

## Java Assignments 1

### 1. Write a Java Program to print Hello World using command line and Eclipse

```
public class Q1 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println(""+args[0]);  
    }  
}
```



## 2. Write a program to find SUM of a given digit.

```
public class Q2 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        int m, n, sum = 0;  
        Scanner s = new Scanner(System.in);  
        System.out.println("Enter the number:");  
        m = s.nextInt();  
        while(m > 0)  
        {  
            n = m % 10;  
            sum = sum + n;  
            m = m / 10;  
        }  
        System.out.println("Sum of Digits:"+sum);  
  
    }  
  
}
```

The screenshot shows an IDE with two tabs: Q1.java and Q2.java. The Q2.java tab is active, displaying the following Java code:

```
package cdac.java.basic;
import java.util.Scanner;
public class Q2 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int m, n, sum = 0;
        Scanner s = new Scanner(System.in);
        System.out.println("Enter the number:");
        m = s.nextInt();
        while (m > 0)
        {
            n = m % 10;
            sum = sum + n;
            m = m / 10;
        }
        System.out.println("Sum of Digits:"+sum);
    }
}
```

Below the code editor, there is a console window with the following output:

```
<terminated> Q2 [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (28-Oct-2020 3:06:14 pm)
Enter the number:
123
Sum of Digits:6
```

### 3. Write a java program to reverse the given number.

```
public class Q3 {

    public static void main(String[] args) {

        int num, reversed = 0;

        Scanner s = new Scanner(System.in);

        System.out.println("enter a no. to swap");

        num = s.nextInt();

        while(num != 0) {

            int digit = num % 10;

            reversed = reversed * 10 + digit;

            num /= 10;

        }

    }

}
```

```

        System.out.println("Reversed Number: " + reversed);
    }
}

```

```

package cdac.java.basic;
import java.util.Scanner;
public class Q3 {
    public static void main(String[] args) {
        int num, reversed = 0;
        Scanner s = new Scanner(System.in);
        System.out.println("enter a no. to swap");
        num = s.nextInt();
        while(num != 0) {
            int digit = num % 10;
            reversed = reversed * 10 + digit;
            num /= 10;
        }
        System.out.println("Reversed Number: " + reversed);
    }
}

```

Problems Javadoc Declaration Console

<terminated> Q3 [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (28-Oct-2020 3:19:21)

enter a no. to swap

83026

Reversed Number: 62038

#### 4. Write a program to swap the given two numbers.

```

public class Q4 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int x, y, t;
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the value of X and Y");
        x = sc.nextInt();
        y = sc.nextInt();
        System.out.println("before swapping numbers: "+x+" "+y);
        t = x;
        x = y;
        y = t;
        System.out.println("After swapping: "+x+" "+y);
        System.out.println( );
    }
}

```

The screenshot shows an IDE with a Java file named Q4.java. The code implements a program to swap two numbers using a temporary variable. The console output shows the program running successfully, taking inputs 54 and 65, and printing the swapped values 65 and 54.

```
Q1.java Q2.java Q3.java Q4.java x
package cdac.java.basic;
import java.util.Scanner;
public class Q4 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int x, y, t;
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the value of X and Y");
        x = sc.nextInt();
        y = sc.nextInt();
        System.out.println("before swapping numbers: "+x + "  "+ y);
        t = x;
        x = y;
        y = t;
        System.out.println("After swapping: "+x + "  " + y);
        System.out.println( );
    }
}
```

Problems @ Javadoc Declaration Console x

<terminated> Q4 [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (28-Oct-2020 3:31:39 pm)

Enter the value of X and Y

54

65

before swapping numbers: 54 65

After swapping: 65 54

5. Write a java program to print prime numbers upto given number.

```
public class Q5 {

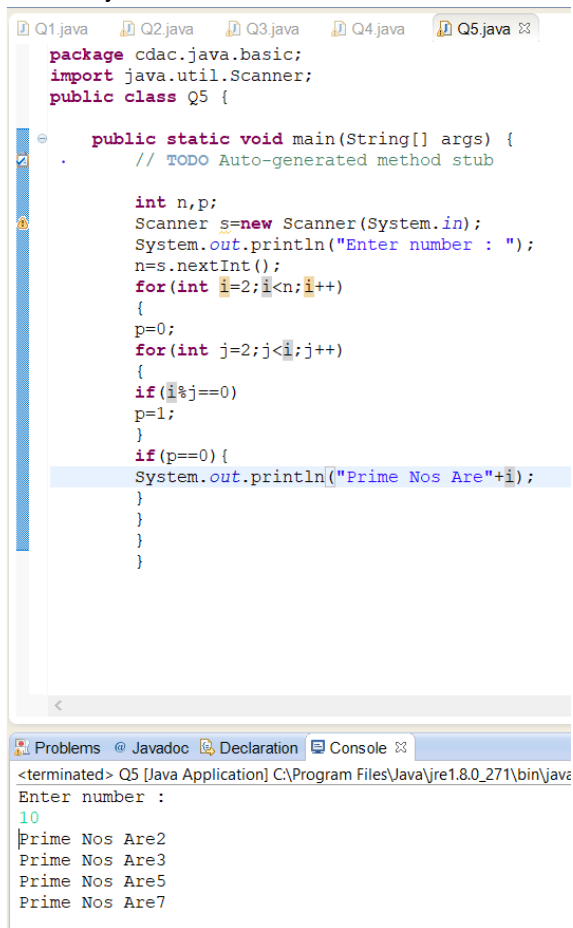
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int n,p;
        Scanner s=new Scanner(System.in);
        System.out.println("Enter number : ");
        n=s.nextInt();
        for(int i=2;i<n;i++)
        {
            p=0;
            for(int j=2;j<i;j++)
            {
                if(i%j==0)
```

```

    p=1;
}
if(p==0){
    System.out.println("Prime Nos Are"+i);
}
}
}
}
}

```



```

package cdac.java.basic;
import java.util.Scanner;
public class Q5 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        int n,p;
        Scanner s=new Scanner(System.in);
        System.out.println("Enter number : ");
        n=s.nextInt();
        for(int i=2;i<n;i++)
        {
            p=0;
            for(int j=2;j<i;j++)
            {
                if(i%j==0)
                {
                    p=1;
                }
            }
            if(p==0) {
                System.out.println("Prime Nos Are"+i);
            }
        }
    }
}

```

Problems @ Javadoc Declaration Console

```

<terminated> Q5 [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\java
Enter number :
10
Prime Nos Are2
Prime Nos Are3
Prime Nos Are5
Prime Nos Are7

```

**6. write a java program using loops to print the pyramid as follows:**

```

*
**
***
****
*****
*****
****
***
**
*

```

```

public class Q6 {

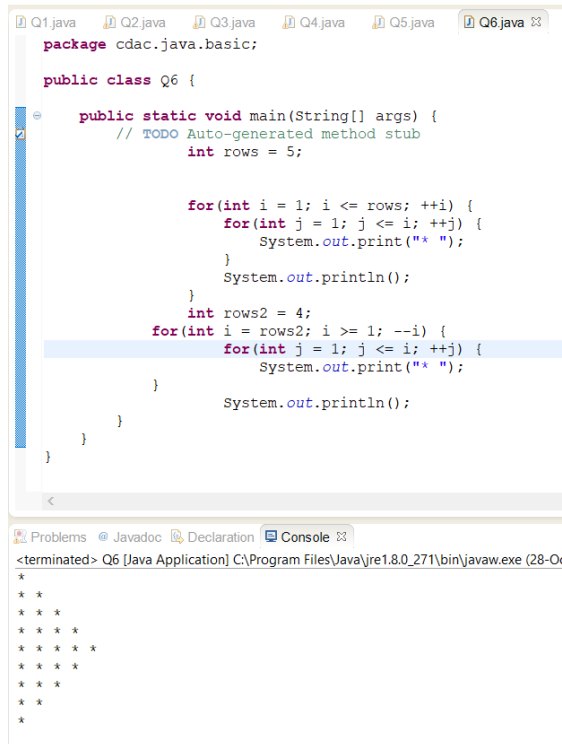
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int rows = 5;
    }
}

```

```

        for(int i = 1; i <= rows; ++i) {
            for(int j = 1; j <= i; ++j) {
                System.out.print("* ");
            }
            System.out.println();
        }
        int rows2 = 4;
        for(int i = rows2; i >= 1; --i) {
            for(int j = 1; j <= i; ++j) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}

```



## 7. Write a java program to print Table of given Number

```

public class Q7 {

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        Scanner s = new Scanner(System.in);

        System.out.print("Enter number:");

        int n=s.nextInt();

        for(int i=1; i <= 10; i++)

        {

            System.out.println(n+" * "+i+" = "+n*i);

```

```

    }
}
}

```

```

package cdac.java.basic;
import java.util.Scanner;
public class Q7 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner s = new Scanner(System.in);
        System.out.print("Enter number:");
        int n=s.nextInt();
        for(int i=1; i <= 10; i++)
        {
            System.out.println(n+" * "+i+" = "+n*i);
        }
    }
}

```

```

<terminated> Q7 [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (28-Oct-20
Enter number:2
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20

```

**8. Write a java program which takes input from 1 to 9 and prints "ONE", "TWO",... , "NINE", "OTHER" respectively. Use switch statement.**

```

package cdac.java.basic;
import java.util.Scanner;
public class Q8 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner in = new Scanner(System.in);
        System.out.print("Enter number: ");
        int n;
        n = in.nextInt();
        switch (n) {
            case 1:

```



```
        System.out.println("ONE");
        break;
    case 2:
        System.out.println("TWO");
        break;
    case 3:
        System.out.println("THREE");
        break;
    case 4:
        System.out.println("FOUR");
        break;
    case 5:
        System.out.println("FIVE");
        break;
    case 6:
        System.out.println("SIX");
        break;
    case 7:
        System.out.println("SEVEN");
        break;
    case 8:
        System.out.println("EIGHTt");
        break;
    case 9:
        System.out.println("NINE");
        break;
    default:
        System.out.println("Invalid Number");
        break;
    }
}
```

```
package cdac.java.basic;
import java.util.Scanner;
public class Q8 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner in = new Scanner(System.in);
        System.out.print("Enter number: ");
        int n;
        n = in.nextInt();
        switch (n) {
            case 1:
                System.out.println("ONE");
                break;
            case 2:
                System.out.println("TWO");
                break;
            case 3:
                System.out.println("THREE");
                break;
            case 4:
                System.out.println("FOUR");
                break;
            case 5:
                System.out.println("FIVE");
                break;
            case 6:
                System.out.println("SIX");
                break;
            case 7:
```

