Java Programming

Strings

Compare Two Strings

1. Write a program to compare two strings lexicographically and also to check whether a given string end with the contents of another string.

```
public class CheckStrings {
   public static void main(String[] args) {
      String str1 = "Password";
      String str2 = "Passkeys";

      System.out.println("String 1: " + str1);
      System.out.println("String 2: " + str2);

      int result = str1.compareTo(str2);
      if (result < 0) {
         System.out.println("String 1 is less than String 2");
      }
      else if (result == 0) {
         System.out.println("String 1 is equal to String 2");
      }
      else {
         System.out.println("String 1 is greater than String 2");
      }

      String end_str = "rd";
      boolean ends1 = str1.endsWith(end_str);
      boolean ends2 = str2.endsWith(end_str);

      // Display the results of the endsWith calls.
      System.out.println("String 1 ends with " + end_str + " = " + ends1);
      System.out.println("String 2 ends with " + end_str + " = " + ends2);
   }
}
```

Output

```
String 1: Password
String 2: Passkeys
String 1 is greater than String 2
String 1 ends with rd = true
String 2 ends with rd = false
```

Maximum Occurring Character

2. Write a program to find the maximum occurring character in a string.

```
import java.util.*;

public class TestString {
   static final int N = 256;
      static char MaxOccuringChar(String str1) {
         int ctr[] = new int[N];
         int l = str1.length();
          int max = -1;
         char result = ' ';

         for (int i = 0; i < l; i++)
            ctr[str1.charAt(i)]++;

         for (int i = 0; i < l; i++) {
            if (max < ctr[str1.charAt(i)]) {
               max = ctr[str1.charAt(i)];
               result = str1.charAt(i);
             }
         }

         return result;
       }
   public static void main(String[] args) {
      String str1 = "test string";
      System.out.println("The given string is: " + str1);
      System.out.println("Max occurring character in the given string is: " + MaxOccuringChar(str1));
    }
}
```

Output

```
The given string is: test string
Max occurring character in the given string is: t
```

Palindrome

3. Write a program to check if the given string is palindrome or not.

```
import java.util.*;

public class PalindromeString {

   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Enter a String to Check:");
      String inputString = input.nextLine();
      //String inputString = "madam";
      String reversedString = reverseString(inputString);

      if (inputString.equals(reversedString)) {
         System.out.println("The inputString '" + inputString + "' is a palindrome String.");
      } else {
         System.out.println("The inputString '" + inputString + "' is not a palindrome String.");
      }
   }

   // a method t reverse the string
   public static String reverseString(String str) {
      String reversedString = "";

      for (int i = str.length() - 1; i >= 0; i--) {
         reversedString += str.charAt(i);

      }
      return reversedString;
   }
}
```

Output 1

```
Enter a String to Check:
radar
The inputString 'radar' is a palindrome String.
```

Output 2

```
Enter a String to Check:
hello
The inputString 'hello' is not a palindrome String.
```

Interface

GeoAnalyzer

4. Create an interface called GeoAnalzer. The interface has a final variable, pi, and an abstract method called area and Perimeter that computes and returns the value area & Perimeter to the class. Create a class called Ellipse and Rectangle that implements GeoAnalyser and displays the values of area & Perimeter. Create an application for Geometry Class that demonstrates the use of the types. The objects are initialized using the constructor of the respective classes.

```
interface GeoAnalyzer {
   final static float pi = 3.142F;
   float area();
   float perimeter();
}
```

```
class Ellipse implements GeoAnalyzer {
   float major;
   float minor;

   Ellipse(float m, float n) {
      major = m;
            minor = n;
   }

   public float area() {
            return (pi * major * minor);
   }

   public float perimeter() {
            return (pi * (major + minor));
   }
}
```

```
class Rectangle implements GeoAnalyzer {
   float length;
   float width;

   Rectangle(float l, float w) {
      length = l;
      width = w;
   }

   public float area() {
            return (length * width);
   }

   public float perimeter() {
            return (2 * (length + width));
   }
}
```

```
class Geometry {
   static void display(float x, float y) {
      System.out.println("Area = " + x + " Perimeter = " + y);
   }
```

```
    public static void main(String args[]) {
        Ellipse e = new Ellipse(4.5f, 3.6f);
        System.out.print("Ellipse: ");
        display(e.area(), e.perimeter());

        Rectangle r = new Rectangle(6.5f, 4.3f);
        System.out.print("Rectangle: ");
        display(r.area(), r.perimeter());
    }
}
```

