**ASSIGNMENT-IV**

1.Write a java program to read a character until a \* is encountered. Also count the number of uppercase, lowercase, and numbers entered by the users.

Sample Input:

Enter \* to exit…

Enter any character: W

Enter any character: d

Enter any character: A

Enter any character: G

Enter any character: g

Enter any character: H

Enter any character: \*

Sample Output:

Total count of lower case:2

Total count of upper case:4

Total count of numbers =0

Test cases:

1. 1,7,6,9,5

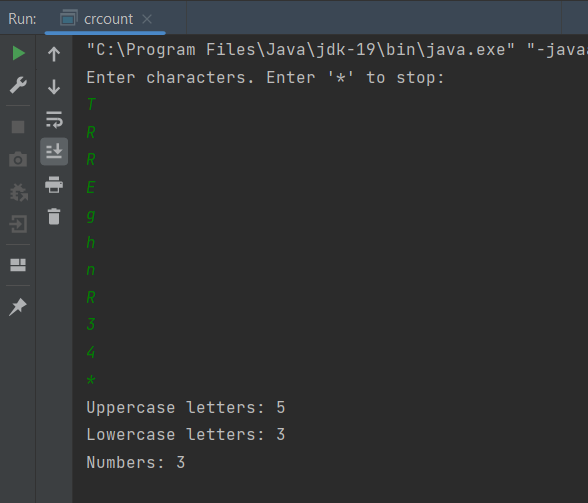
2. S, Q, l, K,7, j, M

3. M, j, L, &, @, G

4. D, K, I, 6, L, \*

5. \*, K, A, e, 1, 8, %, \*

import java.util.\*;  
public class crcount {  
 public static void main(String[] args) {  
 Scanner s= new Scanner(System.*in*);  
 int uc = 0,lc= 0,nc= 0;  
 System.*out*.println("Enter characters. Enter '\*' to stop:");  
 while (true) {  
 char c = s.next().charAt(0);  
 if (c == '\*') {  
 break;  
 } else if(c>='A' && c<='Z') {  
 uc++;  
 } else if(c>='a' && c<='z') {  
 lc++;  
 } else if(Character.*isDigit*(c))  
 {  
 nc++;  
 }  
 }  
 System.*out*.println("Uppercase letters: " + uc);  
 System.*out*.println("Lowercase letters: " + lc);  
 System.*out*.println("Numbers: " + lc);  
 }  
}



2.Bring out the situation in which member names of a subclass hide members by the same name in the super class. How it can be resolved? Write Suitable code in Java and

Implement above scenario with the Parametrized Constructor (accept int type parameter) of the Super Class can be called from Sub Class Using super () and display the input values provided.

Input :

Assign or input values for super class and sub class members.

Pseudo :

Define super class and sub class with one member (has same name)

Define method in super class and sub class with same method signature

Declare the object in main method

Invoke methods using object to display the values

Output :

Sample Input : 100, 200

Sample Output : 100, 200

Test Cases

1. 10, 20

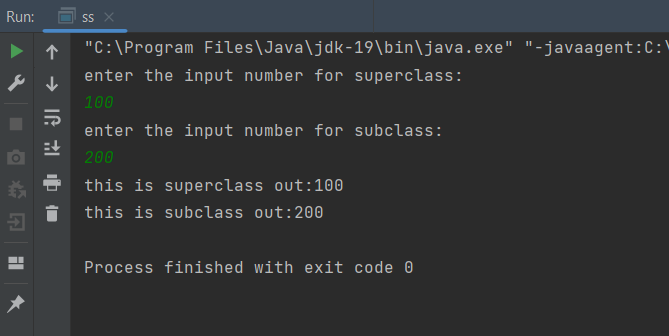
2. -20, -30

3. 0, 0

4. EIGHT FIVE

5. 10.57, 12.58

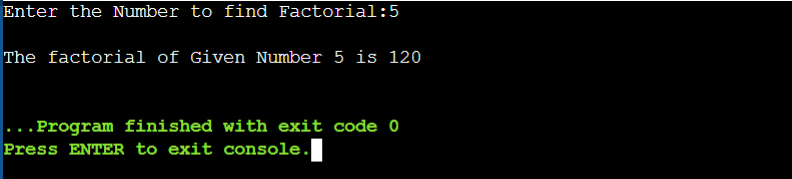
import java.util.\*;  
class supr  
{  
 int n;  
 supr(int n)  
 {  
 this.n=n;  
 }  
 void display()  
 {  
 System.*out*.println("this is superclass out:"+n);  
 }  
}  
  
class sub extends supr  
{  
 int n; // hiding  
 sub(int spn,int sn)  
 {  
 super(spn);  
 this.n=sn;  
 }  
 void display()  
 {  
 super.display();  
 System.*out*.println("this is subclass out:"+n);  
 }  
}  
  
class ss  
{  
 public static void main(String args[])  
 {  
 int n,sn;  
 Scanner s=new Scanner(System.*in*);  
 System.*out*.println("enter the input number for superclass:");  
 n=s.nextInt();  
 System.*out*.println("enter the input number for subclass:");  
 sn= s.nextInt();  
  
 sub o=new sub(n,sn);  
 o.display();  
 }  
}



**Class programms**

1.Write a java program for factorial using OOPS.