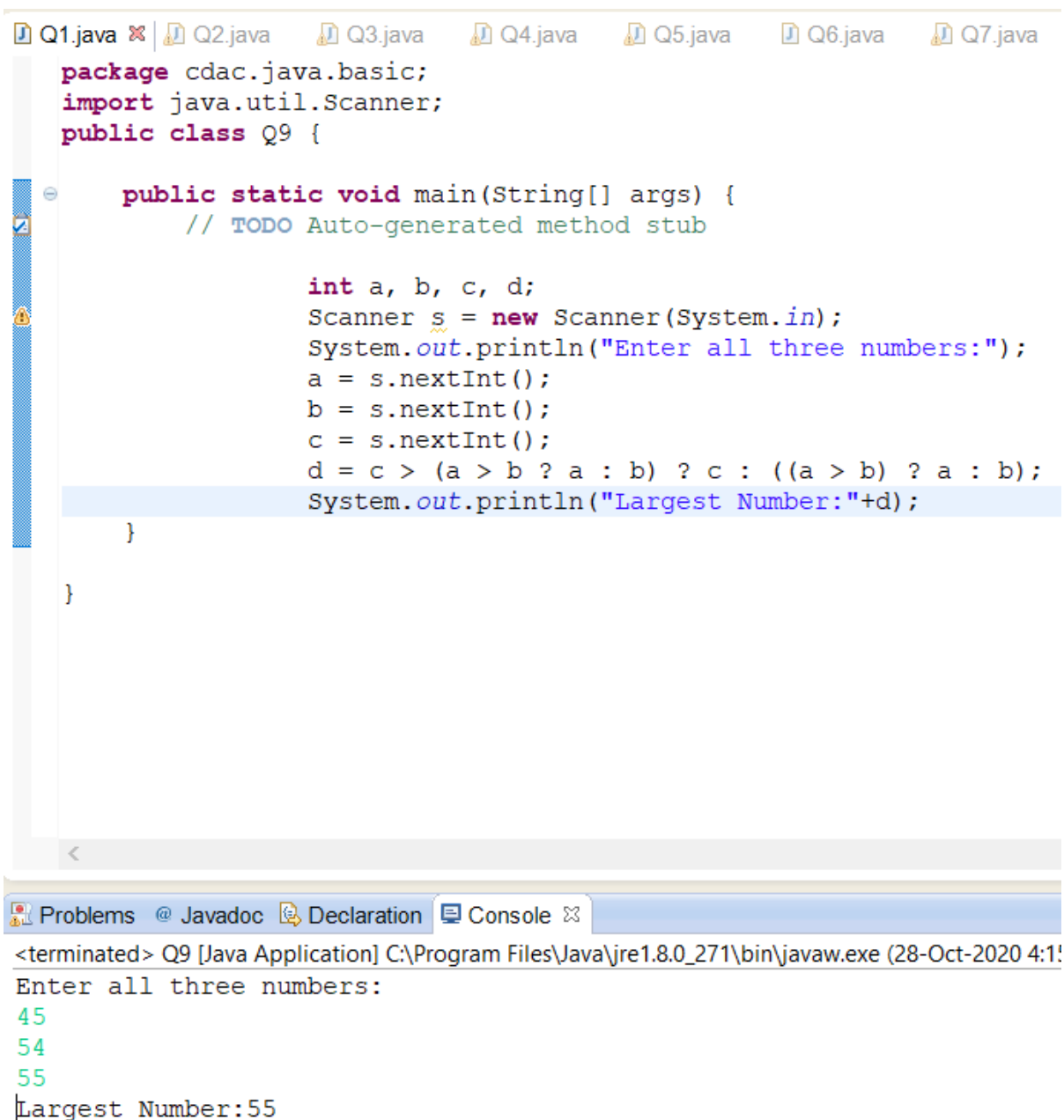
<terminated> Q8 [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (28-Oct-202

Enter number: 5

FIVE

## 9. Write Java Program to Find Largest of Three Numbers

```
public class Q9 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        int a, b, c, d;  
        Scanner s = new Scanner(System.in);  
        System.out.println("Enter all three numbers:");  
        a = s.nextInt();  
        b = s.nextInt();  
        c = s.nextInt();  
        d = c > (a > b ? a : b) ? c : ((a > b) ? a : b);  
        System.out.println("Largest Number:"+d);  
    }  
}
```



The screenshot shows an IDE with a tab bar at the top containing files Q1.java through Q7.java. The active file is Q9.java, which contains the following code:

```
package cdac.java.basic;  
import java.util.Scanner;  
public class Q9 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        int a, b, c, d;  
        Scanner s = new Scanner(System.in);  
        System.out.println("Enter all three numbers:");  
        a = s.nextInt();  
        b = s.nextInt();  
        c = s.nextInt();  
        d = c > (a > b ? a : b) ? c : ((a > b) ? a : b);  
        System.out.println("Largest Number:"+d);  
    }  
}
```

At the bottom, the 'Console' tab is active, displaying the program's execution:

```
<terminated> Q9 [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (28-Oct-2020 4:1!  
Enter all three numbers:  
45  
54  
55  
Largest Number:55
```

10. write a java program to input number of units consumed by a customer using scanner class and generate electricity bill by the following criteria.

number of units	charges
< = 100	Rs.1.20
for the next 200 units	Rs. 2.00
for the next 300 units	Rs. 3.00
for more	Rs. 5.00

ex: input = 320 units output =  $100*1.20 + 200*2.00 + 20*3.00 = \text{Rs. } 580$

```
public class Q10 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        long units;  
        Scanner sc=new Scanner(System.in);  
        System.out.println("enter number of units");  
        units=sc.nextLong();  
        double billpay=0;  
        if(units<100)  
            billpay=units*1.20;  
        else if(units<300)  
            billpay=100*1.20+(units-100)*2;  
        else if(units>300)  
            billpay=100*1.20+200 *2+(units-300)*3;  
        else if(units>500)  
            billpay=100*1.20+200 *2+(units-300)*3+(units-500)*5.00;  
        System.out.println("Bill to pay : " + billpay);  
    }  
}
```

Q1.java Q2.java Q3.java Q4.java Q5.java Q6.java Q7.java Q9.java

```
package cdac.java.basic;
import java.util.Scanner;
public class Q10 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        long units;
        Scanner sc=new Scanner(System.in);
        System.out.println("enter number of units");
        units=sc.nextLong();
        double billpay=0;
        if(units<100)
            billpay=units*1.20;
        else if(units<300)
            billpay=100*1.20+(units-100)*2;
        else if(units>300)
            billpay=100*1.20+200 *2+(units-300)*3;
        else if(units>300)
            billpay=100*1.20+200 *2+(units-300)*3+(units-500)*5.00;
        System.out.println("Bill to pay : " + billpay);
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Q10 [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (28-Oct-2020 8:47:33 pm)

enter number of units

600

Bill to pay : 1420.0

## Java Assignments – 2

### Classes and object

11. Create a class called Emp with data members empno, empname, designation, dept and salary and methods as readEmpData() (to read values to data members) and displayEmpData() (to display data members values to the screen) create an employee instance and display its information.

```
package assignment2;
import java.util.Scanner;
class Employee
{
    int empno;
    String empname,designation,dept;
    long Salary;

    void GetData()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\n\tEnter Employee Id : ");
        empno = Integer.parseInt(sc.nextLine());
        System.out.print("\n\tEnter Employee Name : ");
        empname = sc.nextLine();
        System.out.print("\n\tEnter Employee Designation : ");
        designation = sc.nextLine();
        System.out.print("\n\tEnter Employee Dept : ");
        dept = sc.nextLine();
        System.out.print("\n\tEnter Employee Salary : ");
        Salary = Long.parseLong(sc.nextLine());
    }
    void PutData()
    {
        System.out.print("\n\tEmployee empno : "+empno);
        System.out.print("\n\tEmployee empname : "+empname);
        System.out.print("\n\tEmployee Designation : "+designation);
        System.out.print("\n\tEmployee dept : "+dept);
        System.out.print("\n\tEmployee Salary : "+Salary);
    }
    public static void main(String args[])
    {
        Employee E = new Employee();
        E.GetData();
        E.PutData();
    }
}
```

Q3.java Q4.java Q5.java Q6.java Q7.java Q9.java Q8.java

```
import java.util.Scanner;
class Employee
{
    int empno;
    String empname, designation, dept;
    long Salary;

    void GetData()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\n\tEnter Employee Id : ");
        empno = Integer.parseInt(sc.nextLine());
        System.out.print("\n\tEnter Employee Name : ");
        empname = sc.nextLine();
        System.out.print("\n\tEnter Employee Designation : ");
        designation = sc.nextLine();
        System.out.print("\n\tEnter Employee Dept : ");
        dept = sc.nextLine();
        System.out.print("\n\tEnter Employee Salary : ");
```

Problems @ Javadoc Declaration Console

<terminated> Employee [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (29-Oct-2

```
Enter Employee Id : 83026

Enter Employee Name : Pardeep

Enter Employee Designation : CR

Enter Employee Dept : DBDA

Enter Employee Salary : 4500

Employee empno : 83026
Employee empname : Pardeep
Employee Designation : CR
Employee dept : DBDA
Employee Salary : 4500
```

12. Create a class Electricity bill with data members as customer number, customer name, units consumed and methods as follows:

1. readData() - to read the values of data members.
2. showData - to display the customer details
3. computeBill() - to calculate and return electricity charges to be paid.calculate the bill as specified below

number of units	charges
< = 100	Rs.1.20
for the next 200 units	Rs. 2.00
for the next 300 units	Rs. 3.00
for more	Rs. 5.00

ex: input = 320 units output =  $100 \times 1.20 + 200 \times 2.00 + 20 \times 3.00 = \text{Rs. } 580$

Read customer object values, calculate electricity bill and display the values.

```
package assignment2;

import java.util.Scanner;
class Electricitybill {

    int customerNo;
    String name;
    double units;

    void GetData()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\n\tEnter Customer No. : ");
        customerNo = Integer.parseInt(sc.nextLine());
        System.out.print("\n\tEnter Customer Name : ");
        name = sc.nextLine();
        System.out.print("\n\tEnter Units : ");
        units = Long.parseLong(sc.nextLine());
    }

    void PutData()
    {
        System.out.print("\n\tCustomer No. : "+customerNo);
        System.out.print("\n\tCustomer Name : "+name);
        System.out.print("\n\tunits : "+units);
    }

    double billcalculate ( )
    {
```



```

    double bill;
    if(units<100)
        bill = units * 1.20 ;
    else if(units <= 200)
        bill = 100 * 1.20 + (units - 100) * 2 ;
    else if (units <=300)
        bill = 100 * 1.20 + 200 * 2 + (units - 300) * 3 ;
    else
        bill = 100 * 1.20 + 200 * 2 + (units - 300) * 3 + (units -500) * 5;
    return bill;
}

```

```

        public static void main(String[] args)

        {
            Electricitybill E = new Electricitybill();
            E.GetData();
            E.PutData();
            double billpay = E.billcalculate();
            System.out.println("Bill to pay : " + billpay);
        }
    }
}

```

```
Electricitybill.java
{
    System.out.print("\n\tCustomer No. : "+customerNo);
    System.out.print("\n\tCustomer Name : "+name);
    System.out.print("\n\tunits : "+units);
}
double billcalculate ( )
{
    double bill;
    if(units<100)
        bill = units * 1.20 ;
    else if(units <= 200)
        bill = 100 * 1.20 + (units - 100) * 2 ;
    else if (units <=300)
        bill = 100 * 1.20 + 200 * 2 + (units - 300) * 3 ;
    else
        bill = 100 * 1.20 + 200 * 2 + (units - 300) * 3 + (units -500) * 5;
    return bill;
}

public static void main(String[] args)
{
    Electricitybill E = new Electricitybill();
    E.GetData();
}

<terminated> Electricitybill [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (29-Oct-2020 4:53:34 pm)
```

```
Enter Customer No. : 26
Enter Customer Name : pardeep
Enter Units : 600
Customer No. : 26
Customer Name : pardeep
units : 600.0Bill to pay : 1920.0
```

3) Write a Java program to create Student class with member variable as id, name, mark and result. Use method to initialize the value of name, id and marks. Write a member function to find the result and display the student details with result.