Output

```
Ellipse: Area = 50.9004 Perimeter = 25.450201
Rectangle: Area = 27.95 Perimeter = 21.6
```

Sharing Member Elements

5. Create an interface called SharedConstants. The interface has a final variable NO, YES, MAYBE, LATER, SOON, and NEVER, values ranging from 0 to 5, respectively. Create a class called Question that generates a random value based on which the if-else ladder returns different values of final variables defined in the interface. Create a class AskMe and define a method. Answer receives the value returned from the Question class and uses the switch case & final variables of the interface to evaluate and print values as No, Yes, Maybe, Later, Soon, Never.

```
interface SharedConstants {
   int NO = 0;
   int YES = 1;
   int MAYBE = 2;
   int LATER = 3;
   int SOON = 4;
   int NEVER = 5;
}
```

```
import java.util.*;
class Question implements SharedConstants {
   Random rand = new Random();

   int ask() {
      int prob = (int) (100 * rand.nextDouble());
            if (prob < 30) {
               return NO; // 30%
            } else if (prob < 60) {
               return YES; // 30%
            } else if (prob < 75) {
               return LATER; // 15%
            } else if (prob < 98) {
               return SOON; // 13%
            } else {
               return NEVER; // 2%
            }
   }
}
```

```
class AskMe implements SharedConstants {
   static void answer(int result) {
      switch (result) {
               case NO:
                  System.out.println("No");
                  break;
               case YES:
                  System.out.println("Yes");
                  break;
               case MAYBE:
            System.out.println("Maybe");
            break;
         case LATER:
            System.out.println("Later");
            break;
         case SOON:
            System.out.println("Soon");
            break;
        case NEVER:
```

```
            System.out.println("Never");
        break;
      }
   }

   public static void main(String args[]) {
      Question q = new Question();
      answer(q.ask());
      answer(q.ask());
      answer(q.ask());
   }
}
```

Output 1

No
Yes
Soon

Output 2

Later
Yes
Later

Angles

6. Write a program to create an interface Trigonometry contains abstract methods sine (), cos (), tan (), cosec (), tan (), cosec (), sec (), cot () with one of the parameters as a degree. Create a class TrigonometryDemo that implements the interface. Create an object and invoke all the methods to display the values for 60 degrees.

```
interface Trigonometry {
   float sine(double degrees);
   float cos(double degrees);
   float tan(double degrees);
   double cosec(double degrees);
   double sec(double degrees);
   double cot(double degrees);
}
```

```
public class TrigonometryDemo implements Trigonometry {
   public float sine(double degrees) {
      double s;
      s = (Math.sin(Math.toRadians(degrees)));
      return (float) s;
   }

   public float cos(double degrees) {
      double s;
      s = (Math.cos(Math.toRadians(degrees)));
      return (float) s;
   }

   public float tan(double degrees) {
      double s;
      s = (Math.tan(Math.toRadians(degrees)));
      return (float) s;
   }

   public double cosec(double degrees) {
      double s;
      s = (Math.sin(Math.toRadians(degrees)));
      s = 1 / s;
      return s;
   }

   public double sec(double degrees) {
      double s;
      s = (Math.cos(Math.toRadians(degrees)));
      s = 1 / s;
      return s;
   }

   public double cot(double degrees) {
      double s;
      s = (Math.tan(Math.toRadians(degrees)));
      s = 1 / s;
      return s;
   }

   public static void main(String args[]) {
```

```
        TrigonometryDemo t = new TrigonometryDemo();
        System.out.println("Sin value is:" + t.sine(60));
        System.out.println("Cos value is:" + t.cos(60));
        System.out.println("Tan value is:" + t.tan(60));
        System.out.println("Cosec value is:" + t.cosec(60));
        System.out.println("Sec value is:" + t.sec(60));
        System.out.println("Cot value is:" + t.cot(60));
    }
}
```

Output

```
Sin value is:0.8660254
Cos value is:0.5
Tan value is:1.7320508
Cosec value is:1.1547005383792517
Sec value is:1.9999999999999996
Cot value is:0.577350269189626
```

