ASSIGNMENT

1Q)

```
class HelloWorld {
   public static void main(String[] args) {
      String input="Hello JavaTpoint";

      String rev="";
      for(int i=input.length()-1;i>=0;i--){

         rev=rev+input.charAt(i);

      }
      System.out.println(rev);

   }
}
```

2Q)

```
public class Prime {
   public static void main(String[] args) {
      for (int i = 1; i <=20 ; i++) {
         if(Checkprime(i)){
            System.out.println(i+" is Prime");
         }
      }

   }

   static boolean Checkprime(int num){
      int c=2;
      if(num<=1){
         return false;
      }
      while(c*c<=num){
         if(num%c==0){
            return false;
            }else {
                c = c + 1;
            }
        }
        return true;
    }
}
```

3Q)

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

        for (int i=2;i<=5;i++){
            System.out.println("Printing table "+i);
            for(int j=1;j<=10;j++){
                System.out.println(i+"x"+j+"="+i*j);
            }
            System.out.println("----------------");
        }

    }
}
```

```
4Q)public class Main {
    public static void main(String[] args) {
        for (int i = 1; i <=20; i++) {
            if(check(i)){
                System.out.println(i+" is even");
            }
            else{
                System.out.println(i+" is odd");
            }
        }

    }

    static boolean check(int n) {
        if ((n & 1) == 0) {
            return true;
        } else {
            return false;
        }
    }
}

5Q)import java.util.Scanner;

public class Student {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        System.out.println("enter Student marks");
        int marks= input.nextInt();
        if(marks>90&&marks<=100){
            System.out.println("Distinction");
        } else if (marks>80&&marks<=90) {
            System.out.println("First Division");
        } else if (marks>60&&marks<=80) {
            System.out.println("second division");

        } else if (marks>40&&marks<=60) {
            System.out.println("third division");

        }else {
            System.out.println("Fail");
        }
    }
}
6Q)
7Q)public class Swap {
    public static void main(String[] args) {
        int num1=5;
        int num2=10;
        swaps(num1,num2);

    }
    static void swaps(int a,int b){
        int temp=b;
        b=a;
        a=temp;
        System.out.println(a+" "+b);
    }
}
```

```
7Q)
public class Factorial {
    public static void main(String[] args) {
        int num=8;
        int result=1;
        for (int i = 8; i>=1 ; i--) {

            result=result*i;
        }
        System.out.println("factorial of 8 is "+result);

    }

}
```

```
8Q)import java.util.Scanner;

public class LeapYear {
    public static void main(String[] args) {
        System.out.print("enter year: ");
        Scanner input=new Scanner (System.in);
        int year=input.nextInt();
        if(year%4==0) {
           if(year%100==0){
               if(year%400==0){
                   System.out.println(year+" is a leap year");
               }else {
                   System.out.println(year+" is Not a leap year");
               }
           }else {
               System.out.println(year+" is a leap year");
           }
        }else {
            System.out.println(year+" is Not a leap year");
        }
    }
}
```

```
9Q)  import java.util.Scanner;

public class pattern2 {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);

        int num= input.nextInt();


        for (int row = 1; row <=2*num-1; row++) {
            int numOfSpaces;

              if(row<=5) {
                   numOfSpaces=num-row;
                  for (int i = 1; i <=numOfSpaces ; i++) {
                      System.out.print(" ");
                  }
                  for (int col = 0; col < row; col++) {
                      System.out.print("* ");
                  }
                  System.out.println();
              }
            if (row > 5) {
                numOfSpaces=row-num;
                for (int j = 0; j <numOfSpaces; j++) {
                    System.out.print(" ");
                }
                for (int col = 0; col <(2*num)-row ; col++) {
                    System.out.print("* ");
                }
                System.out.println();
            }
        }

    }
}
```

```
10Q)  import java.util.Scanner;

public class Pattern3 {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);

        int num= 5;

        for (int row = 1; row <=num ; row++) {
            int spacesInRow;
            for (int col = 1; col <=2*num-1; col++) {
//                  spacesInRow=col
                        if(col>row && col<(2*(num)-row)){
                            System.out.print(" ");
                        }
                        else {
                            System.out.print("*");
                        }
            }
            System.out.println();
        }