```
package assignment2;

import java.util.Scanner;

class Student {
    int id;
    String name;
    double english,hindi,science,maths,computers;

    void GetData(){

        Scanner sc = new Scanner(System.in);
```

```

        System.out.print("\n\tEnter Student Id : ");
        id = Integer.parseInt(sc.nextLine());
        System.out.print("\n\tEnter Student Name : ");
        name = sc.nextLine();
        System.out.print("\n\tEnter Makrs English : ");
        english = Double.parseDouble(sc.nextLine());
        System.out.print("\n\tEnter Makrs Hindi : ");
        hindi = Double.parseDouble(sc.nextLine());
        System.out.print("\n\tEnter Makrs Science: ");
        science = Double.parseDouble(sc.nextLine());
        System.out.print("\n\tEnter Maths: ");
        maths = Double.parseDouble(sc.nextLine());
        System.out.print("\n\tEnter Makrs Computers : ");
        computers = Double.parseDouble(sc.nextLine());

    }

    void PutData()
    {
        System.out.print("\n\tEnter Student Id : "+id);
        System.out.print("\n\tEnter Student Name : "+name);
        System.out.print("\n\tEnter Makrs English : "+english);
        System.out.print("\n\tEnter Makrs Hindi : "+hindi);
        System.out.print("\n\tEnter Makrs Science: "+science);
        System.out.print("\n\tEnter Makrs Maths: "+maths);
        System.out.print("\n\tEnter Makrs Computers: "+computers);

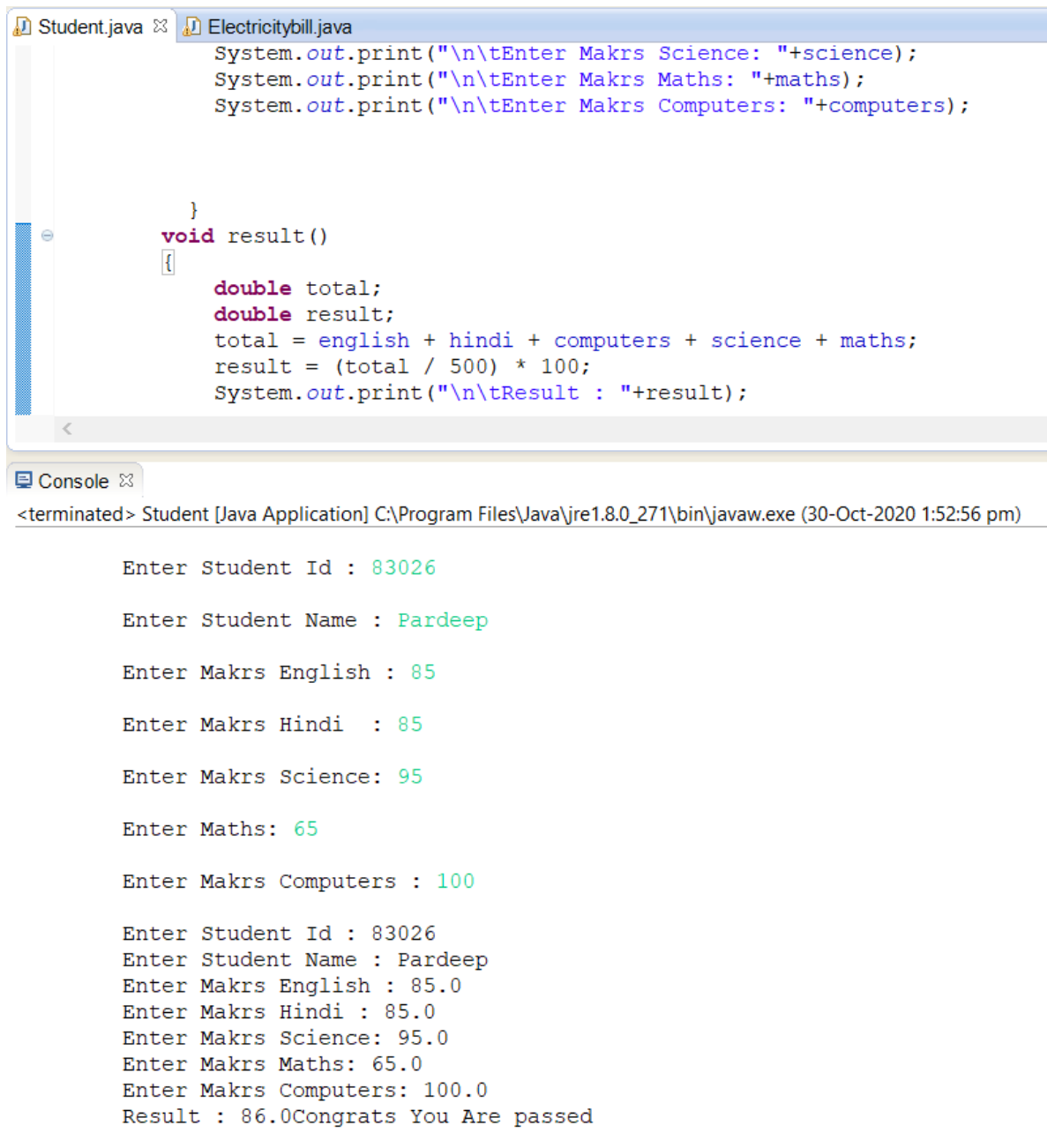
    }

    void result()
    {
        double total;
        double result;
        total = english + hindi + computers + science + maths;
        result = (total / 500) * 100;
        System.out.print("\n\tResult : "+result);

        if
            (result<=40.0);
            System.out.println("Congrats You Are passed");
    }

    public static void main(String[] args) {
        Student s = new Student();
        s.GetData();
        s.PutData();
        s.result();
    }
}

```



The screenshot shows an IDE with two tabs: 'Student.java' and 'Electricitybill.java'. The 'Student.java' tab is active, displaying the following code:

```
System.out.print("\n\tEnter Makrs Science: "+science);
System.out.print("\n\tEnter Makrs Maths: "+maths);
System.out.print("\n\tEnter Makrs Computers: "+computers);

}
void result()
{
    double total;
    double result;
    total = english + hindi + computers + science + maths;
    result = (total / 500) * 100;
    System.out.print("\n\tResult : "+result);
}
```

Below the code editor is a 'Console' window showing the execution output:

```
<terminated> Student [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (30-Oct-2020 1:52:56 pm)

Enter Student Id : 83026

Enter Student Name : Pardeep

Enter Makrs English : 85

Enter Makrs Hindi : 85

Enter Makrs Science: 95

Enter Maths: 65

Enter Makrs Computers : 100

Enter Student Id : 83026
Enter Student Name : Pardeep
Enter Makrs English : 85.0
Enter Makrs Hindi : 85.0
Enter Makrs Science: 95.0
Enter Makrs Maths: 65.0
Enter Makrs Computers: 100.0
Result : 86.0Congrats You Are passed
```

**4) Write a Java program that creates a account classs with instance variable accno,accname,amount and instance method withdraw, deposit, and interest. Create object of account class test all methods.**

```
package assignment2;

public class account {
    int accno=1;
    String accname="pardeep";
    double balance;
```

```

double interest=0.10;
void deposit(int amount)
{
    balance=balance+amount;
}

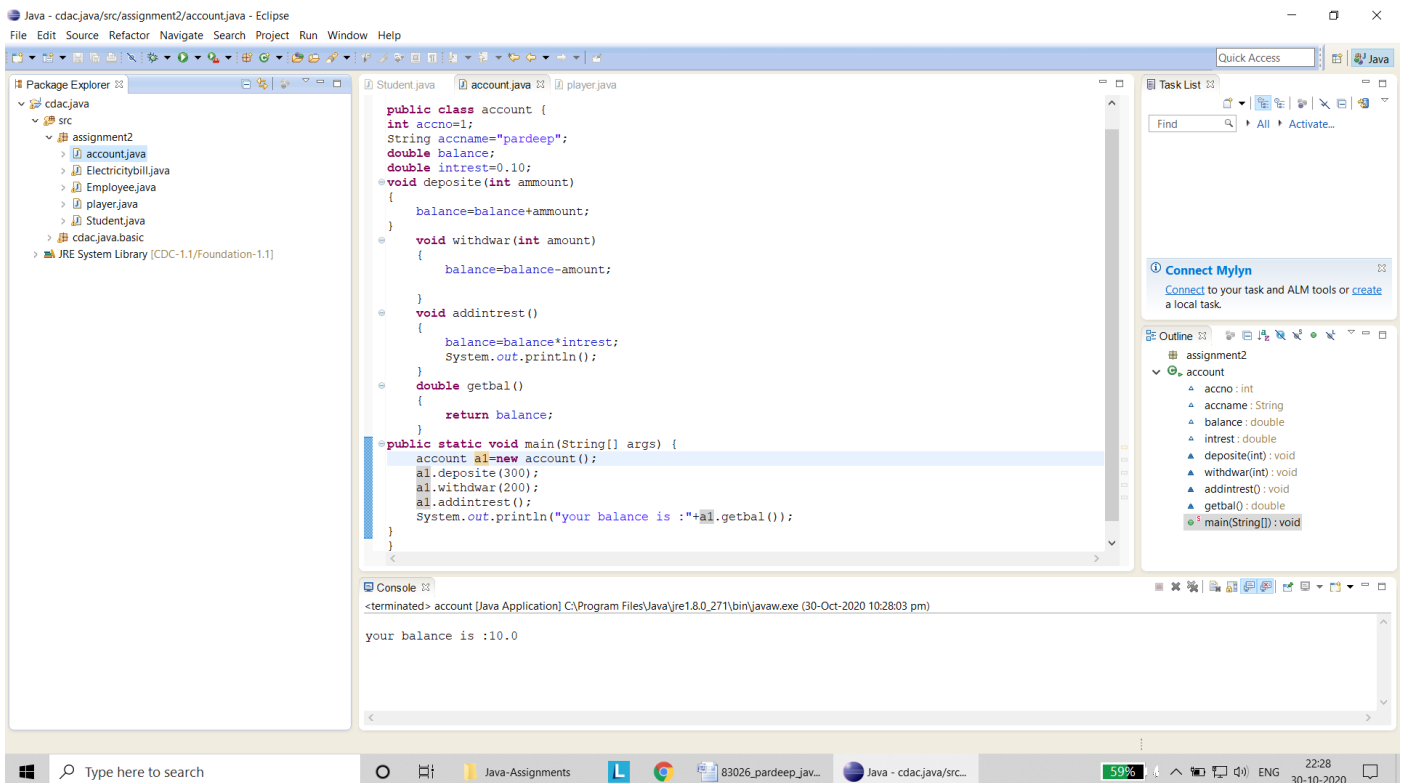
void withdraw(int amount)
{
    balance=balance-amount;
}

void addinterest()
{
    balance=balance*interest;
    System.out.println();
}

double getbal()
{
    return balance;
}

public static void main(String[] args) {
    account a1=new account();
    a1.deposit(300);
    a1.withdraw(200);
    a1.addinterest();
    System.out.println("your balance is :"+a1.getbal());
}
}

```



5) Write a Java program to create a class called player with name, age, country Name, total run as instance member. Create 5 player objects and write instance method to display the details of Player having more than 5000 as total run

```
package assignment2;

public class player {
    String name;
    int age;
    String countryname;
    int totalrun;
    player(String name,int age,String countryname,int totalrun)
    {
        this.name=name;
        this.age=age;
        this.countryname=countryname;
        this.totalrun=totalrun;
    }
    void disp()
    {
        if(this.totalrun>5000)
        {
            System.out.println("Name is :"+name);
            System.out.println("Age is :"+age);
            System.out.println("totalrun are :"+totalrun);
        }
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        player p1=new player("Pardeep",24,"india",4500);
        player p2=new player("Soni",23,"pakistan",2000);
        player p3=new player("Narender",20,"australia",6000);
        p1.disp();
        p2.disp();
        p3.disp();

    }

}
```

```
Student.java  account.java  player.java ✕  
  
    int age;  
    String countryname;  
    int totalrun;  
    player(String name,int age,String countryname,int totalrun)  
    {  
        this.name=name;  
        this.age=age;  
        this.countryname=countryname;  
        this.totalrun=totalrun;  
    }  
    void disp()  
    {  
        if(this.totalrun>5000)  
        {  
            System.out.println("Name is :"+name);  
            System.out.println("Age is :"+age);  
            System.out.println("totalrun are :"+totalrun);  
        }  
    }  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        player p1=new player("Pardeep",24,"india",4500);  
        player p2=new player("Soni",23,"pakistan",2000);  
        player p3=new player("Narender",20,"australia",6000);  
        p1.disp();  
        p2.disp();  
        p3.disp();  
    }  
}
```

```
Console ✕  
<terminated> player [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (30-Oct-2020  
Name is :Narender  
Age is :20  
totalrun are :6000
```

## Assignment 3


---

1. Write a Java Program to create a vehicle class having VehicleName, Model Number and OwnerName. Create parameterized constructor, override toString and override equals method by comparing VehicleName and ModelNumber.

