size()	Gives the number of objects present in vector																												
List.capacity()	This method returns the current capacity of the vector.																												
list.removeElement(item)	Removes the specified item from the list.																												
list.removeElementAt(n)	Removes the item stored in the nth position of the list.																												
list.removeAllElements()	Removes all the elements in the list.																												
list.insertElementAt(item, n)	Inserts the item at nth position.																												
List.contains(object element)	This method checks whether the specified element is present in the Vector. If the element is been found it returns true else false.																												
list.copyInto(array)	Copies all items from list of array.																												



	<pre>public class Main { public static void main(String args[]) { Vector v = new Vector(); v.addElement(new Integer(10)); v.addElement(new Integer(20)); v.addElement(new Integer(30)); v.addElement(new Integer(40)); v.addElement(new Integer(10)); v.addElement(new Integer(20)); System.out.println(v.size()); // display original size System.out.println("Initial Vector: " + v); v.removeElementAt(2); // remove 3rd element System.out.println("Current Vector: " + v); v.removeElementAt(3); // remove 4th element System.out.println("Current Vector: " + v); v.insertElementAt(11,2); // new element inserted at 3rd position System.out.println("Current Vector: " + v); System.out.println("Size of vector after insert delete operations: " + v.size()); } } </pre> <p>Output: 6 Initial Vector: [10, 20, 30, 40, 10, 20] Current Vector: [10, 20, 40, 10, 20] Current Vector: [10, 20, 40, 20] Current Vector: [10, 20, 11, 40, 20] Size of vector after insert delete operations: 5</p>	
b)	Develop and Interest Interface which contains Simple Interest and Compound Interest methods and static final field of rate 25%. Write a class to implement those methods.	6 M
Ans	<pre>import java.util.Scanner; import static java.lang.Math.pow; interface Interest { int roi=25; public void simpleInterest(float principle,float time); public void compoundInterest(float principle,float time); } public class InterestTest implements Interest { public void simpleInterest(float principle,float time) { float si = (principle*roi*time)/100; </pre>	Creating correct interface with-2M Implementing interface-1M Calculating simple interest and compound interest-2M



	<pre>System.out.println("Simple interested calculate by program is : " + si); } public void compoundInterest(float principle,float time) { double ci = principle * (Math.pow((1.0 +(roi/100)), time)) - principle; System.out.println("Compound interested calculate by program is : " + ci); } public static void main(String args[]) { InterestTest i1 = new InterestTest(); i1.simpleInterest(1000,2); i1.compoundInterest(1000,2); } }</pre>	Correct Main method-1M
c)	Write a program that throws an exception called "NoMatchException" when a string is not equal to "India".	6 M
Ans	<pre>import java.io.*; class NoMatchException extends Exception { private String str; NoMatchException(String str1) { str=str1; } public String toString() { return "NoMatchException --> String is not India and string is "+str; } } class Main { public static void main(String args[]) { String str1= new String("India"); String str2= new String("Australia"); try { if(str1.equals("India")) System.out.println(" String is : "+str1); else throw new NoMatchException(str1); if(str2.equals("India")) System.out.println("\n String is : "+str2);</pre>	Any Correct program – 6 M



	<pre>else throw new NoMatchException(str2); } catch(NoMatchException e) { System.out.println("\nCaught.... "+e); } } } OUTPUT: String is : India Caught.... NoMatchException --> String is not India and string is Australlia</pre>	
6.	Attempt any <u>TWO</u> of the following:	12 M
a)	Write a program to print the sum, difference and product of two complex numbers by creating a class named "Complex" with separate methods for each operation whose real and imaginary parts are entered by user.	6 M
Ans	<pre>// Java program to add and subtract two // complex numbers using Class import java.util.*; // User Defined Complex class class Complex { // Declaring variables int real, imaginary; // Empty Constructor Complex() { } // Constructor to accept // real and imaginary part Complex(int tempReal, int tempImaginary) { real = tempReal; imaginary = tempImaginary; } // Defining addComp() method // for adding two complex number Complex addComp(Complex C1, Complex C2) { // creating temporary variable</pre>	Correct program – 6 M



```
Complex temp = new Complex();

// adding real part of complex numbers
temp.real = C1.real + C2.real;

// adding Imaginary part of complex numbers
temp.imaginary = C1.imaginary + C2.imaginary;

// returning the sum
return temp;
}

// Defining subtractComp() method
// for subtracting two complex number
Complex subtractComp(Complex C1, Complex C2)
{
    // creating temporary variable
    Complex temp = new Complex();

    // subtracting real part of complex numbers
    temp.real = C1.real - C2.real;

    // subtracting Imaginary part of complex numbers
    temp.imaginary = C1.imaginary - C2.imaginary;

    // returning the difference
    return temp;
}

Complex productComp(Complex C1, Complex C2)
{
    // creating temporary variable
    Complex temp = new Complex();

    // product of of complex numbers
    //(a + ib) (c + id)= (ac - bd) + i(ad + bc).
    temp.real = ((C1.real*C2.real)-(C1.imaginary*C2.imaginary));
    temp.imaginary = ((C1.real*C2.imaginary) + (C1.imaginary*C2.real));

    // returning the difference
    return temp;
}

// Function for printing complex number
void printComplexNumber()
{
    System.out.println("Complex number: " + real + " + " + imaginary + "i");
}
}
```



```
// Main Class
public class Main
{

    // Main function
    public static void main(String[] args)
    {

        // First Complex number
        Complex C1 = new Complex(3, 2);

        // printing first complex number
        C1.printComplexNumber();

        // Second Complex number
        Complex C2 = new Complex(9, 5);

        // printing second complex number
        C2.printComplexNumber();

        // for Storing the sum
        Complex C3 = new Complex();

        // calling addComp() method
        C3 = C3.addComp(C1, C2);

        // printing the sum
        System.out.print("Sum of ");
        C3.printComplexNumber();

        // calling subtractComp() method
        C3 = C3.subtractComp(C1, C2);

        // printing the difference
        System.out.print("Difference of ");
        C3.printComplexNumber();

        // calling productComp() method
        C3 = C3.productComp(C1, C2);

        // printing the product
        System.out.print("product of ");
        C3.printComplexNumber();

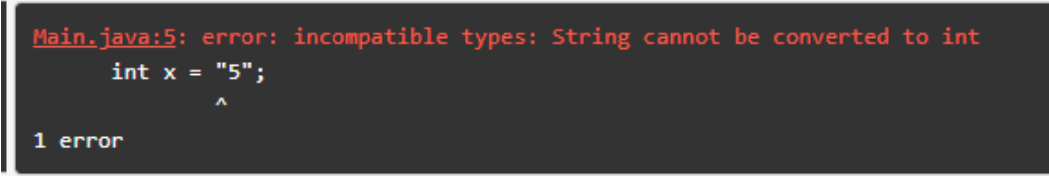
    }
}
```

OUTPUT:

Complex number: 3 + 2i

Complex number: 9 + 5i



	Sum of Complex number: $12 + 7i$ Difference of Complex number: $-6 + -3i$ product of Complex number: $17 + 33i$	
b)	i) Explain Errors and its types in detail. ii) Explain thread methods to set and get priority.	6 M
Ans	<p>An error is an issue in a program that prevents the program from completing its task. There are several types of errors that occur in Java, including syntax errors, runtime errors, and logical errors. They are</p> <ul style="list-style-type: none">● Syntax Errors or Compilation Errors: These occur when the code violates the rules of the Java syntax. These errors are usually caught by the Java compiler during the compilation phase.● Example of compile time error: public class Main { public static void main(String[] args) { int x = "5"; } } <p>OUTPUT:</p>  <pre>Main.java:5: error: incompatible types: String cannot be converted to int int x = "5"; ^ 1 error</pre> <ul style="list-style-type: none">● Runtime Errors: These errors occur when the code encounters an unexpected behaviour during its execution. These errors are usually caused by flawed logic or incorrect assumptions in the code and can be difficult to identify and fix.● The most common run-time errors are:<ol style="list-style-type: none">a) Dividing an integer by zerob) Accessing an element that is out of bounds of an arrayc) Trying to store value into an array of an incompatible class or type Passing parameter that is not in a valid range or value for methodd) Trying to illegally change status of threade) Attempting to use a negative size for an arrayf) Converting invalid string to a numberg) Accessing character that is out of bound of a string <p>These errors can be handled by uexception handling with help of try-catch- final block</p>	Types of errors with example – 3 M and thread methods with any relevant/correct example – 3 M



(i) **Priorities in threads**

To get and set priority of a thread in java following methods are used,

1. **public final int getPriority():** java.lang.Thread.getPriority() method returns priority of given thread.
2. **public final void setPriority(int newPriority):** java.lang.Thread.setPriority() method changes the priority of thread to the value newPriority. This method throws IllegalArgumentException if value of parameter newPriority goes beyond minimum(1) and maximum(10) limit.

Example:

// Java Program to Illustrate Priorities in Multithreading
// via help of getPriority() and setPriority() method

// Importing required classes
import java.lang.*;

// Main class
class ThreadDemo extends Thread {

```
    // Method 1
    // run() method for the thread that is called
    // as soon as start() is invoked for thread in main()
    public void run()
    {
        // Print statement
        System.out.println("Inside run method");
    }
```

```
    // Main driver method
    public static void main(String[] args)
    {
        // Creating random threads
        // with the help of above class
        ThreadDemo t1 = new ThreadDemo();
        ThreadDemo t2 = new ThreadDemo();
        ThreadDemo t3 = new ThreadDemo();

        // Thread 1
        // Display the priority of above thread using getPriority() method
        System.out.println("t1 thread priority : " + t1.getPriority());

        // Thread 2
        // Display the priority of above thread
        System.out.println("t2 thread priority : " + t2.getPriority());

        // Thread 3
        System.out.println("t3 thread priority : " + t3.getPriority());
    }
```



	<pre>// Setting priorities of above threads by passing integer arguments t1.setPriority(2); t2.setPriority(5); t3.setPriority(8); System.out.println("t1 thread priority : "+ t1.getPriority()); System.out.println("t2 thread priority : "+ t2.getPriority()); System.out.println("t3 thread priority : "+ t3.getPriority()); // Main thread // Displays the name of currently executing Thread System.out.println("Currently Executing Thread : " + Thread.currentThread().getName()); System.out.println("Main thread priority : " + Thread.currentThread().getPriority()); // Main thread priority is set to 10 Thread.currentThread().setPriority(10); System.out.println("Main thread priority : " + Thread.currentThread().getPriority()); } }</pre> <p>OUTPUT: t1 thread priority : 5 t2 thread priority : 5 t3 thread priority : 5 t1 thread priority : 2 t2 thread priority : 5 t3 thread priority : 8 Currently Executing Thread : main Main thread priority : 5 Main thread priority : 10</p>	
c)	Write a program to draw a chessboard in Java Applet.	6 M
Ans	<pre>import java.applet.*; import java.awt.*; /*<applet code="Chess" width=600 height=600> </applet>*/ // Extends Applet Class public class Chess extends Applet { static int N = 10; // Use paint() method public void paint(Graphics g) {</pre>	Correct program – 6 M



		<pre>int x, y; for (int row = 0; row < N; row++) { for (int col = 0; col < N; col++) { // Set x coordinates of rectangle // by 20 times x = row * 20; // Set y coordinates of rectangle // by 20 times y = col * 20; // Check whether row and column are in even position // If it is true set Black color if ((row % 2 == 0) == (col % 2 == 0)) g.setColor(Color.BLACK); else g.setColor(Color.WHITE); // Create a rectangle with // length and breadth of 20 g.fillRect(x, y, 20, 20); } } }</pre>	
--	--	--	--



SUMMER – 2023 EXAMINATION
Model Answer – Only for the Use of RAC Assessors

Subject Name: Java Programming

Subject Code: 22412

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.
- 8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English + Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No .	Sub Q. N.	Answer	Marking Scheme
1		Attempt any <u>FIVE</u> of the following:	10 M
	a)	Define the terms with example. i) Class ii) Object	2 M
	Ans	i) <u>Class</u>: Class is a set of object, which shares common characteristics/ behavior and common properties/ attributes. ii) <u>Object</u>: It is a basic unit of Object-Oriented Programming and represents real-life entities. Example: <pre>class Student { int id; String name;</pre>	1 M for any suitable class definition and 1 M for any suitable object definition

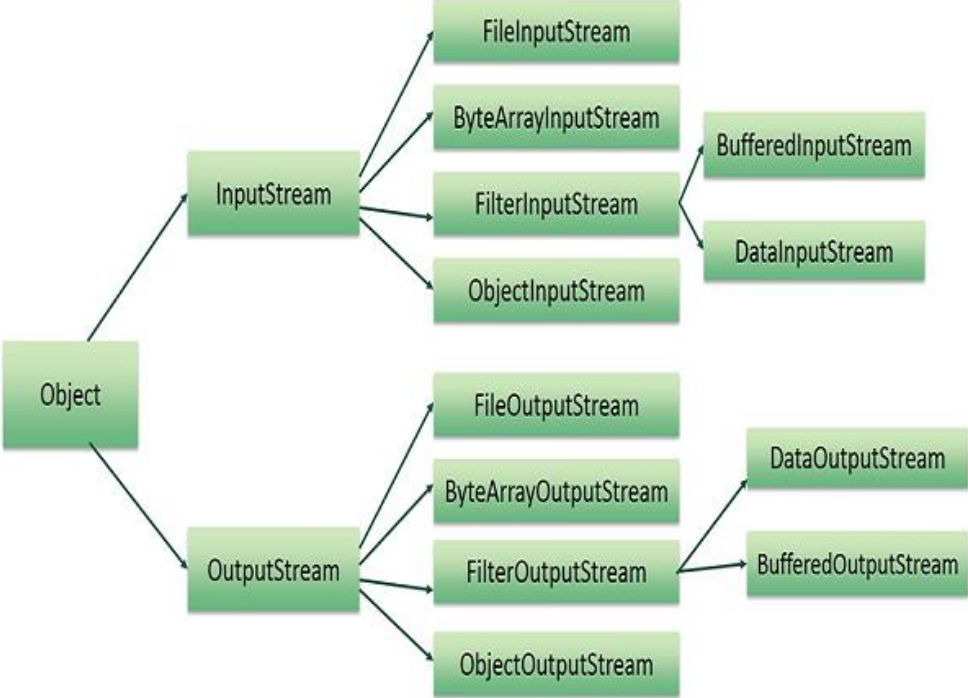


	<pre>public static void main(String args[]) { Student s1=new Student(); //creating an object of Student } }</pre> <p>In this example, we have created a Student class which has two data members id and name. We are creating the object of the Student s1 by new keyword.</p>	
b)	Enlist any two access specifier with syntax.	2 M
Ans	<p>There are 5 types of java access specifier:</p> <ul style="list-style-type: none">• public• private• default (Friendly)• protected• private protected	List any 2 access specifiers - 2M
c)	Give a syntax to create a package and accessing package in java.	2 M
Ans	<p><u>To Create a package follow the steps given below:</u></p> <ul style="list-style-type: none">• Choose the name of the package• Include the package command as the first line of code in your Java Source File.• The Source file contains the classes, interfaces, etc. you want to include in the package• Compile to create the Java packages <p><u>Syntax to create a package:</u></p> <pre>package nameOfPackage;</pre> <p>Example: package p1;</p> <p><u>Accessing Package:</u></p> <ul style="list-style-type: none">• Package can be accessed using keyword import.• There are 2 ways to access java system packages:<ul style="list-style-type: none">○ Package can be imported using import keyword and the wild card(*) but drawback of this shortcut approach is that it is difficult to determine from which package a particular member name. <p>Syntax: import package_name.*;</p>	syntax to create a package-1 M and accessing package-1 M



		For e.g. import java.lang.*; ○ The package can be accessed by using dot(.) operator and can be terminated using semicolon(; Syntax: import package1.package2.classname;	
	d)	Give a syntax of following thread method i) Notify () ii) Sleep ()	2 M
	Ans	i) notify() The notify() method of thread class is used to wake up a single thread. This method gives the notification for only one thread which is waiting for a particular object. Syntax: public final void notify() ii) sleep() Sleep() causes the current thread to suspend execution for a specified period. Syntax: public static void sleep(long milliseconds)	Syntax of notify()-1 M and sleep ()-1 M
	e)	Give a syntax of (param) tag to pass parameters to an applet.	2 M
	Ans	User-define Parameter can be applied in applet using <PARAM...> tags. Each <PARAM...> tag has a name and value attribute. Syntax: <PARAM name = Value = "....." > For example, the param tags for passing name and age parameters looks as shown below: <param name="name" value="Ramesh" /> <param name="age" value="25" /> The <i>getParameter()</i> method of the <i>Applet</i> class can be used to retrieve the parameters passed from the HTML page. The syntax of <i>getParameter()</i> method is as follows: String getParameter(String param-name); Example: public void init() { n = getParameter("name"); a = getParameter("age"); }	Syntax of <param> - 1 M And syntax of getParameter ()- 1 M



f)	Define stream class and list types of stream class.	2 M
Ans	<p>A java stream is a group of objects that can be piped together to produce the desired result. Streams are used in Java to transfer data between programs and I/O devices like a file, network connections, or consoles.</p>  <p style="text-align: center;">Fig: Types of stream classes</p>	<p>Define stream class=1 M</p> <p>and</p> <p>any 2 types of stream class=1 M</p>
g)	Give use of garbage collection in java.	2 M
Ans	<p>The garbage collector provides the following uses:</p> <ol style="list-style-type: none"> 1. Frees developers from having to manually release memory means destroy the unused objects 2. Allocates objects on the managed heap efficiently. 3. Reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. 4. It is automatically done by the garbage collector(a part of JVM), so we don't need extra effort. 	<p>Any 2 uses=2 M</p>
2.	Attempt any <u>THREE</u> of the following:	12 M
a)	Describe type casting in java with example.	4 M
Ans	In Java, type casting is a method or process that converts a data type into another data	Definition of



type in both ways manually and automatically. The automatic conversion is done by the compiler and manual conversion performed by the programmer.

Type casting is of two types: widening, narrowing.

Widening (Implicit)

- The process of assigning a smaller type to a larger one is known as widening or implicit.

Byte → short → int → long → float → double

For e.g.

```
class widening
{
    public static void main(String arg[])
    {
        int i=100;
        long l=i;
        float f=l;
        System.out.println("Int value is"+i);
        System.out.println("Long value is"+l);
        System.out.println("Float value is"+f);
    }
}
```

Narrowing (Explicit)

- The process of assigning a larger type into a smaller one is called narrowing.
- Casting into a smaller type may result in loss of data.
- double → long → int → short → byte

For e.g.

```
class narrowing
{
    Public static void main(String[])
    {
        Double d=100.04;
        Long l=(long) d;
        Int i=(int) l;
    }
}
```

type casting-
1 M

Types of
type

casting-1 M

Example-2
M

(1 M for
each
example)



		<pre>System.out.println("Int value is"+i); System.out.println("Long value is"+l); System.out.println("Float value is" } }</pre>															
	b)	Differentiate between String and String Buffer Class. (any four points)	4 M														
	Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">String class</th> <th style="width: 50%; text-align: center;">StringBuffer class</th> </tr> </thead> <tbody> <tr> <td>String is a major class</td> <td>StringBuffer is a peer class of String</td> </tr> <tr> <td>Length is fixed (immutable)</td> <td>Length is flexible (mutable)</td> </tr> <tr> <td>Contents of object cannot be modified</td> <td>Contents of object can be modified</td> </tr> <tr> <td>Object can be created by assigning String constants enclosed in double quotes.</td> <td>Objects can be created by calling constructor of StringBuffer class using "new"</td> </tr> <tr> <td>Methods of string class: toLowerCase(), toUpperCase(), replace(), trim(), equals(), length(), charAt(), concat(), substring(), compareTo()</td> <td>Methods of StringBuffer class: setCharAt(), append(), insert(), append(), reverse(), delete()</td> </tr> <tr> <td>Ex:- String s="abc";</td> <td>Ex:- StringBuffer s=new StringBuffer ("abc");</td> </tr> </tbody> </table>	String class	StringBuffer class	String is a major class	StringBuffer is a peer class of String	Length is fixed (immutable)	Length is flexible (mutable)	Contents of object cannot be modified	Contents of object can be modified	Object can be created by assigning String constants enclosed in double quotes.	Objects can be created by calling constructor of StringBuffer class using "new"	Methods of string class: toLowerCase(), toUpperCase(), replace(), trim(), equals(), length(), charAt(), concat(), substring(), compareTo()	Methods of StringBuffer class: setCharAt(), append(), insert(), append(), reverse(), delete()	Ex:- String s="abc";	Ex:- StringBuffer s=new StringBuffer ("abc");	Any 4 correct points-4 M
String class	StringBuffer class																
String is a major class	StringBuffer is a peer class of String																
Length is fixed (immutable)	Length is flexible (mutable)																
Contents of object cannot be modified	Contents of object can be modified																
Object can be created by assigning String constants enclosed in double quotes.	Objects can be created by calling constructor of StringBuffer class using "new"																
Methods of string class: toLowerCase(), toUpperCase(), replace(), trim(), equals(), length(), charAt(), concat(), substring(), compareTo()	Methods of StringBuffer class: setCharAt(), append(), insert(), append(), reverse(), delete()																
Ex:- String s="abc";	Ex:- StringBuffer s=new StringBuffer ("abc");																
	c)	Write a program to create a user defined exception in java.	4 M														
	Ans	<p>Following example shows a user defined exception as 'Invalid Age', if age entered by the user is less than eighteen.</p> <pre>import java.lang.Exception; import java.io.*; class myException extends Exception { myException(String msg) { super(msg); } } class agetest { public static void main(String args[]) { BufferedReader br=new BufferedReader(new InputStreamReader(System.in)); //Scanner class is also valid try { System.out.println("enter the age : "); int n=Integer.parseInt(br.readLine()); if(n < 18) throw new myException("Invalid Age"); //user defined exception</pre>	For any Correct program-4 M														



	<pre>else System.out.println("Valid age"); } catch(myException e) { System.out.println(e.getMessage()); } catch(IOException ie) {} } }</pre>	
d)	Write a program for reading and writing character to and from the given files using character stream classes.	4 M
Ans	<pre>import java.io.FileWriter; import java.io.IOException; public class IOStreamsExample { public static void main(String args[]) throws IOException { //Creating FileReader object File file = new File("D:/myFile.txt"); FileReader reader = new FileReader(file); char chars[] = new char[(int) file.length()]; //Reading data from the file reader.read(chars); //Writing data to another file File out = new File("D:/CopyOfmyFile.txt"); FileWriter writer = new FileWriter(out); //Writing data to the file writer.write(chars); writer.flush(); System.out.println("Data successfully written in the specified file"); } }</pre>	4 M (for any correct program and logic)
3.	Attempt any <u>THREE</u> of the following:	12 M
a)	Write a program to print all the Armstrong numbers from 0 to 999	4 M
Ans	<pre>import java.util.Scanner; class ArmstrongWhile { public static void main(String[] arg) { int i=0,arm; System.out.println("Armstrong numbers between 0 to 999"); while(i<1000)</pre>	Correct logic – 4 M



```
{
    arm=armstrongOrNot(i);
    if (arm==i)
        System.out.println(i);
    i++;
}
}

static int armstrongOrNot(int num)
{
    int x,a=0;
    while(num!=0)
    {
        x=num%10;
        a=a+(x*x*x);
        num/=10;
    }
    return a;
}
}

OR

class ArmstrongWhile
{
    public static void main(String[] arg)
    {
        int i=1,a,arm,n,temp;
        System.out.println("Armstrong numbers between 0 to 999 are");
        while(i<500)
        {
            n=i;
            arm=0;
            while(n>0)
            {
                a=n%10;
                arm=arm+(a*a*a);
                n=n/10;
            }
            if (arm==i)
                System.out.println(i);
            i++;
        }
    }
}
```

b) Explain the applet life cycle with neat diagram.