    }
}
```

```
11Q)  import java.util.Scanner;

public class Pattern4 {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);

        int num= 5;

        for (int row = num; row>=1 ; row--) {
            int spacesInRow;
            for (int col = 1; col <=2*num-1; col++) {
//                  spacesInRow=col
                if(col>row && col<(2*(num)-row)){
                    System.out.print(" ");
                }
                else {
                    System.out.print("*");
                }
            }
            System.out.println();
        }
    }
}
```

```
12Q)  public class Maximum {
    public static void main(String[] args) {

        int[] arr={22,100,44,11,22,100,77,44,11};
        max(arr);
    }

    static void max(int[]arr){

        int max1=arr[0];
        for (int i = 0; i < arr.length ; i++) {
            if(arr[i]>max1){
                max1=arr[i];
            }
        }


        int max2=arr[0];
        for (int i = 0; i < arr.length ; i++) {
            if(arr[i]>max2&&arr[i]<max1){
                max2=arr[i];
            }
        }

        int max3=arr[0];
        for (int i = 0; i < arr.length ; i++) {
            if(arr[i]>max3&&arr[i]<max2){
                max3=arr[i];
            }
        }

        System.out.println("3rd max element is "+max3);

    }
}


13Q)  public class Sum {



}
class Class1{
    int add(int a,int b){
        int result=a+b;
        return result;
    }
}

class Class2 extends Class1{

}

class Class3{
    public static void main(String[] args) {
        Class2 obj=new Class2();
        int ans=obj.add(4,4);
        System.out.println(ans);
    }
}
```

14Q)

```
public class Divisor implements div{

    @Override
    public int divisors(int a) {

        if (a > 1000) {
            throw new IllegalArgumentException("Input value must be at most 1000");
        }

        int Sum=0;
        for (int i = 1; i <=a; i++) {
            if(a%i==0){
                Sum=Sum+i;
            }
        }
        return Sum;
    }

    public static void main(String[] args) {
        Divisor divisor = new Divisor();
        int number = 1003;
        int sumOfDivisors = divisor.divisors(number);
        System.out.println("Sum of divisors of " + number + ": " + sumOfDivisors);
    }
}
```

```
public interface div {
   int divisors(int a);


}
```

```
15Q) mport java.util.*;

public class Sorting {
   public static void sort(ArrayList<String> nums) {
      int len = nums.size();
      for (int i = 0; i < len - 1; i++) {
         int min_ind = i;
         String minStr = nums.get(i);
         for (int j = i + 1; j < len; j++) {
            if (Float.parseFloat(nums.get(j)) > Float.parseFloat(minStr)) {
               minStr = nums.get(j);
               min_ind = j;
            }
         }
         if (min_ind != i) {
            String temp = nums.get(min_ind);
            nums.set(min_ind, nums.get(i));
            nums.set(i, temp);
         }
      }
   }

   public static void printArray(ArrayList<String> nums) {

      nums.forEach((num) -> System.out.print(num + " "));

      System.out.println();
   }

   public static void main(String[] args) {
      ArrayList<String> nums = new ArrayList<String>();
      System.out.print("Enter the number of elements:");
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt(); // consuming the <enter> from input above
      System.out.print("Start entering inputs\n");
      sc.nextLine();
```

```
        for (int i = 0; i < n; i++) {
            String str = sc.nextLine();
            nums.add(str);
        }
        sc.close();
        // String nums[] = { "100", "-100", "00000.2222", "22", ".12", "0.12" };
        printArray(nums);
        sorting(nums);
        printArray(nums);
    }
}
```

16Q)

```
import java.util.InputMismatchException;
import java.util.Scanner;

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

        Scanner input=new Scanner(System.in);
        try {

            int num1= input.nextInt();
            int num2= input.nextInt();

            int result=num1/num2;
            System.out.println("result is "+result);

        }
        catch (ArithmeticException e){
            System.out.println(e.getClass().getName());
        }
        catch (InputMismatchException er){
            System.out.println(er.getClass().getName());
//              By accessing the exception class name and printing it, you can display the
desired output without any additional message.
        }