```
package Assignment3;

public class Vehicle {

    private String VehicleName;
    private int ModelNumber;
    private String OwnerName;
    public Vehicle(){
        VehicleName="";
        ModelNumber=0;
        OwnerName="";
    }
    public Vehicle(String name,int ModelNumber,String OwnerName){
        VehicleName=name;
        this.ModelNumber=ModelNumber;
        this.OwnerName=OwnerName;
    }
    public String toString(){
        return "Name:"+VehicleName+"\tModel:"+ModelNumber+"\tOwner:"+OwnerName;
    }
    public static void main(String[] args){
        Vehicle v1=new Vehicle("Santro",2013,"Pardeep");
        System.out.println(v1);
        Vehicle v2=new Vehicle("Indica",2017,"Soni");
        System.out.println(v2);
        if(v1.equals(v2)){
            System.out.println("Both are same.");
        }
        else {
            System.out.println("They are different.");
        }
    }
}
```

 Console 

<terminated> Vehicle [Java Application] C:\Program Files\Java\jre1.8.0\_27

```
Name:Santro      Model:2013      Owner:Pardeep
Name:Indica      Model:2017      Owner:Soni
They are different.
```



## 2. Write a java program to demonstrate PassByValue.

```
package Assignment3;
import java.util.Scanner;
public class passbyvalue {

    public static int calculate(int a,int b){
        int sum=0;
        sum=a+b;
        return sum;
    }
    public static void main(String[] args){
        Scanner scan=new Scanner(System.in);
        System.out.println("Enter a:");
        int a=scan.nextInt();
        System.out.println("Enter b:");
        int b=scan.nextInt();
        System.out.println("Output:"+calculate(a,b));
        scan.close();
    }
}
```

<terminated>

Enter a:

1

Enter b:

2

Output:3

## 3. Create a immutable class “MyString” which contains character Array as a member and following constructors.

a. Public MyString(Char[] str)

b. Public MyString (String s)

Override equal method and return true if first 3 characters are same otherwise false. ( Example: CDAC-Blr, CDAC-Mumbai should return true.

```
package Assignment3;
```

```
import java.util.Scanner;
public class MyString{
    private char[] firstname=new char[10];
    private String lastname;
    public MyString(char[] str){
        for(int i=0;i<str.length;i++){
            firstname[i]=str[i];
        }
    }
}
```

```

    }
    }
    public MyString(String str){
        lastname=str;
    }
    public static void equals(String s1,String s2){
        if(s1.length()>=3 && s2.length()>=3){
            String temp1=s1.substring(0,3);
            String temp2=s2.substring(0,3);
            if(temp1.equals(temp2)){
                System.out.println("True");
            }else{
                System.out.println("False");
            }
        }
    }
    public static void main(String[] args){
        Scanner scan=new Scanner(System.in);
        System.out.println("Enter First Name:");
        char arr[]=scan.next().toCharArray();
        System.out.println("Enter Last Name:");
        String str=scan.next();
        MyString ms=new MyString(arr);
        System.out.println(ms.firstname);
        MyString msg=new MyString(str);
        System.out.println(msg.lastname);
        scan.close();
    }
}

```

Console

```

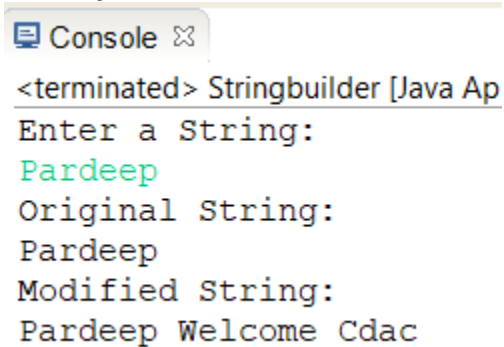
<terminated> MyString [Java Ap
Enter First Name:
Pardeep
Enter Last Name:
Soni
Pardeep
Soni
True

```

#### 4. Write a java program to manipulate a given String using **StringBuilder Class**.

```
package Assignment3;
import java.util.Scanner;
public class Stringbuilder {

    public static void main(String[] args){
        Scanner scan=new Scanner(System.in);
        System.out.println("Enter a String:");
        String sc=scan.next();
        System.out.println("Original String:\n"+sc);
        StringBuilder sb=new StringBuilder(sc);
        sb.append("\tWelcome Cdac ");
        System.out.println("Modified String:\n"+sb);
        scan.close();
    }
}
```



The screenshot shows a console window titled "Console" with the following output:

```
<terminated> Stringbuilder [Java Ap
Enter a String:
Pardeep
Original String:
Pardeep
Modified String:
Pardeep Welcome Cdac
```

#### 5. Write a Java program to reverse a given String.

```
package Assignment3;
import java.util.Scanner;
public class reversestring {

    public static void main(String args[])
    {
        String original, reverse = "";
        Scanner in = new Scanner(System.in);



        System.out.println("Enter a string to reverse");
        original = in.nextLine();

        int length = original.length();

        for (int i = length - 1 ; i >= 0 ; i--)
            reverse = reverse + original.charAt(i);

        System.out.println("Reverse of the string: " + reverse);
    }
}
```

```
}  
}
```

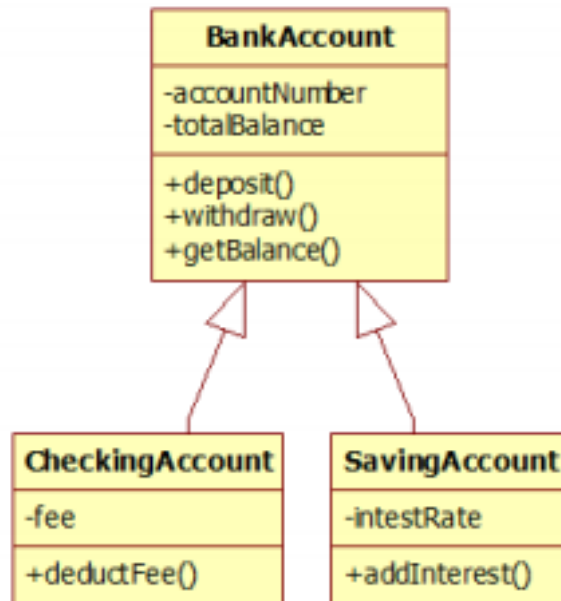
 Console 

```
<terminated> reversestring [Java Application] C  
Enter a string to reverse  
Pardeep  
Reverse of the string: peedraP
```

## Assignment4

### Inheritance

1. Write java program to implement Inheritance with following



example:

```
package Assignment4;

public class BankAccount{
    int accountNumber;
    int totalBalance;

    public static void main(String[] args) {
        SavingAccount s=new SavingAccount(101,200,10);
        System.out.println("account no is:"+s.accountNumber);

        s.deposit(500);
        s.withdraw(250);
        s.addInterest();

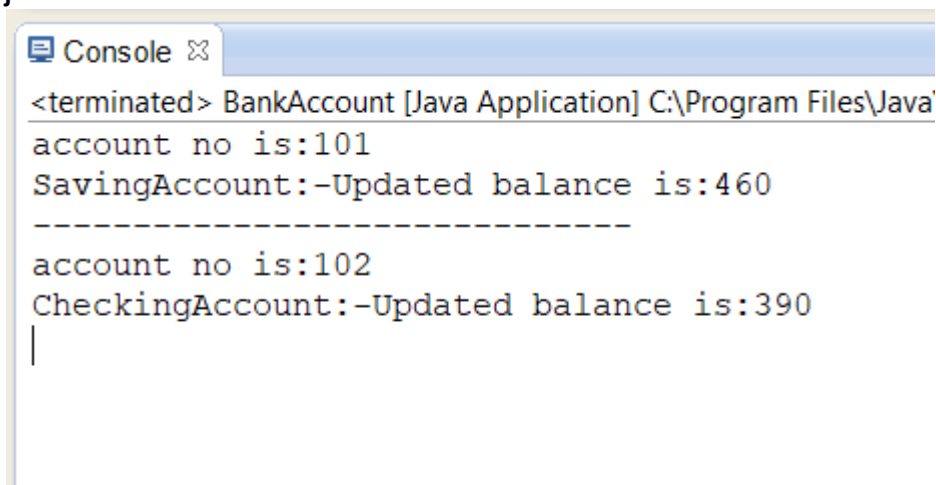
        System.out.println("SavingAccount:-Updated balance is:"+s.getBalance());
        System.out.println("-----");
        CheckingAccount c=new CheckingAccount(102,400,10);
        System.out.println("account no is:"+c.accountNumber);

        c.deposit(200);
        c.withdraw(200);
        c.deductFee();
        System.out.println("CheckingAccount:-Updated balance
is:"+c.getBalance());
    }
}
```

```

    public BankAccount(int a,int b){
        accountNumber=a;
        totalBalance=b;
    }
    public void deposit(int deposited) {
        this.totalBalance=this.totalBalance+deposited;
    }
    public void withdraw(int withdrawAmount) {
        this.totalBalance=this.totalBalance-withdrawAmount;
    }
    public int getBalance() {
        return totalBalance;
    }
}
class CheckingAccount extends BankAccount{
    int fee;
    CheckingAccount(int accountNumber,int totalBalance,int fees)
    {
        super(accountNumber,totalBalance);
        fee=fees;
    }
    public void deduceFee() {
        withdraw(fee);
    }
}
class SavingAccount extends BankAccount {
    int intestRate;
    SavingAccount(int a,int b,int c){
        super(a,b);
        intestRate=c;
    }
    public void addInterest() {
        totalBalance=getBalance()+intestRate;
    }
}

```



```

Console
<terminated> BankAccount [Java Application] C:\Program Files\Java
account no is:101
SavingAccount:-Updated balance is:460
-----
account no is:102
CheckingAccount:-Updated balance is:390
|

```

## 2. Write java program to implement Inheritance with following example:

Person will have name and age as data members. Teacher and employee will inherit data members in the super class and create its own method myProfession() to display their profession. Then create objects of Teacher, Permanent, and Contract employee to display their profession..



