

/*

Assignment-1

1. Write a JAVA Program of Sum of Series for a given x and n
($1+x+x^2+x^3+x^4+\dots$ up to nth terms)
2. Write a program in Java to find the reverse of a given integer
number(take input using command-line argument)
3. Write a program in Java to find the factorial of a given integer
number using Recursion (take input using command-line argument).
4. Write a JAVA program to sort n numbers
(take input using command-line argument)

*/

```
package AssignPkg.Assignment_1;
public class Assignment_1_functions
{
    public final int sumOfSeries (int t_x, int t_n)
    {
        if(t_x <= 0 || t_n <= 0)
        {
            return 1;
        }
        int ans = 1;
        for(int itr = 1; itr <= t_n; ++itr)
        {
            ans += (int) (Math.pow(t_x,itr));
        }
        return ans;
    }
    public final int reverse (int t_num)
    {
        if(t_num <= 0)
        {
            return 0;
        }
        int rev = 0;
        while(t_num >= 1)
        {
            int temp = t_num%10;
            t_num /= 10;
            rev = rev*10+temp;
        }
        return rev;
    }
    public final int factorial (int t_number)
    {
        if(t_number == 0)
            return 1;
        else
            return factorial (t_number-1) * t_number;
    }
    public final String[] sort (String [] t_arg)
    {
```

```

int tmpLen = t_arg.length;
int[] tmpArg = new int[tmpLen];
int itr;
/// converting all string value to int value . tmpArg[] <-- t_arg[]
for(itr = 0; itr < tmpLen; ++itr)
{
    tmpArg[itr] = Integer.parseInt(t_arg[itr]);
}
/// sorting the integer array namely tmpArg[]
for(int i = 0; i < tmpLen-1; ++i)
{
    for(int j = i+1; j < tmpLen; ++j)
    {
        if(tmpArg[i] > tmpArg[j])
        {
            int tmpVar = tmpArg[i];
            tmpArg[i] = tmpArg[j];
            tmpArg[j] = tmpVar;
        }
    }
}
/// converting all int value to string value . t_arg[] <-- tmpArg[]
for(itr = 0; itr < tmpLen; ++itr)
{
    t_arg[itr] = String.valueOf(tmpArg[itr]);
}
/// showing the sorted array tmpArg[]
System.out.print("the sort is assending order-> ");
for(itr = 0; itr < tmpLen; ++itr)
{
    System.out.print(tmpArg[itr] + " ");
}
System.out.println();
return t_arg;
}
}

```

```

package MainClass;
import AssignPkg.Assignment_1.Assignment_1_functions;
public class Main
{
    public static void main (String [] args)
    {
        Assignment_1_functions refObj = new Assignment_1_functions ();
        String[] a = {"-5", "-4", "-7", "4", "2", "9", "-8", "1", "12"};

        System.out.println("reverse --> " + refObj.reverse(34567) + "\nfactorial --> " + refObj.factorial(5));
        System.out.println("sumOfSeries --> " + refObj.sumOfSeries(2,3));

        refObj.sort(a);
    }
}

```

```

Output - collageAssignmentJava (run)
run:
reverse --> 76543
sumOfSeries --> 15
factorial --> 120
the sort is assending order-> -8 -7 -5 -4 1 2 4 9 12
BUILD SUCCESSFUL (total time: 0 seconds)

```

/*Assignment-2.5

5. Create a class called Board that contains two instance variables length and width of integer type and a void method for calculating surface area of the board. Use them from main method declared in different class. (take input using command line argument).

*/

```
package AssignPkg.Assignment_2;
```

```

public class Board
{
    private int m_length;
    private int m_width;

    public Board ()
    {
        setLength(0);
        setWidth(0);
    }
    public Board (int t_valueL, int t_valueW)
    {
        setLength(t_valueL);
        setWidth(t_valueW);
    }
    public Board (Board refObj)
    {
        setLength(refObj.m_length);
        setWidth(refObj.m_width);
    }

    public final int getLength ()
    {
        return m_length;
    }
    public final void setLength (int t_value)
    {
        m_length = (t_value <= 0) ? 0 : t_value;
    }
    public final int getWidth ()
    {
        return m_width;
    }
    public final void setWidth (int t_value)
    {
        m_width = (t_value <= 0) ? 0 : t_value;
    }
}

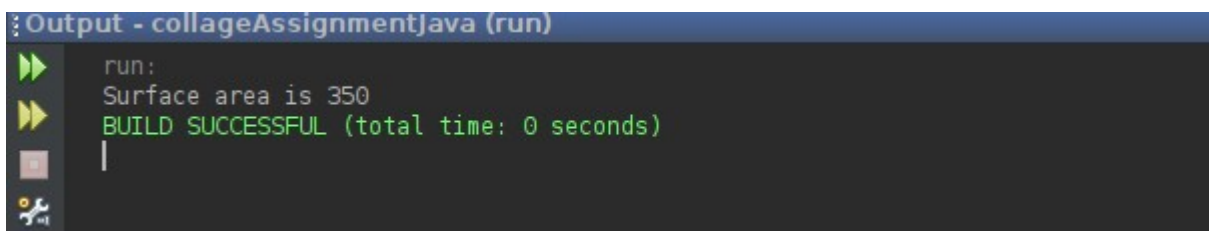
```

```

public int surfaceArea ()
{
    return getLength()*getWidth();
}
public void show ()
{
    System.out.println("\nThe surface area is "+ surfaceArea());
}
}

package MainClass;
import AssignPkg.Assignment_2.Board;
public class Main
{
    public static void main (String [] args)
    {
        Board refObj = new Board (14, 25);
        System.out.println("Surface area is " + refObj.surfaceArea());
    }
}

```



```

: Output - collageAssignmentJava (run)
run:
Surface area is 350
BUILD SUCCESSFUL (total time: 0 seconds)

```

/*Assignment-2.6

6. Write a java program to find out the multiplication of two complex number (take input using Scanner class).

```

*/
package AssignPkg.Assignment_2;

public class multiplyComplexNumber
{
    private int m_real;
    private int m_img;

    public multiplyComplexNumber ()
    {
        setReal(0);
        setImg(0);
    }

    public multiplyComplexNumber (int t_real, int t_img)
    {
        setReal(t_real);
        setImg(t_img);
    }
}