4 M



Ans

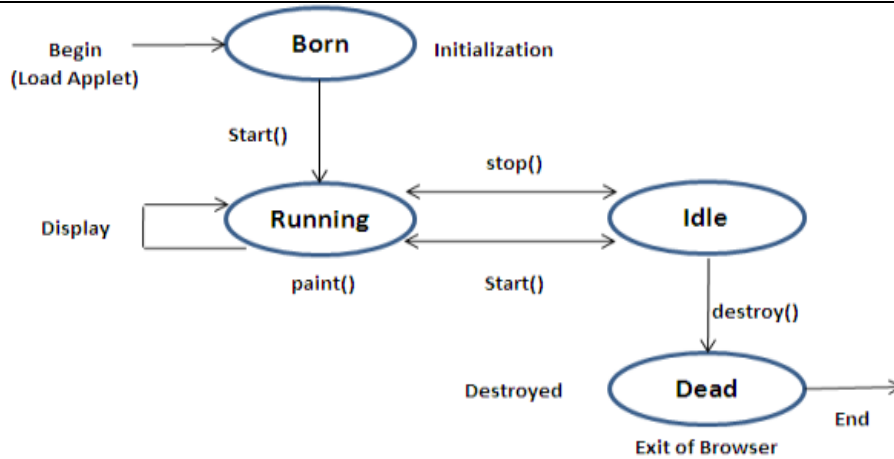


Diagram-2
M and
explanation
– 2 M

Applet Life Cycle: An Applet has a life cycle, which describes how it starts, how it operates and how it ends. The life cycle consists of four methods: `init()`, `start()`, `stop()` and `destroy()`.

Initialization State (The `init()` method):

The life cycle of an Applet is begin on that time when the applet is first loaded into the browser and called the `init()` method. The `init()` method is called only one time in the life cycle on an Applet. The `init()` method is basically called to read the “PARAM” tag in the html file. The `init ()` method retrieve the passed parameter through the “PARAM” tag of html file using `get Parameter()` method All the initialization such as initialization of variables and the objects like image, sound file are loaded in the `init ()` method .After the initialization of the `init()` method user can interact with the Applet and mostly applet contains the `init()` method.

We may do following thing if required.

- Create objects needed by the applet
- Set up initial values
- Load images or fonts
- Set up colors

Running State (The `start()` method): The start method of an Applet is called after the initialization method `init()`. This method may be called multiples time when the Applet needs to be started or restarted. For Example if the user wants to return to the Applet, in this situation the `start()` method of an Applet will be called by the web browser and the user will be back on the applet. In the start method user can interact within the applet.

```

public void start()
{
.....
.....
}
  
```

Idle (The `Stop()` method): An applet becomes idle when it is stopped from running. The `stop()` method stops the applet and makes it invisible. Stopping occurs automatically when we leave the page containing the currently running applet. We can also do so by calling the `stop()` method explicitly. The `stop()` method can be called multiple times in the life cycle of applet like the `start ()` method or should be called at least one time. For example the `stop()` method is called by the web browser on that time When the user leaves one applet to go another applet and the `start()` method is called on that time when the user wants to go back into the first program or Applet.

```

public void stop()
  
```



```
{
.....
.....
}
```

Dead State (The destroy() method): The destroy() method is called to terminate an Applet. An Applet is said to be dead when it is removed from memory. This occurs automatically by invoking the destroy() method when we quit the browser. It is useful for clean-up actions, such as releasing memory after the applet is removed, killing off threads and closing network/database connections.

Thus this method releases all the resources that were initialized during an applet's initialization.

```
public void destroy()
{
.....
.....
}
```

Display State (The paint() method): The paint() method is used for applet display on the screen.

The display includes text, images, graphics and background. This happens immediately after the applet enters into the running state. Almost every applet will have a paint() method and can be called several times during an applet's life cycle. The paint() method is called whenever a window is required to paint or repaint the applet.

```
public void paint(Graphics g)
{
.....
.....
}
```

c)	Describe the package in java with suitable example.	4 M
----	--	------------

Ans	<ul style="list-style-type: none"> • Java provides a mechanism for partitioning the class namespace into more manageable parts called package (i.e package are container for a classes). • The package is both naming and visibility controlled mechanism. Package can be created by including package as the first statement in java source code. • Any classes declared within that file will belong to the specified package. Syntax: package pkg; pkg is the name of the package eg : package mypack; • Java uses file system directories to store packages. The class files of any classes which are declared in a • package must be stored in a directory which has same name as package name. • The directory must match with the package name exactly. • A hierarchy can be created by separating package name and sub package name by a period(.) as pkg1.pkg2.pkg3; which requires a directory structure as pkg1\pkg2\pkg3. • The classes and methods of a package must be public. • To access package In a Java source file, import statements occur immediately following the package. statement (if it exists) and before any class definitions. • Syntax: import pkg1[.pkg2].(classname *); 	Description- 2 M, Example -2 M
------------	---	---



	<ul style="list-style-type: none">• Example: package l; package package1; public class Box { int l= 5; int b = 7; int h = 8; public void display() { System.out.println("Volume is:"+(l*b*h)); } } Source file: import package1.Box; class VolumeDemo { public static void main(String args[]) { Box b=new Box(); b.display(); } }	
d)	Enlist types of Byte stream class and describe input stream class and output stream class.	4 M
Ans	<ul style="list-style-type: none">• Byte streams class: It handles I/O operations on bytes.• InputStream and OutputStream classes are operated on bytes for reading and writing, respectively.• Byte streams are used in a program to read and write 8-bit bytes.• InputStream and OutputStream are the abstract super classes of all byte streams that have a sequential nature.• The stream is unidirectional; they can transmit bytes in only one direction.• InputStream and OutputStream provide the Application program Interface (API) and partial implementation for input streams (streams that read bytes) and output streams (streams that write bytes).• Input Stream Classes: java.io.InputStream is an abstract class that contains the basic methods for reading raw bytes of data from a stream. The InputStream class defines methods for performing the input functions like: reading bytes, closing streams, marking positions in streams, skipping ahead in a stream and finding the number of bytes in a stream.• Input stream class methods:<ol style="list-style-type: none">1. int read ()- Returns an integer representation of next available byte of input.-1 is returned at the stream end.2. int read (byte buffer[])- Read up to buffer.length bytes into buffer & returns actual number	Type – 1 M, Explanation -3 M



	<p>of bytes that are read. At the end returns -1.</p> <p>3. int read(byte buffer[], int offset, int numbytes)- Attempts to read up to numbytes bytes into buffer starting at buffer[offset]. Returns actual number of bytes that are read. At the end returns -1.</p> <p>4. void close()- to close the input stream</p> <p>5. void mark(int numbytes)- places a mark at current point in input stream and remain valid till number of bytes are read.</p> <p>6. void reset()- Resets pointer to previously set mark/ goes back to stream beginning.</p> <p>7. long skip(long numbytes)- skips number of bytes.</p> <p>8. int available()- Returns number of bytes currently available for reading.</p> <ul style="list-style-type: none">• Output Stream Classes:• The java.io.OutputStream class sends raw bytes of data to a target such as the console or a network server. Like InputStream, OutputStream is an abstract class.• The OutputStream includes methods that perform operations like: writing bytes, closing streams, flushing streams etc.• Methods defines by the OutputStream class are<ol style="list-style-type: none">1. void close() - to close the OutputStream2. void write (int b) - Writes a single byte to an output stream.3. void write(byte buffer[]) - Writes a complete array of bytes to an output stream.4. void write (byte buffer[], int offset, int numbytes) - Writes a sub range of numbytes bytes from the array buffer, beginning at buffer[offset].5. void flush() - clears the buffer.	
4.	Attempt any <u>THREE</u> of the following:	12 M
a)	Describe any four features of java.	4 M
Ans	<p>1. Compile & Interpreted: Java is a two staged system. It combines both approaches. First java compiler translates source code into byte code instruction. Byte codes are not machine instructions. In the second stage java interpreter generates machine code that can be directly executed by machine. Thus java is both compile and interpreted language.</p> <p>2. Platform independent and portable: Java programs are portable i.e. it can be easily moved from one computer system to another. Changes in OS, Processor, system resources won't force any change in java programs. Java compiler generates byte code instructions that can be implemented on any machine as well as the size of primitive data type is machine independent.</p> <p>3. Object Oriented: Almost everything in java is in the form of object. All program codes and data reside within objects and classes. Similar to other OOP languages java also has basic OOP properties such as encapsulation, polymorphism, data abstraction, inheritance etc. Java comes with an extensive set of classes (default) in packages.</p> <p>4. Robust & Secure: Java is a robust in the sense that it provides many safeguards to ensure reliable codes. Java incorporates concept of exception handling which captures errors and eliminates any risk of crashing the system. Java system not only verify all memory access but also ensure that no viruses are communicated with an applet. It does not use pointers by which you can gain access to memory locations without proper</p>	Any four each features -1 M each



authorization.

5. Distributed: It is designed as a distributed language for creating applications on network. It has ability to share both data and program. Java application can open and access remote object on internet as easily as they can do in local system.

6. Multithreaded: It can handle multiple tasks simultaneously. Java makes this possible with the feature of multithreading. This means that we need not wait for the application to finish one task before beginning other.

7. Dynamic and Extensible: Java is capable of dynamically linking new class library's method and object. Java program supports function written in other languages such as C, C++ which are called as native methods. Native methods are linked dynamically at run time.

b) Explain any four methods of vector class with example.

4 M

Ans

Vector class is in java.util package of java.

Vector is dynamic array which can grow automatically according to the requirement.

Vector does not require any fix dimension like String array and int array.

Vectors are used to store objects that do not have to be homogeneous.

Vector contains many useful methods.

Vectors are created like arrays. It has three constructor methods

- Vector list = new Vector(); //declaring vector without size
- Vector list = new Vector(3); //declaring vector with size
- Vector list = new Vector(5,2); //create vector with initial size and whenever it need to grows, it grows by value specified by increment capacity.

Method Name	Task performed
list.firstElement()	It returns the first element of the vector.
list.lastElement()	It returns last element of the vector
list.addElement(item)	Adds the item specified to the list at the end.
list.elementAt(n)	Gives the name of the object at nth position
list.size()	Gives the number of objects present in vector
List.capacity()	This method returns the current capacity of the vector.
list.removeElement(item)	Removes the specified item from the list.
list.removeElementAt(n)	Removes the item stored in the nth position of the list.
list.removeAllElements()	Removes all the elements in the list.
list.insertElementAt(item, n)	Inserts the item at nth position.
List.contains(object element)	This method checks whether the specified element is present in the Vector. If the element is been found it returns true else false.
list.copyInto(array)	Copies all items from list of array.

Example:

1 Method – 1
M



```
import java.io.*;
import
java.lang.*;
import
java.util.*;
class vector2
{
public static void main(String args[])
{
vector v=new
vector(); Integer
s1=new Integer(1);
Integer s2=new
Integer(2);String
s3=new
String("fy");String
s4=new
String("sy");
Character s5=new
Character('a');Character
s6=new Character('b');
Float s7=new
Float(1.1f);
Float s8=new Float(1.2f);
v.addElement(s1);
v.addElement(s2);
v.addElement(s3);
v.addElement(s4);
v.addElement(s5);
v.addElement(s6);
v.addElement(s7);
v.addElement(s8);
System.out.println(v);
v.removeElement(s2);
v.removeElementAt(4);
System.out.println(v);
}
}
```

c) **Describe interface in java with suitable example.**

4 M

Ans Java does not support multiple inheritances with only classes. Java provides an alternate approach known as interface to support concept of multiple inheritance. An interface is similar to class which can define only abstract methods and final variables.

Syntax:

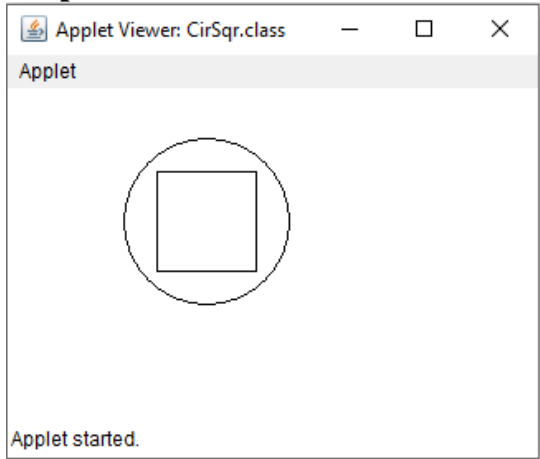
```
access interface InterfaceName
{
Variables declaration;
Methods declaration;
```

Interface explanation – 2 M, any suitable example – 2 M



```
}  
Example:  
interface sports  
{  
int sport_wt=5;  
public void disp();  
}  
class test  
{  
int roll_no;  
String name;  
int m1,m2;  
test(int r, String nm, int m11,int m12)  
{  
roll_no=r;  
name=nm;  
m1=m11;  
m2=m12;  
}  
}  
class result extends test implements sports  
{  
result (int r, String nm, int m11,int m12)  
{  
super (r,nm,m11,m12);  
}  
public void disp()  
{  
System.out.println("Roll no : "+roll_no);  
System.out.println("Name : "+name);  
System.out.println("sub1 : "+m1);  
System.out.println("sub2 : "+m2);  
System.out.println("sport_wt : "+sport_wt);  
int t=m1+m2+sport_wt;  
System.out.println("total : "+t);  
}  
public static void main(String args[])  
{  
result r= new result(101,"abc",75,75);  
r.disp();  
}  
}  
Output :  
D:\>java result  
Roll no : 101  
Name : abc  
sub1 : 75  
sub2 : 75  
sport_wt : 5
```



	total : 155	
d)	Write an applet program for following graphics method. i) Drawoval () ii) Drawline ()	4 M
Ans	<pre>import java.awt.*; import java.applet.*; public class CirSqr extends Applet { public void paint(Graphics g) { g.drawOval(70,30,100,100); g.drawRect(90,50,60,60); } } /*<applet code="CirSqr.class" height=200 width=200> </applet> */</pre> <p>Output:</p> 	Any correct logic can be considered 2 M for drawoval and 2 M for drawline
e)	Enlist any four methods of file input stream class and give syntax of any two methods.	4 M
Ans	<ul style="list-style-type: none">• Input Stream Classes: java.io.InputStream is an abstract class that contains the basic methods for reading raw bytes of data from a stream. The InputStream class defines methods for performing the input functions like: reading bytes, closing streams, marking positions in streams, skipping ahead in a stream and finding the number of bytes in a stream.• Input stream class methods:<ol style="list-style-type: none">1. int read ()- Returns an integer representation of next available byte of input.-1 is returned at the stream end.2. int read (byte buffer[])- Read up to buffer.length bytes into buffer & returns actual number of bytes that are read. At the end returns -1.3. int read(byte buffer[], int offset, int numbytes)- Attempts to read up to numbytes bytes into buffer starting at buffer[offset]. Returns actual number of bytes that are read.	One method – 1 M



		<p>At the end returns -1.</p> <p>4. void close()- to close the input stream</p> <p>5. void mark(int numbytes)- places a mark at current point in input stream and remain valid till number of bytes are read.</p> <p>6. void reset()- Resets pointer to previously set mark/ goes back to stream beginning.</p> <p>7. long skip(long numbytes)- skips number of bytes.</p> <p>8. int available()- Returns number of bytes currently available for reading.</p>	
5.		Attempt any <u>TWO</u> of the following:	12 M
	a)	Write a program to copy all elements of one array into another array.	6 M
	Ans	<pre>public class CopyArray { public static void main(String[] args) { int [] arr1 = new int [] {1, 2, 3, 4, 5}; int arr2[] = new int[arr1.length]; for (int i = 0; i < arr1.length; i++) { arr2[i] = arr1[i]; } System.out.println("Elements of original array: "); for (int i = 0; i < arr1.length; i++) { System.out.print(arr1[i] + " "); } System.out.println(); System.out.println("Elements of new array: "); for (int i = 0; i < arr2.length; i++) { System.out.print(arr2[i] + " "); } } }</pre>	6 M for any correct program and logic



```
}  
}  
}
```

b) Write a program to implement the following inheritance. Refer Fig. No. 1.

