

SECTION A

[40 Marks]

Answer ALL questions in this section.

A1.

- | | | |
|-----|--|-----|
| | [1] | [1] |
| (a) | String[] studname=new String[100]; | |
| | [1] | [1] |
| (b) | float[] value={100.50f,300.30f,56.75,80.50f}; | |
| | [1] | [1] |
| (c) | Employee[] emp=new Employee[5] | |

Each answer carries 2 marks. [Max 6 marks]

- | | | |
|-----|----------------------|------------|
| A2. | Class product | [1] |
| | { | |
| | String name; | [1] |
| | int qty; | [1] |
| | } | |

- | | | | |
|-----|-----|------------------------|------------|
| A3. | (a) | applet viewer | [1] |
| | (b) | selection | [1] |
| | (c) | comment | [1] |
| | (d) | syntax | [1] |
| | (e) | Object-oriented | [1] |
| | (f) | false | [1] |
| | (g) | init | [1] |
| | (h) | sorting | [1] |
| | (i) | initialised | [1] |
| | (j) | inheritance | [1] |

Each answer carries 1 mark. [Max 10 marks]

A4.

- | | | |
|-----|------------|------------|
| (a) | iii | [1] |
| (b) | ii | [1] |
| (c) | iii | [1] |
| (d) | i | [1] |
| (e) | iv | [1] |
| (f) | iii | [1] |
| (g) | iv | [1] |
| (h) | i | [1] |
| (i) | ii | [1] |
| (j) | iii | [1] |

A5.

- | | | |
|-----|--------------|------------|
| (a) | False | [1] |
| (b) | True | [1] |
| (c) | True | [1] |
| (d) | False | [1] |
| (e) | True | [1] |

Each answer carries one mark. [Max 5 marks]

A6.

int sum = 0;

for (count = 1; count <= 99; count += 2)

sum += count;

**Award 1 mark for sum initialization, Further 3 marks for for loop and 2 marks
for sum accumulation by add count each time. [Max 6]**

SECTION B

[60 Marks]

Answer ANY TWO questions from this section.

B1.

(a)

[Award max 4 marks for each program constructs. $4 \times 3 = 12$ marks]

• sequence

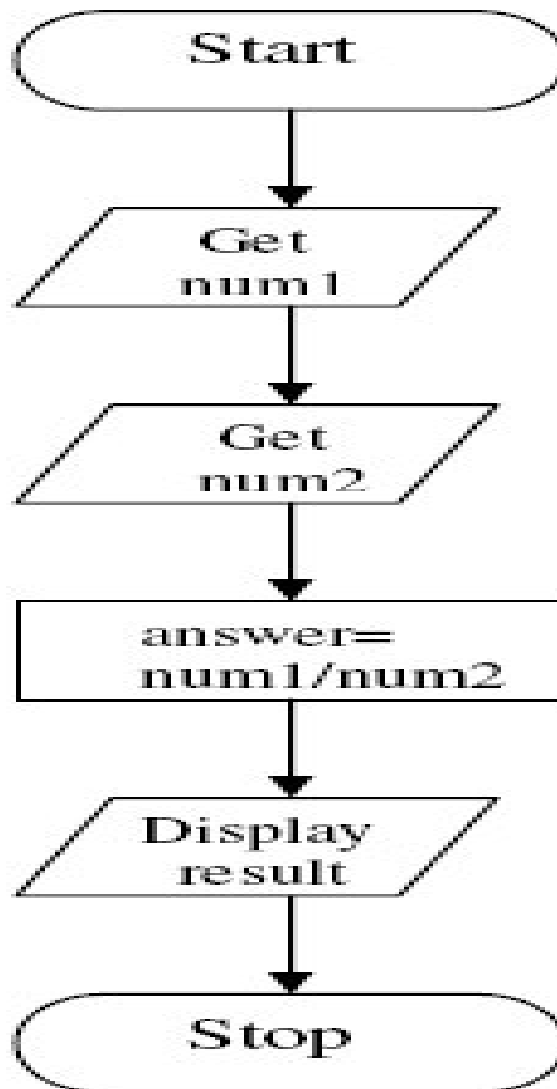
[1]

each instruction is executed in a serial manner one after another.

[1]

Award 2 marks for drawing the flow chart. Accept valid alternative as well.