```

```

public multiplyComplexNumber (multiplyComplexNumber refObj)
{
    setReal(refObj.m_real);
    setImg(refObj.m_img);
}
public final int getReal ()
{
    return m_real;
}
private void setReal (int t_real)
{
    m_real = t_real;
}
public final int getImg ()
{
    return m_img;
}
private void setImg (int t_img)
{
    m_img = t_img;
}
public void multiply (int t_real, int t_img)
{
    calculationOfMultiplication (t_real, t_img);
}
public void multiply ( multiplyComplexNumber t_refObj )
{
    calculationOfMultiplication (t_refObj.m_real, t_refObj.m_img);
}
private void calculationOfMultiplication (int t_real, int t_img)
{
    int a = getReal ();
    int bi = getImg ();
    int c = t_real;
    int di = t_img;
    int tmpRoot1, tmpRoot2;

    /// ==> (a+bi) * (c + di) ==> ac+adi+bci-bd; ==> (ac-bd)+(adi+bci)

    int ac = a*c;
    int adi = a*di;
    int bci = bi*c;
    int bdi2 = -(bi*di);

    tmpRoot1 = ac + bdi2;
    tmpRoot2 = (adi+bci);

    show (tmpRoot1, tmpRoot2);
    setReal (tmpRoot1);
    setImg (tmpRoot2);
}

```

```

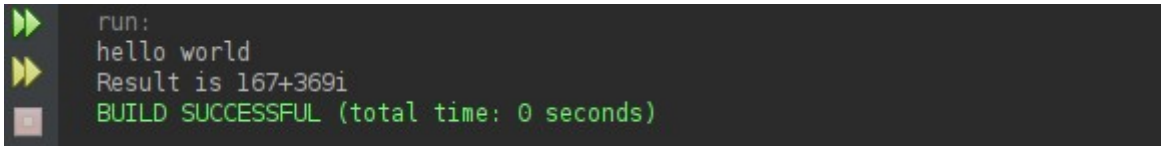
public final void show ()
{
    int t_real = getReal ();
    int t_img = getImg ();
    System.out.println("Result is " + t_real + "" + (t_img > 0 ? ("+" + t_img + "i") : (t_img + "i")));
}

public void show (int t_real, int t_img)
{
    System.out.println("Result is " + t_real + "" + (t_img > 0 ? ("+" + t_img + "i") : (t_img + "i")));
}

}

package MainClass;
import AssignPkg.Assignment_2.multiplyComplexNumber;
public class Main
{
    public static void main (String [] args)
    {
        multiplyComplexNumber obj = new multiplyComplexNumber (7, 12);
        System.out.print("hello world\n");
        obj.multiply(29, 3);
    }
}

```



```

run:
hello world
Result is 167+369i
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment-3.7**

7. Design a class to represent a Bank Account. Include the following things:

Fields

- Name of the depositor
- Account number
- Type of account
- Balance amount in the account.

Constructor

- To assign initial values

Methods

- To deposit an amount
- To withdraw an amount after checking balance
- To display the name and balance

*/

```

package AssignPkg.Assignment_3;

public class BankAccount
{
    private final String m_accountHolderName;
    private int m_accountNumber = -1;
    private String m_typeOfAccount; /// 1 type is savings and 2 is current
    private int m_balance = -1;
    private static int m_totalAccount = 100;

    public BankAccount ()
    {
        m_accountNumber = -1;
        m_accountHolderName = "void";
        m_typeOfAccount = "void";
        m_balance = -1;
    }
    public BankAccount (String t_depositorName, int t_initialBal )
    {
        m_accountHolderName = t_depositorName;
        m_typeOfAccount = "Savings";
        m_balance = (t_initialBal < 0) ? 0 : t_initialBal;
        m_accountNumber = m_totalAccount;
        ++m_totalAccount;
    }
    public BankAccount (String t_depositorName, String t_typeOfAccount, int t_initialBal )
    {
        m_accountHolderName = t_depositorName;
        m_typeOfAccount = (t_typeOfAccount.equals("Current")) ? "Current" : "Savings";
        m_balance = (t_initialBal < 0) ? 0 : t_initialBal;
        m_accountNumber = m_totalAccount;
        ++m_totalAccount;
    }
}
/*
/// this feature can not be implemented beacause one accountHolder can have
/// one account only but this feature breaks the law
public BankAccount (BankAccount t_refObj)
{
    m_accountHolderName = t_refObj.m_accountHolderName;
    m_typeOfAccount = t_refObj.m_typeOfAccount;
    m_balance = t_refObj.m_balance;
    m_accountNumber = (m_accountNumber <= 0)? 100 : ++m_accountNumber;
}
*/
public final String getAccountHolderName ()
{
    return m_accountHolderName;
}
public final int getAccountNumber ()
{
    return m_accountNumber;
}
}

```

```

public final String getTypeOfAccount ()
{
    return m_typeOfAccount;
}
/// this feature will be added after the completion of the class
/// this feature will use when account holder want to change their account
/// type . it will cost him 100 rupees only...
private void setTypeOfAccount ()
{
    m_typeOfAccount = (m_typeOfAccount.equals("Current")) ? "Savings" : "Current";
}
public final int getBalance ()
{
    return m_balance;
}
private void setBalance (int t_balance)
{
    int tmpBal = getBalance();
    m_balance = (t_balance <= 0) ? tmpBal : t_balance;
}

public int getTotalAccount ()
{
    return m_totalAccount;
}

public void showAccountDetails ()
{
    System.out.println();
    System.out.println("Name of the Account Holder is Mr./Mrs. " + getAccountHolderName ());
    System.out.println("Type of Account is *" + getTypeOfAccount () + " account");
    System.out.println("The Account Number is " + getAccountNumber ());
    System.out.println("The Account Balance is " + getBalance ());
}
public void deposit ( int t_depositAmount)
{
    /// setBalance( (t_depositAmount <= 0) ? getBalance() : t_depositAmount);
    if(t_depositAmount > 0)
    {
        setBalance (t_depositAmount+getBalance());
        System.out.println("Successfully Deposit \u20B9" + t_depositAmount);
    }
}
public void withdraw (int t_withdrawAmount)
{
    int tmpBalance = getBalance ();
    if(tmpBalance >= t_withdrawAmount)
    {
        setBalance ((tmpBalance-t_withdrawAmount));
        System.out.println("Successfully Withdrawl \u20B9" + t_withdrawAmount);
    }
    else

```



```

        insufficientBalance ();
    }
    public final void balance ()
    {
        System.out.println("Your Balance is \u20B9" + getBalance () );
    }
    public void changeAccountType ()
    {
        if(getBalance() > 100)
            setTypeOfAccount ();
        insufficientBalance ();
    }
    private void insufficientBalance ()
    {
        System.out.println("Insufficient Balance");
    }
}

