Name- Bishal Ghosh
Scholar- 19212003 69
Subject - Web Development (B25)
Date- 5/03/2021.
Co-ordinator - Prof. GST
Semester - IV

Page- 1

Q.1 Method Overloading:- same as constructors, we can also overload methods. Conditions for method overloading are:-

1. Methods to be overloaded must have the same name.
2. All methods must have different arguments (either different no. of parameters or different type of parameter).

Example:-

```
class Rectangle {
    Public static void print Area(int x,
                                  int y) {
        System.out.println(x*y);
    }
    Public static void print Area(int x) {
        System.out.println(x*x);
    }
    Public static void print Area(double x,
                                  double y) {
        system.out.println(x*y);
    }
    Public static void Print Area(double x) {
        system.out.println(x*x);
    }
}
```

```
Public static void main (String [] args) {
    PrintArea (2,4);
    PrintArea (2, 5.1);
    PrintArea (10);
    PrintArea (2,3);

}

}.
```

Here we have defined four methods with the same name 'PrintArea' but different parameters. In the main class, firstly the function PrintArea is called with 2 and 4 passed to it. Since, both 2 and 4 are integers, so the method named PrintArea with both its parameter of type int (int x, int y) will be called.

Method overriding:

```
class Animals {
    Public void Sound () {
        System.out.println ("This is the
                            Parent class");
    }
}
class Dog extends Animals {
    Public void sound () {
        System.out.println ("Dogs Bark");
    }
}.
```

```
class m {
    Public static void main (String[] args) {
        Dogs d = new Dogs();
        d.sound();
    }
}
```

In the above Example, the class 'Dogs' and its parent class 'Animals' share the same method "public void sound()". When the object 'd' of the class Dogs call this method, then the method in the child class 'Dogs' is called, not that in the parent class. Thus, the method in the child class overrides the method in the parent class.

B.2 Java program to print counting 1 to 100. in tabular form:-

```
import java.io.*;
class Tabular_form {
    Public static void main (String [] args) {
        int i, j;
        for (i=1; i<=10; i++) {
            for (int j=0; j<=9; ++j) {
```

```
System.out.print( j*10 + i );
System.out.println(" ");
    }
  }
}
```

}.

Q.3 The process by which one class acquires the properties (data members) and functionalities (methods) of another class is called inheritance. The aim of inheritance is to provide the reusability of code so that a class has to write only the unique features and rest of the common properties and functionalities can be extended from another class.

Child class: The ~~child~~ class that extends the features of another class is known as child class, Sub class or derived class.

Parent class: The class whose properties and functionalities are used (inherited) by another class is known as parent class, super class, or Base class.

```
class A {
    int x, y;
    public void show () {
        System.out.println ("numbers are "
                            + x + ", " + y);
    }
}.
```

Name- Bithal Bhosh Scholar- 19212069.

```
class B extends A{ Page 8
    void get (int a, int b)
    { super.x = a;
      super.y = b;
    }
}
class Test {
    Public static void main (String [] args)
    { B. obj = new B();
      obj.get (10,20); obj.show (); }}.
```

Q.4 2D Array: Multidimensional Arrays can be defined in simple words as . arrays of arrays. Data in multidimensional arrays are stored in tabular form (in row mayor order).

eg: int [][] twoDarray = new int [10][20];

```
import java.io.*;
class square of Matrix {
    static void print Matrix ( int M[][],
                               int row, int col)
    {
        for (int i=0 ; i < row ; i++) {
            for (int j=0 ; j < col ; j++) {
                System.out.print (M[i][j]+ " ");
            }
            System.out.println ();
        }
    }
```

```
static void multiplyMatrix ( int row1, int col1, int A[][],
                             int row2, int col2; int B[][])
{
    int i, j, k ;
    if (row2 != col1) {
        System.out.println ( " Multiplication cannot be
                              done");

        return ;
    }
    int C[][] = new int [row1] [col2];
    for (int i=0; i<row1 ; i++) {
        for int j=0 ; j<col2 ; j++) {
            for (k= 0; k<row2 ; k++){
                c[i][j] += A[i] [k] * B [k][j];
        }
    }
    System.out.println ("Resultant Matrix ");
    Print Matrix (C, row1, col2);
}
Public static void main (String[]args){
    int row1 = 4, col1 = 3, row2 = 3, col2=4;
    int A[][] = { {1, 1, 1},
                  {2, 2, 2},
                  {3, 3, 3},
                  {4, 4, 4} };
```

```
int B[][] = { { 1, 1, 1, 1}
              { 2, 2, 2, 2}
              { 4, 5, 6, 7} };

multiplyMatrix ( row1, col1, A, row2, col2, B);
}
}
```

5. operator precedence determines the order in which the operators in an expression are evaluated -

The table below lists the precedence of operators in java ; higher it appear in the table, the higher its precedence -

operators	Precedence
1) Postfix Increment and Decrement	++, --
2) Prefix increment and decrement, and unary	++, --, +, -, ~, !
3) Multiplicative	*, /, %
4) additive	+, -
5) shift	<<, >>, >>>

Name- Bithal Ghosh Scholar- 19212069.

operators	Precedence
6) Relational	<, >, <=, >=, instanceof
7) equality	==, !=
8) bitwise AND	&
9) bitwise exclusive OR	^
10) bitwise inclusive OR	|
11) logical AND	&&
12) logical OR	||
13) ternary	? :
14) assignment	=, +=, -=, *=, /=, %=, &=, ^=, |=, <<=, >>=, >>>=