6 M

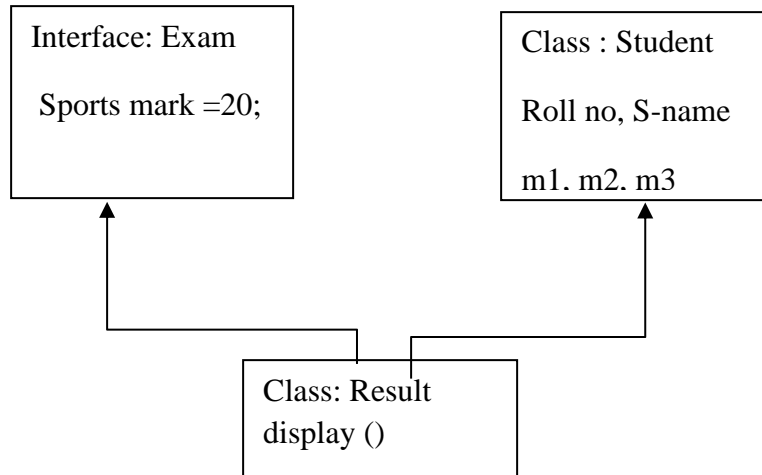


Fig. No. 1

Ans

```
interface Exam
{
    int sports_mark=20;
}

class Student
{
    String S-name;
    int Roll_no, m1, m2, m3;
    Student(String n, int a, int b, int c, int d)
}

S-name = n;
Roll_no = a;
```

6 M for
correct
program



```
m1 = b;

m2 = c;

m3 = d;

}

void showdata()

{

System.out.println("Name of student :"+S-name);

System.out.println("Roll no. of the students :"+Roll_no);

System.out.println("Marks of subject 1:"+m1);

System.out.println("Marks of subject 2:"+m2);

System.out.println("Marks of subject 3:"+m3);

}

}

class Result extends Student implements Exam

{

Result(String n, int a, int b, int c, int d)

{

super(n, a, b, c, d );

}

void dispaly()

{

super.showdata();

int total=(m1+m2+m3);

float result=(total+Sports_mark)/total*100;
```



	<pre>System.out.println("result of student is:"+result); } } class studentsDetails { public static void main(String args[]) { Result r=new Result("Sachin",14, 78, 85, 97); r.display(); } }</pre>	
c)	Write a program to print even and odd number using two threads with delay of 1000ms after each number.	6 M
Ans	<pre>class odd extends Thread { public void run() { for(int i=1;i<=20;i=i+2) { System.out.println("ODD="+i); try { sleep(1000); } catch(Exception e) { System.out.println("Error"); }</pre>	6 M for correct program



```
}  
}  
}  
}  
class even extends Thread  
{  
public void run()  
{  
for(int i=0;i<=20;i=i+2)  
{  
System.out.println("EVEN="+i);  
try  
{  
sleep(1000);  
}  
catch(Exception e)  
{  
System.out.println("Error");  
}  
}  
}  
}  
class oddeven  
{  
public static void main(String arg[])  
{  
odd o=new odd();  
even e=new even();
```



```
o.start();  
e.start();  
}  
}
```

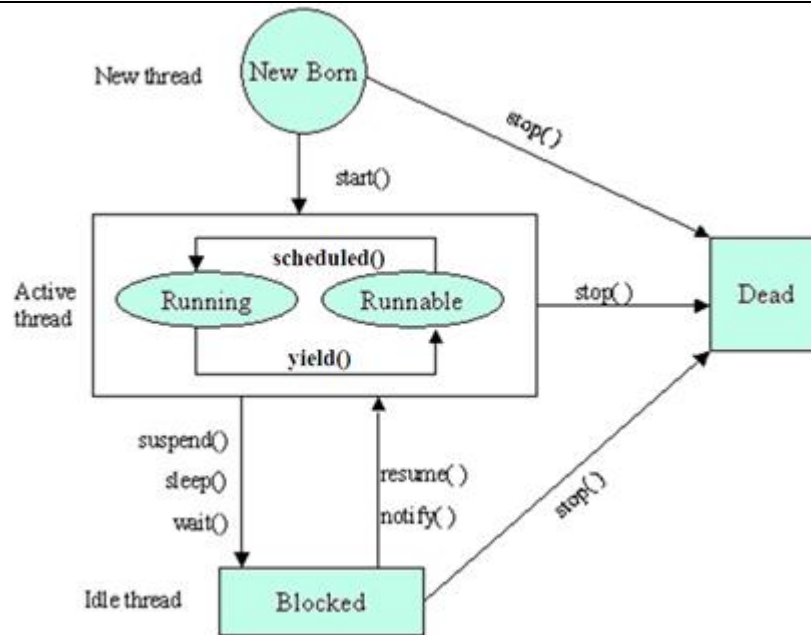
6. Attempt any TWO of the following:

12 M

a) Explain thread life cycle with neat diagram.

6 M

Ans



2 M for diagram, 4 M for explanation

Thread Life Cycle Thread has five different states throughout its life.

1) Newborn State

When a thread object is created it is said to be in a new born state. When the thread is in a new born state it is not scheduled running from this state it can be scheduled for running by `start()` or killed by `stop()`. If put in a queue it moves to runnable state.

2) Runnable State

It means that thread is ready for execution and is waiting for the availability of the processor i.e. the thread has joined the queue and is waiting for execution. If all threads have equal priority then they are given time slots for execution in round robin fashion. The thread that relinquishes control joins the queue at the end and again waits for its turn. A thread can relinquish the control to another before its turn comes by `yield()`.



3) Running State

It means that the processor has given its time to the thread for execution. The thread runs until it relinquishes control on its own or it is pre-empted by a higher priority thread.

4) Blocked State

A thread can be temporarily suspended or blocked from entering into the runnable and running state by using either of the following thread method.

o suspend() : Thread can be suspended by this method. It can be rescheduled by resume().

o wait(): If a thread requires to wait until some event occurs, it can be done using wait method and can be scheduled to run again by notify().

o sleep(): We can put a thread to sleep for a specified time period using sleep(time) where time is in ms. It reenters the runnable state as soon as period has elapsed /over.

5) Dead State

Whenever we want to stop a thread from running further we can call its stop(). The stop() causes the thread to move to a dead state. A thread will also move to dead state automatically when it reaches to end of the method. The stop method may be used when the premature death is required

Thread should be in any one state of above and it can be move from one state to another by different methods and ways.

b) **Write a program to generate following output using drawline () method. Refer Fig. No. 2.**

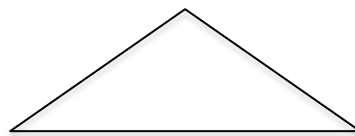


Fig. No. 2

6 M

Ans

Using drawLine() method

```
import java.applet.*;
import java.awt.*;
public class Triangle extends Applet
{
public void paint(Graphics g)
{
g.drawLine(100,200,200,100);
```

6 M for
correct
program



```
g.drawLine(200,100,300,200);  
g.drawLine(300,200,100,200);  
}  
}  
/*<applet code="Triangle.class" height=300 width=200> </applet>*/
```

OR

Using drawPolygon() method

```
import java.applet.*;  
import java.awt.*;  
public class Triangle extends Applet  
{  
public void paint(Graphics g)  
{  
int a[]={ 100,200,300,100};  
int b[]={ 200,100,200,200};  
int n=4;  
g.drawPolygon(a,b,n);  
}  
}  
/*<applet code="Triangle.class" height=300 width=200> </applet>*/
```

c) **Explain constructor with its type. Give an example of parameterized constructor.**

6 M

Ans **Constructor:** A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.

Types of constructors are:

Default constructor : It is constructor which is inserted by Java compiler when no constructor is provided in class. Every class has constructor within it. Even abstract class have default constructor.

By default, Java compiler, insert the code for a zero parameter constructor.

2 M for definition of constructor and types of constructors

4 M for example (for any correct suitable program of parameterize



Default constructor is the no arguments constructor automatically generated unless you define another constructor.

The default constructor automatically initializes all numeric members to zero and other types to null or spaces.

```
class Rect
```

```
{  
    int length, breadth;
```

```
    Rect() //constructor
```

```
{  
    length=4;  
    breadth=5;  
}
```

```
public static void main(String args[])
```

```
{  
    Rect r = new Rect();  
    System.out.println("Area : " +(r.length*r.breadth));  
}
```

Parameterized constructor: Constructor which have arguments are known as parameterized constructor.

When constructor method is defined with parameters inside it, different value sets can be provided to different constructor with the same name.

Example of Parameterized Constructor

```
class Rect
```

```
{  
    int length, breadth;
```

```
    Rect(int l, int b) // parameterized constructor
```

```
{  
    length=l;  
    breadth=b;  
}
```

```
public static void main(String args[])
```

```
{  
    Rect r = new Rect(4,5); // constructor with parameters  
    Rect r1 = new Rect(6,7);  
    System.out.println("Area : " +(r.length*r.breadth));
```

d
constructor)



		<pre>System.out.println("Area : " +(r1.length*r1.breadth)); } }</pre>	
--	--	---	--



WINTER – 2022 EXAMINATION

Subject Name: Java Programming

Model Answer

Subject Code: 22412

Important Instructions to examiners:

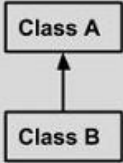
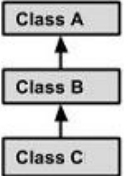
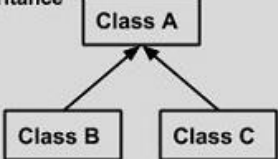
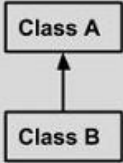
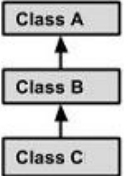
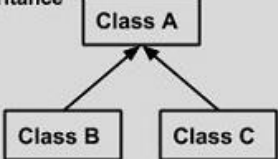
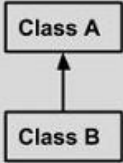
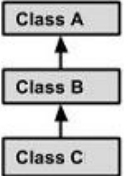
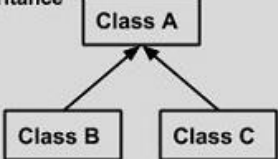
- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.
- 8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q. N.	Answer	Marking Scheme														
1		Attempt any FIVE of the following:	10 M														
	a)	State any four relational operators and their use.	2 M														
	Ans	<table border="1"> <thead> <tr> <th>Operator</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td><</td> <td>Less than</td> </tr> <tr> <td>></td> <td>Greater than</td> </tr> <tr> <td><=</td> <td>Less than or equal to</td> </tr> <tr> <td>>=</td> <td>Greater than or equal to</td> </tr> <tr> <td>= =</td> <td>Equal to</td> </tr> <tr> <td>!=</td> <td>Not equal to</td> </tr> </tbody> </table>	Operator	Meaning	<	Less than	>	Greater than	<=	Less than or equal to	>=	Greater than or equal to	= =	Equal to	!=	Not equal to	2M (1/2 M each) Any Four
Operator	Meaning																
<	Less than																
>	Greater than																
<=	Less than or equal to																
>=	Greater than or equal to																
= =	Equal to																
!=	Not equal to																
	b)	Enlist access specifiers in Java.	2 M														
	Ans	<p>The access specifiers in java specify accessibility (scope) of a data member, method, constructor or class. There are 5 types of java access specifier:</p> <ul style="list-style-type: none"> • public • private 	2M (1/2 M each) Any Four														



	<ul style="list-style-type: none">• default (Friendly)• protected• private protected	
c)	Explain constructor with suitable example.	2 M
Ans	<p>Constructors are used to assign initial value to instance variable of the class. It has the same name as class name in which it resides and it is syntactically similar to any method. Constructors do not have return value, not even 'void' because they return the instance of class. Constructor called by new operator.</p> <p>Example:</p> <pre>class Rect { int length, breadth; Rect() //constructor { length=4; breadth=5; } public static void main(String args[]) { Rect r = new Rect(); System.out.println("Area : " +(r.length*r.breadth)); } }</pre> <p>Output : Area : 20</p>	1M- Explanation 1M- Example
d)	List the types of inheritance which is supported by java.	2 M
Ans		Any two 1 M each



	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%; padding: 5px;">Single Inheritance</td> <td style="padding: 5px;">  <pre> graph BT B[Class B] --> A[Class A] </pre> </td> <td style="padding: 5px;"> <pre> public class A { } public class B extends A { } </pre> </td> </tr> <tr> <td style="padding: 5px;">Multi Level Inheritance</td> <td style="padding: 5px;">  <pre> graph BT C[Class C] --> B[Class B] B --> A[Class A] </pre> </td> <td style="padding: 5px;"> <pre> public class A {} public class B extends A {.....} public class C extends B {.....} </pre> </td> </tr> <tr> <td style="padding: 5px;">Hierarchical Inheritance</td> <td style="padding: 5px;">  <pre> graph BT B[Class B] --> A[Class A] C[Class C] --> A </pre> </td> <td style="padding: 5px;"> <pre> public class A {} public class B extends A {.....} public class C extends A {.....} </pre> </td> </tr> </table>	Single Inheritance	 <pre> graph BT B[Class B] --> A[Class A] </pre>	<pre> public class A { } public class B extends A { } </pre>	Multi Level Inheritance	 <pre> graph BT C[Class C] --> B[Class B] B --> A[Class A] </pre>	<pre> public class A {} public class B extends A {.....} public class C extends B {.....} </pre>	Hierarchical Inheritance	 <pre> graph BT B[Class B] --> A[Class A] C[Class C] --> A </pre>	<pre> public class A {} public class B extends A {.....} public class C extends A {.....} </pre>	
Single Inheritance	 <pre> graph BT B[Class B] --> A[Class A] </pre>	<pre> public class A { } public class B extends A { } </pre>									
Multi Level Inheritance	 <pre> graph BT C[Class C] --> B[Class B] B --> A[Class A] </pre>	<pre> public class A {} public class B extends A {.....} public class C extends B {.....} </pre>									
Hierarchical Inheritance	 <pre> graph BT B[Class B] --> A[Class A] C[Class C] --> A </pre>	<pre> public class A {} public class B extends A {.....} public class C extends A {.....} </pre>									
e)	Define thread. Mention 2 ways to create thread.	2 M									
Ans	<p>1. Thread is a smallest unit of executable code or a single task is also called as thread.</p> <p>2. Each tread has its own local variable, program counter and lifetime.</p> <p>3. A thread is similar to program that has a single flow of control.</p> <p><u>There are two ways to create threads in java:</u></p> <ol style="list-style-type: none"> By extending thread class Syntax: - <pre> class Mythread extends Thread { ---- } </pre> Implementing the Runnable Interface Syntax: <pre> class MyThread implements Runnable { public void run() { ----- } </pre> 	<p>1 M- Define Thread</p> <p>1M -2ways to create thread</p>									
f)	Distinguish between Java applications and Java Applet (Any 2 points)	2 M									



Ans	<table border="1"><thead><tr><th>Applet</th><th>Application</th></tr></thead><tbody><tr><td>Applet does not use main() method for initiating execution of code</td><td>Application use main() method for initiating execution of code</td></tr><tr><td>Applet cannot run independently</td><td>Application can run independently</td></tr><tr><td>Applet cannot read from or write to files in local computer</td><td>Application can read from or write to files in local computer</td></tr><tr><td>Applet cannot communicate with other servers on network</td><td>Application can communicate with other servers on network</td></tr><tr><td>Applet cannot run any program from local computer.</td><td>Application can run any program from local computer.</td></tr><tr><td>Applet are restricted from using libraries from other language such as C or C++</td><td>Application are not restricted from using libraries from other language</td></tr><tr><td>Applets are event driven.</td><td>Applications are control driven.</td></tr></tbody></table>	Applet	Application	Applet does not use main() method for initiating execution of code	Application use main() method for initiating execution of code	Applet cannot run independently	Application can run independently	Applet cannot read from or write to files in local computer	Application can read from or write to files in local computer	Applet cannot communicate with other servers on network	Application can communicate with other servers on network	Applet cannot run any program from local computer.	Application can run any program from local computer.	Applet are restricted from using libraries from other language such as C or C++	Application are not restricted from using libraries from other language	Applets are event driven.	Applications are control driven.	1 M for each point (any 2 Points)
Applet	Application																	
Applet does not use main() method for initiating execution of code	Application use main() method for initiating execution of code																	
Applet cannot run independently	Application can run independently																	
Applet cannot read from or write to files in local computer	Application can read from or write to files in local computer																	
Applet cannot communicate with other servers on network	Application can communicate with other servers on network																	
Applet cannot run any program from local computer.	Application can run any program from local computer.																	
Applet are restricted from using libraries from other language such as C or C++	Application are not restricted from using libraries from other language																	
Applets are event driven.	Applications are control driven.																	

g)	Draw the hierarchy of stream classes.	2 M
-----------	--	------------

Ans	<pre>graph LR; Object --> InputStream; Object --> OutputStream; InputStream --> FileInputStream; InputStream --> ByteArrayInputStream; InputStream --> FilterInputStream; InputStream --> ObjectInputStream; FilterInputStream --> BufferedInputStream; FilterInputStream --> DataInputStream; OutputStream --> FileOutputStream; OutputStream --> ByteArrayOutputStream; OutputStream --> FilterOutputStream; OutputStream --> ObjectOutputStream; FilterOutputStream --> DataOutputStream; FilterOutputStream --> BufferedOutputStream;</pre>	2M-Correct diagram
------------	---	--------------------

Fig: hierarchy of stream classes



2.		Attempt any THREE of the following:	12 M
	a)	Write a program to check whether the given number is prime or not.	4 M
	Ans	<p>Code:</p> <pre>class PrimeExample { public static void main(String args[]){ int i,m=0,flag=0; int n=7;//it is the number to be checked m=n/2; if(n==0 n==1){ System.out.println(n+" is not prime number"); }else{ for(i=2;i<=m;i++){ if(n%i==0){ System.out.println(n+" is not prime number"); flag=1; break; } } if(flag==0) { System.out.println(n+" is prime number"); } } } } } </pre> <p>Output:</p> <p>7 is prime number</p>	4M (for any correct program and logic)



	b) Define a class employee with data members 'empid , name and salary. Accept data for three objects and display it	4 M
Ans	<pre>class employee { int empid; String name; double salary; void getdata() { BufferedReader obj = new BufferedReader (new InputStreamReader(System.in)); System.out.print("Enter Emp number : "); empid=Integer.parseInt(obj.readLine()); System.out.print("Enter Emp Name : "); name=obj.readLine(); System.out.print("Enter Emp Salary : "); salary=Double.parseDouble(obj.readLine()); } void show() { System.out.println("Emp ID : " + empid); System.out.println("Name : " + name); System.out.println("Salary : " + salary); } } classEmpDetails { public static void main(String args[]) { employee e[] = new employee[3]; for(inti=0; i<3; i++) { e[i] = new employee(); e[i].getdata(); } System.out.println(" Employee Details are : "); for(inti=0; i<3; i++) e[i].show(); } }</pre>	4M (for correct program and logic)



c)	Describe Life cycle of thread with suitable diagram.	4 M
Ans	<p>1) Newborn State A NEW Thread (or a Born Thread) is a thread that's been created but not yet started. It remains in this state until we start it using the start() method.</p> <p>The following code snippet shows a newly created thread that's in the NEW state:</p> <pre>Runnable runnable = new NewState(); Thread t = new Thread(runnable);</pre> <p>2) Runnable State It means that thread is ready for execution and is waiting for the availability of the processor i.e. the thread has joined the queue and is waiting for execution. If all threads have equal priority, then they are given time slots for execution in round robin fashion. The thread that relinquishes control joins the queue at the end and again waits for its turn. A thread can relinquish the control to another before its turn comes by yield().<pre>Runnable runnable = new NewState(); Thread t = new Thread(runnable); t.start();</pre><p>3) Running State It means that the processor has given its time to the thread for execution. The thread runs until it relinquishes control on its own or it is pre-empted by a higher priority thread.</p><p>4) Blocked State A thread can be temporarily suspended or blocked from entering into the runnable and running state by using either of the following thread method.</p><ul style="list-style-type: none">○ suspend() : Thread can be suspended by this method. It can be rescheduled by resume().○ wait(): If a thread requires to wait until some event occurs, it can be done using wait method and can be scheduled to run again by notify().○ sleep(): We can put a thread to sleep for a specified time period using sleep(time) where time is in ms. It reenters the runnable state as soon as period has elapsed /over.<p>5) Dead State Whenever we want to stop a thread form running further we can call its stop(). The stop() causes the thread to move to a dead state. A thread will also move to dead state automatically when it reaches to end of the method. The stop method may be used when the premature death is required</p></p>	1M-digram of life cycle 3M- explanation

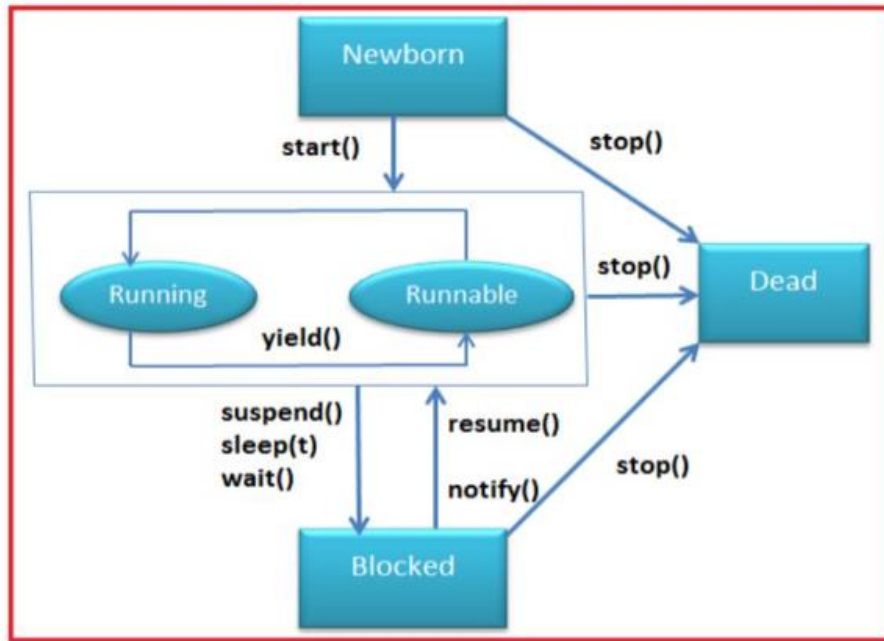


Fig: Life cycle of Thread

d) Write a program to read a file (Use character stream)

4 M

Ans

```
import java.io.FileWriter;
import java.io.IOException;
public class IOStreamsExample {
    public static void main(String args[]) throws IOException {
        //Creating FileReader object
        File file = new File("D:/myFile.txt");
        FileReader reader = new FileReader(file);
        char chars[] = new char[(int) file.length()];
        //Reading data from the file
        reader.read(chars);
        //Writing data to another file
        File out = new File("D:/CopyOfmyFile.txt");
        FileWriter writer = new FileWriter(out);
        //Writing data to the file
        writer.write(chars);
        writer.flush();
        System.out.println("Data successfully written in the specified file");
    }
}
```