```
package assignmentsexten;
```

```
class Person{
    int age;
    String name;
    Person(int age,String name){
        this.age=age;
        this.name=name;
    }
    public static void main(String[] args) {
        teacher t=new teacher(24,"Pardeep");
        System.out.println(t);
        Employee e=new Employee(23,"Soni");
        System.out.println(e);
        contractEmployee c=new contractEmployee(22,"Aman");
        System.out.println(c);
    }
}

class Employee extends Person{
    Employee(int age,String name){
        super(age,name);
    }
    String myProfession() {
        return "Employee";
    }
    public String toString() {
        return "Peson[name="+name+",age="+age+" My profession is"+myProfession()+"]";
    }
}
```

```

class Permanent extends Employee{
    Permanent(int age,String name)
    {
        super(age,name);
    }
    String myProfession() {
        return "Permanent";
    }
    public String toString() {
        return "Peson[name="+name+",age="+age+" My profession is"+myProfession()+"]";
    }
}
class contractEmployee extends Employee{
    contractEmployee(int age,String name){
        super(age,name);
    }
    String myProfession() {
        return "contractEmployee";
    }
    public String toString() {
        return "Peson[name="+name+",age="+age+" My profession is"+myProfession()+"]";
    }
}
class teacher extends Person{
    teacher(int age,String name)
    {
        super(age,name);
    }
    String myProfession() {
        return "teacher";
    }
    public String toString() {
        return "Peson[name="+name+",age="+age+" My profession is"+myProfession()+"]";
    }
}
}

```

Console

```

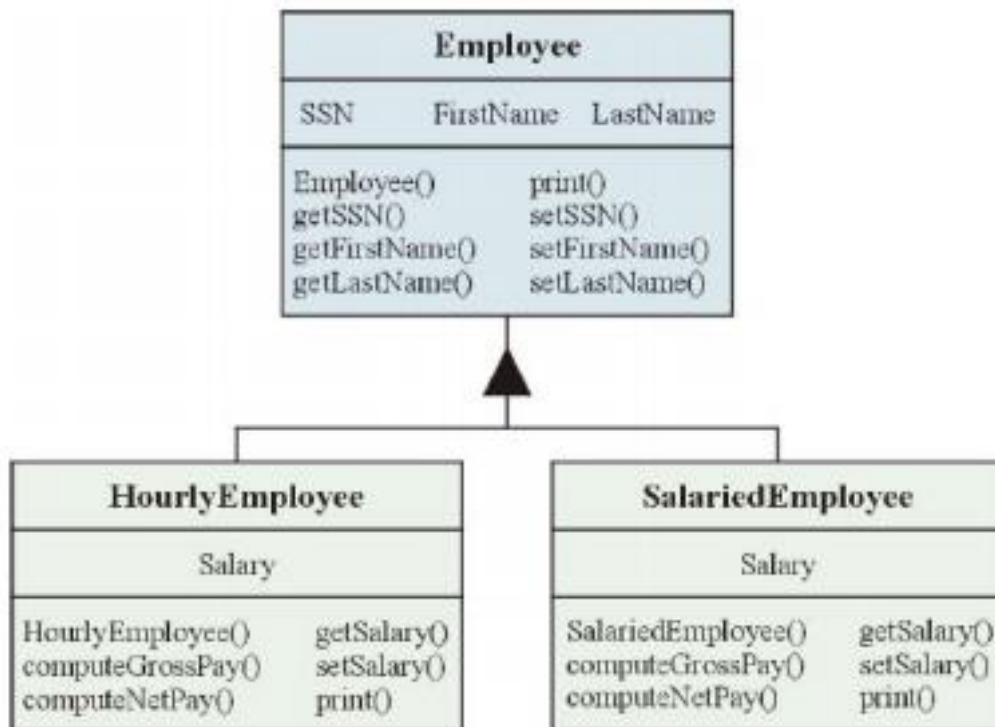
<terminated> Person [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (04-N
Peson[name=Pardeep,age=24 My profession isteacher]
Peson[name=Soni,age=23 My profession isEmployee]
Peson[name=Aman,age=22 My profession iscontractEmployee]

```



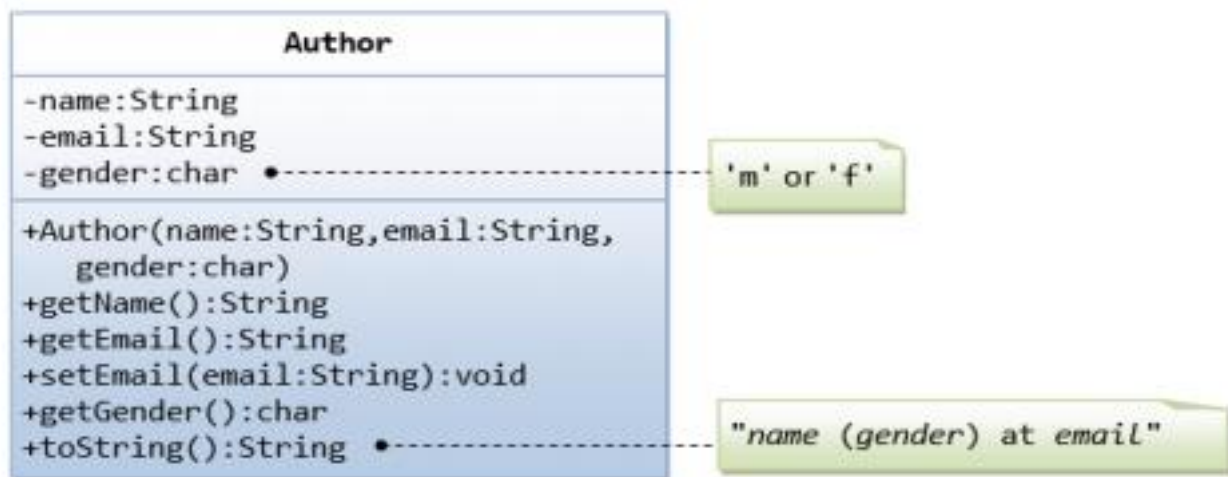
3. Write java program to implement Inheritance with following example:

## Inheritance Concepts - Example



### Inheritance using has a relationship

· Author class

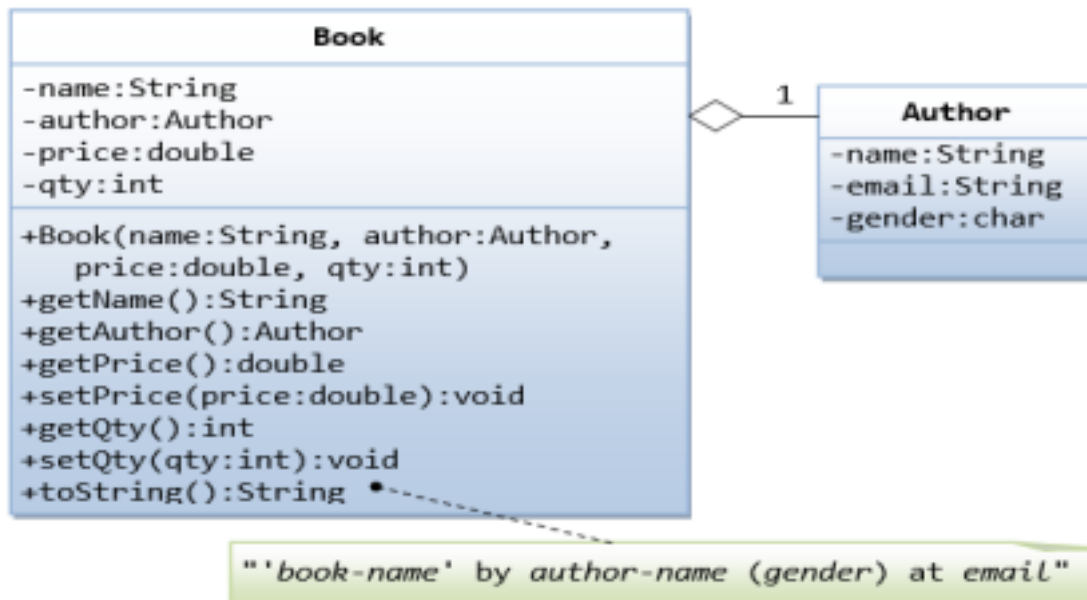


A class called Author is designed as shown in the class diagram. It contains:

- Three private member variables: name (String), email (String), and gender (char of either 'm' or 'f' - you might also use a boolean variable called isMale having value of true or false).

- A constructor to initialize the name, email and gender with the given values. (There is no *default constructor*, as there is no default value for name, email and gender.)
- Public getters/setters: getName(), getEmail(), setEmail(), and getGender(). (There are no setters for name and gender, as these properties are not designed to be changed.)

A Book is written by one Author - Using an "Object" Member Variable



Let's design a Book class. Assume that a book is written by one (and exactly one) author. The Book class (as shown in the class diagram) contains the following members:

- Four private member variables: name (String), author (an *instance* of the Author class we have just created, assuming that each book has exactly one author), price (double), and qty (int).
- The public getters and setters: getName(), getAuthor(), getPrice(), setPrice(), getQty(), setQty().

```

class Employee{
    private int SSN;
    private String FirstName;
    private String LastName;
    Employee(int SSN,String FirstName,String LastName)
    {
        this.SSN=SSN;
        this.FirstName=FirstName;
        this.LastName=LastName;
    }
    public int getSSN() {
        return this.SSN;
    }
    public String getFirstName() {
        return this.FirstName;
    }
    public String getLastName() {
        return this.LastName;
    }
    public void setSSN(int SSN) {
        this.SSN=SSN;
    }
    public void setFirstName(String FirstName) {
        this.FirstName=FirstName;
    }
    public void setLastName(String LastName) {
        this.LastName=LastName;
    }
}

class HourlyEmployee extends Employee{
    private int salary;
    HourlyEmployee(int SSN,String FirstName,String LastName,int salary){
        super(SSN,FirstName,LastName);
        this.salary=salary;
    }
}
  
```



```

    }
    public void setSalary(int salary) {
        this.salary=salary;
    }
    public int getSalary() {
        return this.salary;
    }
    public void print() {
        toString();
    }
    public String toString() {
        return "[FirstName:="+getFirstName()+"LastName:="+getLastName()+"Salary is:-"+getSalary();
    }
}
class SalaryEmployee extends Employee{
    private int salary;
    SalaryEmployee(int SSN,String FirstName,String LastName,int salary){
        super(SSN,FirstName,LastName);
        this.salary=salary;
    }
    public void setSalary(int salary) {
        this.salary=salary;
    }
    public int getSalary() {
        return this.salary;
    }
    public void print() {
        toString();
    }
    public String toString() {
        return "[FirstName:="+getFirstName()+"LastName:="+getLastName()+"Salary is:-"+getSalary();
    }
}
public class Main5 {
    public static void main(String[] args) {
        HourlyEmployee h=new HourlyEmployee(101,"Pardeep","Soni",45000);
        h.setFirstName("Soni");
        h.setLastName("Aman");
        System.out.println(h);
        SalaryEmployee s=new SalaryEmployee(102,"Soni","Aman",50000);
        System.out.println(s);
    }
}

```

#### 4. Write a program to interface with folowing example.

```
interface CreditCard{
    public void Rupee();
    public void Dollars();
    public void Pounds();
}
class BankAccount3 implements CreditCard {
    public void Rupee() {
        System.out.println("I am rupee");
    }
    public void Dollars() {
        System.out.println("I am Dollars");
    }
    public void Pounds() {
        System.out.println("I am Pounds");
    }
}
class Main4{
    public static void main(String[] args) {
        BankAccount3 b=new BankAccount3();
        b.Rupee();
        b.Dollars();
        b.Pounds();
    }
}
```

 Console 

<terminated> CreditCa

I am rupee

I am Dollars

I am Pounds

## Java Assignment -5

### ( Exception Handling )

1. Write a program to demonstrate the use of try, catch, finally throw and throws keywords and demonstrate the following points in the program

#### a. Multiple catch blocks.

```
package Assignment5;

public class Multi {
    public static void main(String[] args) {
        try {
            int array[] = new int[5];
            array[5] = 45 / 0;
        } catch (ArithmeticException | ArrayIndexOutOfBoundsException e) {
            System.out.println("Arithmic Exception");
        }
    }
}
```

 Console 



```
<terminated> Multi [Java App
Arithmic Exception
```

#### b. Try-catch-finally combination.