Package

Player Interface

7. Create an interface called Player. The interface has an abstract method called play() that displays a message describing the meaning of "play" to the class. Create classes called Child, Musician, and Actor that all implement Player. Create an application that demonstrates the use of the classes (UsePlayer.java). Save the files as Player.java and keep inside package p1; Child.java, Actor.java, and Musician.java and keep them inside package p2; and UsePlayer.java in package p3.

```
package p1;
public interface Player {
   void play ();
}
```

```
package p2;
import p1.Player;
public class Child implements Player {
   @Override
   public void play () {
      System.out.println("Child plays with Lego.");
   }
}
```

```
package p2;
import p1.Player;
public class Musician implements Player {
   @Override
   public void play () {
      System.out.println("Musician plays a piano.");
   }
}
```

```
package p2;
 import p1.Player;
 public class Actor implements Player {
   @Override
   public void play () {
       System.out.println("Actor plays in a film.");
   }
}
```

```
package p3;
import p1.Player;
import p2.*;
public class UsePlayer {
   public static void main (String [] args) {
      Player player;
      player = new Child();
      player.play();
      player = new Musician();
      player.play();
      player = new Actor();
      player.play();
   }
}
```

Output:
Child plays with Lego.
Musician plays a piano.
Actor plays in a film.

Check Balance

8. Create a class called Balance with instance variables name, bal, and two-parameter constructor to initialize the instance variables. Create classes called Saving, with inherits the members of the Balance class and has its own members branch, customerID, a four-parameter constructor to initialize the variables, and a method show () that checks the value of Balance is less than zero. Create an array of 3 objects for the Saving class to test its function in the Demo class. Save the files as Balance.java inside package p4, Saving.java, and Demo.java inside package p5.

```
package p4;

public class Balance {
   public String name;
   public double bal;

   protected Balance(String n, double b) {
      name = n;
      bal = b;
   }
}
```

```
package p5;
import p4.Balance;

public class Savings extends Balance{
    String branch;
    int customerID;
    Savings(String n, double bal, String br, int id) {
       super(n, bal);
       branch = br;
       customerID = id;
    }

   void show() {
            if (bal < 0) {
            System.out.print("Sorry! Negative balance ");
            }
            System.out.println(name + ", Rs." + bal + ", "+ branch + ", "+customerID);
   }
}
```

```
package p5;

class Demo{
   public static void main(String args[]) {
      Savings current[] = new Savings[3];
      current[0] = new Savings("D. Samanta", 123.23, "Kolkata", 555);
      current[1] = new Savings("T. Ahmed", 157.02, "Bangalore", 727);
      current[2] = new Savings("N. Sinhababu", -12.33, "Mumbai", 999);
      for (int i = 0; i < 3; i++) {
         current[i].show();
      }
   }
}
```

Output

```
D. Samanta, Rs.123.23, Kolkata, 555
T. Ahmed, Rs.157.02, Bangalore, 727
Sorry! Negative balance N. Sinhababu, Rs.-12.33, Mumbai, 999
```

Access Control Modifier

9. Write a program to demonstrate all combinations of the access control modifiers by using a set of instance variables. In the program, you should define two packages and five classes, as shown below

```
package p6:
    class X                 // same class
    class Y extends X // same package & subclass
    class A                 // same package & non-subclass
package p7:
    class Z extends X// different package & subclass
    class B                 // different package & non-subclass
```

package p6;
 	
public class X {
  int h = 1;
  private int i = 2;
  protected int j = 3;
  public int k = 4;
 	
  public void display(){
  System.out.println("same class");
  System.out.println("I am from class X:");
  System.out.println("default h = "+h);
  System.out.println("private i = "+i);
  System.out.println("protected j = "+j);
  System.out.println("public k = "+k);
  }
  public static void main(String args[]){
  X x = new X();
  x.display();
  }
}
	Output
	same class
	I am from class X:
	default h = 1
	private i = 2
	protected j = 3
	public k = 4

package p6;
 	
class Y extends X {
  public void display(){
  System.out.println("same package & subclass");
  System.out.println("I am from class Y:");
  System.out.println("default h = "+h);
  //System.out.println("private i = "+i); // Error
  System.out.println("protected j = "+j);
  System.out.println("public k = "+k);
  }
  public static void main(String args[]){
  Y y = new Y();
  y.display();
  }
}
	Output
	same package & subclass
	I am from class Y:
	default h = 1
	protected j = 3
	public k = 4

package p6;
 	
class A {
  public static void main(String args[]){
  X x = new X();
  System.out.println("same package non-subclass");
  System.out.println("I am from class A");
	Output
	same package non-subclass
	I am from class A
	default h = 1
	protected j = 3
	public k = 4