4M (for correct program and logic)

3. Attempt any **THREE** of the following:

12 M



	a) Write a program to find reverse of a number.	4 M
Ans	<pre>public class ReverseNumberExample1 { public static void main(String[] args) { int number = 987654, reverse =0; while(number !=0) { int remainder = number % 10; reverse = reverse * 10 + remainder; number = number/10; } System.out.println("The reverse of the given number is: " + reverse); } } </pre>	Any Correct program with proper logic -4M
	b) State the use of final keyword with respect to inheritance.	4 M
Ans	<p>Final keyword : The keyword final has three uses. First, it can be used to create the equivalent of a named constant.(in interface or class we use final as shared constant or constant.)</p> <p>Other two uses of final apply to inheritance</p> <p>Using final to Prevent Overriding While method overriding is one of Java's most powerful features,</p> <p>To disallow a method from being overridden, specify final as a modifier at the start of its declaration. Methods declared as final cannot be overridden.</p> <p>The following fragment illustrates final:</p> <pre>class A { final void meth() { System.out.println("This is a final method."); } }</pre>	Use of final keyword-2 M Program-2 M



	<pre>} } class B extends A { void meth() { // ERROR! Can't override. System.out.println("Illegal!"); } }</pre> <p>As base class declared method as a final , derived class can not override the definition of base class methods.</p>	
c)	Give the usage of following methods i) drawPolygon () ii) DrawOval () iii) drawLine () iv) drawArc ()	4 M
Ans	i) drawPolygon (): <ul style="list-style-type: none">• drawPolygon() method is used to draw arbitrarily shaped figures.• Syntax: void drawPolygon(int x[], int y[], int numPoints)• The polygon's end points are specified by the co-ordinates pairs contained within the x and y arrays. The number of points define by x and y is specified by numPoints. <p>Example: int xpoints[]={ 30,200,30,200,30}; int ypoints[]={ 30,30,200,200,30}; int num=5; g.drawPolygon(xpoints,ypoints,num);</p> ii) drawOval (): <ul style="list-style-type: none">• To draw an Ellipses or circles used drawOval() method can be used.• Syntax: void drawOval(int top, int left, int width, int height) The ellipse is drawn within a bounding rectangle whose upper-left corner is specified by top and left and whose width and height are specified by width and height to draw a circle or filled circle, specify the same width and height the following program draws several ellipses and circle.• Example: g.drawOval(10,10,50,50); ii) drawLine (): <ul style="list-style-type: none">• The drawLine() method is used to draw line which take two pair of coordinates,	Method use with description 1 M



		<p>(x1,y1) and (x2,y2) as arguments and draws a line between them.</p> <ul style="list-style-type: none"> • The graphics object g is passed to paint() method. • Syntax: g.drawLine(x1,y1,x2,y2); • Example: g.drawLine(100,100,300,300;) <p>iv) drawArc () drawArc() It is used to draw arc.</p> <p>Syntax: void drawArc(int x, int y, int w, int h, int start_angle, int sweep_angle);</p> <p>where x, y starting point, w & h are width and height of arc, and start_angle is starting angle of arc sweep_angle is degree around the arc</p> <p>Example: g.drawArc(10, 10, 30, 40, 40, 90);</p>															
	d)	Write any four methods of file class with their use.	4 M														
	Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 35%;">public String getName()</td> <td>Returns the name of the file or directory denoted by this abstract pathname.</td> </tr> <tr> <td>public String getParent()</td> <td>Returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory</td> </tr> <tr> <td>public String getPath()</td> <td>Converts this abstract pathname into a pathname string.</td> </tr> <tr> <td>public boolean isAbsolute()</td> <td>Tests whether this abstract pathname is absolute. Returns true if this abstract pathname is absolute, false otherwise</td> </tr> <tr> <td>public boolean exists()</td> <td>Tests whether the file or directory denoted by this abstract pathname exists. Returns true if and only if the file or directory denoted by this abstract pathname exists; false otherwise</td> </tr> <tr> <td>public boolean isDirectory()</td> <td>Tests whether the file denoted by this abstract pathname is a directory. Returns true if and only if the file denoted by this abstract pathname exists and is a directory; false otherwise.</td> </tr> <tr> <td>public boolean isFile()</td> <td>Tests whether the file denoted by this abstract pathname is a normal file. A file is normal if it is not a directory and, in addition, satisfies other system-dependent criteria. Any nondirectory file created by a Java application is guaranteed to be a normal file. Returns true if and only if the file denoted by this abstract pathname exists and is a normal file; false otherwise.</td> </tr> </table>	public String getName()	Returns the name of the file or directory denoted by this abstract pathname.	public String getParent()	Returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory	public String getPath()	Converts this abstract pathname into a pathname string.	public boolean isAbsolute()	Tests whether this abstract pathname is absolute. Returns true if this abstract pathname is absolute, false otherwise	public boolean exists()	Tests whether the file or directory denoted by this abstract pathname exists. Returns true if and only if the file or directory denoted by this abstract pathname exists; false otherwise	public boolean isDirectory()	Tests whether the file denoted by this abstract pathname is a directory. Returns true if and only if the file denoted by this abstract pathname exists and is a directory; false otherwise.	public boolean isFile()	Tests whether the file denoted by this abstract pathname is a normal file. A file is normal if it is not a directory and, in addition, satisfies other system-dependent criteria. Any nondirectory file created by a Java application is guaranteed to be a normal file. Returns true if and only if the file denoted by this abstract pathname exists and is a normal file; false otherwise.	<p>One method</p> <p>1 M</p>
public String getName()	Returns the name of the file or directory denoted by this abstract pathname.																
public String getParent()	Returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory																
public String getPath()	Converts this abstract pathname into a pathname string.																
public boolean isAbsolute()	Tests whether this abstract pathname is absolute. Returns true if this abstract pathname is absolute, false otherwise																
public boolean exists()	Tests whether the file or directory denoted by this abstract pathname exists. Returns true if and only if the file or directory denoted by this abstract pathname exists; false otherwise																
public boolean isDirectory()	Tests whether the file denoted by this abstract pathname is a directory. Returns true if and only if the file denoted by this abstract pathname exists and is a directory; false otherwise.																
public boolean isFile()	Tests whether the file denoted by this abstract pathname is a normal file. A file is normal if it is not a directory and, in addition, satisfies other system-dependent criteria. Any nondirectory file created by a Java application is guaranteed to be a normal file. Returns true if and only if the file denoted by this abstract pathname exists and is a normal file; false otherwise.																
	4.	Attempt any <u>THREE</u> of the following:	12 M														
	a)	Write all primitive data types available in Java with their storage Sizes in	4 M														



		bytes.																			
	Ans	<table border="1"> <thead> <tr> <th>Data Type</th> <th>Size</th> </tr> </thead> <tbody> <tr> <td>Byte</td> <td>1 Byte</td> </tr> <tr> <td>Short</td> <td>2 Byte</td> </tr> <tr> <td>Int</td> <td>4 Byte</td> </tr> <tr> <td>Long</td> <td>8 Byte</td> </tr> <tr> <td>Double</td> <td>8 Byte</td> </tr> <tr> <td>Float</td> <td>4 Byte</td> </tr> <tr> <td>Char</td> <td>2 Byte</td> </tr> <tr> <td>boolean</td> <td>1 Bit</td> </tr> </tbody> </table>	Data Type	Size	Byte	1 Byte	Short	2 Byte	Int	4 Byte	Long	8 Byte	Double	8 Byte	Float	4 Byte	Char	2 Byte	boolean	1 Bit	Data type name, size and default value and description carries 1 M
Data Type	Size																				
Byte	1 Byte																				
Short	2 Byte																				
Int	4 Byte																				
Long	8 Byte																				
Double	8 Byte																				
Float	4 Byte																				
Char	2 Byte																				
boolean	1 Bit																				
	b)	Write a program to add 2 integer, 2 string and 2 float values in a vector. Remove the element specified by the user and display the list.	4 M																		
	Ans	<pre>import java.io.*; import java.lang.*; import java.util.*; class vector2 { public static void main(String args[]) { vector v=new vector(); Integer s1=new Integer(1); Integer s2=new Integer(2); String s3=new String("fy"); String s4=new String("sy"); Float s7=new Float(1.1f); Float s8=new Float(1.2f); v.addElement(s1); v.addElement(s2); v.addElement(s3); v.addElement(s4); v.addElement(s7); v.addElement(s8); System.out.println(v); v.removeElement(s2); v.removeElementAt(4); System.out.println(v); } }</pre>	Correct program- 4 M, stepwise can give marks																		
	c)	Develop a program to create a class 'Book' having data members author, title and price. Derive a class 'BookInfo' having data member 'stock position' and	4 M																		



	method to initialize and display the information for three objects.	
Ans	<pre>class Book { String author, title, publisher; Book(String a, String t, String p) { author = a; title = t; publisher = p; } } class BookInfo extends Book { float price; int stock_position; BookInfo(String a, String t, String p, float amt, int s) { super(a, t, p); price = amt; stock_position = s; } void show() { System.out.println("Book Details:"); System.out.println("Title: " + title); System.out.println("Author: " + author); System.out.println("Publisher: " + publisher); System.out.println("Price: " + price); System.out.println("Stock Available: " + stock_position); } } class Exp6_1 { public static void main(String[] args) { BookInfo ob1 = new BookInfo("Herbert Schildt", "Complete Reference", "ABC Publication", 359.50F,10); BookInfo ob2 = new BookInfo("Ulman", "system programming", "XYZ Publication", 359.50F, 20); BookInfo ob3 = new BookInfo("Pressman", "Software Engg", "Pearson Publication", 879.50F, 15); ob1.show();</pre>	Correct program- 4 M



	<pre>ob2.show(); ob3.show(); } }</pre> <p>OUTPUT Book Details: Title: Complete Reference Author: Herbert Schildt Publisher: ABC Publication Price: 2359.5 Stock Available: 10 Book Details: Title: system programming Author: Ulman Publisher: XYZ Publication Price: 359.5 Stock Available: 20 Book Details: Title: Software Engg Author: Pressman Publisher: Pearson Publication Price: 879.5 Stock Available: 15</p>	
d)	Mention the steps to add applet to HTML file. Give sample code.	4 M
Ans	<p>Adding Applet to the HTML file: Steps to add an applet in HTML document</p> <ol style="list-style-type: none">1. Insert an <APPLET> tag at an appropriate place in the web page i.e. in the body section of HTML file.2. Specify the name of the applet's .class file.3. If the .class file is not in the current directory then use the codebase parameter to specify:-<ol style="list-style-type: none">a. the relative path if file is on the local system, orb. the uniform resource locator(URL) of the directory containing the file if it is on a remote computer.4. Specify the space required for display of the applet in terms of width and height in pixels.5. Add any user-defined parameters using <param> tags6. Add alternate HTML text to be displayed when a non-java browser is used.7. Close the applet declaration with the </APPLET> tag. <p>Open notepad and type the following source code and save it into file name</p>	<p>Steps – 2M Example – 2 M</p>



```
“Hellojava.java”
import java.awt.*;
import java.applet.*;
public class Hellojava extends Applet
{
    public void paint (Graphics g)
    {
        g.drawString("Hello Java",10,100);
    }
}
```

Use the java compiler to compile the applet “Hellojava.java” file.
C:\jdk> javac Hellojava.java

After compilation “Hellojava.class” file will be created. Executable applet is nothing but the .class file of the applet, which is obtained by compiling the source code of the applet. If any error message is received, then check the errors, correct them and compile the applet again.

We must have the following files in our current directory.

- o Hellojava.java
- o Hellojava.class
- o HelloJava.html

If we use a java enabled web browser, we will be able to see the entire web page containing the applet.

We have included a pair of <APPLET..> and </APPLET> tags in the HTML body section. The <APPLET...> tag supplies the name of the applet to be loaded and tells the browser how much space the applet requires. The <APPLET> tag given below specifies the minimum requirements to place the HelloJava applet on a web page. The display area for the applet output as 300 pixels width and 200 pixels height. CENTER tags are used to display area in the center of the screen.

```
<APPLET CODE = hellojava.class WIDTH = 400 HEIGHT = 200 > </APPLET>
```

Example: Adding applet to HTML file:
Create Hellojava.html file with following code:

```
<HTML>
<! This page includes welcome title in the title bar and displays a welcome message. Then it specifies the applet to be loaded and executed.
>
<HEAD> <TITLE> Welcome to Java Applet </TITLE> </HEAD>
<BODY> <CENTER> <H1> Welcome to the world of Applets </H1> </CENTER> <BR>
<CENTER>
```



		<APPLET CODE=HelloJava.class WIDTH = 400 HEIGHT = 200 > </APPLET> </CENTER> </BODY> </HTML>	
	e)	Write a program to copy contents of one file to another.	4 M
	Ans	<pre>import java.io.*; class copyf { public static void main(String args[]) throws IOException { BufferedReader in=null; BufferedWriter out=null; try { in=new BufferedReader(new FileReader("input.txt")); out=new BufferedWriter(new FileWriter("output.txt")); int c; while((c=in.read())!=-1) { out.write(c); } System.out.println("File copied successfully"); } finally { if(in!=null) { in.close(); } if(out!=null) { out.close(); } } } }</pre>	Correct program- 4 M
5.		Attempt any <u>TWO</u> of the following:	12 M
	a)	Compare array and vector. Explain elementAT() and addElement() methods.	6 M
	Ans		



Sr. No.	Array	Vector
1	An array is a structure that holds multiple values of the same type.	The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.
2	An array is a homogeneous data type where it can hold only objects of one data type.	Vectors are heterogeneous. You can have objects of different data types inside a Vector.
3	After creation, an array is a fixed-length structure.	The size of a Vector can grow or shrink as needed to accommodate adding and removing items after the Vector has been created.
4	Array can store primitive type data element.	Vector are store non-primitive type data element
5	Array is unsynchronized i.e. automatically increase the size when the initialized size will be exceed.	Vector is synchronized i.e. when the size will be exceeding at the time; vector size will increase double of initial size.
6	Declaration of an array : <code>int arr[] = new int [10];</code>	Declaration of Vector: <code>Vector list = new Vector(3);</code>
7	Array is the static memory allocation.	Vector is the dynamic memory allocation
8	Array allocates the memory for the fixed size ,in array there is wastage of memory.	Vector allocates the memory dynamically means according to the requirement no wastage of memory.
9	No methods are provided for adding and removing elements.	Vector provides methods for adding and removing elements.
10	In array wrapper classes are not used.	Wrapper classes are used in vector
11	Array is not a class.	Vector is a class.

4 M for any 4 correct points

1 M for elementAt()

1 M for addElement()
()

elementAT():

The **elementAt()** method of Java Vector class is used to get the element at the specified



	<p>index in the vector. Or The elementAt() method returns an element at the specified index.</p> <p>addElement():</p> <p>The addElement() method of Java Vector class is used to add the specified element to the end of this vector. Adding an element increases the vector size by one.</p>	
b)	<p>Write a program to create a class 'salary with data members empid', 'name' and 'basicsalary'. Write an interface 'Allowance' which stores rates of calculation for da as 90% of basic salary, hra as 10% of basic salary and pf as 8.33% of basic salary. Include a method to calculate net salary and display it.</p>	6 M
Ans	<pre>interface allowance { double da=0.9*basicsalary; double hra=0.1*basicsalary; double pf=0.0833*basicsalary; void netSalary(); } class Salary { int empid; String name; float basicsalary; Salary(int i, String n, float b) { empid=i; name=n; basicsalary =b; } void display() { System.out.println("Empid of Emplyee="+empid); System.out.println("Name of Employee="+name); System.out.println("Basic Salary of Employee="+ basicsalary); } } class net_salary extends salary implements allowance { float ta; net_salary(int i, String n, float b, float t) {</pre>	6 M for correct program



	<pre>super(i,n,b); ta=t; } void disp() { display(); System.out.println("da of Employee="+da); } public void netsalary() { double net_sal=basicsalary+ta+hra+da; System.out.println("netSalary of Employee="+net_sal); } } class Empdetail { public static void main(String args[]) { net_salary s=new net_salary(11, "abcd", 50000); s.disp(); s.netsalary(); } }</pre>	
c)	Define an exception called 'No Match Exception' that is thrown when the password accepted is not equal to 'MSBTE'. Write the program.	6 M
Ans	<pre>import java.io.*; class NoMatchException extends Exception { NoMatchException(String s) { super(s); } } class test1 { public static void main(String args[]) throws IOException { BufferedReader br= new BufferedReader(new InputStreamReader(System.in)); System.out.println("Enter a word:"); String str= br.readLine(); try {</pre>	6 M for correct program



	<pre>if (str.compareTo("MSBTE")!=0) // can be done with equals() throw new NoMatchException("Strings are not equal"); else System.out.println("Strings are equal"); } catch(NoMatchException e) { System.out.println(e.getMessage()); } } }</pre>	
6.	Attempt any <u>TWO</u> of the following:	12 M
a)	Write a program to check whether the string provided by the user is palindrome or not.	6 M
Ans	<pre>import java.lang.*; import java.io.*; import java.util.*; class palindrome { public static void main(String arg[]) throws IOException { BufferedReader br=new BufferedReader(new InputStreamReader(System.in)); System.out.println("Enter String:"); String word=br.readLine(); int len=word.length()-1; int l=0; int flag=1; int r=len; while(l<=r) {</pre>	6 M for correct program



	<pre>if(word.charAt(l)==word.charAt(r)) { l++; r--; } else { flag=0; break; } } if(flag==1) { System.out.println("palindrome"); } else { System.out.println("not palindrome"); } } }</pre>	
b)	Define thread priority ? Write default priority values and the methods to set and change them.	6 M
Ans	<p>Thread Priority:</p> <p>In java each thread is assigned a priority which affects the order in which it is scheduled for running. Threads of same priority are given equal treatment by the java scheduler.</p> <p>Default priority values as follows</p>	<p>2 M for define Thread priority</p> <p>2 M for</p>



The thread class defines several priority constants as: -

MIN_PRIORITY =1

NORM_PRIORITY = 5

MAX_PRIORITY = 10

Thread priorities can take value from 1-10.

getPriority(): The java.lang.Thread.getPriority() method returns the priority of the given thread.

setPriority(int newPriority): The java.lang.Thread.setPriority() method updates or assign the priority of the thread to newPriority. The method throws IllegalArgumentException if the value newPriority goes out of the range, which is 1 (minimum) to 10 (maximum).

```
import java.lang.*;
```

```
public class ThreadPriorityExample extends Thread
```

```
{
```

```
    public void run()
```

```
{
```

```
    System.out.println("Inside the run() method");
```

```
}
```

```
    public static void main(String argsv[])
```

```
{
```

```
    ThreadPriorityExample th1 = new ThreadPriorityExample();
```

```
    ThreadPriorityExample th2 = new ThreadPriorityExample();
```

```
    ThreadPriorityExample th3 = new ThreadPriorityExample();
```

```
    System.out.println("Priority of the thread th1 is : " + th1.getPriority());
```

```
    System.out.println("Priority of the thread th2 is : " + th2.getPriority());
```

```
    System.out.println("Priority of the thread th2 is : " + th2.getPriority());
```

```
    th1.setPriority(6);
```

```
    th2.setPriority(3);
```

```
    th3.setPriority(9);
```

```
    System.out.println("Priority of the thread th1 is : " + th1.getPriority());
```

```
    System.out.println("Priority of the thread th2 is : " + th2.getPriority());
```

```
    System.out.println("Priority of the thread th3 is : " + th3.getPriority());
```

```
    System.out.println("Currently Executing The Thread : " + Thread.currentThread().getName());
```

```
    System.out.println("Priority of the main thread is : " + Thread.currentThread().getPriority());
```

```
    Thread.currentThread().setPriority(10);
```

```
    System.out.println("Priority of the main thread is : " + Thread.currentThread().getPriority());
```

```
}
```

default
priority
values

2 M for
method to
set and
change



		}	
c)	Design an applet to perform all arithmetic operations and display the result by using labels, textboxes and buttons.		6 M
Ans	<pre>import java.awt.*; import java.awt.event.*; public class sample extends Frame implements ActionListener { Label l1, l2,l3; TextField tf1, tf2, tf3; Button b1, b2, b3, b4; sample() { l1=new Lable("First No."); l1.setBounds(10, 10, 50, 20); tf1 = new TextField(); tf1.setBounds(50, 50, 150, 20); l2=new Lable("Second No."); l2.setBounds(10, 60, 50, 20); tf2 = new TextField(); tf2.setBounds(50, 100, 150, 20); l3=new Lable("Result"); l3.setBounds(10, 110, 150, 20); tf3 = new TextField(); tf3.setBounds(50, 150, 150, 20); tf3.setEditable(false); b1 = new Button("+"); b1.setBounds(50, 200, 50, 50); b2 = new Button("-"); b2.setBounds(120,200,50,50); b3 = new Button("*"); b3.setBounds(220, 200, 50, 50); b4 = new Button("/"); b4.setBounds(320,200,50,50); b1.addActionListener(this); b2.addActionListener(this); b3.addActionListener(this); b4.addActionListener(this); add(tf1); add(tf2); add(tf3); add(b1); add(b2); add(b3); add(b4);</pre>	6 M for correct program	



```
setSize(400,400);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e) {
String s1 = tf1.getText();
String s2 = tf2.getText();
int a = Integer.parseInt(s1);
int b = Integer.parseInt(s2);
int c = 0;
if (e.getSource() == b1){
    c = a + b;
}
else if (e.getSource() == b2){
    c = a - b;
else if (e.getSource() == b3){
    c = a * b;
else if (e.getSource() == b4){
    c = a / b;
}
String result = String.valueOf(c);
tf3.setText(result);
}
public static void main(String[] args) {
    new sample();
}
}
```




MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for anyequivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.
- 8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No	Sub Q.N.	Answer	Marking Scheme
1.	a) Ans.	Attempt any <u>FIVE</u> of the following: Enlist the logical operators in Java. && : Logical AND : Logical OR ! : Logical NOT	10 2M <i>1M each</i> <i>Any two operators</i>
	b) Ans.	Give the syntax and example for the following functions i) min () ii) Sqrt () i) min() Syntax: (Any one of the following) static int min(int x, int y) Returns minimum of x and y static long min(long x, long y) Returns minimum of x and y static float min(float x, float y) Returns minimum of x and y static double min(double x, int y) Returns minimum of x and y	2M <i>1M for each function with example</i>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>Example: int y= Math.min(64,45);</p> <p>ii)Sqrt()</p> <p>Syntax: static double sqrt(double arg) Returns square root of arg.</p> <p>Example: double y= Math.sqrt(64);</p>	
	<p>c) Ans.</p>	<p>Define the interface in Java. Interface is similar to a class. It consist of only abstract methods and final variables. To implement an interface a class must define each of the method declared in the interface. It is used to achieve fully abstraction and multiple inheritance in Java.</p>	<p>2M</p> <p><i>1M for each point, Any two points</i></p>
	<p>d) Ans.</p>	<p>Enlist any four inbuilt packages in Java. 1.java.lang 2.java.util 3.java.io 4.java.awt 5.java.net 6.java.applet</p>	<p>2M</p> <p><i>1/2 M for each package Any four packages</i></p>
	<p>e) Ans.</p>	<p>Explain any two methods of File Class</p> <p>1. boolean createNewFile(): It creates a new, empty file named by this abstract pathname automatically, if and only if no file with the same name exists. if(file.createNewFile()) System.out.println("A new file is successfully created.");</p> <p>2. String getName(): It returns the name of the file or directory denoted by the object's abstract pathname. System.out.println("File name : " + file.getName());</p> <p>3. String getParent(): It returns the parent's pathname string of the object's abstract pathname or null if the pathname does not name a parent directory. System.out.println("Parent name : " + file.getParent());</p>	<p>2M</p> <p><i>1M for each method Any two methods</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>4. boolean <code>isFile()</code>: It returns True if the file denoted by the abstract pathname is a normal file, and False if it is not a normal file. <code>System.out.println("File size (bytes) : " + file.isFile());</code></p> <p>5. boolean <code>canRead()</code>: It returns True if the application can read the file denoted by the abstract pathname, and returns False otherwise. <code>System.out.println("Is file readable : " + file.canRead());</code></p> <p>6. boolean <code>canWrite()</code>: It returns True if the application can modify the file denoted by the abstract pathname, and returns False otherwise. <code>System.out.println("Is file writeable : " + file.canWrite());</code></p> <p>7. boolean <code>canExecute()</code>: It returns True if the application can execute the file denoted by the abstract pathname, and returns False otherwise. <code>System.out.println("Is file executable : " + file.canExecute());</code></p>	
	<p>f) Ans.</p>	<p>Write syntax of ellipse. Syntax: <code>void fillOval(int top, int left, int width, int height)</code> The filled ellipse is drawn within a bounding rectangle whose upper-left corner is specified by top and left and whose width and height are specified by width and height</p> <p>OR</p> <p>Syntax: <code>void drawOval(int top, int left, int width, int height)</code> The empty ellipse is drawn within a bounding rectangle whose upper-left corner is specified by top and left and whose width and height are specified by width and height</p>	<p>2M 2M for correct syntax</p>
	<p>g) Ans.</p>	<p>Enlist any four compile time errors.</p> <ol style="list-style-type: none">1)Missing semicolon2)Missing of brackets in classes and methods3)Misspelling of variables and keywords.4)Missing double quotes in Strings.5)Use of undeclared variable.6)Incompatible type of assignment/initialization.7)Bad reference to object.	<p>2M ½ M for each error</p> <p>Any four can be considered</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

2.	a) Ans.	<p>Attempt any <u>THREE</u> of the following: Explain any four features of Java</p> <p>1.Object Oriented: In Java, everything is an Object. Java can be easily extended since it is based on the Object model.</p> <p>2.Platform Independent: Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.</p> <p>3.Simple: Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.</p> <p>4.Secure: With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.</p> <p>5.Architecture-neutral: Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.</p> <p>6.Multithreaded: With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.</p> <p>7.Interpreted: Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.</p>	12 4M <i>1M for each feature Any four features</i>
----	------------	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	b) Ans.	Write a Java program to copy the content of one file into another. <pre>import java.io.*; class filecopy { public static void main(String args[]) throws IOException { FileReader fr= new FileReader("file1.txt"); FileWriter fo= new FileWriter("file2.txt"); int ch; try { while((ch=fr.read())!= -1) { fo.write(ch); } System.out.println("file copied successfully"); fr.close(); fo.close(); } finally { if(fr!=null) fr.close(); if(fo!=null) fo.close(); }}}</pre>	4M <i>2M for correct logic,</i> <i>2M for code</i>									
	c) Ans.	Write the difference between vectors and arrays. (any four points) <table border="1" data-bbox="378 1514 1300 1841"><thead><tr><th data-bbox="378 1514 480 1549">S.No</th><th data-bbox="480 1514 813 1549">Array</th><th data-bbox="813 1514 1300 1549">Vector</th></tr></thead><tbody><tr><td data-bbox="378 1549 480 1698">1</td><td data-bbox="480 1549 813 1698">An array is a structure that holds multiple values of the same type.</td><td data-bbox="813 1549 1300 1698">The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.</td></tr><tr><td data-bbox="378 1698 480 1841">2</td><td data-bbox="480 1698 813 1841">An array is a homogeneous data type where it can hold only objects of one data type</td><td data-bbox="813 1698 1300 1841">Vectors are heterogeneous. You can have objects of different data types inside a Vector.</td></tr></tbody></table>	S.No	Array	Vector	1	An array is a structure that holds multiple values of the same type.	The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.	2	An array is a homogeneous data type where it can hold only objects of one data type	Vectors are heterogeneous. You can have objects of different data types inside a Vector.	4M <i>1M for each point</i> <i>Any four points</i>
S.No	Array	Vector										
1	An array is a structure that holds multiple values of the same type.	The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.										
2	An array is a homogeneous data type where it can hold only objects of one data type	Vectors are heterogeneous. You can have objects of different data types inside a Vector.										



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		3	After creation, an array is a fixed-length structure	The size of a Vector can grow or shrink as needed to accommodate adding and removing items after the Vector has been created	
		4	Array can store primitive type data element.	Vector are store non primitive type data element.	
		5	Declaration of an array int arr[] = new int [10];	Declaration of Vector: Vector list = new Vector(3)	
		6	Array is the static memory allocation.	Vector is the dynamic memory allocation	
	d)	Explain exception handling mechanism w.r.t. try, catch, throw and finally.			4M
	Ans.	try: Program statements that you want to monitor for exceptions are contained within a try block. If an exception occurs within the try block, it is thrown. Syntax: try { // block of code to monitor for errors } catch: Your code can catch this exception (using catch) and handle it in some rational manner. System-generated exceptions are automatically thrown by the Java runtime system. A catch block immediately follows the try block. The catch block can have one or more statements that are necessary to process the exception. Syntax: catch (ExceptionType1 exOb) { // exception handler for ExceptionType1 }			1M for each



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>throw: It is mainly used to throw an instance of user defined exception. Example: throw new myException("Invalid number"); assuming myException as a user defined exception</p> <p>finally: finally block is a block that is used to execute important code such as closing connection, stream etc. Java finally block is always executed whether exception is handled or not. Java finally block follows try or catch block.</p> <p>Syntax: finally { // block of code to be executed before try block ends }</p>	
3.	a) Ans.	<p>Attempt any <u>THREE</u> of the following: Write a Java Program to find out the even numbers from 1 to 100 using for loop.</p> <pre>class test { public static void main(String args[]) { System.out.println("Even numbers from 1 to 100 :"); for(int i=1;i<=100; i++) { if(i%2==0) System.out.print(i+" "); } }</pre>	12 4M <i>2M for Program logic</i> <i>2M for program syntax</i>
	b) Ans.	<p>Explain any four visibility controls in Java.</p> <p>Four visibility control specifiers in Java are public, default, private and protected. The visibility control in java can be seen when concept of package is used with the java application.</p> <ol style="list-style-type: none">1) private :The access level of a private specifier is only within the class. It cannot be accessed from outside the class.2) default :When no specifier is used in the declaration, it is called as default specification. Default scope for anything declared in java	4M <i>3M for Explanatio n</i>



SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>is implicit public. With this it can be accessed anywhere within the same package.</p> <p>3) protected :The access level of a protected specifier is within the package and outside the package through derived class.</p> <p>4) public :The access level of a public specifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.</p> <p>5) private protected access: The visibility level is between protected access and private access. The fields are visible in all subclasses regardless of what package they are in.</p> <p>These five access specifiers can be mapped with four categories in which packages in java can be managed with access specification matrix as:</p> <table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="border: none;">Access Modifier</th> <th>Public</th> <th>Protected</th> <th>Friendly (default)</th> <th>Private protected</th> <th>private</th> </tr> <tr> <th style="border: none;">Access Location</th> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </thead> <tbody> <tr> <td style="border: none;">Same Class</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> </tr> <tr> <td style="border: none;">Sub class in same package</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> <td>No</td> </tr> <tr> <td style="border: none;">Other classes in same package</td> <td>Yes</td> <td>Yes</td> <td>Yes</td> <td>No</td> <td>No</td> </tr> <tr> <td style="border: none;">Sub class in other packages</td> <td>Yes</td> <td>Yes</td> <td>No</td> <td>Yes</td> <td>No</td> </tr> <tr> <td style="border: none;">Non sub classes in other packages</td> <td>Yes</td> <td>No</td> <td>No</td> <td>No</td> <td>No</td> </tr> </tbody> </table>	Access Modifier	Public	Protected	Friendly (default)	Private protected	private	Access Location						Same Class	Yes	Yes	Yes	Yes	Yes	Sub class in same package	Yes	Yes	Yes	Yes	No	Other classes in same package	Yes	Yes	Yes	No	No	Sub class in other packages	Yes	Yes	No	Yes	No	Non sub classes in other packages	Yes	No	No	No	No	<p><i>1M for access specification table</i></p>
Access Modifier	Public	Protected	Friendly (default)	Private protected	private																																								
Access Location																																													
Same Class	Yes	Yes	Yes	Yes	Yes																																								
Sub class in same package	Yes	Yes	Yes	Yes	No																																								
Other classes in same package	Yes	Yes	Yes	No	No																																								
Sub class in other packages	Yes	Yes	No	Yes	No																																								
Non sub classes in other packages	Yes	No	No	No	No																																								
	<p>c) Ans.</p>	<p>Explain single and multilevel inheritance with proper example.</p> <p>Single level inheritance: In single inheritance, a single subclass extends from a single superclass.</p> <div style="text-align: center; margin: 10px 0;"> <pre> classDiagram class A class B B -- > A : extends </pre> </div> <p>Example : class A {</p>	<p style="text-align: center;">4M</p> <p style="text-align: center;"><i>1M for each explanation</i></p> <p style="text-align: center;"><i>1M for each example</i></p>																																										



SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

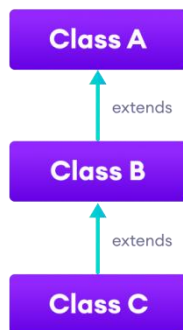
Subject Code: 22412

```
void display()
{
System.out.println("In Parent class A");
}
}
class B extends A //derived class B from A
{
void show()
{
System.out.println("In child class B");
}
public static void main(String args[])
{
B b= new B();
b.display(); //super class method call
b.show(); // sub class method call
}
}
```

Note : any other relevant example can be considered.

Multilevel inheritance:

In multilevel inheritance, a subclass extends from a superclass and then the same subclass acts as a superclass for another class. Basically it appears as derived from a derived class.



Example:

```
class A
{
void display()
```

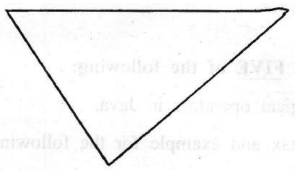


MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre>{ System.out.println("In Parent class A"); } } class B extends A //derived class B from A { void show() { System.out.println("In child class B"); } } class C extends B //derived class C from B { public void print() { System.out.println("In derived from derived class C"); } public static void main(String args[]) { C c= new C(); c.display(); //super class method call c.show(); // sub class method call c.print(); //sub-sub class method call } } } </pre> <p><i>Note : any other relevant example can be considered.</i></p>	
d)	<p>Write a java applet to display the following output in Red color. Refer Fig. No. 1.</p>  <p>Fig No. 1.</p>	4M
Ans.	<pre>import java.awt.*; import java.applet.*; public class myapplet extends Applet</pre>	2M for correct logic



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>{ public void paint(Graphics g) { int x[]={ 10,200,70}; int y[]={ 10,10,100}; g.setColor(Color.red); g.drawPolygon(x,y,3); } } /*<applet code=myapplet height=400 width=400> </applet>*/</pre>	<p><i>2M for correct syntax</i></p>
4.	a) Ans.	<p>Attempt any <u>THREE</u> of the following: Explain switch case and conditional operator in java with suitable example. switch...case statement: The switch...case statement allows us to execute a block of code among many alternatives. Syntax : switch (expression) { case value1: // code break; case value2: // code break; default: // default statements }</p> <p>The expression is evaluated once and compared with the values of each case. If expression matches with value1, the code of case value1 are executed. Similarly, the code of case value2 is executed if expression matches with value2.</p>	<p>12 4M</p> <p><i>1M for explanation n switch case statement</i></p> <p><i>1M for example</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>break is a required statement, which is used to take break from switch block, if any case is true. Otherwise even after executing a case, if break is not given, it will go for the next case. If there is no match, the code of the default case is executed.</p> <p>Example : // Java Program to print day of week // using the switch...case statement class test1 { public static void main(String[] args) { int number = 1; String day; switch (number) { case 1: day = "Monday"; break; case 2: day= "Tuesday"; break; case 3: day = "Wednesday"; break; case 4: day= "Thursday"; break; case 5: day = "Friday"; break; case 6: day= "Saturday"; break; case 7: day = "Sunday"; break; default: day= "Invalid day"; } }</p>	
--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

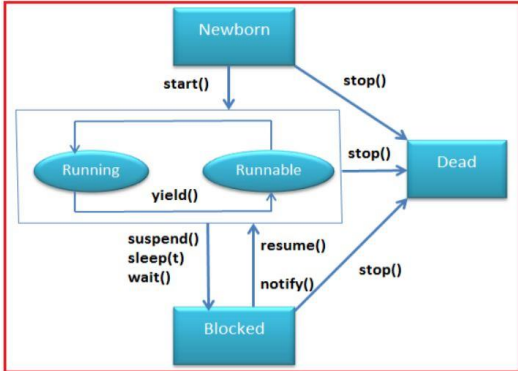
	<pre>System.out.println(day); } }</pre> <p><i>Note : any other relevant example can be considered.</i></p> <p>Conditional Operator: The Conditional Operator is used to select one of two expressions for evaluation, which is based on the value of the first operands. It is used to handling simple situations in a line. Syntax: expression1 ? expression2:expression3; The above syntax means that if the value given in Expression1 is true, then Expression2 will be evaluated; otherwise, expression3 will be evaluated.</p> <p>Example class test { public static void main(String[] args) { String result; int a = 6, b = 12; result = (a==b ? "equal":"Not equal"); System.out.println("Both are "+result); } }<p><i>Note : any other relevant example can be considered.</i></p></p>	<p><i>1M for explanation</i> <i>n</i> <i>Conditiona</i> <i>l operator</i></p> <p><i>1M for example</i></p>
<p>b) Ans.</p>	<p>Draw and explain life cycle of thread. Life cycle of thread includes following states :</p> <ol style="list-style-type: none">1.Newborn2. Runnable3. Running4. Blocked5. Dead	<p>4M</p>



SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

			<p><i>2M for diagram</i></p>
		<p>New – A new thread begins its life cycle in the new state. It is also referred to as a born thread. This is the state where a thread has been created, but it has not yet been started. A thread is started by calling its start() method.</p> <p>Runnable – The thread is in the runnable state after the invocation of the start() method, but the scheduler has not selected it to be the running thread. It is in the Ready-to-run state by calling the start method and waiting for its turn.</p> <p>Running – When the thread starts executing, then the state is changed to a “running” state. The method invoked is run ().</p> <p>Blocked–This is the state when the thread is still alive but is currently not eligible to run. This state can be implemented by methods such as suspend()-resume(), wait()-notify() and sleep(time in ms).</p> <p>Dead – This is the state when the thread is terminated. The thread is in a running state and as soon as it is completed processing it is in a “dead state”. Once a thread is in this state, the thread cannot even run again.</p>	<p><i>2M for explanation</i></p>
	<p>c) Ans.</p>	<p>Write a java program to sort an 1-d array in ascending order using bubble-sort.</p> <pre>public class BubbleSort { public static void main(String[] args)</pre>	<p>4M</p> <p><i>2M for correct logic</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre>{ int arr[]={3,60,35,2,45,320,5}; System.out.println("Array Before Bubble Sort"); for(int i=0; i<arr.length; i++) { System.out.print(arr[i] + " "); } System.out.println(); int n = arr.length; int temp = 0; for(int i=0; i< n; i++) { for(int j=1; j < (n-i); j++) { if(arr[j-1] >arr[j]) { //swap elements temp = arr[j-1]; arr[j-1] = arr[j]; arr[j] = temp; } } } System.out.println("Array After Bubble Sort"); for(int i=0; i<arr.length; i++) { System.out.print(arr[i] + " "); } }</pre>	<p><i>2M for correct syntax</i></p>
<p>d) Ans.</p>	<p>Explain how to create a package and how to import it To create package following steps can be taken: 1) Start the code by keyword 'package' followed by package name. Example : package mypackage; 2) Complete the code with all required classes inside the package with appropriate access modifiers. 3) Compile the code with 'javac' to get .class file.</p>	<p>4M <i>3M for steps to create</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>Example: javac myclass.java to get myclass.class</p> <p>4) Create a folder which is same as package name and make sure that class file of package is present inside it. If not, copy it inside this folder.</p> <p>To import the package inside any other program : Make use of import statement to include package in your program. It can be used with '*' to gain full access to all classes within package or just by giving class name if just one class access is required. Example : import mypackage.myclass; or importmypackage.*;</p>	<p><i>1M to import</i></p>
	<p>e)</p> <p>Explain</p> <p>i) drawLine ii) drawOval iii) drawRect iv) drawArc</p> <p>Ans.</p> <p>i) drawLine(): It is a method from Graphics class and is used to draw line between the points(x1, y1) and (x2, y2). Syntax : drawLine(int x1, int y1, int x2, int y2)</p> <p>ii) drawOval():Its is a method from Graphics class and is used to draw oval or ellipse and circle. Syntax : drawOval(int x, ,int y, int width, int height) It is used to draw oval with the specifiedwidth and height. If width and height are given equal, then it draws circle otherwise oval/ellipse.</p> <p>iii) drawRect():It is a method from Graphics class and it draws a rectangle with the specified widthand height. Syntax : drawRect(int x, int y, int width, int height)</p> <p>iv) drawArc():It is a method from Graphics class and is used to draw a circular or elliptical arc. Syntax : drawArc(int x, int y, int width, int height, intstartAngle, intswEEPAngle)</p>	<p>4M</p> <p><i>1M for each</i></p>	