[2]



• selection

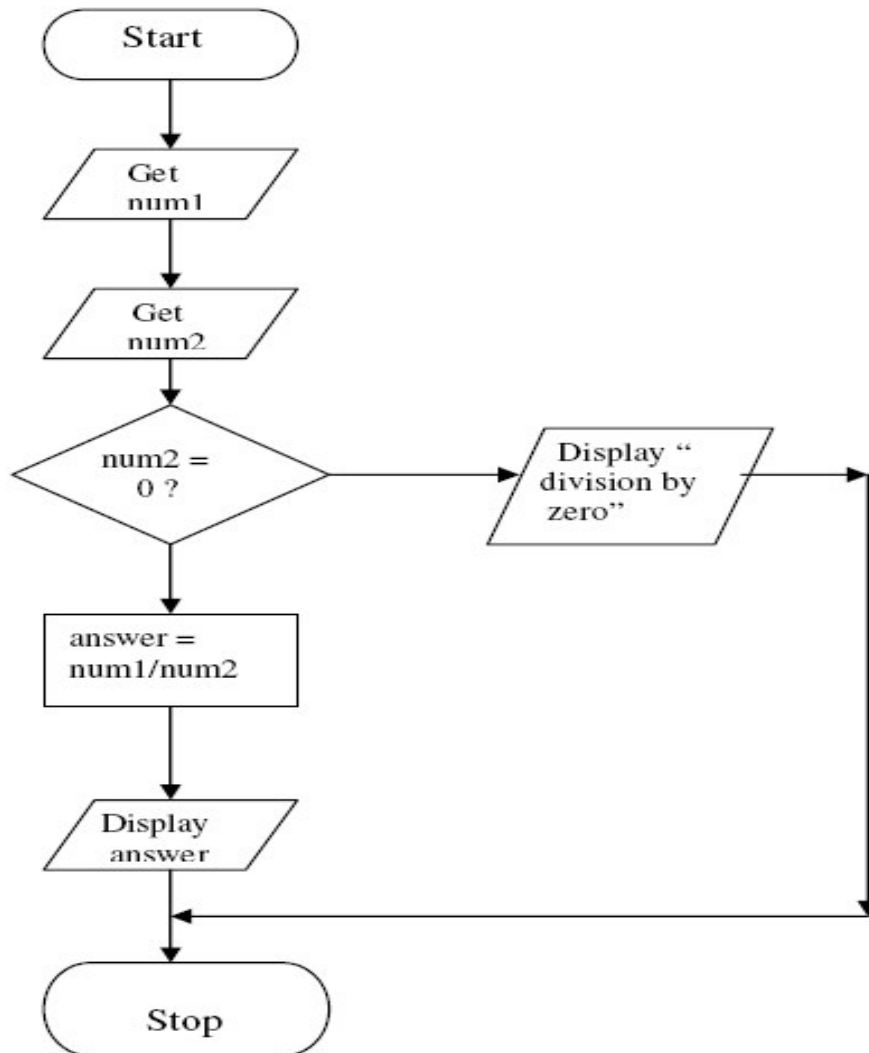
[1]

provides a decision point that allows one of the two choices to be chosen based on some value within a program.

[1]

Award 2 marks for drawing the flow chart. Accept valid alternative as well.

[2]



• iteration

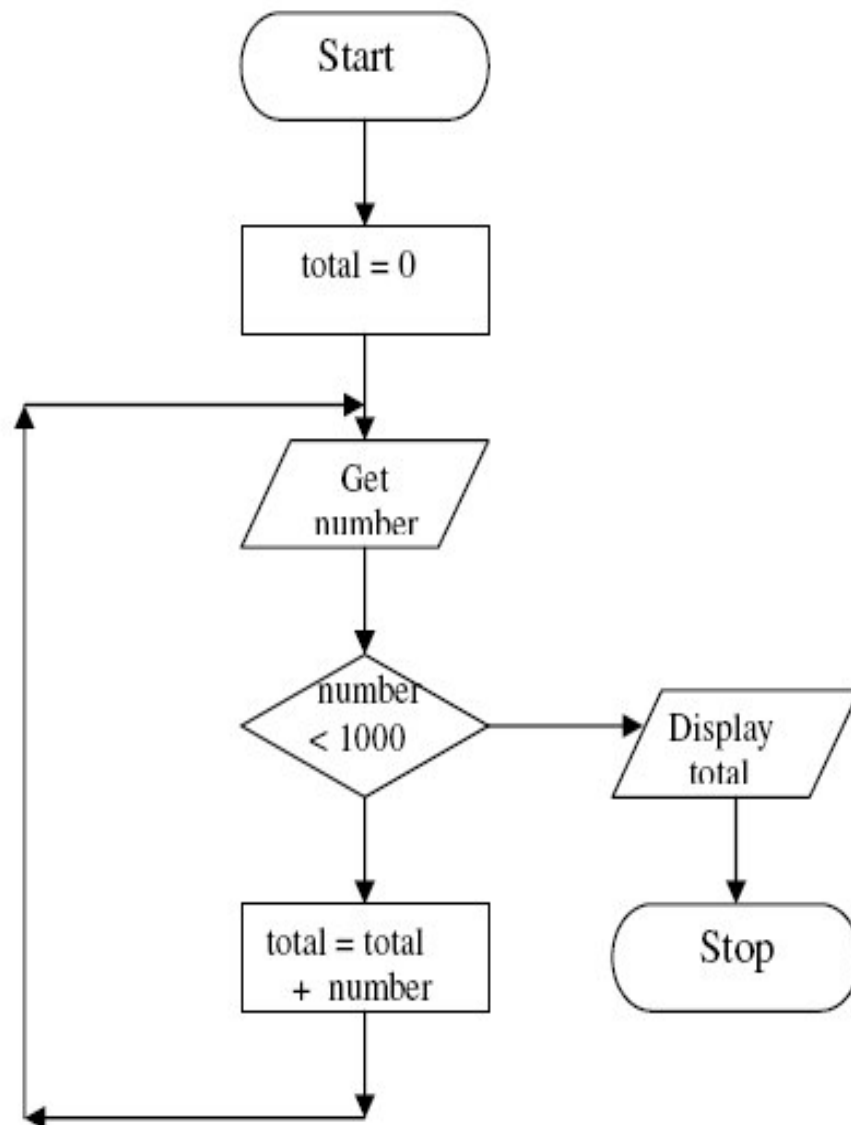
[1]

