Exercise - 1 (Basics)

a) Displaying default value of all primitive data types

Aim: To write a JAVA program to display default value of all primitive data type of JAVA

Program:

```
class defaultdemo
{
static byte b;
static short s;
static int i;
static long l;
static float f;
static double d;
static char c;
static boolean bl;
public static void main(String[] args)
{
System.out.println("The default values of primitive data types are:");
System.out.println("Byte :"+b);
System.out.println("Short :"+s);
System.out.println("Int :"+i);
System.out.println("Long :"+l);
System.out.println("Float :"+f);
System.out.println("Double :"+d);
System.out.println("Char :"+c);
System.out.println("Boolean :"+bl);
}
}
```

b) Roots of a quadratic equation

Aim: To write a java program that display the roots of a quadratic equation ax2+bx=0. Calculate the discriminate D and basing on value of D, describe the nature of root.

Program:

```
import java.util.*;
class quadraticdemo
{
public static void main(String[] args)
{
int a, b, c;
double r1, r2, D;
Scanner s = new Scanner(System.in);
System.out.println("Given quadratic equation:ax^2 + bx + c");
System.out.print("Enter a:");
a = s.nextInt();
System.out.print("Enter b:");
b = s.nextInt();
System.out.print("Enter c:");
c = s.nextInt();
D = b * b - 4 * a * c;
if(D > 0)
{
System.out.println("Roots are real and unequal");
r1 = ( - b + Math.sqrt(D))/(2*a);
r2 = (-b - Math.sqrt(D))/(2*a);
System.out.println("First root is:"+r1);
System.out.println("Second root is:"+r2);
}
else if(D == 0)
{
System.out.println("Roots are real and equal");
r1 = (-b+Math.sqrt(D))/(2*a);
System.out.println("Root:"+r1);
}
else
{
System.out.println("Roots are imaginary");
}
}
}
```

c) Bike Race

Aim: Five Bikers Compete in a race such that they drive at a constant speed which may or may not be the same as the other. To qualify the race, the speed of a racer must be more than the average speed of all 5 racers. Take as input the speed of each racer and print back the speed of qualifying racers.

Program:

```
import java.util.*;
class racedemo
{
public static void main(String[] args)
{
float s1,s2,s3,s4,s5,average;
Scanner s = new Scanner(System.in);
System.out.println("Enter speed of first racer:");
s1 = s.nextFloat();
System.out.println("Enter speed of second racer:");
s2 = s.nextFloat();
System.out.println("Enter speed of third racer:");
s3 = s.nextFloat();
System.out.println("Enter speed of fourth racer:");
s4 = s.nextFloat();
System.out.println("Enter speed of fifth racer:");
s5 = s.nextFloat();
average=(s1+s2+s3+s4+s5)/5;
if(s1>average)
System.out.println("First racer is qualify racer:");
else if(s2>average)
System.out.println("Second racer is qualify racer:");
else if(s3>average)
System.out.println("Third racer is qualify racer:");
else if(s4>average)
System.out.println("Fourth racer is qualify racer:");
else if(s5>average)
System.out.println("Fifth racer is qualify racer:");
}
}
```

d) A case study

Aim: A case study on public static void main(250 words)

Case study:

The program structure of a simple java program is given below with different steps

Step-1: Click start+run and then type notepad in run dialog box and click OK. It displays Notepad.

Step-2: In run dialogbox type cmd and click OK. It displays command prompt.

Step-3: Type the following program in the Notepad and save the program as “example.java” in a current working directory.

```
class example

{

public static void main(String args[])

{

System.out.println(“Welcome”);

}

}
```

Step-4 (Compilation): To compile the program type the following in current working directory and then click enter.

```
c:\xxxx >javac example.java
```

Step-5 (Execution): To run the program type the following in current working directory and then click enter.

```
c:\xxxx>java example
```

Explanation:

Generally the file name and class name should be same. If it is not same then the java file can be compiled but it cannot be executed. That is when execution it gives the following error

```
Exception in thread "main" java.lang.NoClassDefFoundError: ex
```

In “public static void main(String args[])” statement

public is an access specifier. If a class is visible to all classes then public is used

main() must be declared as public since it must be called by outside of its class.