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		where first four are x, y, width and height as in case of oval or rect. The next two are start angle and sweep angle. When sweep angle is positive, it moves in anticlockwise direction. It is given as negative, It moves in clockwise direction.	
5.	a)	Attempt any <u>TWO</u> of the following: How to create user defined package in Java. Explain with an suitable example.	12 6M
	Ans.	<p>A java package is a group of similar types of classes, interfaces and sub-packages It also provides access protection and removes name collisions.</p> <p>Creation of user defined package: To create a package a physical folder by the name should be created in the computer. Example: we have to create a package myPack, so we create a folder d:\myPack The java program is to be written and saved in the folder myPack. To add a program to the package, the first line in the java program should be package <name>; followed by imports and the program logic.</p> <pre>package myPack; import java.util; public class Myclass { //code }</pre> <p>Access user defined package: To access a user defined package, we need to import the package in our program. Once we have done the import we can create the object of the class from the package and thus through the object we can access the instance methods.</p> <pre>import mypack.*; public class MyClassExample{ public static void main(String a[]) { Myclass c= new Myclass();</pre>	<p>3M Package creation</p> <p><i>(Note: Code snippet can be used for describing)</i></p> <p>3M for Example</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>} }</pre> <p>Example:</p> <pre>package package1; public class Box { int l= 5; int b = 7; int h = 8; public void display() { System.out.println("Volume is:"+(l*b*h)); } }</pre> <p>Source file:</p> <pre>import package1.Box; class volume { public static void main(String args[]) { Box b=new Box(); b.display(); } }</pre>	<p>(Note Any other similar example can be considered)</p>
	<p>b)</p> <p>Ans.</p>	<p>Write a Java program in which thread A will display the even numbers between 1 to 50 and thread B will display the odd numbers between 1 to 50. After 3 iterations thread A should go to sleep for 500ms.</p> <pre>Import java.lang.*; class A extends Thread { public void run() { try { for(int i=2;i<=50;i=i+2) {</pre>	<p>6M</p> <p>3M Correct program with syntax</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre> System.out.println("\t A thread :"+i); if(i == 6) // for 3rd iteration sleep(500); } } catch(Exception e) { System.out.println("A thread interrupted"); } } } class B extends Thread { public void run() { try { for(int i=1;i<50;i=i+2) { System.out.println("\t B thread :"+i); } } catch(Exception e) { System.out.println("B thread interrupted"); } } } class OddEven { public static void main(String args[]) { new A().start(); new B().start(); } }</pre>	<p><i>3M Correct logic</i></p>
--	--	--



SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>c) Ans.</p>	<p>What is constructor? List types of constructor. Explain parameterized constructor with suitable example.</p> <p>Constructor: A constructor is a special member which initializes an object immediately upon creation.</p> <ul style="list-style-type: none">• It has the same name as class name in which it resides and it is syntactically similar to any method.• When a constructor is not defined, java executes a default constructor which initializes all numeric members to zero and other types to null or spaces.• Once defined, constructor is automatically called immediately after the object is created before new operator completes. <p>Types of constructors:</p> <ol style="list-style-type: none">1. Default constructor2. Parameterized constructor3. Copy constructor4. Constructor with no arguments or No-Arg Constructor or Non-Parameterized constructor. <p>Parameterized constructor: When constructor method is defined with parameters inside it, different value sets can be provided to different constructor with the same name.</p> <p>Example</p> <pre>class Student { int roll_no; String name; Student(int r, String n) // parameterized constructor { roll_no = r; name=n; } void display() { System.out.println("Roll no is: "+roll_no); System.out.println("Name is : "+name); } }</pre>	<p>6M</p> <p><i>2M for Definition</i></p> <p><i>1M List types</i></p> <p><i>(Any 3)</i></p> <p><i>1M parameterized constructor</i></p> <p><i>2M Example</i></p> <p><i>(Any Other Example Can be</i></p>
--	--------------------	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>public static void main(String a[]) { Student s = new Student(20,"ABC"); // constructor with parameters s.display(); }</pre>	<i>considered</i>)
6.	a)	Attempt any <u>TWO</u> of the following: Write a Java Program to count the number of words from a text file using stream classes.	12 6M (Note : Any other relevant logic shall be considered)
	Ans.	<pre>import java.io.*; public class FileWordCount { public static void main(String are[]) throws IOException { File f1 = new File("input.txt"); int wc=0; FileReader fr = new FileReader (f1); int c=0; try { while(c!=-1) { c=fr.read(); if(c==(char) ' ') wc++; } System.out.println("Number of words :"+(wc+1)); } finally { if(fr!=null) fr.close(); } }</pre>	3M Correct program with syntax 3M Correct logic
	b)	Explain the difference between string class and string buffer class. Explain any four methods of string class	6M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

Ans.	Sr. No.	String	StringBuffer	
	1	String is a major class	StringBuffer is a peer class of String	<i>1M each Any 2 points</i>
	2	Length is fixed	Length is flexible	
	3	Contents of object cannot be modified	Contents of object can be modified	
	4	Object can be created by assigning String constants enclosed in double quotes.	Objects can be created by calling constructor of StringBuffer class using new operator.	
	5	String s="MSBTE"	StringBuffer s=new StringBuffer ("MSBTE")	
		Methods of string class		<i>1M each Any 4 Methods correct explanation</i>
		1)toLowerCase (): Converts all of the characters in this String to lower case. Syntax: s1.toLowerCase() Example: String s="Sachin"; System.out.println(s.toLowerCase()); Output: sachin		
		2) toUpperCase(): Converts all of the characters in this String to upper case Syntax: s1.toUpperCase() Example: String s="Sachin"; System.out.println(s.toUpperCase()); Output: SACHIN		
		3)trim (): Returns a copy of the string, with leading and trailing whitespace omitted. Syntax: s1.trim() Example: String s=" Sachin "; System.out.println(s.trim());		



SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>Output:Sachin</p> <p>4)replace ():Returns a new string resulting from replacing all occurrences of old Char in this string with new Char.</p> <p>Syntax: s1.replace('x','y')</p> <p>Example: String s1="Java is a programming language. Java is a platform.";</p> <pre>String s2=s1.replace("Java","Kava"); //replaces all occurrences of "Java" to "Kava" System.out.println(s2);</pre> <p>Output: Kava is a programming language. Kava is a platform</p> <p>5. length():</p> <p>Syntax: int length()</p> <p>It is used to return length of given string in integer.</p> <p>Eg. String str="INDIA"</p> <pre>System.out.println(str.length()); // Returns 5</pre> <p>6. charAt():</p> <p>Syntax: char charAt(int position)</p> <p>The charAt() will obtain a character from specified position .</p> <p>Eg. String s="INDIA"</p> <pre>System.out.println(s.charAt(2)); // returns D</pre> <p>7. substring():</p> <p>Syntax:</p> <p>String substring (int startindex)</p> <p>startindex specifies the index at which the substring will begin.It will returns a copy of the substring that begins at startindex and runs to the end of the invoking string</p> <p>Example:</p> <pre>System.out.println(("Welcome".substring(3)); //come</pre> <p>(OR)</p> <p>String substring(int startindex,int endindex)</p> <p>Here startindex specifies the beginning index, and endindex specifies the stopping point. The string returned all the characters from the</p>	
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>beginning index, upto, but not including, the ending index. <i>Example :</i> System.out.println(("Welcome".substring(3,5));//co</p> <p>8. compareTo(): Syntax: int compareTo(Object o) or int compareTo(String anotherString) There are two variants of this method. First method compares this String to another Object and second method compares two strings lexicographically. Example. String str1 = "Strings are immutable"; String str2 = "Strings are immutable"; String str3 = "Integers are not immutable"; int result = str1.compareTo(str2); System.out.println(result); result = str2.compareTo(str3); System.out.println(result);</p>											
<p>c)</p>	<p>Write a Java applet to draw a bar chart for the following values.</p> <table border="1"><thead><tr><th>Year</th><th>2011</th><th>2012</th><th>2013</th><th>2014</th></tr></thead><tbody><tr><td>Turnover (Rs. crores)</td><td>110</td><td>120</td><td>170</td><td>160</td></tr></tbody></table> <p>Ans. import java.awt.*; import java.applet.*;</p> <p>/* <applet code=BarChart width=400 height=400> <param name=c1 value=110> <param name=c2 value=120> <param name=c3 value=170> <param name=c4 value=160> <param name=label1 value=2011> <param name=label2 value=2012> <param name=label3 value=2013></p>	Year	2011	2012	2013	2014	Turnover (Rs. crores)	110	120	170	160	<p>6M</p> <p>2M for Applet tag</p> <p>2M for</p>
Year	2011	2012	2013	2014								
Turnover (Rs. crores)	110	120	170	160								



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre><param name=label4 value=2014> <param name=Columns value=4> </applet> */ public class BarChart extends Applet { int n=0; String label[]; int value[]; public void init() { setBackground(Color.yellow); try { int n = Integer.parseInt(getParameter("Columns")); label = new String[n]; value = new int[n]; label[0] = getParameter("label1"); label[1] = getParameter("label2"); label[2] = getParameter("label3"); label[3] = getParameter("label4"); value[0] = Integer.parseInt(getParameter("c1")); value[1] = Integer.parseInt(getParameter("c2")); value[2] = Integer.parseInt(getParameter("c3")); value[3] = Integer.parseInt(getParameter("c4")); } catch(NumberFormatException e){} } public void paint(Graphics g) { for(int i=0;i<4;i++) { g.setColor(Color.black); g.drawString(label[i],20,i*50+30); g.setColor(Color.red); g.fillRect(50,i*50+10,value[i],40); } }</pre>	<p><i>Syntax</i></p> <p>2M <i>Correct</i> <i>Logic</i></p>
--	---	---



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code:

22412

		}}	
--	--	----	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

Winter – 19 EXAMINATION

Subject Name: Java Programming

Model Answer

Subject Code: 22412

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1.		Attempt any Five of the following:	10M
	a	Define Constructor. List its types.	2M
	Ans	Constructor: A constructor is a special member which initializes an object immediately upon creation. It has the same name as class name in which it resides and it is syntactically similar to any method. When a constructor is not defined, java executes a default constructor which initializes all numeric members to zero and other types to null or spaces. Once defined, constructor is automatically called immediately after the object is created before new operator completes. Types of constructors: 1. Default constructor 2. Parameterized constructor 3. Copy constructor	Definition: 1Mark Types: 1 Mark
	b	Define Class and Object.	2M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	Ans	Class: A class is a user defined data type which groups data members and its associated functions together. Object: It is a basic unit of Object Oriented Programming and represents the real life entities. A typical Java program creates many objects, which as you know, interact by invoking methods.	Definition 1 Mark each
	c	List the methods of File Input Stream Class.	2M
	Ans	<ul style="list-style-type: none">• void close()• int read()• int read(byte[] b)• read(byte[] b, int off, int len)• int available()	Any Two Each for 1 Mark
	d	Define error. List types of error.	2M
	Ans	<ul style="list-style-type: none">• Errors are mistakes that can make a program go wrong. Errors may be logical or may be typing mistakes. An error may produce an incorrect output or may terminate the execution of the program abruptly or even may cause the system to crash. <p>Errors are broadly classified into two categories:</p> <ol style="list-style-type: none">1. Compile time errors2. Runtime errors	Definition: 1m List: 1m
	e	List any four Java API packages.	2M
	Ans	<ol style="list-style-type: none">1.java.lang2.java.util3.java.io4.java.awt5.java.net6.java.applet	1/2 Marks for one Package
	f	Define array. List its types.	2M
	Ans	An array is a homogeneous data type where it can hold only objects of one data type. Types of Array:	Definition 1 Mark, List 1 Mark



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		1)One-Dimensional 2)Two-Dimensional													
	g	List access specifiers in Java.	2M												
	Ans	1)public 2)private 3)friendly 4)protected 5)Private Protected	Any 2, 1M for each												
2.		Attempt any Three of the following:	12M												
	a	Differentiate between String and String Buffer.	4M												
	Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">String</th> <th style="text-align: center;">String Buffer c</th> </tr> </thead> <tbody> <tr> <td>String is a major class</td> <td>String Buffer is a peer class of String</td> </tr> <tr> <td>Length is fixed (immutable)</td> <td>Length is flexible (mutable)</td> </tr> <tr> <td>Contents of object cannot be modified</td> <td>Contents of object can be modified</td> </tr> <tr> <td>Object can be created by assigning String constants enclosed in double quotes.</td> <td>Objects can be created by calling constructor of String Buffer class using "new"</td> </tr> <tr> <td>Ex:- String s="abc";</td> <td>Ex:- StringBuffer s=new StringBuffer ("abc");</td> </tr> </tbody> </table>	String	String Buffer c	String is a major class	String Buffer is a peer class of String	Length is fixed (immutable)	Length is flexible (mutable)	Contents of object cannot be modified	Contents of object can be modified	Object can be created by assigning String constants enclosed in double quotes.	Objects can be created by calling constructor of String Buffer class using "new"	Ex:- String s="abc";	Ex:- StringBuffer s=new StringBuffer ("abc");	Any 4 Points 4 Marks
String	String Buffer c														
String is a major class	String Buffer is a peer class of String														
Length is fixed (immutable)	Length is flexible (mutable)														
Contents of object cannot be modified	Contents of object can be modified														
Object can be created by assigning String constants enclosed in double quotes.	Objects can be created by calling constructor of String Buffer class using "new"														
Ex:- String s="abc";	Ex:- StringBuffer s=new StringBuffer ("abc");														
	b	Define a class circle having data members pi and radius. Initialize and display values of data members also calculate area of circle and display it.													
	Ans	class abc {	correct Program with correct logic 4 Mark												



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<pre>float pi,radius; abc(float p, float r) { pi=p; radius=r; } void area() { float ar=pi*radius*radius; System.out.println("Area="+ar); } void display() { System.out.println("Pi="+pi); System.out.println("Radius="+radius); } } class area { public static void main(String args[]) { abc a=new abc(3.14f,5.0f); a.display();</pre>	
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		<pre>a.area(); } }</pre>	
	c	Define exception. State built-in exceptions.	4M
	Ans	<p>An exception is a problem that arises during the execution of a program.</p> <p>Java exception handling is used to handle error conditions in a program systematically by taking the necessary action</p> <p>Built-in exceptions:</p> <ul style="list-style-type: none">• Arithmetic exception: Arithmetic error such as division by zero.• ArrayIndexOutOfBoundsException: Array index is out of bound• ClassNotFoundException• FileNotFoundException: Caused by an attempt to access a nonexistent file.• IO Exception: Caused by general I/O failures, such as inability to read from a file.• NullPointerException: Caused by referencing a null object.• NumberFormatException: Caused when a conversion between strings and number fails.• StringIndexOutOfBoundsException: Caused when a program attempts to access a nonexistent character position in a string.• OutOfMemoryException: Caused when there's not enough memory to allocate a new object.• SecurityException: Caused when an applet tries to perform an action not allowed by the browser's security setting.• StackOverflowException: Caused when the system runs out of stack space.	Definition 2 Marks, List: 2 Marks
	d	Write syntax and example of :	4M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		1) drawRect() 2)drawOval()	
	Ans	1)drawRect() : drawRect () method display an outlined rectangle. Syntax: void drawRect(int top,int left, int width,int height) The upper-left corner of the Rectangle is at top and left. The dimension of the Rectangle is specified by width and height. Example: g.drawRect(10,10,60,50); 2) drawOval(): Drawing Ellipses and circles: To draw an Ellipses or circles used drawOval () method can be used. Syntax: void drawOval(int top, int left, int width, int height) The ellipse is drawn within a bounding rectangle whose upper-left corner is specified by top and left and whose width and height are specified by width and height to draw a circle or filled circle, specify the same width and height the following program draws several ellipses and circle. Example: g.drawOval(10,10,50,50);	drawRect: 2Marks, drawOval: 2 Marks
3.		Attempt any Three of the following:	
	a	Explain the following classes. i)Byte stream class ii)Character Stream Class	4M
	Ans	i)Byte stream class: 1) InputStream and OutputStream are designed for byte streams 2) Use the byte stream classes when working with bytes or other binary objects. 3) Input Stream is an abstract class that defines Java's model of streaming byte input	2M for any two points

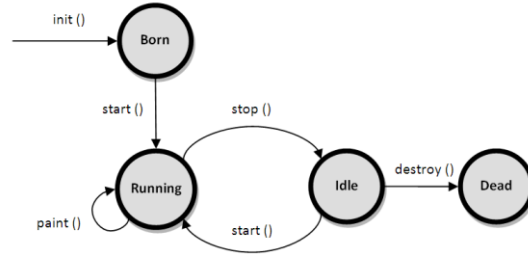


MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<p>4)The Input stream class defines methods for performing input function such as reading bytes, closing streams, Marking position in stream.</p> <p>5) Output Stream is an abstract class that defines streaming byte output.</p> <p>6) The output stream class defines methods for performing output function such as writing bytes, closing streams</p> <p>ii)Character Stream Class:</p> <ol style="list-style-type: none">1. Reader and Writer are designed for character streams.2. Use character stream classes when working with characters or strings.3. Writer stream classes are designed to write characters.4. Reader stream classes are designed to read characters. <p>5)The two subclasses used for handling characters in file are FileReader (for reading characters) and FileWriter (for writing characters).</p>	
b	Explain life cycle of Applet.	4M
Ans	<p>When an applet begins, the AWT calls the following methods, in this sequence:</p> <ol style="list-style-type: none">1. init()2. start()3. paint() <p>When an applet is terminated, the following sequence of method calls takes place:</p> <ol style="list-style-type: none">4. stop()5. destroy()	1M for diagram ,3M for explanation



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
 (Autonomous)
 (ISO/IEC - 27001 - 2013 Certified)



init ():The **init()** method is the first method to be called. This is where you should initialize Variables. This method is called only once during the run time of your applet.

start():The **start()** method is called after **init()**.It is also called to restart an applet after it has Been stopped. Whereas **init()** is called once—the first time an applet is loaded—**start()** is called each time an applet’s HTML document is displayed onscreen.

Paint (): The **paint ()** method is called each time your applet’s output must be redrawn. Paint () is also called when the applet begins execution. Whatever the cause, whenever the applet must redraw its output, paint() is called. The paint () method has one parameter of type Graphics.

Stop (): When stop () is called, the applet is probably running. You should use stop () to suspend threads that don’t need to run when the applet is not visible.

destroy(): The destroy () method is called when the environment determines that your applet needs to be removed completely from memory.

c	Differentiate between class and interfaces.	4M								
Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Class</th> <th style="width: 50%; text-align: center;">Interface</th> </tr> </thead> <tbody> <tr> <td>1)doesn’t Supports multiple inheritance</td> <td>1) Supports multiple inheritance</td> </tr> <tr> <td>2)”extend ” keyword is used to inherit</td> <td>2)”implements ” keyword is used to inherit</td> </tr> <tr> <td>3) class contain method body</td> <td>3) interface contains abstract method(method without body)</td> </tr> </tbody> </table>	Class	Interface	1)doesn’t Supports multiple inheritance	1) Supports multiple inheritance	2)”extend ” keyword is used to inherit	2)”implements ” keyword is used to inherit	3) class contain method body	3) interface contains abstract method(method without body)	1M for each point
Class	Interface									
1)doesn’t Supports multiple inheritance	1) Supports multiple inheritance									
2)”extend ” keyword is used to inherit	2)”implements ” keyword is used to inherit									
3) class contain method body	3) interface contains abstract method(method without body)									



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	4)contains any type of variable	4)contains only final variable	
	5)can have constructor	5)cannot have constructor	
	6)can have main() method	6)cannot have main() method	
	7)syntax Class classname { Variable declaration, Method declaration }	7)syntax Inteface Innterfacename { Final Variable declaration, abstract Method declaration }	
d	Define type casting. Explain its types with syntax and example.		4M
Ans	<p>1. The process of converting one data type to another is called casting or type casting.</p> <p>2. If the two types are compatible, then java will perform the conversion automatically.</p> <p>3. It is possible to assign an int value to long variable.</p> <p>4. However, if the two types of variables are not compatible, the type conversions are not implicitly allowed, hence the need for type casting.</p> <p>There are two types of conversion:</p> <p>1.Implicit type-casting:</p> <p>2.Explicit type-casting:</p> <p>1. Implicit type-casting:</p> <p>Implicit type-casting performed by the <i>compiler automatically</i>; if there will be no loss of precision.</p> <p>Example:</p> <pre>int i = 3; double f; f = i;</pre>		1M for definition,3M for types explanation

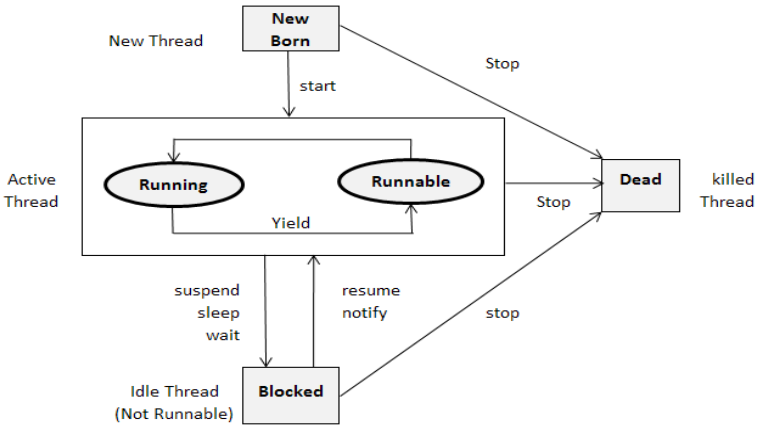


MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		<p>output: f = 3.0</p> <p>Widening Conversion:</p> <p>The rule is to promote the smaller type to bigger type to prevent loss of precision, known as Widening Conversion.</p> <p>2. Explicit type-casting:</p> <ul style="list-style-type: none">• Explicit type-casting performed via a type-casting operator in the prefix form of (<i>new-type</i>) operand.• Type-casting forces an explicit conversion of type of a value. Type casting is an operation which takes one operand, operates on it and returns an equivalent value in the specified type. <p>Syntax:</p> <p>newValue = (typecast)value;</p> <p>Example:</p> <p>double f = 3.5;</p> <p>int i; i = (int)f; // it cast double value 3.5 to int 3.</p> <p>Narrowing Casting: Explicit type cast is requires to Narrowing conversion to inform the compiler that you are aware of the possible loss of precision.</p>	
4.		Attempt any Three of the following:	
	a	Explain life cycle of thread.	4M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

