Mohd. Manzar Iqbal
192120017
MCA - IV sem

Date:
Page:

Mid-Term Exam
MCA - 625

(Q1) **2D Array :-** A 2D array has type such as int[][] or String[][], with two pairs of square brackets. The elements of a 2D array are arranged in a rows and columns, and the new operator for 2D arrays specifies both numbers of rows and the number of columns.

Ex:- int[][] A ;
A = new int [3][4];

```
Public class Example {
    Public static void main (String args[])
    {
    int  a[][] = {{ 1,1,1.}, {2,2,2}, {3,3,3}};
    int  b[][] = {{ 1,1,1 }, {2,2,2}, {3,3,3}};

    int c[][] = new int [3][3];

    for (int i=0; i<3; i++) {
     for (int j=0; j<3 ; j++) {
       c[i][j] = 0;
     for (int k = 0; k<3; k++)
      {
        c[i][j] += a[i][k]* b[k][j];
      }
      System.out.print (c[i][j]+" ");
     }
      System.out.println();
    }
```

192120017

(Q.3) Inheritance :- Inheritance can be defined as the process where one class acquires the properties (methods and fields) of another.
The class which inherits the property of other is known as subclass (desired class, child class) and the class whose properties are inherited is known as superclass (base class, parent class).

```
Class A {
   void show ()
    {
      System.out.printIn( " Inheritance");
    }}
Class B extends A
      {
        void display ()
         {
           System.out.println( " B extends A ");
         }
      }
Class TestInheritance {
      public static void main (Strings args []){
           d B  b=new B();
                b.display();
                b.show();
            }}
```

Output :- B extends A
Output :- Inheritance

B

19212017

Date:
Page:

(Q.1) Method overloading

```
public class Add
 {
   public int add (int a, int b)'
    {
      return (a+b);
    }
   public int add (int a, int b, int c){
      return (a+b+c);
    }
   public double add (double a, double b)
    {
      return (a+b);
    }
   public static void main (String args[])
   {
    Add ob = new Add();
     ob. add (15, 25);
     ob. add ( 15, 25, 35);
     ob. add ( 10.2, 3.2);
   }
 }
```

Output :-

```
40
70
13.2
```

19/12/2017

Method overriding :-

```
class A {
  void show {
    System.out.println ("parent class method");
  }
  class B extends A {
    void. show () {
    super.show ();
    System.out.println ("child class method");
    }
  public static void main (String args[])
    {
     A ob = new B ();
      ob.show ();
     }
    }.
```

(Q.5) :- Operator precedence :-

Operator precedence determines the grouping of terms in an expression and decides how an expression is evaluated. Certain operators have higher precedence than others; for example multiplication operator has higher precedence than addition operator.

Date:
Page:

192120017

(Q.5) Operator Precedence

Operator	Precedence
postfix increment & decrement	++, --
prefix increment & decrement and unary	++, --, +, -, ~, !
multiplicative	*, /, %
additive	+ -
shift	<<, >>, >>>
relational	< > <= >=
equality	== !=
Bitwise AND	&
Bitwise XOR	^
Bitwise inclusive OR	|
logical AND	&&
logical OR	||
ternary	?:
assignment	=, +=, -=, *=, /=, &=, ^=, |=, <<=, >>=, >>>=

Scanned with CamScanner

Date:
Page:

192120017

(Q.2)

```
class Solution
{
  public static void main(string [] args)
   {
       int columns, rows, number;
       rows = 1;
        do{
            columns = 0;
          do{
            System.out.printf("%d\t", column*10+rows);
          }  while (columns++ < 9);
             System.out.println();
           } while (rows++ < 10);
    }
  }
```