The keyword **static** allows **main()** to be called without creating object of the class.

The keyword **void** represents that **main( )** does not return a value.

The main method contains one parameter String args[].

We can send some input values (arguments) at run time to the String args[] of the main method . These arguments are called command line arguments. These command line arguments are passed at the command prompt.

In System.out.println("Welcome"); statement

System is a predefined class that provides access to the system.

out is the output stream.

println() method display the output in different lines. If we use **print()** method it display the output in the same line

Exercise - 2 (Operations, Expressions, Control-flow, Strings)

**a). Write a JAVA program to search for an element in a given list of elements using binary
search mechanism.**

```
import java.util.Scanner;
import java.io.*;
class Binarys
{
  public static void main(String args[])
   {
      int c,first,last,middle,n,search,array[];
      boolean status=false;
       Scanner s=new Scanner(System.in);
       System.out.println("Enter number of elements:");
       n=s.nextInt();
       array=new int[n];
        System.out.println("Enter"+n+"integer:");
      for(c=0;c<n;c++)
        array[c]=s.nextInt();
     for (int i = 0; i<array.length; i++)
         {
           for (int j = i + 1; j <array.length; j++)
           {
               if (array[i] > array[j])
               {
                 int  temp = array[i];
                   array[i] = array[j];
                   array[j] = temp;
               }
           }
      }
        System.out.println("Enter value to find:");
        search=s.nextInt();
        first=0;
        last=n-1;
        middle=(first+last)/2;
    for(int i=0;i<n;i++)
     {
        if(first<=last)
         {
            if(array[middle]<search)
                 first=middle+1;
            else if(array[middle]==search)
            {
               status=true;
            }
          else
           {
             last=middle-1;
           }
              middle=(first+last)/2;
        }
    }
      if(status==true)
      {
         System.out.println(search+"found at location"+(middle+1));
      }
    else
       System.out.println(search+"is not found in the list");
   }
}
```

b). Write a JAVA program to sort for an element in a given list of elements using bubble sort.

```
import java.util.Scanner;
public class Bubblesort
{
 public static void main(String[] args)
 {
       Scanner s=new Scanner(System.in);
       System.out.println("enter the size of the array:");
       int size=s.nextInt();
       System.out.println("enter the values into the array:");
       int arr[]=new int[size];
       for(int i=0;i<size;i++)
       {
           arr[i]=s.nextInt();
       }
       sorting(arr);
    }
    public static void sorting(int arr[])
    {
       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])
        {
          temp=arr[j-1];
          arr[j-1]=arr[j];
          arr[j]=temp;
        }
       }
     }
      System.out.println("the sorted arraty is:");
      for(int i=0;i<arr.length;i++)
      {
        System.out.print(arr[i]+"\t");
      }
   }
}
```

(c). Write a JAVA program to sort for an element in a given list of elements using merge sort.

```
import java.util.Scanner;
class merge
{
 public static void main(String args[])
 {
  Scanner s=new Scanner(System.in);
  System.out.println("enter size of the array:");
  int size=s.nextInt();
  int arr[]=new int[size];
  System.out.println("enter elements into the array:");
  for(int i=0;i<size;i++)
  {
   arr[i]=s.nextInt();
  }
 int start=0;
 int end=size-1;
 sort(arr,start,end);
System.out.println("the sorted array is:");
 for(int j=0;j<size;j++)
 {
  System.out.print(arr[j]+"\t");
 }
 }
 public static void sort(int arr[],int start,int end)
 {
  if(start<end)
  {
    int middle=(start+end)/2;
    sort(arr,start,middle);
    sort(arr,middle+1,end);
    merge(arr,start,middle,end);
  }
}
public static void merge(int arr[],int start,intmiddle,int end)
{
  int n1 = middle -start + 1;
   int n2 = end - middle;
   int L[] = new int [n1];
   int R[] = new int [n2];
   for (int i=0; i<n1; ++i)
   {
       L[i] = arr[start + i];
    }
    for (int j=0; j<n2; ++j)
    {
        R[j] = arr[middle + 1+ j];
     }
     int i = 0, j = 0;
     int k = start;
     while (i< n1 && j < n2)
     {
         if (L[i] <= R[j])
         {
            arr[k] = L[i];
             i++;
         }
        else
        {
          arr[k] = R[j];
          j++;
         }
         k++;
    }
    while (i< n1)
    {
        arr[k] = L[i];
         i++;
         k++;
     }
     while (j < n2)
     {
        arr[k] = R[j];
         j++;
         k++;
       }

  }
}
```