<p>Ans</p>	 <p>The diagram illustrates the Thread Life Cycle with five states: New Born, Running, Runnable, Blocked, and Dead. Transitions are labeled with methods: start (New Born to Running), Stop (New Born to Dead), suspend/sleep/wait (Running to Blocked), resume/notify (Blocked to Runnable), Yield (Running to Runnable), and Stop (Runnable to Dead). A box labeled 'Active Thread' encloses the Running and Runnable states, while 'Idle Thread (Not Runnable)' encloses the Blocked state. 'killed Thread' is associated with the Dead state.</p> <p>Thread Life Cycle Thread has five different states throughout its life.</p> <ol style="list-style-type: none">1. Newborn State2. Runnable State3. Running State4. Blocked State5. Dead State <p>Thread should be in any one state of above and it can be move from one state to another by different methods and ways.</p> <p>Newborn state: When a thread object is created it is said to be in a new born state. When the thread is in a new born state it is not scheduled running from this state it can be scheduled for running by start() or killed by stop(). If put in a queue it moves to runnable state.</p> <p>Runnable State: It means that thread is ready for execution and is waiting for the availability of the processor i.e. the thread has joined the queue and is waiting for execution. If all threads have equal priority, then they are given time slots for execution in round robin fashion. The thread that relinquishes control joins the queue at the end and again waits for its turn. A thread can relinquish the control to another before its turn comes by yield().</p>	<p>2M for diagram, 2M for explanation</p>
-------------------	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<p>Running State: It means that the processor has given its time to the thread for execution. The thread runs until it relinquishes control on its own or it is pre-empted by a higher priority thread.</p> <p>Blocked state: A thread can be temporarily suspended or blocked from entering into the runnable and running state by using either of the following thread method.</p> <ol style="list-style-type: none">1) suspend() : Thread can be suspended by this method. It can be rescheduled by resume().2) wait(): If a thread requires to wait until some event occurs, it can be done using wait method and can be scheduled to run again by notify().3) sleep(): We can put a thread to sleep for a specified time period using sleep(time) where time is in ms. It re-enters the runnable state as soon as period has elapsed /over <p>Dead State: Whenever we want to stop a thread form running further we can call its stop().The statement causes the thread to move to a dead state. A thread will also move to dead state automatically when it reaches to end of the method. The stop method may be used when the premature death is required.</p>	
b	Describe final variable and final method.	4M
Ans	<p>Final method: making a method final ensures that the functionality defined in this method will never be altered in any way, ie a final method cannot be overridden.</p> <p>Syntax:</p> <pre>final void findAverage() { //implementation }</pre> <p>Example of declaring a final method:</p> <pre>class A {</pre>	2M for definition,2M for example



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<pre>final void show() { System.out.println("in show of A"); } } class B extends A { void show() // can not override because it is declared with final { System.out.println("in show of B"); }} Final variable: the value of a final variable cannot be changed. Final variable behaves like class variables and they do not take any space on individual objects of the class. Example of declaring final variable: final int size = 100;</pre>									
c	Explain any two logical operator in java with example.	4M								
Ans	<p>Logical Operators: Logical operators are used when we want to form compound conditions by combining two or more relations. Java has three logical operators as shown in table:</p> <table border="1"><thead><tr><th>Operator</th><th>Meaning</th></tr></thead><tbody><tr><td>&&</td><td>Logical AND</td></tr><tr><td> </td><td>Logical OR</td></tr><tr><td>!</td><td>Logical NOT</td></tr></tbody></table> <p>Program demonstrating logical Operators</p> <pre>public class Test</pre>	Operator	Meaning	&&	Logical AND		Logical OR	!	Logical NOT	2M for each operator with eg.
Operator	Meaning									
&&	Logical AND									
	Logical OR									
!	Logical NOT									



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
 (Autonomous)
 (ISO/IEC - 27001 - 2013 Certified)

		<pre> { public static void main(String args[]) { boolean a = true; boolean b = false; System.out.println("a && b = " + (a&&b)); System.out.println("a b = " + (a b)); System.out.println(!(a && b) = " + !(a && b)); } } Output: a && b = false a b = true !(a && b) = true </pre>					
	d	Differentiate between array and vector.	4M				
	Ans	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Array</th> <th style="width: 50%; text-align: center;">Vector</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">1) An array is a structure that holds multiple values of the same type.</td> <td style="padding: 5px;">1)The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.</td> </tr> </tbody> </table>	Array	Vector	1) An array is a structure that holds multiple values of the same type.	1)The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.	any four points 1m for each point
Array	Vector						
1) An array is a structure that holds multiple values of the same type.	1)The Vector is similar to array holds multiple objects and like an array; it contains components that can be accessed using an integer index.						



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<p>2) An array is a homogeneous data type where it can hold only objects of one data type.</p> <p>3) After creation, an array is a fixed-length structure.</p> <p>4) Array can store primitive type data element.</p> <p>5) Declaration of an array : int arr[] = new int [10];</p> <p>6) Array is the static memory allocation.</p>	<p>2) Vectors are heterogeneous. You can have objects of different data types inside a Vector.</p> <p>3) The size of a Vector can grow or shrink as needed to accommodate adding and removing items after the Vector has been created.</p> <p>4) Vector are store non-primitive type data element.</p> <p>5) Declaration of Vector: Vector list = new Vector(3);</p> <p>6) Vector is the dynamic memory allocation.</p>	
e	List any four methods of string class and state the use of each.		4M
Ans	<p>The java.lang.String class provides a lot of methods to work on string. By the help of these methods,</p> <p>We can perform operations on string such as trimming, concatenating, converting, comparing, replacing strings etc.</p> <p>1) to Lowercase (): Converts all of the characters in this String to lower case.</p> <p>Syntax: s1.toLowerCase() Example: String s="Sachin"; System.out.println(s.toLowerCase());</p> <p>Output: sachin</p> <p>2) to Uppercase(): Converts all of the characters in this String to upper case</p>		any four methods of string class can be considered



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		<p>Syntax: <code>s1.toUpperCase()</code></p> <p>Example:</p> <pre>String s="Sachin"; System.out.println(s.toUpperCase());</pre> <p>Output: SACHIN</p> <p>3) trim (): Returns a copy of the string, with leading and trailing whitespace omitted.</p> <p>Syntax: <code>s1.trim()</code></p> <p>Example:</p> <pre>String s=" Sachin "; System.out.println(s.trim());</pre> <p>Output:Sachin</p> <p>4) replace ():Returns a new string resulting from replacing all occurrences of old Char in this string with new Char.</p> <p>Syntax: <code>s1.replace('x','y')</code></p> <p>Example:</p> <pre>String s1="Java is a programming language. Java is a platform." String s2=s1.replace("Java","Kava");//replaces all occurrences of "Java" to "Kava" System.out.println(s2);</pre> <p>Output: Kava is a programming language. Kava is a platform.</p>	
5.		Attempt any Three of the following:	12-Total Marks
	a	Write a program to create a vector with five elements as (5, 15, 25, 35, 45). Insert new element at 2nd position. Remove 1st and 4th element from vector.	6M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

Ans	<pre>import java.util.*; class VectorDemo { public static void main(String[] args) { Vector v = new Vector(); v.addElement(new Integer(5)); v.addElement(new Integer(15)); v.addElement(new Integer(25)); v.addElement(new Integer(35)); v.addElement(new Integer(45)); System.out.println("Original array elements are "); for(int i=0;i<v.size();i++) { System.out.println(v.elementAt(i)); } v.insertElementAt(new Integer(20),1); // insert new element at 2nd position v.removeElementAt(0); //remove first element v.removeElementAt(3); //remove fourth element System.out.println("Array elements after insert and remove operation "); for(int i=0;i<v.size();i++) { System.out.println(v.elementAt(i)); } } }</pre>	<p><i>(Vector creation with elements – 2 M,</i></p> <p><i>Insert new element – 2M,</i></p> <p><i>Remove elements 2 M,</i></p> <p>(Any other logic can be considered)</p>
b	Define package. How to create user defined package? Explain with example.	6M
Ans	Java provides a mechanism for partitioning the class namespace into more manageable parts. This mechanism is the package. The package is both naming and visibility controlled mechanism. Package can be created by including package as the first statement in java source code. Any classes declared within that file will belong to the specified package. Package defines a namespace in	(Definition of package - 1M,



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<p>which classes are stored.</p> <p>The syntax for defining a package is: package <i>pkg</i>; Here, <i>pkg</i> is the name of the package eg : package mypack;</p> <p>Packages are mirrored by directories. Java uses file system directories to store packages. The class files of any classes which are declared in a package must be stored in a directory which has same name as package name. The directory must match with the package name exactly. A hierarchy can be created by separating package name and sub package name by a period(.) as <i>pkg1.pkg2.pkg3</i>; which requires a directory structure as <i>pkg1\pkg2\pkg3</i>.</p> <p>Syntax: To access package In a Java source file, import statements occur immediately following the package statement (if it exists) and before any class definitions.</p> <p>Syntax: import <i>pkg1</i>[.<i>pkg2</i>].(<i>classname</i> *);</p> <p>Example: package package1; public class Box { int l= 5; int b = 7; int h = 8; public void display() { System.out.println("Volume is:"+(l*b*h)); } }</p> <p>Source file: import package1.Box; class volume {</p>	<p>Package creation - 2M</p> <p>Example - 3M</p> <p>(Note Any other example can be considered)</p>
--	---	---



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<pre>public static void main(String args[]) { Box b=new Box(); b.display(); } }</pre>	
c	Write a program to create two threads one thread will print even no. between 1 to 50 and other will print odd number between 1 to 50.	6M
Ans	<pre>import java.lang.*; class Even extends Thread { public void run() { try { for(int i=2;i<=50;i=i+2) { System.out.println("\t Even thread :"+i); sleep(500); } } catch(InterruptedException e) {System.out.println("even thread interrupted"); } } } class Odd extends Thread { public void run() { try { for(int i=1;i<50;i=i+2) { System.out.println("\t Odd thread :"+i); sleep(500); } } } }</pre>	Creation of two threads 4M Creating main to create and start objects of 2 threads: 2M (Any other logic can be considered)



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		<pre> } } catch(InterruptedException e) {System.out.println("odd thread interrupted"); } } } class EvenOdd { public static void main(String args[]) { new Even().start(); new Odd().start(); } } </pre>	
6.		Attempt any Three of the following:	12 M
	a	Explain how to pass parameter to an applet ? Write an applet to accept username in the form of parameter and print “Hello <username>”.	6M
	Ans	<p>Passing Parameters to Applet</p> <ul style="list-style-type: none"> • User defined parameters can be supplied to an applet using <PARAM.....> tags. • PARAM tag names a parameter the Java applet needs to run, and provides a value for that parameter. • PARAM tag can be used to allow the page designer to specify different colors, fonts, URLs or other data to be used by the applet. <p>To set up and handle parameters, two things must be done.</p> <p>1. Include appropriate <PARAM..>tags in the HTML document. The Applet tag in HTML document allows passing the arguments using param tag. The syntax of <PARAM...> tag</p> <pre> <Applet code="AppletDemo" height=300 width=300> <PARAM NAME = name1 VALUE = value1> </Applet> NAME:attribute name VALUE: value of attribute named by corresponding PARAM NAME. </pre>	<p>(Explanation for parameter passing - 3M,</p> <p>Correct Program – 3M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<p>2. Provide code in the applet to parse these parameters. The Applet access their attributes using the getParameter method.</p> <p>The syntax is : String getParameter(String name);</p> <p>Program</p> <pre>import java.awt.*; import java.applet.*; public class hellouser extends Applet { String str; public void init() { str = getParameter("username"); str = "Hello "+ str; } public void paint(Graphics g) { g.drawString(str,10,100); } } <HTML> <Applet code = hellouser.class width = 400 height = 400> <PARAM NAME = "username" VALUE = abc> </Applet> </HTML></pre> <p>(OR)</p> <pre>import java.awt.*; import java.applet.*; /*<Applet code = hellouser.class width = 400 height = 400> <PARAM NAME = "username" VALUE = abc> </Applet>*/ public class hellouser extends Applet { String str; public void init() { str = getParameter("username"); str = "Hello "+ str; } }</pre>	
--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<pre>public void paint(Graphics g) { g.drawString(str,10,100); } }</pre>	
b	Write a program to perform following task (i) Create a text file and store data in it. (ii) Count number of lines and words in that file.	6M
Ans	<pre>import java.util.*; import java.io.*; class Model6B { public static void main(String[] args) throws Exception { int lineCount=0, wordCount=0; String line = ""; BufferedReader br1 = new BufferedReader(new InputStreamReader(System.in)); FileWriter fw = new FileWriter("Sample.txt"); //create text file for writing System.out.println("Enter data to be inserted in file: "); String fileData = br1.readLine(); fw.write(fileData); fw.close(); BufferedReader br = new BufferedReader(new FileReader("Sample.txt")); while ((line = br.readLine()) != null) { lineCount++; // no of lines count String[] words = line.split(" "); wordCount = wordCount + words.length; // no of words count } System.out.println("Number of lines is : " + lineCount); System.out.println("Number of words is : " + wordCount); } }</pre>	Create file and store data : 3M, Get lines and word count : 3M) (Any other logic can be considered)



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

		}	
	c	Implement the following inheritance <pre>classDiagram class Salary { Basic Salary Basic_Sal() } class Employee { Name, age Display() } class Gross_Salary { TA, DA, HRA Total_Sal() } Employee -- > Gross_Salary Employee .. > Salary Gross_Salary .. > Salary</pre>	6M
	Ans	<pre>interface Salary { double Basic Salary=10000.0; void Basic Sal(); } class Employee { String Name; int age; Employee(String n, int b) { Name=n; age=b; } void Display() { System.out.println("Name of Employee :"+Name); System.out.println("Age of Employee :"+age); } } class Gross_Salary extends Employee implements Salary { double HRA,TA,DA; Gross_Salary(String n, int b, double h,double t,double d) { super(n,b); HRA=h;</pre>	(Interface: 1M, Employee class: 2M, Gross_Salary class: 3M)



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

	<pre>TA=t; DA=d; } public void Basic_Sal() { System.out.println("Basic Salary :"+Basic_Salary); } void Total_Sal() { Display(); Basic_Sal(); double Total_Sal=Basic_Salary + TA + DA + HRA; System.out.println("Total Salary :"+Total_Sal); } } class EmpDetails { public static void main(String args[]) { Gross_Salary s=new Gross_Salary("Sachin",20,1000,2000,7000); s.Total_Sal(); } }</pre>	<p>(Any other logic considered)</p>
--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No	Sub Q.N.	Answer	Marking Scheme
1.	a) Ans.	Attempt any <u>FIVE</u> of the following: List any eight features of Java. Features of Java: 1. Data Abstraction and Encapsulation 2. Inheritance 3. Polymorphism 4. Platform independence 5. Portability 6. Robust 7. Supports multithreading 8. Supports distributed applications 9. Secure 10. Architectural neutral 11. Dynamic	10 2M <i>Any eight features 2M</i>
	b) Ans.	State use of finalize() method with its syntax. Use of finalize(): Sometimes an object will need to perform some action when it is	2M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>destroyed. Eg. If an object holding some non java resources such as file handle or window character font, then before the object is garbage collected these resources should be freed. To handle such situations java provide a mechanism called finalization. In finalization, specific actions that are to be done when an object is garbage collected can be defined. To add finalizer to a class define the finalize() method. The java run-time calls this method whenever it is about to recycle an object.</p> <p>Syntax: protected void finalize() { } }</p>	<p><i>Use 1M</i></p> <p><i>Syntax 1M</i></p>
	<p>c) Ans.</p>	<p>Name the wrapper class methods for the following: (i) To convert string objects to primitive int. (ii) To convert primitive int to string objects. (i) To convert string objects to primitive int: String str="5"; int value = Integer.parseInt(str); (ii) To convert primitive int to string objects: int value=5; String str=Integer.toString(value);</p>	<p>2M</p> <p><i>1M for each method</i></p>
	<p>d) Ans.</p>	<p>List the types of inheritances in Java. <i>(Note: Any four types shall be considered)</i> Types of inheritances in Java: i. Single level inheritance ii. Multilevel inheritance iii. Hierarchical inheritance iv. Multiple inheritance v. Hybrid inheritance</p>	<p>2M</p> <p><i>Any four types ½M each</i></p>
	<p>e) Ans.</p>	<p>Write the syntax of try-catch-finally blocks. try{ //Statements to be monitored for any exception } catch(ThrowableInstance1 obj) { //Statements to execute if this type of exception occurs } catch(ThrowableInstance2 obj2) { //Statements }finally{</p>	<p>2M</p> <p><i>Correct syntax 2M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		//Statements which should be executed even if any exception happens }	
f) Ans.	Give the syntax of < param > tag to pass parameters to an applet. Syntax: <param name="name" value="value"> Example: <param name="color" value="red">	2M <i>Correct syntax 2M</i>	
g) Ans.	Define stream class. List its types. Definition of stream class: An I/O Stream represents an input source or an output destination. A stream can represent many different kinds of sources and destinations, including disk files, devices, other programs, and memory arrays. Streams support many different kinds of data, including simple bytes, primitive data types, localized characters, and objects. Java's stream based I/O is built upon four abstract classes: InputStream, OutputStream, Reader, Writer. Types of stream classes: i. Byte stream classes ii. Character stream classes.	2M <i>Definitio n 1M</i> <i>Types 1M</i>	
2. a) Ans.	Attempt any <u>THREE</u> of the following: Explain the concept of platform independence and portability with respect to Java language. <i>(Note: Any other relevant diagram shall be considered).</i> Java is a platform independent language. This is possible because when a java program is compiled, an intermediate code called the byte code is obtained rather than the machine code. Byte code is a highly optimized set of instructions designed to be executed by the JVM which is the interpreter for the byte code. Byte code is not a machine specific code. Byte code is a universal code and can be moved anywhere to any platform. Therefore java is portable, as it can be carried to any platform. JVM is a virtual machine which exists inside the computer memory and is a simulated computer within a computer which does all the functions of a computer. Only the JVM needs to be implemented for each platform. Although the details of the JVM will defer from platform to platform, all interpret the same	12 4M <i>Explana tion 3M</i>	