System.out.println("default h = "+x.h);
//System.out.println("private i = "+x.i); // Error
System.out.println("protected j = "+x.j);
System.out.println("public k = "+x.k);
}
}	

package p7;
import p6.X;
 	
class Z extends X {
public void display(){
System.out.println("different package & subclass");
System.out.println("I am from class Z:");
//System.out.println("default h = "+h); // Error
//System.out.println("private i = "+i); // Error
System.out.println("protected j = "+j);
System.out.println("public k = "+k);
}
public static void main(String args[]){
Z z = new Z();
z.display();
}
}
	Output
	different package & subclass
	I am from class Z:
	protected j = 3
	public k = 4

package p7;
import p6.X;
 	
class B {
public static void main(String args[]){
X x = new X();
System.out.println("different package & non-subclass");
System.out.println("I am from class B");
//System.out.println("default h = "+x.h); // Error
//System.out.println("private i = "+x.i); // Error
//System.out.println("protected j = "+x.j); // Error
System.out.println("public k = "+x.k);
}
}
	Output
	different package & non-subclass
	I am from class B
	public k = 4

Exceptions

Sqrt Exception

10. Write an application that throws and catches an ArithmaticException when you attempt to take the square root of a negative value. Prompt the user for an input value and try the Math.sqrt() method on it. The application either displays the square root or catchesThe thrown Exception and displays an appropriate message.

```
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.Scanner;

public class SqrtException {
   public static void main(String[] args) {
   NumberFormat formatted = new DecimalFormat("#0.00");
   int input;
   String output;
   Scanner in = new Scanner(System.in);
   System.out.println("Enter a positive integer");
   input = in.nextInt();
   try{
      if(input>0)
         System.out.println("The square root of " + input + " is " + (formatted.format(Math.sqrt(input))));
      else if(input<0)
         throw(new ArithmeticException());

   }
      catch(ArithmeticException e){
         System.out.println("The integer cannot be negative!");
      }
   }
}
```

Output 1

```
Enter a positive integer
7
The square root of 7 is 2.65
```

Output 2

```
Enter a positive integer
-2
The integer cannot be negative!
```

Input Mismatch Exception

11. Write a program that prompts the user to read two integers and displays their sum. The program should prompt the user to read the number again if the input is incorrect. Use the exception concept to demonstrate the program.

```
public class InputException {
  public static void main(String[] args) {
    Scanner input = new Scanner(System.in);

    boolean done = false;
    int number1 = 0;
    int number2 = 0;

    // Enter two integers
    System.out.print("Enter two integers: ");

    while (!done) {
      try {
        number1 = input.nextInt();
        number2 = input.nextInt();
        done = true;
      }
      catch (Exception ex) {
        ex.printStackTrace();
        System.out.print("Incorrect input and re-enter two integers: ");
        input.nextLine(); // Discard input
      }
    }
    System.out.println("Sum is " + (number1 + number2));
  }
}
```

Output 1

```
Enter two integers:
7
11
Sum is 18
```

Output 2

```
Enter two integers: a
Incorrect input and re-enter two integers:
2
3
Sum is 5
```

Sum Above 100 Exception

12. Create a custom Exception class. Consider two integer inputs that the user must supply. The program must display the sum of the integers if and only if the sum is less than 100. If it is not less than 100, the program must throw your custom exception.

```
class SumGreaterThan100Exception  extends Exception {
   String msg;
   SumGreaterThan100Exception(String msg) {
         this.msg=msg;
   }
}
```

```
class DemoClass {
   public int sum(int x,int y) throws SumGreaterThan100Exception {
      int sumofIntegers=x+y;
      if(sumofIntegers<=100) {
       return sumofIntegers;
      }
      else {
         throw new SumGreaterThan100Exception ("Sum is greater than 100");
      }
   }
}
```

```
import java.util.Scanner;
public class CustomExceptionEx {
   public static void main(String args[]) {
      Scanner in= new Scanner(System.in);
      int number1,number2;
      System.out.println("Enter first integer");
      number1= Integer.parseInt(in.nextLine());
      System.out.println("Enter second integer");
      number2= Integer.parseInt(in.nextLine());

      DemoClass demo=new DemoClass();
      try {
         int result=demo.sum(number1, number2);
         System.out.println("Sum of the numbers is "+ result);
      }
      catch(SumGreaterThan100Exception e) {
         System.out.println( "Caught the custom exception : "+e);
         //e.printStackTrace();
      }
   }
}
```