    }
}
```

```
17Q) import java.util.Random;

class MyException extends Exception {

    public MyException(String s) {
        super(s);

    }
}

class SBI {
    double balance;
    String name;

    public void openAccount() {

        Random random = new Random();
        name = "" + Math.round(random.nextFloat() * Math.pow(10, 12));
        balance = 0;
        System.out.println("Account created");
        System.out.println("Name: " + name);
        System.out.println("Current Balance: " + balance);
    }

    public void deposit(int amount) {
        balance += amount;
        System.out.println("Added " + amount + " to your account");
        System.out.println("Current Balance: " + balance);

    }

    public void withdraw(int amount) {

        if (balance - amount < 0) {
            System.out.println("Insufficient Balance to withdraw");
        }

        else {

            balance -= amount;
            System.out.println("Withdrawn " + amount + " to your account");
            System.out.println("Current Balance: " + balance);
        }
    }
    public static void main(String[] args) {
        Jandhan j = new Jandhan();
        j.openAccount();
        Premium p = new Premium();
        p.openAccount();

        j.deposit(1000);
        p.deposit(100);
        j.withdraw(1000);
        p.withdraw(1000);
    }
}

class Jandhan extends SBI {

}

class Premium extends SBI {
    public void openAccount() {

        Random random = new Random();
        this.name = "" + Math.round(random.nextFloat() * Math.pow(10, 12));
        balance = 5000;
        System.out.println("Account created - Premium");
        System.out.println("Name: " + name);
        System.out.println("Current Balance: " + balance);
    }

    public void deposit(int amount) {
        balance += amount;
        System.out.println("Added " + amount + " to your account");
        System.out.println("Current Balance: " + balance);

    }

    public void withdraw(int amount) {

        if (balance - amount < 5000) {
            System.out.println("Insufficient Balance to withdraw");
        }
        else {

            balance -= amount;
            System.out.println("Withdrawn " + amount + " to your account");
            System.out.println("Current Balance: " + balance);
        }

    }
}
```

18Q)

```
import java.util.ArrayList;
import java.util.List;

public class BankingSystem {
    public static void main(String[] args) {
            Bank bank = new Bank();

            SavingsAccount savingsAccount = new SavingsAccount();
            CurrentAccount currentAccount = new CurrentAccount();

            bank.addAccount(savingsAccount);
            bank.addAccount(currentAccount);


            savingsAccount.deposit(1000);
            savingsAccount.withdraw(600);
            savingsAccount.calculateInterest();

            currentAccount.deposit(2000);
            currentAccount.withdraw(100);


            System.out.println("Savings Account Balance: " +
savingsAccount.viewBalance());
            System.out.println("Current Account Balance: " +
currentAccount.viewBalance());

    }

}
class Bank{

  List<Accounts> accounts = new ArrayList<>();
```

```
    public void addAccount(Accounts account) {
        accounts.add(account);
    }
}

class SavingsAccount  implements Accounts{

    private int balance;

    public SavingsAccount() {
        balance = 0;
    }

    @Override
    public void deposit(int amount) {
        balance =balance+ amount;
    }

    @Override
    public void withdraw(int amount) {
        if (amount <= balance) {
            balance =balance- amount;
        } else {
            System.out.println("Insufficient balance");
        }
    }



    @Override
    public void calculateInterest() {
        System.out.println("Interest calculated for savings account");
    }


    @Override
    public int viewBalance() {
        return balance;
    }
}
class CurrentAccount implements Accounts{
    private int balance;

    public CurrentAccount() {
        balance = 0;
    }

    @Override
    public void deposit(int amount) {
        balance =balance+amount;
    }

    @Override
    public void withdraw(int amount) {
        if (amount <= balance) {
            balance =balance-amount;
        } else {
            System.out.println("Insufficient balance");
        }
    }



    @Override
    public void calculateInterest() {
        System.out.println("Interest calculated for savings account");
    }

    @Override
    public int viewBalance() {
        return balance;
    }

}

public interface Accounts {
    void deposit(int amount) ;
    void withdraw(int amount);

    void calculateInterest();

    int viewBalance();
}
```