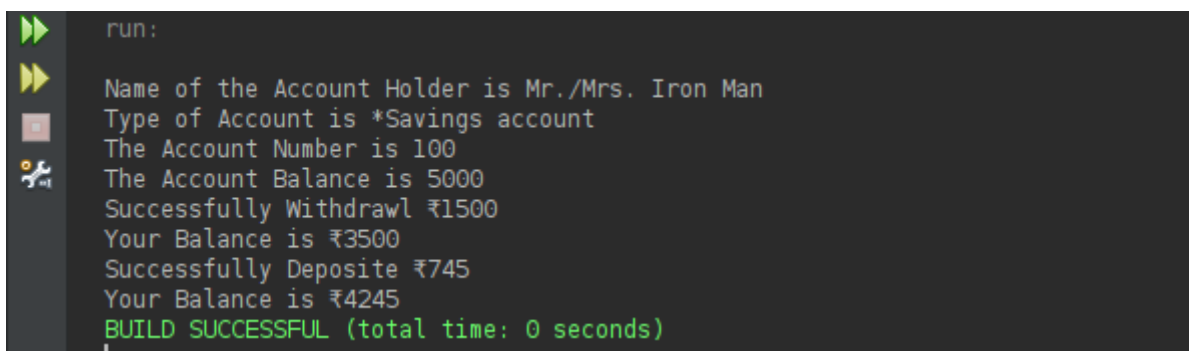
```

```

package MainClass;
import AssignPkg.Assignment_3.BankAccount;
public class Main
{
    public static void main (String [] args)
    {
        BankAccount refObj = new BankAccount ("Iron Man", 5000);

        refObj.showAccountDetails();
        refObj.withdraw(1500);
        refObj.balance();
        refObj.deposit(745);
        refObj.balance();
    }
}

```



```

run:
Name of the Account Holder is Mr./Mrs. Iron Man
Type of Account is *Savings account
The Account Number is 100
The Account Balance is 5000
Successfully Withdrawl ₹1500
Your Balance is ₹3500
Successfully Deposit ₹745
Your Balance is ₹4245
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* Assignment-3.8

8. Create a class named Shape. Make Circle, Triangle and Rectangle as object of the Shape class and calculate their area by concept of method overloading(take input using scanner class).

*/

```

package AssignPkg.Assignment_3;
public class Shape
{
    /// area of circle
    public double area(double t_re dius)
    {
        return (t_re dius*t_re dius*3.1415);
    }

    /// area of tringle by (b*h)/2 formula
    public double area(double t_base, double t_height)
    {
        return (t_base*t_height)/2;
    }

    ///area of tringle by heron's formula
    public double area(double t_a, double t_b, double t_c)
    {
        double s = (t_a + t_b + t_c) / 2;
        double tmpVar = s*(s+t_a)*(s+t_b)*(s+t_c);
        return Math.sqrt(tmpVar);
    }

    /// area of a rentangle
    public float area(float t_length, float t_bredth)
    {
        return (float)(t_length*t_bredth);
    }
}

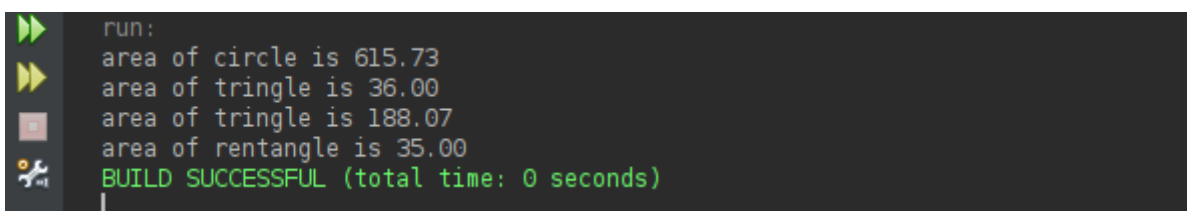
```

```

package MainClass;
import AssignPkg.Assignment_3.Shape;

public class Main
{
    public static void main (String [] args)
    {
        Shape refObj = new Shape ();
        System.out.println("area of circle is " + String.format("%.2f",refObj.area(14)));
        System.out.println("area of tringle is " + String.format("%.2f",refObj.area(12,3)));
        System.out.println("area of tringle is " + String.format("%.2f",refObj.area(9,8,2)));
        System.out.println("area of rentangle is " + String.format("%.2f",refObj.area(5,7)));
    }
}

```



```

run:
area of circle is 615.73
area of tringle is 36.00
area of tringle is 188.07
area of rentangle is 35.00
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment-4.9**

9. Assume that a bank maintains two kinds of account for its customers, one called "savings account" and other called "current account". The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance (say Rs. 1000) and if the balance falls below this level a service charge is imposed (say Rs. 100).

Create a class Account that stores customer name, account number and type of account. From this class derive two classes Curr_Acct and Savn_Acct respectively to make them more specific to their requirements. Include the necessary methods to achieve the following tasks:

Accept deposit from a customer and update the balance.

Display the balance.

Compute and deposit interest.

Permit withdrawal and update the balance.

Check for min. balance, impose penalty, if necessary, and update the balance.

Do not use any constructors. Use methods to initialize the class members.

*/

```
package AssignPkg.Assignment_4;
```

```
public class Account
```

```
{
    private String m_accountHolderName;
    private int m_accountNumber;
    private int m_accountBalance;
    private String m_typeOfAccount;
```

```
    public Account ()
```

```
    {
        m_accountHolderName = "";
        m_accountNumber = -1;
        m_accountBalance = 0;
        m_typeOfAccount = "";
    }
```

```
    public final String getAccountHolderName ()
```

```
    {
        return m_accountHolderName;
    }
```

```
    public final void setAccountHolderName (String t_acHolName)
```

```
    {
        if( m_accountHolderName.equals("") && !t_acHolName.equals(""))
        {
            m_accountHolderName = t_acHolName;
            System.out.println("\nYour Account is successfully created");
        }
    }
}
```

```

public final int getAccountNumber ()
{
    return m_accountNumber;
}
protected final void setAccountNumber (int t_acNumber)
{
    if(getAccountNumber() < 0 && (t_acNumber > 1000))
    {
        m_accountNumber = t_acNumber;
    }
}

public final int getAccountBalance()
{
    return m_accountBalance;
}
protected final void setAccountBalance(int t_acBal)
{
    if(t_acBal > 0)
    {
        m_accountBalance = t_acBal;
    }
}

public final String getTypeOfAccount ()
{
    return m_typeOfAccount;
}
protected void setTypeOfAccount (int t_check)
{
    m_typeOfAccount = (t_check == 1) ? "Savings Account" : "Current Account";
}

public void deposit ( int t_depositAmount)
{
    /// setBalance( (t_depositAmount <= 0) ? getBalance() : t_depositAmount);
    if(t_depositAmount > 0)
    {
        setAccountBalance (t_depositAmount+getAccountBalance());
        System.out.println("\nStatus::Successfully Deposit \u20B9" + t_depositAmount);
    }
}

public void withdraw (int t_withdrawAmount)
{
    int tmpBalance = getAccountBalance ();

    if(tmpBalance >= t_withdrawAmount)
    {
        setAccountBalance ((tmpBalance-t_withdrawAmount));
        System.out.println("Successfully Withdrawl \u20B9" + t_withdrawAmount);
    }
}

```