iteration construct allows a set of instructions to be repeated a number of times based on the condition stated.

[1]

Award 2 marks for drawing the flow chart. Accept valid alternative as well.

[2]



(d) **public static void area(float length) [2]**
{
System.out.println("The area is" + length*length); [1]
}

public static void area(float length, float width) [2]
{
System.out.println("The area is" + length*width); [1]
}

[Max 6 marks] [Award full credit for other suitable examples]

B2.

(a) **public class FindVowels**
{
public static void main (String args[]) throws Exception
{
char[] vowels = {'a','e','i','o','u', 'A','E','I','O','U'}; [1]
int i = 0;
char g = ' ';
boolean isVowel = false; [1]
System.out.print("Enter a character: ");
g = (char)System.in.read(); [1]
System.in.read(); System.in.read();
for (i = 0; i < 10; i++) { [1]
if (g == vowels[i]) [1]
isVowel = true; [1]
}
if (isVowel == true) [1]
System.out.println(g + " is a vowel"); [1]
else
System.out.println(g + " is NOT a vowel"); [1]
}
}
Correct use of {} at class, method and for..loop [1]

- (b) **An exception is an event that occurs during the execution of program** [1]
that prevents the continuation of the normal flow of instructions. [1]
The division by zero operation is an exception because it prevents the code from
continuing. [1]

Example:

```
import java.io.*; [1]  
public class compute [1]  
{  
public static void main(String[] args)throws IOException [1]  
{  
int x = 0; [1]  
int y = 7; [1]  
System.out.println(y/x); [1]  
}  
}  
Correct use of {} [1]
```

- (c) (i) **Invalid** [1]
(ii) **Valid** [1]
(iii) **Invalid** [1]
(iv) **Valid** [1]
(v) **Invalid** [1]
- (d) (i) **concat()** [1]
(ii) **length()** [1]
(iii) **Integer.parseInt()** [1]
(iv) **equals()** [1]
(v) **toCharArray()** [1]

B3.

(a)

import java.applet.Applet; [1]

import java.awt.Graphics; [1]

public class drawline extends Applet [1]

{

public void paint(Graphics g) [1]

{

g.drawLine(10, 10, 230, 95); [2]

}

}

(b) **Button, Label, Menu, TextField, TextArea etc.** [5]

Award 1 mark for each component. [Max 5 marks]

(c)

switch(course)

{

case 'C' :

case 'c' : System.out.println("Computing course");

break;

case 'B' :

case 'b' : System.out.println("Business course");

break;

default : System.out.println("Tourism course");

}

Award 1 mark for switch() {} [1]

Award 1 mark for each case given D/d/A/a along with break keyword [4]

Award 1 mark for print statement given [1]

[Max 6 marks]

- (d) (i) **Two-dimensional arrays store information using a rows and columns format.** [1]
It requires the use of two indexes to access element inside the array. [1]
- (ii) **public class Finance**
{
String managerName; [1]
double[][] payment ; [1]
}
- (iii) **Finance(String manager_name) {** [1]
this.managerName = managerName; [1]
payment = new double[12][31]; [1]
}
- (e)
- (i) **Error : miss the closing right brace of the while body.** [1]
Correction: add closing right brace after the statement ++c; [1]
- (ii) **Error: semicolon after else results in a logic error. The second output statement will always be executed.** [1]
Correction: remove the semicolon after else [1]
- (f) **Syntax error is a compilation error occurs if the language is used incorrectly.** [1]
Logical error occurs if there is a misinterpretation of the requirements [1]

-END OF PAPER-