Output 1

```
Enter first integer
30
Enter second integer
15
Sum of the numbers is 45
```

Output 2

```
Enter first integer
55
Enter second integer
72
Caught the custom exception : SumGreaterThan100Exception
```

Files

Process Scores

13. Suppose a text file contains an unspecified number of scores separated by blanks. Write a program that prompts the user to enter the file, reads the scores from it, and displays their total and average. Scores are separated by blanks.

```
import java.util.*;
import java.io.*;

public class FileScore {
   public static void main(String[] args) throws Exception {
      Scanner input = new Scanner(System.in);
      System.out.print("Enter a file of scores: ");
      File file = new File(input.nextLine());

      int count = 0;
      double total = 0;

      try {
         Scanner inputFile = new Scanner(file);
         while (inputFile.hasNext()) {
            total += inputFile.nextInt();
            count++;
         }
      } catch(FileNotFoundException e) {
         System.out.println("File " + file + " does not exist");
         System.exit(1);
      }

      System.out.println("File Name: " + file.getName());
      System.out.println("Total scores: " + total);
      System.out.println("Average scores: " + (total / count));
   }
}
```

Output 1

```
Enter a file of scores: input
File input does not exist
```

Output 2 (Suppose we have a file named Input.txt with the following contents, 2 4 6 8 10)

```
Enter a file of scores: Input.txt
File Name: Input.txt
Total scores: 30.0
Average scores: 6.0
```

Copy File

14. Write a program to copy the content of one file (Sample.txt) to another file (Sample_Copt.txt) using FileStream Class.

```
import java.io.*;

public class CopyFile {
   public static void main(String args[]) {

      try {
         File fileName1 = new File("Sample.txt");
         FileInputStream inFile = new FileInputStream("Sample.txt");
         int fileLength = (int) fileName1.length();
         byte Bytes[] = new byte[fileLength];

         System.out.println("File size is: " + inFile.read(Bytes));

         String file1 = new String(Bytes);
         System.out.println("File content is: " + file1);

         File fileName2 = new File("Sample_Copy.txt");
         fileName2.createNewFile();
         FileOutputStream outFile = new FileOutputStream(fileName2);

         for (int i = 0; i < fileLength; i++) {
           outFile.write(Bytes[i]);
         }

         System.out.println("File copied.");
         inFile.close();
         outFile.close();
      } catch(IOException e) {
         System.out.println("File Not Copied");
         System.out.println(e);
      }
   }
}
```

Output 1

File Not Copied
java.io.FileNotFoundException: **hello.txt** (No such file or directory)

Output 2 *(Suppose we have a file named **Sample.txt** with the following contents, "This is a sample file")*

File size is: 23
File content is: This is a sample file.
File copied.

Longest Word

15. Suppose that a text file contains a few sentences. Write a program to find the longest word in a text file and display the results.

```
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;

public class LongestWordDemo {
   public static void main(String [ ] args) throws FileNotFoundException {
      LongestWordDemo lword = new LongestWordDemo();
      lword.findLongestWords();
   }

    public void findLongestWords() throws FileNotFoundException {
      String longest_word = "";
      String current;
      try {
          File input = new File("Sample");
          Scanner sc = new Scanner(input);

         while (sc.hasNext()) {
             current = sc.next();
             if (current.length() > longest_word.length()) {
             longest_word = current;
             }
          }
       System.out.println("Longest Word in the file:"+longest_word);
       sc.close();
       } catch (FileNotFoundException e) {
         System.out.println("An error occurred, File not found");
         //e.printStackTrace();
       }
   }
}
```

Output 1

An error occurred, File not found

*Output 2 (Suppose we have a file named **Sample.txt** with the following contents, "This is a sample file to test longest word in a file using programming")*