```

        else
            insufficientBalance ();
    }

    public final void balance ()
    {
        System.out.println("Your Balance is \u20B9" + getAccountBalance () );
    }
    protected void insufficientBalance ()
    {
        System.out.println("Insufficient Balance");
    }

    public void showDetails ()
    {
        System.out.println();
        System.out.println("Name of the Account Holder is Mr./Mrs. " + getAccountHolderName ());
        System.out.println("The Account Number is " + getAccountNumber ());
    }
}

/*
    The savings account provides compound interest and withdrawal facilities
    but no cheque book facility.
*/
package AssignPkg.Assignment_4;
public class Savn_Acct extends Account
{
    private static int m_totalAccounts = 1001;

    public Savn_Acct ()
    {
        setTypeOfAccount(1);
        setAccountNumber(getTotalAccount());
        m_totalAccounts+=2;
    }
    public final int getTotalAccount ()
    {
        return m_totalAccounts;
    }
    @Override
    public void showDetails ()
    {
        System.out.print("\nAccount Details ---> \n");
        System.out.println("Name of the Account Holder is Mr./Mrs. " + getAccountHolderName ());
        System.out.println("The Account Number is " + getAccountNumber ());
        System.out.println("The Account Type is *" + getTypeOfAccount ());
        System.out.println("The Account Balance is \u20B9" + getAccountBalance ());
        System.out.print("The Benefits::Your ccount provides compound interest and withdrawal facilities\n");
        System.out.println("The Drawbacks::Your savings account does not provides cheque book facility");
    }
}

```

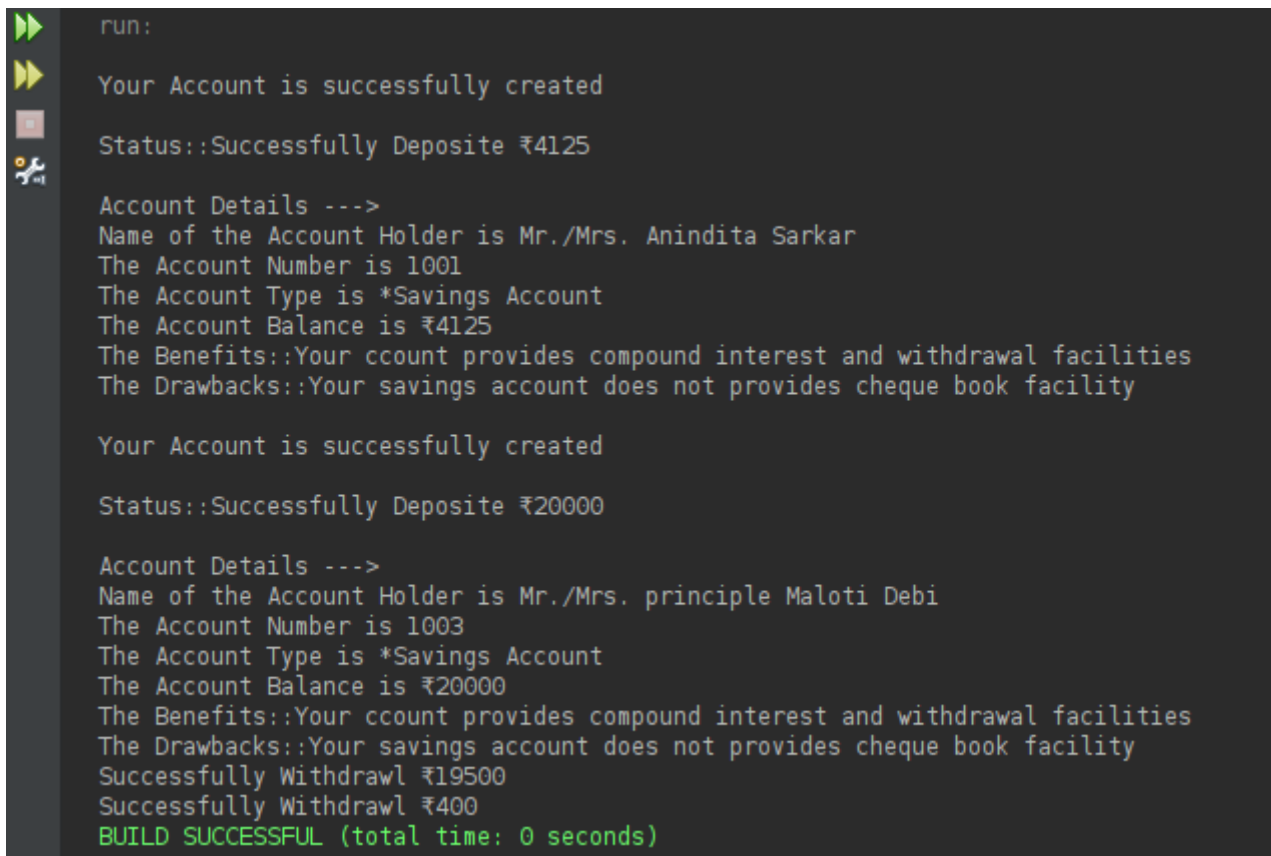
```

package MainClass;
import AssignPkg.Assignment_4.Savn_Acct;
public class Main
{
    public static void main (String [] args)
    {
        Savn_Acct svAc1 = new Savn_Acct();
        Savn_Acct svAc2 = new Savn_Acct();

        svAc1.setAccountHolderName("Anindita Sarkar");
        svAc1.deposit(4125);
        svAc1.showDetails();

        svAc2.setAccountHolderName("principle Maloti Debi");
        svAc2.deposit(20000);
        svAc2.showDetails();
        svAc2.withdraw(19500);
        svAc2.withdraw(400);
    }
}

```



```

run:
Your Account is successfully created
Status::Successfully Deposit ₹4125
Account Details --->
Name of the Account Holder is Mr./Mrs. Anindita Sarkar
The Account Number is 1001
The Account Type is *Savings Account
The Account Balance is ₹4125
The Benefits::Your account provides compound interest and withdrawal facilities
The Drawbacks::Your savings account does not provide cheque book facility
Your Account is successfully created
Status::Successfully Deposit ₹20000
Account Details --->
Name of the Account Holder is Mr./Mrs. principle Maloti Debi
The Account Number is 1003
The Account Type is *Savings Account
The Account Balance is ₹20000
The Benefits::Your account provides compound interest and withdrawal facilities
The Drawbacks::Your savings account does not provide cheque book facility
Successfully Withdraw ₹19500
Successfully Withdraw ₹400
BUILD SUCCESSFUL (total time: 0 seconds)

```