import java.util.\*;  
class FactusingOOP{  
 int n, fact=1, i, res;  
 void fact(){  
 Scanner s=new Scanner(System.*in*);  
 System.*out*.print("Enter the Number to find Factorial:");  
 n=s.nextInt();  
 for(i=1;i<=n;i++){  
 fact=fact\*i;  
 }  
 res=fact;  
 }  
 void result(){  
 System.*out*.println("The factorial of Given Number "+n+" is "+res);  
 }  
}  
class FACT{  
 public static void main(String[] args){  
 FactusingOOP obj=new FactusingOOP();  
 obj.fact();  
 obj.result();  
 }  
}



2. Write a java program for Simple Interest using OOPS Argument Passing.

**Program:-**

import java.util.\*;

class SI{

    int p, t, r, intr=0;

    SI(){

        Scanner s=new Scanner(System.in);

        System.out.print("Enter Principal Amount:");

        p=s.nextInt();

        System.out.print("Enter Time Period:");

        t=s.nextInt();

        System.out.print("Enter the Rate of Interest:");

        r=s.nextInt();

    }

    void intr(){

        intr=(p\*t\*r)/100;

        System.out.println("The Simple Interest for Principal Amount "+p+" for Time Period "+t+" is : "+intr);

    }

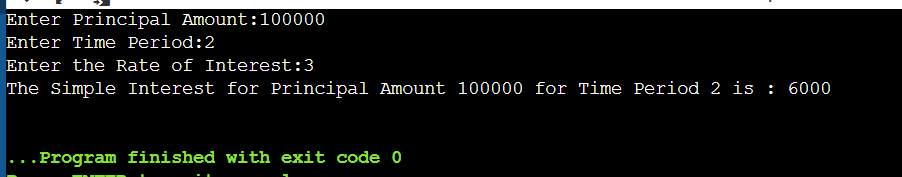
    public static void main(String[] args){

        SI obj=new SI();

        obj.intr();

    }

}

****

3. Write a java program that has a overload method. The first method should accept no arguments, the second method will accept a string and third method will accept a string and an integer. The first method should display the message “Welcome to java.” once. The second method should display the message “Welcome to Polymorphism.” twice. The third method should display “Welcome to Overloading.” thrice.

**Program:-**

import java.util.\*;

class MethodOverloading{

    void arg(){

        System.out.print("Welcome to java!\n");

    }

    void arg(String a){

        for(int i=1;i<=2;i++){

            System.out.println(a);

        }

    }

    void arg(String b,int c)

    {

        for(int i=1;i<=3;i++){

            System.out.println(b);

        }

    }

    public static void main(String[] args){

        MethodOverloading t=new MethodOverloading();

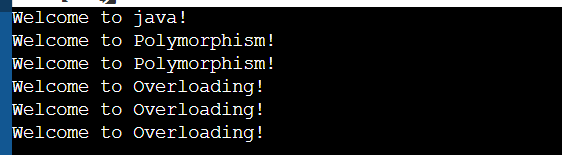
        t.arg();

        t.arg("Welcome to Polymorphism!");

        t.arg("Welcome to Overloading!", 4);

    }

}

****

4. Write a java program for finding :

1. Area of circle :- pi\*radius\*radius
2. Volume of cylinder :- pi\*radius\*radius\*height

Using Inheritance concept{especially:- Single Inheritance}

**Program:-**

import java.util.\*;

class Circle{

    protected double area;

    private int r;

    void get(int a){

        r=a;

    }

    void cal(){

        area=3.14\*r\*r;

    }

}

class Cylinder extends Circle{

    protected double volume;

    private int h;

    void get1(int b){

        h=b;

    }

    void cal1(){

        volume=area\*h;

    }

    void display(){

        System.out.println("The area of Circle is :"+area);

        System.out.println("The volume of Cylinder is: "+volume);

    }

}

class Inher{

    public static void main(String[] args){

        int x,y;

        Scanner s=new Scanner(System.in);

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

        x=s.nextInt();

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

        y=s.nextInt();

        Cylinder obj=new Cylinder();

        obj.get(x);

        obj.cal();

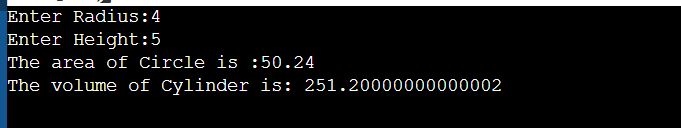
        obj.get1(y);

        obj.cal1();

        obj.display();

    }

}

****