```
package Assignment5;

public class trycatchfinally {
    public static void main(String[] args) {
        try {
            int divideByZero = 45/0;
            System.out.println("TRY block is executed");

        } catch (ArithmeticException e) {
            System.out.println("ArithmeticException => " + e.getMessage());
        }
        finally {
            System.out.println("Finally block is always executed");
        }
    }
}
```

 Console 

```
<terminated> trycatchfinally [Java Application] C:\I
ArithmeticException => / by zero
Finally block is always executed
```

### c. Try-finally combination.

```
package Assignment5;

public class trycatchfinally {
    public static void main(String[] args) {
        try {
            System.out.println("TRY block is executed");
        }
        finally {
            System.out.println("Finally block is always executed");
        }
    }
}
```

 Console 

```
<terminated> trycatchfinally [Java Application] C:\Pro
TRY block is executed
Finally block is always executed
```

### d. Exception propagation among many methods.

```
package Assignment5;

public class Propgation {
    public void division(int num1, int num2) {
        System.out.println(num1/num2);
    }
    public void m1(int num1, int num2) {
        division(num1, num2);
    }
    public void m2(int num1, int num2){
        try{
            m1(num1, num2);}
        catch(Exception e){
            System.out.println("Exception Handled");
        }
    }
    public static void main(String args[]){
        Propgation obj = new Propgation();
        obj.m2(20, 0);
    }
}
```

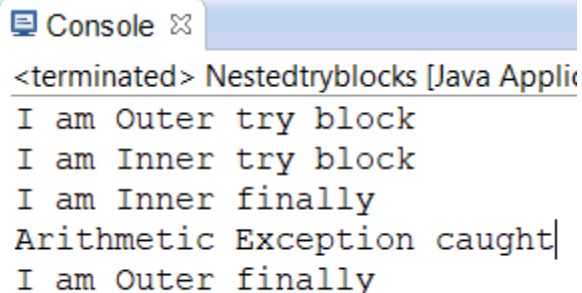
 Console 

```
<terminated> Propgation [.
Exception Handled
```

## e. Nested try blocks

```
package Assignment5;

public class Nestedtryblocks {
    public static void main(String[] args) {
        try{
            System.out.println("I am Outer try block");
            try{
                System.out.println("I am Inner try block");
                int a = 5 / 0;
            }
            catch (IllegalArgumentException e) {
                System.out.println("IllegalArgumentException caught");
            }
            finally{
                System.out.println("I am Inner finally");
            }
        }
        catch (ArithmeticException e) {
            System.out.println("Arithmetic Exception caught");
        }
        finally {
            System.out.println("I am Outer finally");
        }
    }
}
```



```
<terminated> Nestedtryblocks [Java Applic
I am Outer try block
I am Inner try block
I am Inner finally
Arithmetic Exception caught
I am Outer finally
```

## 2. Write a java program to illustrate user defined Exception.

- a. Create a class SavingsAccount, having two methods – withdraw() and getBalance ()
- b. Write a user defined exception (InsufficientFundsException) to demonstrate the insufficient fund.

```
package Assignment5;
```

```
public class BankAccount {
    private int balance = 4000;
```

```
    public int balance()
    {
        return balance;
```

```

    }

    public class InsufficientFundException extends RuntimeException {

        private String message;

        public InsufficientFundException(String message) {
            this.message = message;
        }

        public InsufficientFundException(Throwable cause, String message) {
            super(cause);
            this.message = message;
        }

        public String getMessage() {
            return message;
        }
    }

    public void withdraw(int amount) throws InsufficientFundException {
        if (amount > balance) {

            System.out.print("\nCurrent balance is less than requested amount i.e 3000");
        }
        balance = balance - amount;
    }

    public void deposit(int amount) {
        if (amount <= 0) {
            throw new IllegalArgumentException(String.format(
                "Invalid deposit amount %s", amount));
        }
    }

    public static void main(String args[]) {
        BankAccount acc = new BankAccount();

        System.out.println("Current balance : " + acc.balance());

        System.out.println("Withdrawing 2000");
        acc.withdraw(2000);

        System.out.println("Current balance : " + acc.balance());
        acc.withdraw(3500);
    }
}

```



Console

```
<terminated> BankAccount (1) [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.  
Current balance : 4000  
Withdrawing 2000  
Current balance : 2000  
  
Current balance is less than requested amount i.e 3000
```

## ( Exception Handling )

1. Create a sample annotation and demonstrate the use of the created annotation in a sample method for demonstration.

```
package Assignment5;  
  
public class Annotation {  
    void displayInfo() {  
        System.out.println("I am an Human.");  
    }  
    public static class Dog extends Annotation {  
        @Override  
        void displayInfo() {  
            System.out.println("I am a Pardeep.");  
        }  
    }  
    public static void main(String[] args) {  
        Dog d1 = new Dog();  
        d1.displayInfo();  
    }  
}
```

Console

```
<terminated> Annotation  
I am Pardeep.
```

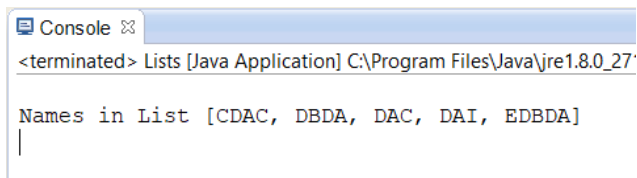
## Java Assignment 6

(Collection Framework)

### 1. Write a java program to do following operations on List.

a. Add element

```
public class Lists {  
    public static void main(String[] args) {  
        ArrayList<String> names = new ArrayList<String>();  
        //ADDING  
        names.add("CDAC");  
        names.add("DBDA");  
        names.add("DAC");  
        names.add("DAI");  
        names.add(4,"EDBDA");  
        System.out.println("\nNames in List "+names);  
    }  
}
```



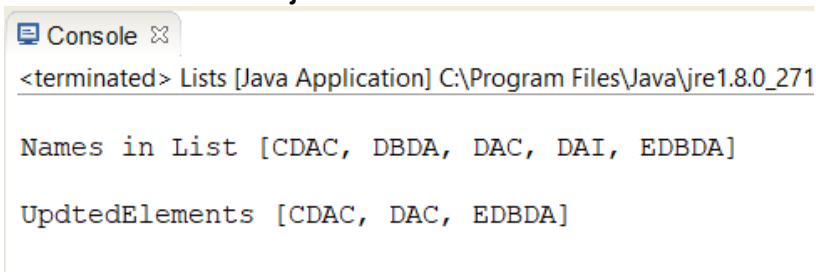
Console

<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0\_271

Names in List [CDAC, DBDA, DAC, DAI, EDBDA]

b. Remove a particular element

```
public class Lists {  
    public static void main(String[] args) {  
        ArrayList<String> names = new ArrayList<String>();  
        //ADDING  
        names.add("CDAC");  
        names.add("DBDA");  
        names.add("DAC");  
        names.add("DAI");  
        names.add(4,"EDBDA");  
        System.out.println("\nNames in List "+names);  
        //REMOVED  
        names.remove("DBDA");  
        names.remove(2);  
        System.out.println("\nUpdtedElements "+names);  
    }  
}
```



Console

<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0\_271

Names in List [CDAC, DBDA, DAC, DAI, EDBDA]

UpdtedElements [CDAC, DAC, EDBDA]

### c. Modify

```
public class Lists {  
    public static void main(String[] args) ;  
    ArrayList<String> names = new ArrayList<String>();  
    //ADDING  
    names.add("CDAC");  
    names.add("DBDA");  
    names.add("DAC");  
    names.add("DAI");  
    names.add(4,"EDBDA");  
    System.out.println("\nNames in List "+names);  
    ListIterator<String> iterator = names.listIterator();  
    while (iterator.hasNext()){  
        String s = iterator.next();  
        iterator.set(s.toLowerCase());  
    }  
    System.out.println("List After Modification :"+names);  
}
```

 Console 

<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin\javaw.exe (05-Nov-2020 3:26:54)

```
Names in List [CDAC, DBDA, DAC, DAI, EDBDA]  
List After Modification :[cdac, dbda, dac, dai, edbda]
```

### d. View All elements(Use Iterator)

```
public class Lists {  
    public static void main(String[] args) {  
        ArrayList<String> names = new ArrayList<String>();  
        //ADDING  
        names.add("CDAC");  
        names.add("DBDA");  
        names.add("DAC");  
        names.add("DAI");  
        names.add(4,"EDBDA");  
        for(String str:names)  
            System.out.println("Courses : "+str);  
        System.out.println("\nNames in List "+names);  
    }  
}
```

```
Console
<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0_27

Names in List [CDAC, DBDA, DAC, DAI, EDBDA]
Courses : CDAC
Courses : DBDA
Courses : DAC
Courses : DAI
Courses : EDBDA
```

#### e. View a Particular element (get() )

```
public class Lists {
    private static final int String = 1;
    public static void main(String[] args) {
        ArrayList<String> names = new ArrayList<String>();
        //ADDING
        names.add("CDAC");
        names.add("DBDA");
        names.add("DAC");
        names.add("DAI");
        names.add(4,"EDBDA");
        for(String str:names)
            System.out.println("Courses : "+str);
        System.out.println("\nNames in List "+names);
        System.out.println("\nElement No.1: "+names.get(String));
    }
}
```

```
Console
<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0_27

Names in List [CDAC, DBDA, DAC, DAI, EDBDA]

Element No.1: DBDA
```

#### f. Sort (Collections.sort)

```
public class Lists {
    private static final int String = 1;
    public static void main(String[] args) {
        ArrayList<String> names = new ArrayList<String>();
        //ADDING
        names.add("CDAC");
        names.add("DBDA");
        names.add("DAC");
        names.add("DAI");
        names.add(4,"EDBDA");
        for(String str:names)
            System.out.println("Courses : "+str);
        System.out.println("\nNames in List "+names);
        Collections.sort(names);
        System.out.println("\nSorted List "+names);
    }
}
```

}}

Console

<terminated> Lists [Java Application] C:\Program Files\Java\jre1.8.0\_2

Names in List [CDAC, DBDA, DAC, DAI, EDBDA]

Sorted List [CDAC, DAC, DAI, DBDA, EDBDA]

## 2. Write a java program to do following operations on Set.

### a. Add element

```
import java.util.HashSet;
import java.util.Set;
import java.util.*;
import javax.swing.text.html.HTMLDocument.Iterator;
public class SetSample{
    public static void main(String args[]) {
        //Creating HashSet and adding elements
        Set<String> set = new HashSet<>();
        set.add("One");
        set.add("Two");
        set.add("Three");
        set.add("Four");
        set.add("Five");
        System.out.println("\nSET ELEMENT : "+set);
    }
}
```

Console

<terminated> SetSample [Java Application] C:\Program Files\Java\jre1.8.0\_271\bin

SET ELEMENT : [Five, One, Four, Two, Three]

### b. Remove a particular element

```
import java.util.HashSet;
import java.util.Set;
import java.util.*;
import javax.swing.text.html.HTMLDocument.Iterator;
public class SetSample{
    public static void main(String args[]) {
        //Creating HashSet and adding elements
        Set<String> set = new HashSet<>();
        set.add("One");
```

```

        set.add("Two");
        set.add("Three");
        set.add("Four");
        set.add("Five");
        System.out.println("\nSET ELEMENT : "+set);
        set.remove("One");
        set.remove("Three");
        set.remove("Five");
        System.out.println("\nSET ELEMENT After Remove : "+set);
    }
}

```

Console