```

/*
The current account provides cheque book facility but no
interest. Current account holders should also maintain a minimum balance
(say Rs. 1000) and if the balance falls below this level a service charge
is imposed (say Rs. 100).
*/

```

```

package AssignPkg.Assignment_4;
import AssignPkg.Assignment_4.Savn_Acct;

public class Curr_Acct extends Account
{
    private static int m_totalAccounts = 1002;

    public Curr_Acct ()
    {
        setTypeOfAccount(2);
        setAccountNumber(getTotalAccount());
        m_totalAccounts+=2;
    }

    public final int getTotalAccount ()
    {
        return m_totalAccounts;
    }

    @Override
    public void withdraw (int t_withdrawlAmmount)
    {
        int tmpBalance = getAccountBalance ();

        if(tmpBalance >= t_withdrawlAmmount+100)
        {
            if(tmpBalance <= 1000)
            {
                setAccountBalance ((tmpBalance-t_withdrawlAmmount-100));
                System.out.println("Due to insufficient Minimum Balance");
                System.out.println("Service charge \u20B9100 is deducted from your account");
            }
            else
                setAccountBalance ((tmpBalance-t_withdrawlAmmount));
            System.out.println("Successfully Withdrawl \u20B9" + t_withdrawlAmmount);
        }
        else
            insufficientBalance ();
    }

    @Override
    public void showDetails ()
    {
        System.out.println();
        System.out.println("Name of the Account Holder is Mr./Mrs. " + getAccountHolderName ());
        System.out.println("The Account Number is " + getAccountNumber ());
        System.out.println("The Account Type is *" + getTypeOfAccount ());
        System.out.println("The Account Balance is " + getAccountBalance ());
        System.out.print("The Benefits:: Your savings account provides you cheque book facility\n");
        System.out.println("The Drawbacks:: Your savings account does not provides you compound interest");
    }
}

```

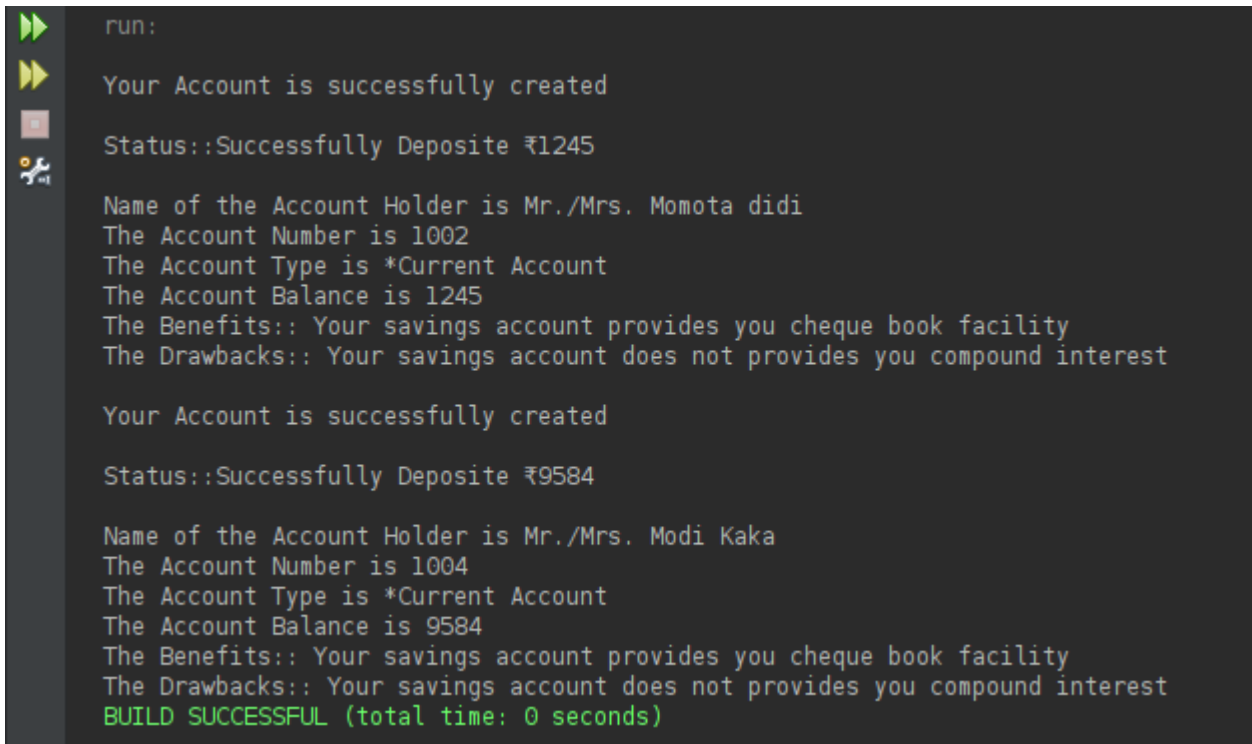
```

package MainClass;
public class Main
{
    public static void main (String [] args)
    {
        Curr_Acct cvAc1 = new Curr_Acct();
        Curr_Acct cvAc2 = new Curr_Acct();

        cvAc1.setAccountHolderName("Momota didi");
        cvAc1.deposit(1245);
        cvAc1.showDetails();

        cvAc2.setAccountHolderName("Modi Kaka");
        cvAc2.deposit(9584);
        cvAc2.showDetails();
    }
}

```



```

run:
Your Account is successfully created
Status::Successfully Deposit ₹1245
Name of the Account Holder is Mr./Mrs. Momota didi
The Account Number is 1002
The Account Type is *Current Account
The Account Balance is 1245
The Benefits:: Your savings account provides you cheque book facility
The Drawbacks:: Your savings account does not provides you compound interest

Your Account is successfully created
Status::Successfully Deposit ₹9584
Name of the Account Holder is Mr./Mrs. Modi Kaka
The Account Number is 1004
The Account Type is *Current Account
The Account Balance is 9584
The Benefits:: Your savings account provides you cheque book facility
The Drawbacks:: Your savings account does not provides you compound interest
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* Assignment-5.10

10. Show that static block execute before any object creation.

*/

```

package AssignPkg.Assignment_5;
public class StaticBlockExecutionBeforeObj
{
    public StaticBlockExecutionBeforeObj ()
    {
        infoPrint();
    }
    private void infoPrint ()
    {
        System.out.println("1.Obj is successfully created (StaticBlockExecutionBeforeObj)");
    }
}

```



```

static
{
    System.out.println("1.Static block is executed successfully (StaticBlockExecutionBeforeObj)");
}
}

```

```

package MainClass;
import AssignPkg.Assignment_5.StaticBlockExecutionBeforeObj;
public class Main
{
    public static void main (String [] args)
    {
        StaticBlockExecutionBeforeObj refObj = new StaticBlockExecutionBeforeObj();
    }
}

