Q1. 1. Stack Operations using Interface: Create an interface Stack with a variable size and abstract methods push(), pop(), display(), overflow(), and underflow(). Implement a subclass IntegerStack that implements the Stack interface. Create a test class to check the working of all methods in the IntegerStack class.

```
import    java.util.*;
interface    Stack    {
int size = 5;      void
push();      int pop();
void          display();
void       overflow();
void underflow();
}
class Integerstack implements Stack {
int[] stack = new int[size];     int top =
-1;
   Scanner sc = new Scanner(System.in);


   public void push() {
if (top == size - 1) {
overflow();
      } else {
         System.out.print("Enter element to push: ");
int value = sc.nextInt();              stack[++top] =
value;
         System.out.println(value + " pushed to stack");
      }
   }    public int
pop() {        if (top
== -1) {
underflow();
return -1;        }
else {
         int poppedv = stack[top--];
         System.out.println(poppedv + " popped from stack.");
return poppedv;
      }
   }
   public void display() {
if (top == -1) {
```

```
            System.out.println("Stack is empty.");
        } else {
            System.out.print("Stack: ");
for (int i = top; i >= 0; i--) {
                System.out.print(stack[i] + " ");
            }
            System.out.println();
        }
    }
    public void overflow() {
        System.out.println("Stack Overflow! Cannot push more elements.");
    }
    public void underflow() {
        System.out.println("Stack Underflow! Cannot pop.");
    }
}
public class test {     public static void
main(String[] args) {         Scanner sc = new
Scanner(System.in);         Integerstack stack
= new Integerstack();         while (true) {
            System.out.println("\n1: Push");
            System.out.println("2: Pop");
            System.out.println("3: Display");
            System.out.println("4: Exit");
System.out.print("Enter your choice: ");
int choice = sc.nextInt();             switch
(choice) {                 case 1:
stack.push();                   break;                 case
2:                   stack.pop();                   break;
case 3:                   stack.display();
break;                 case 4:
                    System.out.println("Exiting program...");
return;                 default:
                    System.out.println("Invalid choice! Try again.");
            }
        }
    } }
```

Output:

```
1: Push
```

```
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter element to push: 1
1 pushed to stack
1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 1
Enter element to push: 2
2 pushed to stack
1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 2
2 popped from stack.
1: Push
2: Pop
3: Display
4: Exit
Enter your choice: 3
Stack: 1
```

Q2. Shape Interface with Rectangle and Triangle: Implement the following: a. Create an interface Shape with an abstract method area(). b. Create two classes, Rectangle and

Triangle, that implement the Shape interface. c. Calculate and display the area of both Rectangle and Triangle.

```
import java.util.Scanner;
interface Shape {     void
area();
}
class Rectangle implements Shape {
int  length,  breadth;        public  void
area() {
        Scanner sc = new Scanner(System.in);
System.out.print("Enter length of rectangle: ");         length
= sc.nextInt();
        System.out.print("Enter breadth of rectangle: ");
breadth = sc.nextInt();
        System.out.println("Area of Rectangle: " + (length * breadth));
    }
}
class Triangle implements Shape {
int  base,  height;         public  void
area() {
        Scanner sc = new Scanner(System.in);
System.out.print("Enter base of triangle: ");
base = sc.nextInt();
        System.out.print("Enter height of triangle: ");
height = sc.nextInt();
        System.out.println("Area of Triangle: " + (0.5 * base * height));
    }
}
public class Shapetest {     public static
void main(String[] args) {
Rectangle rect = new Rectangle();
rect.area();
        Triangle tri = new Triangle();
tri.area();
    }
} output:
```

```
Enter length of rectangle: 2

Enter breadth of rectangle: 3
Area of Rectangle: 6
```

```
Enter base of triangle: 4

Enter height of triangle: 5

Area of Triangle: 10.0
```

Q3. Student Exam Results Using Inheritance and Interface in: Implement the following hierarchy: a. Create a class Student with a variable rollNo and methods getRollNo() and setRollNo(). b. Create a class Test that inherits Student and has variables sub1 and sub2 with methods getMarks() and setMarks(). c. Create an interface Sports with a variable sMarks and a method set(). d. Create a class Result that inherits Test and implements the Sports interface. It should display the marks. e. Demonstrate the functionality of these classes in a test application.

```
import java.util.*;
class Student {     int
rollno;     void
setRollNo() {
        Scanner  sc  =  new  Scanner(System.in);
System.out.print("Enter   Roll   Number:   ");
rollno = sc.nextInt();
    }
    void getRollNo() {
        System.out.println("Roll Number: " + rollno);
    }
}
class Test extends Student {
int  sub1,  sub2;           void
setMarks() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter marks for Subject 1: ");
sub1 = sc.nextInt();
        System.out.print("Enter marks for Subject 2: ");
sub2 = sc.nextInt();
    }

    void getMarks() {
        System.out.println("Subject 1 Marks: " + sub1);
        System.out.println("Subject 2 Marks: " + sub2);
    }
}
interface Sports {     void
setSportsMarks();
}
class Result extends Test implements Sports {
int sportsMarks;


   public void setSportsMarks() {
Scanner sc = new Scanner(System.in);
System.out.print("Enter Sports Marks: ");
sportsMarks = sc.nextInt();
   }
   void displayResult() {
      System.out.println("\n--- Student Exam Results ---");
getRollNo();         getMarks();
      System.out.println("Sports Marks: " + sportsMarks);
int total = sub1 + sub2 + sportsMarks;
      System.out.println("Total Marks: " + total);
   }
}
public class Testapplication {     public
static void main(String[] args) {
Result student = new Result();
student.setRollNo();
student.setMarks();
student.setSportsMarks();
student.displayResult();
   }
} output:
```

```
Enter Roll Number: 37

Enter marks for Subject 1: 90

Enter marks for Subject 2: 90

Enter Sports Marks: 12

--- Student Exam Results ---

Roll Number: 37

Subject 1 Marks: 90

Subject 2 Marks: 90

Sports Marks: 12

Total Marks: 192
```