(d) Write a JAVA program using StringBufferto delete, remove character.

```
import java.util.Scanner;
import java.lang.*;
class stringbuffer
{
  public static void main(String args[])
  {
     Scanner s=new Scanner(System.in);
     System.out.println("enter a string you like:");
     String str=s.nextLine();
     StringBuffer b=new StringBuffer(str);
     System.out.println("enter the index you want to delete from:");
      int i1=s.nextInt();
     System.out.println("to:");
     int i2=s.nextInt();
     b.delete(i1,i2);
     System.out.println("after deletion the new buffer is:"+b);
  }
}
```

Exercise - 3 (Class, Objects)

a). Write a JAVA program to implement class mechanism. – Create a class, methods and invoke them inside main method.

```
class Student
{
   String name="Raju";
   String _class="B.Tech 1st year";
   int rollno=178;
   public void display()
   {
     System.out.println("Name of the student is:"+name);
     System.out.println("Present qualification of the student is:"+_class);
     System.out.println("Roll number of the student is:"+rollno);
   }

}
class Details
{
  public static void main(String args[])
  {
    Student s1=new Student();
    s1.display();
  }
}
```

b). Write a JAVA program to implement constructor.

```
import java.util.Scanner;
class Sports_person
{
 int height=164;
  int weight=62;
  Sports_person()
  {
		System.out.print("Enter name of the country:");
		Scanner sc=new Scanner(System.in);
		String country=sc.nextLine();
    System.out.println("Height of the sports person is:"+height+"CM");
    System.out.println("Weight of the sports person is:"+weight+"KG");
System.out.println("He/She belongs to the country:"+country);

  }
}
class Olimpics
{
  public static void main(String args[])
  {
     Sports_personsp=new Sports_person();
  }

}
```

Exercise - 4 (Methods)

a) Constructor Overloading

Aim: To write a JAVA program to implement constructor overloading

Program:

```
class A
{
int l,b;
A()
{
l=10;
b=20;
}
A(int u,int v)
{
l=u;
b=v;
}
int area()
{
return l*b;
}
}
class overconstructdemo
{
public static void main(String args[])
{
A a1=new A();
int r1=a1.area();
System.out.println("The area is: "+r1);
A a2=new A(30,40);
int r2=a2.area();
System.out.println("The area is: "+r2);
}
}
```

b) Method Overloading

Aim: To write a JAVA program implement method overloading

Program:

```
class A
{
int l=10,b=20;
int area()
{
return l*b;
}
int area(int l,int b)
{
return l*b;
}
}
class overmethoddemo
{
public static void main(String args[])
{
A a1=new A();
int r1=a1.area();
System.out.println("The area is: "+r1);
int r2=a1.area(5,20);
System.out.println("The area is: "+r2);
}
}
```

Exercise - 5 (Inheritance)

a)Implementing Single Inheritance

Aim: To write a JAVA program to implement Single Inheritance

Program:

```
class A
{
A()
{
System.out.println("Inside A's Constructor");
}
}
class B extends A
{
B()
{
System.out.println("Inside B's Constructor");
}
}
class singledemo
{
public static void main(String args[])
{
B b1=new B();
}
}
```

b)Multi level Inheritance

Aim: To write a JAVA program to implement multi level Inheritance

Program:

```
class A
{
A()
{
System.out.println("Inside A's Constructor");
}
}
class B extends A
{
B()
{
System.out.println("Inside B's Constructor");
}
}
class C extends B
{
C()
{
System.out.println("Inside C's Constructor");
}
}
class multidemo
{
public static void main(String args[])
{
C c1=new C();
}
}
```

c)Abstract Class

Aim: To write a java program for abstract class to find areas of different shapes