```

```

run:
1.Static block is executed successfully (StaticBlockExecutionBeforeObj)
1.Obj is successfully created (StaticBlockExecutionBeforeObj)
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment- 5.11**

11. Show that static blocks are executed in order of inheritance.

*/

```

package AssignPkg.Assignment_5;

```

```

public class StaticBlockExecutionOnInheritance extends StaticBlockExecutionBeforeObj
{
    public StaticBlockExecutionOnInheritance ()
    {
        infoPrint();
    }
    private void infoPrint ()
    {
        System.out.println("2.Obj is successfully created (from StaticBlockExecutionOnInheritance)");
    }
    static
    {
        System.out.println("2.Static block is executed successfully (from StaticBlockExecutionOnInheritance)");
    }
}

```

```

package MainClass;
import AssignPkg.Assignment_5.StaticBlockExecutionOnInheritance;
public class Main
{
    public static void main (String [] args)
    {
        StaticBlockExecutionOnInheritance refObj1 = new StaticBlockExecutionOnInheritance();
    }
}

```

```

run:
1.Static block is executed successfully (StaticBlockExecutionBeforeObj)
2.Static block is executed successfully (from StaticBlockExecutionOnInheritance)
1.Obj is successfully created (StaticBlockExecutionBeforeObj)
2.Obj is successfully created (from StaticBlockExecutionOnInheritance)
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment- 5.12**

12. Create a class with an inner class that has no default constructor (one that take arguments). Create a second class with an inner class that inherits from the first inner class

*/

```
package AssignPkg.Assignment_5.Assignment_5_12;
```

```
public class InnerClass
{
    private int m_a;

    public InnerClass (int t_a)
    {
        setA(t_a);
        System.out.println("Inside of InnerClass Constructor");
    }
    public int getA ()
    {
        return m_a;
    }
    protected void setA (int t_a)
    {
        m_a = t_a;
    }
    public void show ()
    {
        System.out.println("A = " + getA());
    }
}
```

```
package AssignPkg.Assignment_5.Assignment_5_12;
```

```
public class DemoClass
{
    public class InnerClassInsideDemo extends InnerClass
    {
        private int m_b;

        public InnerClassInsideDemo (int t_a, int t_b)
        {
            super(t_a);
            setB(t_b);
            System.out.println("Inside of InnerClassInsideDemo Constructor");
        }
    }
}
```

```

    public final int getB ()
    {
        return m_b;
    }
    public final void setB (int t_b)
    {
        m_b = t_b;
    }

    @Override
    public void show ()
    {
        System.out.println("A = " + getA());
        System.out.println("B = " + getB());
    }
}

```

/// -----

```

private int m_c;
private final InnerClassInsideDemo m_obj;

public DemoClass (int t_a, int t_b, int t_c)
{
    m_obj = new InnerClassInsideDemo(t_a, t_b);
    setC(t_c);
    System.out.println("Inside of DemoClass Constructor");
}

public int getC ()
{
    return m_c;
}
public final void setC (int t_c)
{
    m_c = t_c;
}

public InnerClassInsideDemo getObj ()
{
    return m_obj;
}
public void show ()
{
    InnerClassInsideDemo tmpObj = getObj();
    System.out.println("A = " + tmpObj.getA());
    System.out.println("B = " + tmpObj.getB());
    System.out.println("C = " + getC ());
}
}

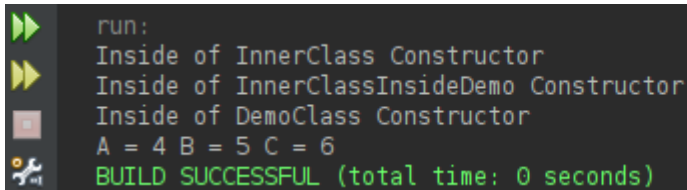
```

```

package MainClass;
import AssignPkg.Assignment_5.Assignment_5_12.DemoClass;

public class Main
{
    public static void main (String [] args)
    {
        DemoClass refObj = new DemoClass(4, 5, 6);
        refObj.show();
    }
}

```



```

run:
Inside of InnerClass Constructor
Inside of InnerClassInsideDemo Constructor
Inside of DemoClass Constructor
A = 4 B = 5 C = 6
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment- 5.13**

13. Consider an example of book shop which sells books and video tapes. These two classes inherited from the super class called media. The media class has data member such as title and publications. The book class has data members for storing number of pages in a book and the tape class has the playing time in tape. Each class will have member function such as read() and show(). Write a program which models the class hierarchy for book shop.

*/

```

package AssignPkg.Assignment_5.Assignment_5_13;
public class Media
{
    private String m_title = "";
    private String m_publication = "";

    public Media (){}
    public Media (String t_title, String t_publication)
    {
        setTitle (t_title);
        setPublication (t_publication);
    }
    public final String getTitle ()
    {
        return m_title;
    }
    protected final void setTitle (String t_title)
    {
        if (m_title.equals("") && !t_title.equals(""))
        {
            m_title = t_title;
        }
    }
}

```

```

public final String getPublication ()
{
    return m_publication;
}
protected final void setPublication (String t_publication)
{
    if (m_publication.equals("") && !t_publication.equals(""))
    {
        m_publication = t_publication;
    }
}

public void read (String t_title, String t_publication)
{
    setTitle (t_title);
    setPublication (t_publication);
}
public void show ()
{
    System.out.println("Title is " + getTitle());
    System.out.println("Publication is " + getPublication());
}
}

package AssignPkg.Assignment_5.Assignment_5_13;

public class VideoTape extends Media
{
    private int m_playTime;

    public VideoTape ()
    {
        super ();
        m_playTime = 0;
    }
    public VideoTape (String t_title, String t_publication, int t_playTime)
    {
        super (t_title, t_publication);
        setPlayTime(t_playTime);
    }

    public int getPlayTime ()
    {
        return m_playTime;
    }
    private void setPlayTime (int t_playTime)
    {
        if(t_playTime >= 1)
        {
            m_playTime = t_playTime;
        }
    }
}

```

```

public void read (String t_title, String t_publication, int t_playTime)
{
    super.setTitle (t_title);
    super.setPublication (t_publication);
    setPlayTime(t_playTime);
}

@Override
public void show ()
{
    System.out.println("Title of the Video Tape is \"" + getTitle() + "\"");
    System.out.println("Publication of the Video Tape is \"" + getPublication() + "\"");
    System.out.println("Total time \"" + String.format("%.2f", (float)(getPlayTime()/60.0)) + " hours\"");
    System.out.println();
}
}

```