```

<terminated> SetSample [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\java.exe
SET ELEMENT : [Five, One, Four, Two, Three]
SET ELEMENT After Remove : [Four, Two]

```

### c. Remove all elements

```

import java.util.HashSet;
import java.util.Set;
import java.util.*;
import javax.swing.text.html.HTMLDocument.Iterator;
public class SetSample{
    public static void main(String args[]) {
        //Creating HashSet and adding elements
        Set<String> set = new HashSet<>();
        set.add("One");
        set.add("Two");
        set.add("Three");
        set.add("Four");
        set.add("Five");
        System.out.println("\nSET ELEMENT : "+set);
        set.clear();
        System.out.println("\nSET ELEMENT After Remove all : "+set);
    }
}

```

Console

```

<terminated> SetSample [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\java.exe
SET ELEMENT : [Five, One, Four, Two, Three]
SET ELEMENT After Remove all : []

```

#### d. View All elements(Use Iterator)

```
public class SetSample{
    public static void main(String args[]) {
        //Creating HashSet and adding elements
        Set<String> set = new HashSet<>();
        set.add("One");
        set.add("Two");
        set.add("Three");
        set.add("Four");
        set.add("Five");
        System.out.println("\nSET ELEMENT : "+set);
        for(String str:set)
            System.out.println(" Set : "+str);
    }
}
```

```
SET ELEMENT : [Five, One, Four, Two, Three]
Set : Five
Set : One
Set : Four
Set : Two
Set : Three
```

### 3. Write a java program to do following operations on Map.

#### a. Put elements

```
public class MapSample {
    public static void main(String ...args){

        Map<Integer,String> m1= new HashMap<>();
        m1.put(1,"Samsung");
        m1.put(2,"LG");
        m1.put(3,"Whirlpool");
        m1.put(4,"Lenovo");
        System.out.println("Map Elements Are:"+m1);
    }
}
```

Console

```
<terminated> MapSample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (05-No
Map Elements Are:{1=Samsung, 2=LG, 3=Whirlpool, 4=Lenovo}
```

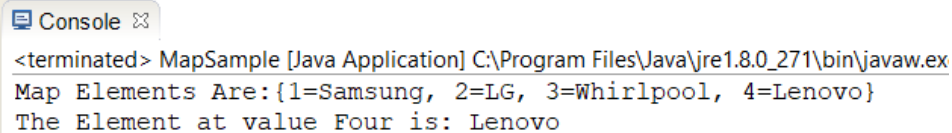
#### b. Get elements

```
public class MapSample {
    public static void main(String ...args){
```

```

        Map<Integer,String> m1= new HashMap<>();
        m1.put(1,"Samsung");
        m1.put(2,"LG");
        m1.put(3,"Whirlpool");
        m1.put(4,"Lenovo");
        System.out.println("Map Elements Are:"+m1);
    System.out.println("The Element at value Four is: " + m1.get(4));
}
}

```



```

Console
<terminated> MapSample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe
Map Elements Are:{1=Samsung, 2=LG, 3=Whirlpool, 4=Lenovo}
The Element at value Four is: Lenovo

```

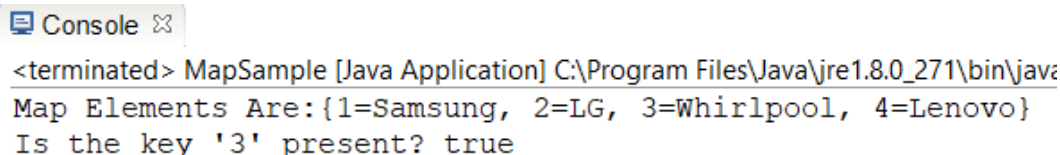
### c. Check whether element present or not

```

public class MapSample {
    public static void main(String ...args){

        Map<Integer,String> m1= new HashMap<>();
        m1.put(1,"Samsung");
        m1.put(2,"LG");
        m1.put(3,"Whirlpool");
        m1.put(4,"Lenovo");
        System.out.println("Map Elements Are:"+m1);
        System.out.println("Is the key '3' present? " + m1.containsKey(3));
    }
}

```



```

Console
<terminated> MapSample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\java
Map Elements Are:{1=Samsung, 2=LG, 3=Whirlpool, 4=Lenovo}
Is the key '3' present? true

```

### d. Remove element

```

public class MapSample {
    public static void main(String ...args){

        Map<Integer,String> m1= new HashMap<>();
        m1.put(1,"Samsung");
        m1.put(2,"LG");
        m1.put(3,"Whirlpool");
        m1.put(4,"Lenovo");
        System.out.println("Map Elements Are:"+m1);
        String returned_value = (String)m1.remove(2);
        System.out.println("Removed Element is: "+ returned_value);
    }
}

```



```

Console
<terminated> MapSample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe
Map Elements Are:{1=Samsung, 2=LG, 3=Whirlpool, 4=Lenovo}
Returned value is: LG

```

e. Run through the map.

```

public class MapSample {
    public static void main(String ...args){

        Map<Integer,String> m1= new HashMap<>();
        m1.put(1,"Samsung");
        m1.put(2,"LG");
        m1.put(3,"Whirlpool");
        m1.put(4,"Lenovo");
        System.out.println("Map Elements Are:"+m1);
        Set<Map.Entry<Integer,String>> set= m1.entrySet();
        Iterator<Map.Entry<Integer,String>> itr= set.iterator();
        while(itr.hasNext()){
            System.out.println(itr.next());
        }
    }
}

```

```

Console
<terminated> MapSample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw
Map Elements Are:{1=Samsung, 2=LG, 3=Whirlpool, 4=Lenovo}
1=Samsung
2=LG
3=Whirlpool
4=Lenovo

```

4. Write a java program to store customer objects with details, cust id, cust name, cust location. Override the equals method to check whether two customer are equal. Sort the customers name wise(use comparable).

```

class Customer {

    int CustID;
    String Custname;
    public Customer(int CustID, String Custname) {
        this.CustID = CustID;
        this.Custname = Custname;
    }
}

```

```

// Overriding equals() to compare two Customer objects
@Override
public boolean equals(Object o) {
    if (o == this) {
        return true;
    }
    if (!(o instanceof Customer)) {
        return false;
    }
    Customer c = (Customer) o;
    return int.Customer(CustID, c.CustID) == 0 && String.Customer(Custname,
c.Custname) == 0;
}
}

public static void main(String[] args) {
    Customer c1 = new Customer(5, "Pardeep");
    Customer c2 = new Customer(5, "Pardeep");
    if (c1.equals(c2)) {
        System.out.println("Equal ");
    } else {
        System.out.println("Not Equal ");
    }
}
}

```

## 5. Write a java program for performing operations such as multiplication and subtraction using generics.



```

package assignment6;
public class Generic<T extends Number> {
    private T number1, number2, sum;
    int difference;
    int product;
    public void MultNumbers (T number1, T number2){
        this.number1 = number1;
        this.number2 = number2;
    }
    public int getProd (){
        product = (int) (number1.doubleValue() * number2.doubleValue());
        return product;
    }
    public void subtractNumbers (T number1, T number2){
        this.number1 = number1;
        this.number2 = number2;
    }
}

```

```
public int getDifference(){
    difference = (int) (number1.doubleValue() - number2.doubleValue());
    return difference;
}

public static void main(String[] args) {
    Generic<Integer> integerNumbers = new Generic<>();
    integerNumbers.MultNumbers(45, 55);
    System.out.println("Multiplication of 45 and 55 is: "+integerNumbers.getProd());
    integerNumbers.subtractNumbers(999, 99);
    System.out.println("Substraction of 999 and 99 is: "+integerNumbers.getDifference());
}
```

 Console 

```
<terminated> Generic [Java Application] C:\Program Files\Java
Multiplication of 45 and 55 is: 2475
Substraction of 999 and 99 is: 900
```

## Java Assignment -7

### 1. Write a Java program to create a thread using the following methods

#### a. Extending Thread class

```
class MultithreadingDemo extends Thread
{
    public void run()
    {
        try
        {
            System.out.println ("Thread " +
                                Thread.currentThread().getId() +
                                " is running");
        }
        catch (Exception e)
        {
            System.out.println ("Exception is caught");
        }
    }
}

// Main Class
public class Multithread
{
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i=0; i<n; i++)
        {
            MultithreadingDemo object = new MultithreadingDemo();
            object.start();
        }
    }
}

<terminated> Multithread [Java .
Thread 11 is running
Thread 14 is running
Thread 13 is running
Thread 12 is running
Thread 16 is running
Thread 15 is running
Thread 17 is running
Thread 18 is running
```

## b. Implementing the Runnable interface

```
class TaskA implements Runnable{

    @Override
    public void run() {
        for ( int i=0; i<10; i++) {

            System.out.println(Thread.currentThread().getName() + " value
            : " +i);

            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}
```

```
public class RunnableSample {


    public static void main(String[] args) {

        TaskA task = new TaskA();
        Thread tA = new Thread(task);
        tA.start();

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        System.out.println(" I am getting Excuted by" +
        Thread.currentThread().getName());
    }
}
```

```
}
```

```
}
```

 Console 

<terminated> RunnableSample [Java Applicatic

Thread-0 value :1

Thread-0 value :2

Thread-0 value :3

Thread-0 value :4

Thread-0 value :5

Thread-0 value :6

Thread-0 value :7

Thread-0 value :8

Thread-0 value :9

I am getting Excuted bymain

- c. **Set thread name using the constructor argument and using the setName method.**

```
class ThreadNaming extends Thread
```

```
{
```

```
    @Override
```

```
    public void run()
```

```
    {
```

```
        System.out.println("Thread is running.....");
```

```
    }
```

```
}
```

```
class GFG
```

```
{
```

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

```
    {
```

```
        ThreadNaming t1 = new ThreadNaming();
```

```
        ThreadNaming t2 = new ThreadNaming();
```

```
        System.out.println("Thread 1: " + t1.getName());
```

```
        System.out.println("Thread 2: " + t2.getName());
```

```
        t1.start();
```

```
        t2.start();
```

```
        t1.setName("Cdac");
```

```
        t2.setName("DBDA");
```

```

System.out.println("Thread names after changing the "+
"thread names");
System.out.println("New Thread 1 name: " + t1.getName());
System.out.println("New Thread 2 name: " + t2.getName());

```

```

}
}

```

Console

```

<terminated> GFG [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\jre
Thread 1: Thread-0
Thread 2: Thread-1
Thread names after changing the thread names
New Thread 1 name: Cdac
New Thread 2 name: DBDA
Thread is running.....
Thread is running.....

```

## 2. Write a Java program to simulate Daemon and User thread

- Create a class which extends Thread
- Create its run method with an Infinite loop.
- Start the thread and analyse the output.
- Set the thread to daemon.
- Start the thread and analyze the output.

### // A. class who is extending Thread

```

public class DaemonThread extends Thread
{

```

```

    public DaemonThread(String name){
        super(name);
    }

```

### //B. Creating Run method

```

public void run()
{

```

```

    // Checking whether the thread is Daemon or not

```

```

    if(Thread.currentThread().isDaemon())

```

```

    {
        System.out.println(getName() + " is Daemon thread");
    }

```

```

    else

```

```

    {
        System.out.println(getName() + " is User thread");
    }
}

```

```

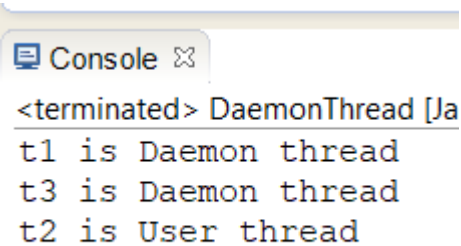
public static void main(String[] args)
{
    // C. Getting output after starting thread and analysing
    Oupput.
    DaemonThread t1 = new DaemonThread("t1");
    DaemonThread t2 = new DaemonThread("t2");
    DaemonThread t3 = new DaemonThread("t3");

    // D. Setting the Thread As daemon
    t1.setDaemon(true);

    //E. starting first 2 threads
    t1.start();
    t2.start();

    //D. Setting user thread t3 to Daemon
    t3.setDaemon(true);
    t3.start();
}
}

```



```

<terminated> DaemonThread [Ja
t1 is Daemon thread
t3 is Daemon thread
t2 is User thread

```

3. Write a Java program to compute  $f(x) = X! + Y! + Z!$  Implement runnable interface for computing factorial. Assign independent task to different threads. Perform join operation to compute summation in main thread.