Program:

```
abstract class shape
{
abstract double area();
}
class rectangle extends shape
{
double l=12.5,b=2.5;
double area()
{
return l*b;
}
}
class triangle extends shape
{
double b=4.2,h=6.5;
double area()
{
return 0.5*b*h;
}
}
class square extends shape
{
double s=6.5;
double area()
{
return 4*s;
}
}
class shapedemo
{
public static void main(String[] args)
{
rectangle r1=new rectangle();
triangle t1=new triangle();
square s1=new square();
System.out.println("The area of rectangle is: "+r1.area());
System.out.println("The area of triangle is: "+t1.area());
System.out.println("The area of square is: "+s1.area());
}
}
```

Exercise - 6 (Inheritance - Continued)

a)super keyword implementation

Aim: Write a JAVA program give example for “super” keyword

Programs:

(i)Using super to call super class constructor (Without parameters)

```
class A
{
int l,b;
A()
{
l=10;
b=20;
}
}
class B extends A
{
int h;
B()
{
super();
h=30;
}
int volume()
{
return l*b*h;
}
}
class superdemo
{
public static void main(String args[])
{
B b1=new B();
int r=b1.volume();
System.out.println("The vol. is: "+r);
}
}
```

(ii)Using super to call super class constructor (With parameters)

```
class A
{
int l,b;
A(int u,int v)
{
l=u;
b=v;
}
}

class B extends A
{
int h;
B(int u,int v,int w)
{
super(u,v);
h=w;
}
int volume()
{
return l*b*h;
}
}
class superdemo
{
public static void main(String args[])
{
B b1=new B(30,20,30);
int r=b1.volume();
System.out.println("The vol. is: "+r);
}
}
```

b) Implementing interface

Aim: To write a JAVA program to implement Interface.

Programs:

(i) First form of interface implementation

```
interface A
{
void display();
}
class B implements A
{
public void display()
{
System.out.println("B's method");
}
}
class C extends B
{
public void callme()
{
System.out.println("C's method");
}
}
class interfacedemo
{
public static void main(String args[])
{
C c1=new C();
c1.display();
c1.callme();
}
}
```

(ii) Second form of interface implementation

```
interface D
{
void display();
}
interface E extends D
{
void show();
}
class A
{
void callme()
{
System.out.println("This is in callme method");
}
}

class B extends A implements E
{
public void display()
{
System.out.println("This is in display method");
}
public void show()
{
System.out.println("This is in show method");
}
}
class C extends B
{
void call()
{
System.out.println("This is in call method");
}
}
class interfacedemo
{
public static void main(String args[])
{
C c1=new C();
c1.display();
c1.show();
c1.callme();
c1.call();
}
}
```

(iii) Third form of interface implementation

```
interface A
{
void display();
}
class B implements A
{
public void display()
{
System.out.println("This is in B's method");
}
}
class C implements A
{
public void display()
{
System.out.println("This is C's method");
}
}
class interfacedemo
{
public static void main(String args[])
{
B b1=new B();
C c1=new C();
b1.display();
c1.display();
}
}
```

(iv) Fourth form of interface implementation

```
interface A
{
void display();
}
interface B
{
void callme();
}
interface C extends A,B
{
void call();
}
class D implements C
{
public void display()
{
System.out.println("interface A");
}
public void callme()
{
System.out.println("interface B");
}
public void call()
{
System.out.println("interface C");
}
}
class interfacedemo
{
public static void main(String args[])
{
D d1=new D();
d1.display();
d1.callme();
d1.call();
}
}
```