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>byte code.</p> <pre> graph TD SC[Source Code] --> JC[Java Compiler] JC --> BC[Byte code] BC --> JVM1[Java Virtual Machine JVM] BC --> JVM2[Java Virtual Machine JVM] JVM1 --> WOS[Window Operating System] JVM2 --> LOS[Linux Operating System] </pre>	<p><i>Diagram</i> 1M</p>
<p>b)</p> <p>Ans.</p>	<p>Explain the types of constructors in Java with suitable example. <i>(Note: Any two types shall be considered).</i></p> <p>Constructors are used to initialize an object as soon as it is created. Every time an object is created using the ‘new’ keyword, a constructor is invoked. If no constructor is defined in a class, java compiler creates a default constructor. Constructors are similar to methods but with to differences, constructor has the same name as that of the class and it does not return any value.</p> <p>The types of constructors are:</p> <ol style="list-style-type: none"> 1. Default constructor 2. Constructor with no arguments 3. Parameterized constructor 4. Copy constructor <p>1. Default constructor: Java automatically creates default constructor if there is no default or parameterized constructor written by user. Default constructor in Java initializes member data variable to default values (numeric values are initialized as 0, Boolean is initialized as false and references are initialized as null).</p> <pre> class test1 { int i; boolean b; byte bt; float ft; String s; </pre>	<p style="text-align: center;">4M</p> <p style="text-align: center;"><i>Explana tion of the two types of construc tors</i> 2M</p> <p style="text-align: center;"><i>Example</i> 2M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre>public static void main(String args[]) { test1 t = new test1(); // default constructor is called. System.out.println(t.i); System.out.println(t.s); System.out.println(t.b); System.out.println(t.bt); System.out.println(t.ft); } }</pre> <p>2. Constructor with no arguments: Such constructors does not have any parameters. All the objects created using this type of constructors has the same values for its datamembers.</p> <p>Eg:</p> <pre>class Student { int roll_no; String name; Student() { roll_no = 50; name="ABC"; } void display() { System.out.println("Roll no is: "+roll_no); System.out.println("Name is : "+name); } public static void main(String a[]) { Student s = new Student(); s.display(); } }</pre> <p>3. Parametrized constructor: Such constructor consists of parameters. Such constructors can be used to create different objects with datamembers having different values.</p> <pre>class Student { int roll_no; String name; Student(int r, String n) { roll_no = r;</pre>	
--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre>name=n; } void display() { System.out.println("Roll no is: "+roll_no); System.out.println("Name is : "+name); } public static void main(String a[]) { Student s = new Student(20,"ABC"); s.display(); } }</pre> <p>4. Copy Constructor : A copy constructor is a constructor that creates a new object using an existing object of the same class and initializes each instance variable of newly created object with corresponding instance variables of the existing object passed as argument. This constructor takes a single argument whose type is that of the class containing the constructor.</p> <pre>class Rectangle { int length; int breadth; Rectangle(int l, int b) { length = l; breadth= b; } //copy constructor Rectangle(Rectangle obj) { length = obj.length; breadth= obj.breadth; } public static void main(String[] args) { Rectangle r1= new Rectangle(5,6); Rectangle r2= new Rectangle(r1); System.out.println("Area of First Rectangle : "+ (r1.length*r1.breadth));</pre>	
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>System.out.println("Area of First Second Rectangle : "+ (r1.length*r1.breadth)); } }</pre>	
	<p>c) Ans.</p>	<p>Explain the two ways of creating threads in Java. Thread is a independent path of execution within a program. There are two ways to create a thread: 1. By extending the Thread class. Thread class provide constructors and methods to create and perform operations on a thread. This class implements the Runnable interface. When we extend the class Thread, we need to implement the method run(). Once we create an object, we can call the start() of the thread class for executing the method run(). Eg: class MyThread extends Thread { public void run() { for(int i = 1;i<=20;i++) { System.out.println(i); } } public static void main(String a[]) { MyThread t = new MyThread(); t.start(); } } a. By implementing the runnable interface. Runnable interface has only on one method- run(). Eg: class MyThread implements Runnable { public void run() { for(int i = 1;i<=20;i++) { System.out.println(i); } } } public static void main(String a[]) { MyThread m = new MyThread(); Thread t = new Thread(m); t.start(); } }</p>	<p>4M</p> <p><i>2M each for explaining of two types with example</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		}																						
d) Ans.	<p>Distinguish between Input stream class and output stream class. Java I/O (Input and Output) is used to process the input and produce the output. Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations. A stream is a sequence of data. In Java, a stream is composed of bytes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Sr. No.</th> <th style="width: 45%;">Input stream class</th> <th style="width: 45%;">Output stream class</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>Java application uses an input stream to read data from a source;</td> <td>Java application uses an output stream to write data to a destination;</td> </tr> <tr> <td style="text-align: center;">2</td> <td>It may read from a file, an array, peripheral device or socket</td> <td>It may be a write to file, an array, peripheral device or socket</td> </tr> <tr> <td style="text-align: center;">3</td> <td>Input stream classes reads data as bytes</td> <td>Output stream classes writes data as bytes</td> </tr> <tr> <td style="text-align: center;">4</td> <td>Super class is the abstract inputStream class</td> <td>Super class is the abstract OutputStream class</td> </tr> <tr> <td style="text-align: center;">5</td> <td>Methods: public int read() throws IOException public int available() throws IOException public void close() throws IOException</td> <td>Methods: public void write(int b) throws IOException public void write(byte[] b) throws IOException public void flush() throws IOException public void close() throws IOException</td> </tr> <tr> <td style="text-align: center;">6</td> <td>The different subclasses of Input Stream are: File Input stream, Byte Array Input Stream, Filter Input Stream, Piped Input Stream, Object Input Stream, DataInputStream.</td> <td>The different sub classes of Output Stream class are: File Output Stream, Byte Array Output Stream , Filter output Stream, Piped Output Stream, Object Output Stream, DataOutputStream</td> </tr> </tbody> </table>		Sr. No.	Input stream class	Output stream class	1	Java application uses an input stream to read data from a source;	Java application uses an output stream to write data to a destination;	2	It may read from a file, an array, peripheral device or socket	It may be a write to file, an array, peripheral device or socket	3	Input stream classes reads data as bytes	Output stream classes writes data as bytes	4	Super class is the abstract inputStream class	Super class is the abstract OutputStream class	5	Methods: public int read() throws IOException public int available() throws IOException public void close() throws IOException	Methods: public void write(int b) throws IOException public void write(byte[] b) throws IOException public void flush() throws IOException public void close() throws IOException	6	The different subclasses of Input Stream are: File Input stream, Byte Array Input Stream, Filter Input Stream, Piped Input Stream, Object Input Stream, DataInputStream.	The different sub classes of Output Stream class are: File Output Stream, Byte Array Output Stream , Filter output Stream, Piped Output Stream, Object Output Stream, DataOutputStream	<p>4M</p> <p><i>Any four points for input stream class and output stream class 1M each</i></p>
Sr. No.	Input stream class	Output stream class																						
1	Java application uses an input stream to read data from a source;	Java application uses an output stream to write data to a destination;																						
2	It may read from a file, an array, peripheral device or socket	It may be a write to file, an array, peripheral device or socket																						
3	Input stream classes reads data as bytes	Output stream classes writes data as bytes																						
4	Super class is the abstract inputStream class	Super class is the abstract OutputStream class																						
5	Methods: public int read() throws IOException public int available() throws IOException public void close() throws IOException	Methods: public void write(int b) throws IOException public void write(byte[] b) throws IOException public void flush() throws IOException public void close() throws IOException																						
6	The different subclasses of Input Stream are: File Input stream, Byte Array Input Stream, Filter Input Stream, Piped Input Stream, Object Input Stream, DataInputStream.	The different sub classes of Output Stream class are: File Output Stream, Byte Array Output Stream , Filter output Stream, Piped Output Stream, Object Output Stream, DataOutputStream																						



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

3.	a) Ans.	<p>Attempt any <u>THREE</u> of the following: Define a class student with int id and string name as data members and a method void SetData (). Accept and display the data for five students.</p> <pre>import java.io.*; class student { int id; String name; BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); void SetData() { try { System.out.println("enter id and name for student"); id=Integer.parseInt(br.readLine()); name=br.readLine(); } catch(Exception ex) {} } void display() { System.out.println("The id is " + id + " and the name is "+ name); } public static void main(String are[]) { student[] arr; arr = new student[5]; int i; for(i=0;i<5;i++) { arr[i] = new student(); } for(i=0;i<5;i++) { arr[i].SetData(); } }</pre>	<p>12 4M</p> <p><i>Correct logic 4M</i></p>
----	----------------	---	---



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre> for(i=0;i<5;i++) { arr[i].display(); } } }</pre>	
b) Ans.	Explain dynamic method dispatch in Java with suitable example. Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time. <ul style="list-style-type: none">• When an overridden method is called through a superclass reference, Java determines which version (superclass/subclasses) of that method is to be executed based upon the type of the object being referred to at the time the call occurs. Thus, this determination is made at run time.• At run-time, it depends on the type of the object being referred to (not the type of the reference variable) that determines which version of an overridden method will be executed• A superclass reference variable can refer to a subclass object. This is also known as upcasting. Java uses this fact to resolve calls to overridden methods at run time. Therefore, if a superclass contains a method that is overridden by a subclass, then when different types of objects are referred to through a superclass reference variable, different versions of the method are executed. Here is an example that illustrates dynamic method dispatch: // A Java program to illustrate Dynamic Method // Dispatch using hierarchical inheritance class A { void m1() { System.out.println("Inside A's m1 method"); } } class B extends A { // overriding m1() void m1()	4M <i>Explanation 2M</i> <i>Example 2M</i>	



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre>{ System.out.println("Inside B's m1 method"); } class C extends A { // overriding m1() void m1() { System.out.println("Inside C's m1 method"); } } // Driver class class Dispatch { public static void main(String args[]) { // object of type A A a = new A(); // object of type B B b = new B(); // object of type C C c = new C(); // obtain a reference of type A A ref; // ref refers to an A object ref = a; // calling A's version of m1() ref.m1(); // now ref refers to a B object ref = b;</pre>	
--	---	--



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>// calling B's version of m1() ref.m1(); // now ref refers to a C object ref = c; // calling C's version of m1() ref.m1(); } }</pre>	
	<p>c)</p> <p>Ans.</p>	<p>Describe the use of following methods:</p> <p>(i) Drawoval () (ii) getFont () (iii) drawRect () (iv) getFamily ()</p> <p>(i) Drawoval (): Drawing Ellipses and circles: To draw an Ellipses or circles used drawOval() method can be used. Syntax: void drawOval(int top, int left, int width, int height) The ellipse is drawn within a bounding rectangle whose upper-left corner is specified by top and left and whose width and height are specified by width and height. To draw a circle or filled circle, specify the same width and height.</p> <p><i>Example:</i> g.drawOval(10,10,50,50);</p> <p>(ii) getFont (): It is a method of Graphics class used to get the font property Font f = g.getFont(); String fontName = f.getName(); Where g is a Graphics class object and fontName is string containing name of the current font.</p> <p>(iii) drawRect (): The drawRect() method display an outlined rectangle. Syntax: void drawRect(int top,int left,int width,int height) The upper-left corner of the Rectangle is at top and left. The dimension of the Rectangle is specified by width and height. <i>Example:</i> g.drawRect(10,10,60,50);</p>	<p>4M</p> <p><i>Each method 1M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		(iv) getFamily () : The getfamily() method Returns the family of the font. String family = f.getFamily(); Where f is an object of Font class	
	d) Ans.	Write a program to count number of words from a text file using stream classes. <i>(Note : Any other relevant logic shall be considered)</i> import java.io.*; public class FileWordCount { public static void main(String are[]) throws IOException { File f1 = new File("input.txt"); int wc=0; FileReader fr = new FileReader (f1); int c=0; try { while(c!=-1) { c=fr.read(); if(c==(char)' ') wc++; } System.out.println("Number of words :"+(wc+1)); } finally { if(fr!=null) fr.close(); } } }	4M <i>Correct program 4M</i>
4.	a) Ans.	Attempt any <u>THREE</u> of the following: Describe instance Of and dot (.) operators in Java with suitable example. Instance of operator: The java instance of operator is used to test whether the object is an instance of the specified type (class or subclass or interface).	12 4M



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>The instance of in java is also known as type comparison operator because it compares the instance with type. It returns either true or false. If we apply the instance of operator with any variable that has null value, it returns false.</p> <p><i>Example</i></p> <pre>class Simple1{ public static void main(String args[]){ Simple1 s=new Simple1(); System.out.println(s instanceof Simple1);//true } }</pre> <p>dot (.) operator:</p> <p>The dot operator, also known as separator or period used to separate a variable or method from a reference variable. Only static variables or methods can be accessed using class name. Code that is outside the object's class must use an object reference or expression, followed by the dot (.) operator, followed by a simple field name.</p> <p><i>Example</i></p> <p>this.name="john"; where name is a instance variable referenced by 'this' keyword</p> <p>c.getdata(); where getdata() is a method invoked on object 'c'.</p>	<p><i>Description and example of each operator</i></p> <p>2M</p>
<p>b) Ans.</p>	<p>Explain the four access specifiers in Java.</p> <p>There are 4 types of java access modifiers:</p> <p>1. private 2. default 3. Protected 4. public</p> <p>1) private access modifier: The private access modifier is accessible only within class.</p> <p>2) default access specifier: If you don't specify any access control specifier, it is default, i.e. it becomes implicit public and it is accessible within the program.</p> <p>3) protected access specifier: The protected access specifier is accessible within package and outside the package but through inheritance only.</p> <p>4) public access specifier: The public access specifier is accessible everywhere. It has the widest scope among all other modifiers.</p>	<p>4M</p> <p><i>Each access specifiers</i></p> <p>1M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

c)	Differentiate between method overloading and method overriding.	4M	Ans.	Sr. No.	Method overloading	Method overriding
				1	Overloading occurs when two or more methods in one class have the same method name but different parameters.	Overriding means having two methods with the same method name and parameters (i.e., method signature)
				2	In contrast, reference type determines which overloaded method will be used at compile time.	The real object type in the run-time, not the reference variable's type, determines which overridden method is used at runtime
				3	Polymorphism not applies to overloading	Polymorphism applies to overriding
				4	overloading is a compile-time concept.	Overriding is a run-time concept
d)	Differentiate between Java Applet and Java Application (any four points)	4M	Ans.	Sr. No.	Java Applet	Java Application
				1	Applets run in web pages	Applications run on stand-alone systems.
				2	Applets are not full featured application programs.	Applications are full featured programs.
				3	Applets are the small programs.	Applications are larger programs.
				4	Applet starts execution with its init().	Application starts execution with its main ().
				5	Parameters to the applet are given in the HTML file.	Parameters to the application are given at the command prompt
				6	Applet cannot access the local file system and resources	Application can access the local file system and resources.
				7	Applets are event driven	Applications are control driven.

*Any four points
1M each*

*Any four points
1M each*



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>e) Ans.</p>	<p>Write a program to copy content of one file to another file.</p> <pre>class fileCopy { public static void main(String args[]) throws IOException { FileInputStream in= new FileInputStream("input.txt"); FileOutputStream out= new FileOutputStream("output.txt"); int c=0; try { while(c!=-1) { c=in.read(); out.write(c); } System.out.println("File copied to output.txt...."); } finally { if(in!=null) in.close(); if(out!=null) out.close(); } } }</pre>	<p>4M</p> <p><i>Correct logic 2M</i></p> <p><i>Correct Syntax 2M</i></p>
5.	<p>a) Ans.</p>	<p>Attempt any <u>TWO</u> of the following:</p> <p>Describe the use of any methods of vector class with their syntax. <i>(Note: Any method other than this but in vector class shall be considered for answer).</i></p> <ul style="list-style-type: none">• boolean add(Object obj)-Appends the specified element to the end of this Vector.• Boolean add(int index,Object obj)-Inserts the specified element at the specified position in this Vector.• void addElement(Object obj)-Adds the specified component to the end of this vector, increasing its size by one.• int capacity()-Returns the current capacity of this vector.• void clear()-Removes all of the elements from this vector.• Object clone()-Returns a clone of this vector.	<p>12 6M</p> <p><i>Any 6 methods with their use 1M each</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<ul style="list-style-type: none">• boolean contains(Object elem)-Tests if the specified object is a component in this vector.• void copyInto(Object[] anArray)-Copies the components of this vector into the specified array.• Object firstElement()-Returns the first component (the item at index 0) of this vector.• Object elementAt(int index)-Returns the component at the specified index.• int indexOf(Object elem)-Searches for the first occurrence of the given argument, testing for equality using the equals method.• Object lastElement()-Returns the last component of the vector.• Object insertElementAt(Object obj,int index)-Inserts the specified object as a component in this vector at the specified index.• Object remove(int index)-Removes the element at the specified position in this vector.• void removeAllElements()-Removes all components from this vector and sets its size to zero.	
<p>b)</p> <p>Ans.</p>	<p>Explain the concept of Dynamic method dispatch with suitable example.</p> <p>Method overriding is one of the ways in which Java supports Runtime Polymorphism. Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time.</p> <p>When an overridden method is called through a superclass reference, Java determines which version (superclass/subclasses) of that method is to be executed based upon the type of the object being referred to at the time the call occurs. Thus, this determination is made at run time. At run-time, it depends on the type of the object being referred to (not the type of the reference variable) that determines which version of an overridden method will be executed</p> <p>A superclass reference variable can refer to a subclass object. This is also known as upcasting. Java uses this fact to resolve calls to overridden methods at run time.</p> <p>If a superclass contains a method that is overridden by a subclass, then when different types of objects are referred to through a superclass reference variable, different versions of the method are executed. Here is an example that illustrates dynamic method dispatch:</p>	<p>6M</p> <p><i>Explanation 3M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre data-bbox="386 445 1023 1866">/ A Java program to illustrate Dynamic Method // Dispatch using hierarchical inheritance class A { void m1() { System.out.println("Inside A's m1 method"); } } class B extends A { // overriding m1() void m1() { System.out.println("Inside B's m1 method"); } } class C extends A { // overriding m1() void m1() { System.out.println("Inside C's m1 method"); } } // Driver class class Dispatch { public static void main(String args[]) { // object of type A A a = new A(); // object of type B B b = new B(); // object of type C C c = new C(); } }</pre>	<p data-bbox="1312 850 1429 913"><i>Example 3M</i></p>
--	---	--



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<pre>// obtain a reference of type A A ref; // ref refers to an A object ref = a; // calling A's version of m1() ref.m1(); // now ref refers to a B object ref = b; // calling B's version of m1() ref.m1(); // now ref refers to a C object ref = c; // calling C's version of m1() ref.m1(); } }</pre> <p>Output: Inside A's m1 method Inside B's m1 method Inside C's m1 method</p> <p>Explanation: The above program creates one superclass called A and it's two subclasses B and C. These subclasses overrides m1() method.</p> <ol style="list-style-type: none">1. Inside the main() method in Dispatch class, initially objects of type A, B, and C are declared.2. A a = new A(); // object of type A3. B b = new B(); // object of type B C c = new C(); // object of type C	
--	--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>c)</p> <p>Ans.</p> <pre>class Ascending extends Thread { public void run() { for(int i=1; i<=15;i++) { System.out.println("Ascending Thread : " + i); } } } class Descending extends Thread { public void run() { for(int i=15; i>0;i--) { System.out.println("Descending Thread : " + i); } } } public class AscendingDescending Thread { public static void main(String[] args) { Ascending a=new Ascending(); a.start(); Descending d=new Descending(); d.start(); } }</pre>	<p>6M</p> <p><i>Creation of two threads 4M</i></p> <p><i>Creating main to create and start objects of 2 threads: 2M</i></p>
6.	<p>a)</p> <p>Ans.</p> <p>Java Command Line Argument: The java command-line argument is an argument i.e. passed at the time of running the java program.</p>	<p>12</p> <p>6M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<p>The arguments passed from the console can be received in the java program and it can be used as an input. So, it provides a convenient way to check the behaviour of the program for the different values. You can pass N (1,2,3 and so on) numbers of arguments from the command prompt.</p> <p>Command Line Arguments can be used to specify configuration information while launching your application. There is no restriction on the number of java command line arguments. You can specify any number of arguments Information is passed as Strings. They are captured into the String args of your main method</p> <p>Simple example of command-line argument in java</p> <p>In this example, we are receiving only one argument and printing it. To run this java program, you must pass at least one argument from the command prompt.</p> <pre>class CommandLineExample { public static void main(String args[]){ System.out.println("Your first argument is: "+args[0]); } }</pre> <p>compile by > javac CommandLineExample.java run by > java CommandLineExample sonoo</p>	<p><i>4M for explanation</i></p> <p><i>2M for example</i></p>
<p>b) Ans.</p>	<p>Write a program to input name and salary of employee and throw user defined exception if entered salary is negative.</p> <pre>import java.io.*; class NegativeSalaryException extends Exception { public NegativeSalaryException (String str) { super(str); } } public class S1</pre>	<p>6M</p> <p><i>Extended Exception class with constructor 2M</i></p>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

	<pre> { public static void main(String[] args) throws IOException { BufferedReader br= new BufferedReader(new InputStreamReader(System.in)); System.out.print("Enter Name of employee"); String name = br.readLine(); System.out.print("Enter Salary of employee"); int salary = Integer.parseInt(br.readLine()); Try { if(salary<0) throw new NegativeSalaryException("Enter Salary amount isnegative"); System.out.println("Salary is "+salary); } catch (NegativeSalaryException a) { System.out.println(a); } } } </pre>	<p style="text-align: center;"><i>Acceptin g data 1M</i></p> <p style="text-align: center;"><i>Throwin g user defining Exceptio n with try catch and throw 3M</i></p>
<p>c) Ans.</p>	<p>Describe the applet life cycle in detail.</p> <div style="text-align: center;"> <pre> graph TD Start(()) -- init() --> Born((Born)) Born -- start() --> Running((Running)) Running -- stop() --> Idle((Idle)) Idle -- start() --> Running Running -- paint() --> Running Idle -- destroy() --> Dead((Dead)) </pre> </div> <p>Below is the description of each applet life cycle method:</p> <p>init(): The init() method is the first method to execute when the applet is executed. Variable declaration and initialization operations</p>	<p style="text-align: center;">6M</p> <p style="text-align: center;"><i>2M Diagram</i></p>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Java Programming

Subject Code: 22412

		<p>are performed in this method.</p> <p>start(): The start() method contains the actual code of the applet that should run. The start() method executes immediately after the init() method. It also executes whenever the applet is restored, maximized or moving from one tab to another tab in the browser.</p> <p>stop(): The stop() method stops the execution of the applet. The stop() method executes when the applet is minimized or when moving from one tab to another in the browser.</p> <p>destroy(): The destroy() method executes when the applet window is closed or when the tab containing the webpage is closed. stop() method executes just before when destroy() method is invoked. The destroy() method removes the applet object from memory.</p> <p>paint(): The paint() method is used to redraw the output on the applet display area. The paint() method executes after the execution of start() method and whenever the applet or browser is resized.</p> <p>The method execution sequence when an applet is executed is:</p> <ul style="list-style-type: none">• init()• start()• paint() <p>The method execution sequence when an applet is closed is:</p> <ul style="list-style-type: none">• stop()• destroy()	<p><i>4M descripti on</i></p>
--	--	--	---------------------------------------