```

package AssignPkg.Assignment_5.Assignment_5_13;

```

```

public class Books extends Media
{
    private int m_numberOfPages;

    public Books ()
    {
        super ();
        m_numberOfPages = 0;
    }

    public Books (String t_title, String t_publication, int t_noOfPage)
    {
        super (t_title, t_publication);
        setNoOfPage(t_noOfPage);
    }

    public int getNoOfPage ()
    {
        return m_numberOfPages;
    }

    private void setNoOfPage (int t_noOfPage)
    {
        if(t_noOfPage >= 1)
        {
            m_numberOfPages = t_noOfPage;
        }
    }
}

```

```

public void read (String t_title, String t_publication, int t_noOfPage)
{
    super.setTitle (t_title);
    super.setPublication (t_publication);
    setNoOfPage(t_noOfPage);
}

@Override
public void show ()
{
    System.out.println("Title of the Book is \"\" + getTitle() + "\"");
    System.out.println("Publication of the Book is \"\" + getPublication() + "\"");
    System.out.println("No of pages the Book have is \"\" + getNoOfPage() + "\"");
    System.out.println();
}
}

```

```

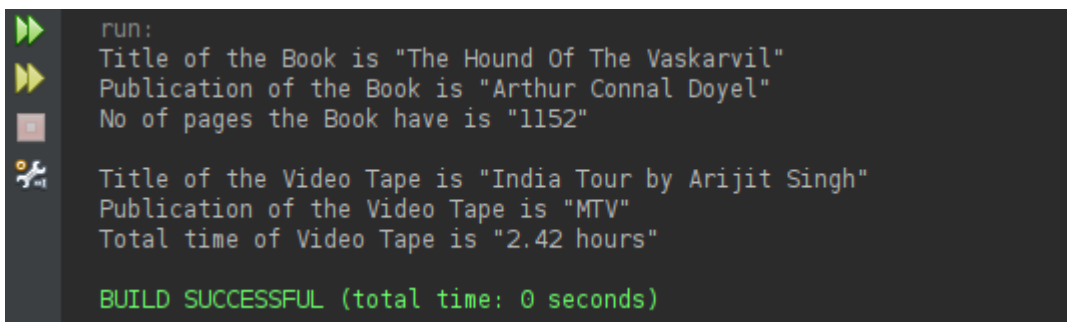
package MainClass;
import AssignPkg.Assignment_5.Assignment_5_13.Books;
import AssignPkg.Assignment_5.Assignment_5_13.VideoTape;

public class Main
{
    public static void main (String [] args)
    {
        Books refBook = new Books("The Hound Of The Vaskarvil", "Arthur Connal Doyel", 1152);
        VideoTape refVideoTape = new VideoTape("India Tour by Arijit Singh", "MTV", 145);

        refBook.read ("The Hound Of The Vaskarvil", "Arthur Connal Doyel", 1152);
        refBook.show();

        refVideoTape.read("India Tour by Arijit Singh", "MTV", 145);
        refVideoTape.show();
    }
}

```



```

run:
Title of the Book is "The Hound Of The Vaskarvil"
Publication of the Book is "Arthur Connal Doyel"
No of pages the Book have is "1152"

Title of the Video Tape is "India Tour by Arijit Singh"
Publication of the Video Tape is "MTV"
Total time of Video Tape is "2.42 hours"

BUILD SUCCESSFUL (total time: 0 seconds)

```

/* Assignment- 6.14

14. Create an abstract class Shape with two abstract methods, area() & disp(). Now design three concrete classes Rectangle, Circle & Triangle can compute area and display its separately.

***/**

```
package AssignPkg.Assignment_6.Assignment_6_14;
```

```
public abstract class Shape
```

```
{  
    public abstract double area ();  
    public abstract void disp ();  
}
```

```
package AssignPkg.Assignment_6.Assignment_6_14;
```

```
public class Rectangle extends Shape
```

```
{  
    private double m_length = 0;  
    private double m_bredth = 0;
```

```
    public Rectangle ()
```

```
{
```

```
}
```

```
    public Rectangle (double t_length, double t_bredth)
```

```
{  
        setLength (t_length);  
        setBredth (t_bredth);  
}
```

```
    public final double getLength ()
```

```
{  
        return m_length;  
}
```

```
    protected final void setLength (double t_length)
```

```
{  
        if(t_length > 0)  
        {  
            m_length = t_length;  
        }  
}
```

```
    public final double getBredth ()
```

```
{  
        return m_bredth;  
}
```

```
    protected final void setBredth (double t_bredth)
```

```
{  
        if(t_bredth > 0)  
        {  
            m_bredth = t_bredth;  
        }  
}
```



```

public final void setData (double t_a, double t_b)
{
    setLength (t_a);
    setBredth (t_b);
}

@Override
public double area ()
{
    return (getLength () * getBredth ());
}

@Override
public void disp ()
{
    System.out.println("Area of Rectangle is " + String.format("%.2f",area ()));
}
}

```

```

package AssignPkg.Assignment_6.Assignment_6_14;
public class Circle extends Shape
{
    private double m_re dius = 0;

    public Circle (){}

    public Circle (double t_re dius)
    {
        setRedius (t_re dius);
    }
    public double getRedius ()
    {
        return m_re dius;
    }
    public final void setRedius (double t_re dius)
    {
        if (t_re dius > 0)
        {
            m_re dius = t_re dius;
        }
    }
    @Override
    public double area ()
    {
        return (Math.pow(m_re dius, 2) * 3.1415);
    }
    @Override
    public void disp ()
    {
        System.out.println("Area of Circle is " + String.format("%.2f",area ()));
    }
}

```

```
package AssignPkg.Assignment_6.Assignment_6_14;
public class Triangle extends Shape
{
    private double m_a = 0;
    private double m_b = 0;
    private double m_c = 0;

    public Triangle (){}
    public Triangle (double t_base, double t_height)
    {
        setA (t_base);
        setB (t_height);
    }
    public Triangle (double t_a, double t_b, double t_c)
    {
        setA (t_a);
        setB (t_b);
        setC (t_c);
    }
    public final double getA ()
    {
        return m_a;
    }
    protected final void setA (double t_a)
    {
        if(t_a > 0)
        {
            m_a = t_a;
        }
    }
    public final double getB ()
    {
        return m_b;
    }
    protected final void setB (double t_b)
    {
        if(t_b > 0)
        {
            m_b = t_b;
        }
    }
    public final double getC ()
    {
        return m_c;
    }
    protected final void setC (double t_c)
    {
        if(t_c > 0)
        {
            m_c = t_c;
        }
    }
}
```

```

public final void setData (double t_a, double t_b, double t_c)
{
    setA (t_a);
    setB (t_b);
    setC (t_c);
}
public final void setData (double t_a, double t_b)
{
    setA (t_a);
    setB (t_b);
}