```

class Threadx implements Runnable {

    int x,r=1,i;

    public Threadx(int x) {
        super();
        this.x = x;
    }
}

```



```

public void run()
{
    for(i=2;i<=x;i++)
    {
        r=r*i;
    }
    System.out.printf("%s : %d \n", Thread.currentThread().getName(), r);
}

}

class Thready implements Runnable{

    int y,r=1,i;
    public Thready(int y) {
        super();
        this.y = y;
    }

    public void run() {

        for(int i=2;i<=y;i++)
        {
            r = r*i;
        }

        System.out.printf("%s : %d \n", Thread.currentThread().getName(), r);
    }

}

class Threadz implements Runnable{
    int z,r=1,i;

    public Threadz(int z) {
        super();
        this.z = z;
    }

    public void run() {

        for(int i=2;i<=z;i++)
        {
            r =r*i;
        }

        System.out.printf("%s : %d \n", Thread.currentThread().getName(), r);
    }

}

}

public class FactorialRunnable{

```

```

public static void main(String[] args) {

    double totaladd;
    Threadx x = new Threadx(5);
        Thread t1 = new Thread(x);

        Thready y = new Thready(2);
        Thread t2 = new Thread(y);

        Threadz z = new Threadz(6);
        Thread t3 = new Thread(z);
        t1.start();
        t2.start();
        t3.start();


    System.out.println("... Multithreading is over ");

    try {
        // other thread join t1 after it completes its execution
        // tA.join();
        // other thread join t1 after 1000 millisecond



        t1.join();
        t2.join();
        t3.join();
        totaladd = x.r+y.r+z.r;
        System.out.println("f(x)= x!+y!+z! = "+ totaladd);

    } catch (InterruptedException ex) {
        Logger.getLogger(FactorialRunnable.class.getName()).log(Level.SEVERE, null, ex);
        //ex.printStackTrace();
    }

}

}

```

 Console 

```

<terminated> FactorialRunnable [Java Applicatic
... Multithreading is over
Thread-1 : 2
Thread-0 : 120
Thread-2 : 720
f(x)= x!+y!+z! = 842.0

```

4. Develop a multithreaded application to compute  $p = \sin(x) + \cos(y) + \tan(z)$ .
- Create thread by implementing Runnable interface.
  - Set thread name for different functions by using setName method.
  - Compute results P in main method.

```
class MathSin implements Runnable {  
    double deg,result;  
    public MathSin(double degree)  
    {  
        deg=degree;  
    }  
    public void run()  
    {  
        result=Math.sin(deg);  
    }  
}
```

```
class MathCos implements Runnable {  
  
    double deg,result;  
    public MathCos(double degree)  
    {  
        deg=degree;  
    }  
    public void run()  
    {  
        result=Math.cos(deg);  
    }  
}
```

```
class MathTan implements Runnable {  
    double deg,result;  
    public MathTan(double degree)  
    {  
        deg=degree;  
    }  
    public void run()  
    {  
        result=Math.tan(deg);  
    }  
}
```

```
public class MathThread {  
  
    public static void main(String[] args) {  
        double totaladd;  
        MathSin sin=new MathSin(45.0);
```

```

Thread t1 = new Thread(sin);

MathCos cos=new MathCos(45.0);
Thread t2 = new Thread(cos);

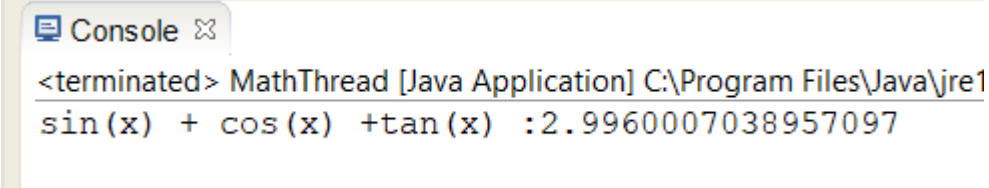
MathTan tan=new MathTan(45.0);
Thread t3 = new Thread(tan);
t1.start();
t2.start();
t3.start();
try{
    t1.join();
    t2.join();
    t3.join();
    totaladd=sin.result+cos.result+tan.result;

    System.out.println("sin(x) + cos(x) +tan(x) :"+totaladd);

}catch(InterruptedException IntExp){

}
}
}

```



The screenshot shows a console window titled "Console" with the following output:

```

<terminated> MathThread [Java Application] C:\Program Files\Java\jre1
sin(x) + cos(x) +tan(x) :2.9960007038957097

```

- 5. Write a Java program to Simulate following Synchronization problems**
- The bank account is getting accessed by multiple threads simultaneously.**
  - Perform operation, i.e. withdrawal without Synchronization constructs**

```

class NetworkStatus extends Thread {
    String ip;
    public NetworkStatus(String ip) {
        this.ip = ip;
    }
    @Override
    public void run() {
        while (true) {

```

```

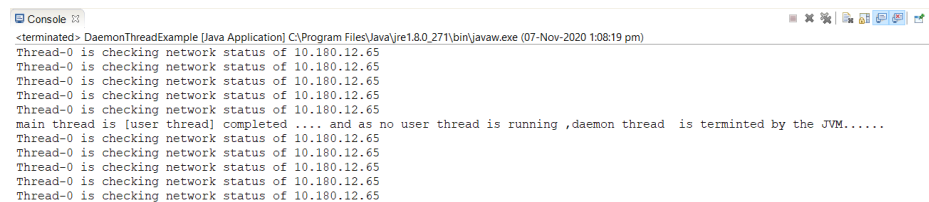
        System.out.println(Thread.currentThread().getName() + " is checking network status of
" + ip);
        try {
            Thread.sleep(100);
        } catch (InterruptedException ex) {
            Logger.getLogger(NetworkStatus.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
}
}

```

```

public class DaemonThreadExample {
    public static void main(String[] args) throws InterruptedException {
        NetworkStatus ns = new NetworkStatus("10.180.12.65");
        ns.setDaemon(false);
        ns.start();
        Thread.sleep(500);
        System.out.println(Thread.currentThread().getName() + " thread is [user thread]
completed .... and as no user thread is running ,daemon thread is terminated by the
JVM.....");
    }
}

```



```

<terminated> DaemonThreadExample [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (07-Nov-2020 1:08:19 pm)
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
main thread is [user thread] completed .... and as no user thread is running ,daemon thread is terminated by the JVM.....
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65
Thread-0 is checking network status of 10.180.12.65

```

### c. Perform withdrawal operation with synchronization constructs.

```

class Account {

    private int balance = 5000;

    public synchronized void withdraw(int amount) {
        System.out.println(Thread.currentThread().getName() + " is going to withdraw : " +
amount);
        if (balance < amount) {
            System.out.println(Thread.currentThread().getName() + " is waiting for deposit [
Insufficient balance ] "+balance);
            try {
                wait();
            } catch (InterruptedException ex) {
                Logger.getLogger(Account.class.getName()).log(Level.SEVERE, null, ex);
            }
            balance = balance - amount;
        }
    }
}

```

```

        System.out.println(Thread.currentThread().getName() + " withdrawn and available
balance : " + balance);

    } else {
        balance = balance - amount;
        System.out.println(Thread.currentThread().getName() + " withdrawn and available
balance : " + balance);
    }

}

public synchronized void deposit(int amount) {

    balance = balance + amount;
    System.out.println(Thread.currentThread().getName() + " deposited and balance : " +
balance);
    notify();
}

}

public class BankTransaction {

    public static void main(String[] args) {

        final Account ac = new Account();

        new Thread() {

            @Override
            public void run() {
                ac.withdraw(10000);
            }

        }.start();

        new Thread() {

            @Override
            public void run() {
                ac.deposit(10000);
            }

        }.start();
    }
}

```

Console

```

<terminated> BankTransaction [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe
Thread-0 is going to withdraw : 10000
Thread-0 is waiting for deposit [ Insufficient balance ] 5000
Thread-1 deposited and balance : 15000
Thread-0 withdrawn and available balance : 5000

```

## 6. Stack operations in multithreaded environment – push and pop

- Perform operations without synchronization constructs
- Perform operation with synchronization construct

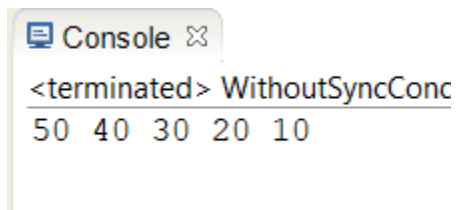
```
public class WithoutSyncConductor {  
  
    private int maxSize;  
    private long[] stackArray;  
    private int top;  
  
    public WithoutSyncConductor(int s) {  
        maxSize = s;  
        stackArray = new long[maxSize];  
        top = -1;  
    }  
  
    public void push(long j) {  
        stackArray[++top] = j;  
    }  
  
    public long pop() {  
        return stackArray[top--];  
    }  
  
    public long peek() {  
        return stackArray[top];  
    }  
  
    public boolean isEmpty() {  
        return (top == -1);  
    }  
  
    public boolean isFull() {  
        return (top == maxSize - 1);  
    }  
  
    public static void main(String[] args) {  
        WithoutSyncConductor theStack = new WithoutSyncConductor(10);  
        theStack.push(10);  
        theStack.push(20);  
        theStack.push(30);  
        theStack.push(40);  
        theStack.push(50);  
  
        while (!theStack.isEmpty()) {  
            long value = theStack.pop();  
            System.out.print(value);  
            System.out.print(" ");  
        }  
    }  
}
```

```

    }

    System.out.println("");
}
}

```



```

Console
<terminated> WithoutSyncConc
50 40 30 20 10

```

## 7. Develop a multi-threaded Java program to print all numbers below 100 that are both prime and Fibonacci numbers.

```

class PrimeAndFib
{
    Boolean isSquare(int n)
    {
        int sr = (int)Math.sqrt(n);
        return (sr * sr == n);
    }
    static void printPrimeAndFib(int n)
    {
        Boolean[] prime = new Boolean[n + 1];

        for (int p = 0; p <= n; p++)
            prime[p] = true;
        for (int p = 2; p * p <= n; p++) {

            if (prime[p] == true) {

                for (int i = p * 2; i <= n; i += p)
                    prime[i] = false;
            }
        }

        for (int i=2; i<=n; i++) {
            double sqrt = Math.sqrt(5 * i * i + 4);
            double sqrt1 = Math.sqrt(5 * i * i - 4);

            int x = (int) sqrt;
            int y = (int) sqrt1;

            if (prime[i]==true && (Math.pow(sqrt,2) ==
                Math.pow(x,2) || Math.pow(sqrt1,2) ==
                Math.pow(y,2)))
                System.out.print(i+" ");
        }
    }
}

```



```
public static void main(String s[])
{
    int n = 100;
    printPrimeAndFib(n);
}
```

Console

<terminated> PrimeA  
2 3 5 13 89