/// area of tringle by (b*h)/2 formula and by heron's formula
@Override
public double area()
{
    if(getC () == 0)
    {
        return (getA () * getB ())/2;
    }
    else
    {
        double s = (getA() + getB() + getC()) / 2;
        double tmpVar = s*(s + getA()*(s + getB()*(s + getC()));
        return Math.sqrt(tmpVar);
    }
}

@Override
public void disp ()
{
    System.out.println("Area of Tringle is " + String.format("%.2f", area ());
}
}

```

```

package MainClass;
import AssignPkg.Assignment_6.Assignment_6_14.Shape;
import AssignPkg.Assignment_6.Assignment_6_14.Rectangle;
import AssignPkg.Assignment_6.Assignment_6_14.Circle;
import AssignPkg.Assignment_6.Assignment_6_14.Triangle;

public class Main
{
    public static void main (String [] args)
    {
        Rectangle refRectangle = new Rectangle (5, 7);
        Circle refCircle = new Circle (4);
        Triangle refTringle1 = new Triangle (3, 4);
        Triangle refTringle2 = new Triangle (3, 4, 5);

        Shape obj[] = {refRectangle, refCircle, refTringle1, refTringle2};
    }
}

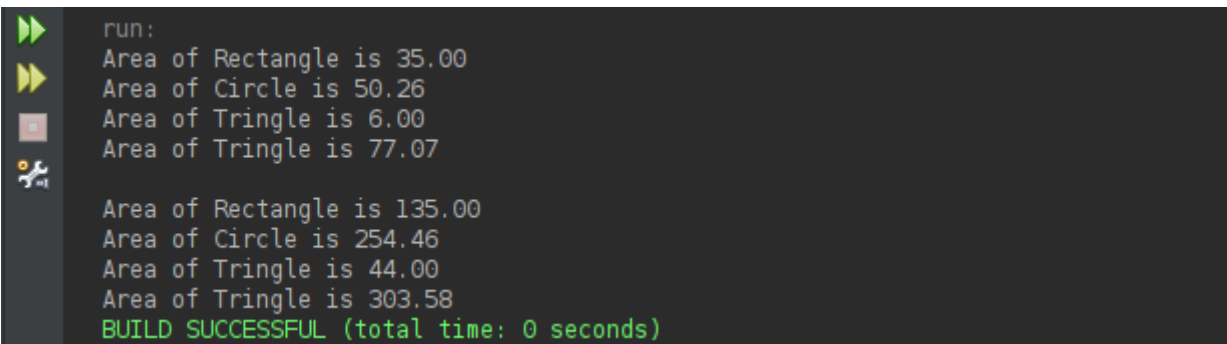
```

```

        for (Shape x : obj)
        {
            x.disp ();
        }
        refCircle.setRedius (9);
        refRectangle.setData (15,9);
        refTringle1.setData(8, 11);
        refTringle2.setData(8, 4, 12);

        System.out.println();
        for (Shape x : obj)
        {
            x.disp ();
        }
    }
}

```



```

run:
Area of Rectangle is 35.00
Area of Circle is 50.26
Area of Tringle is 6.00
Area of Tringle is 77.07

Area of Rectangle is 135.00
Area of Circle is 254.46
Area of Tringle is 44.00
Area of Tringle is 303.58
BUILD SUCCESSFUL (total time: 0 seconds)

```

/*Assignment- 6.15

15. Create two interface, each with one method, inherit a new interface from the two, adding a new method. Create a class by implementing the new interface also inheriting from a concrete class . Now write three methods, each of which takes one of the three interface as an argument. In main() create an object of your class and pass it to each of the methods.

*/

```

package AssignPkg.Assignment_6.Assignment_6_15.interfacePkg;
public interface Interface1
{
    public void method1 ();
}

```

```

package AssignPkg.Assignment_6.Assignment_6_15.interfacePkg;
public interface Interface2
{
    public void method2 ();
}

```

```
package AssignPkg.Assignment_6.Assignment_6_15.interfacePkg;
public interface InheritedInterface extends Interface1, Interface2
{
    public void method3 ();
}
```

```
package AssignPkg.Assignment_6.Assignment_6_15.concreteClass;
import AssignPkg.Assignment_6.Assignment_6_15.interfacePkg.InheritedInterface;
```

```
public class ImplementClass implements InheritedInterface
{
    @Override
    public void method1 ()
    {
        System.out.println("Method1 implements by concrete class (method1 is in Interface1)");
    }
    @Override
    public void method2 ()
    {
        System.out.println("Method2 implements by concrete class (method1 is in Interface2)");
    }
    @Override
    public void method3 ()
    {
        System.out.println("Method3 implements by concrete class (method3 is in InheritedInterface)");
    }
}
```

```
package AssignPkg.Assignment_6.Assignment_6_15.concreteClass;
public class Concrete
{
    public void concreteMethod ()
    {
        System.out.println("concrete Method from concrete class");
    }
}
```

/* Assignment – 7.16

16. Write a Java code to create a package named “testpackage”, create a class named “Add” under this package which contains a method named ”display”. Import this package in another Java program

*/

```
package AssignPkg.Assignment_7.testpackage;
public class Add
{
    public Add ()
    {
        display ();
    }
}
```

```

    public final void display ()
    {
        System.out.println("Inside Add class which is Inside of testpackage");
    }
}

```

```

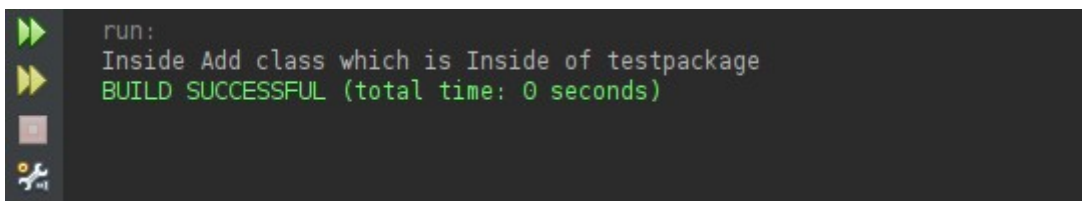
package MainClass;
import AssignPkg.Assignment_7.testpackage.Add;

```

```

public class Main
{
    public static void main (String [] args)
    {
        Add refObj = new Add();
    }
}

```



```

run:
Inside Add class which is Inside of testpackage
BUILD SUCCESSFUL (total time: 0 seconds)

```

/* **Assignment – 7.17**

17. Create an interface with at least one method in its own package. Create a class in a separate package. Add a protected inner class that implements the interface. In a third package, inherit from your class and inside a method, return an object of the protected inner class up-casting to the interface during the return.

*/

```

package AssignPkg.Assignment_7.pkg1;
public interface Interface1
{
    public void show ();
}

```

```

package AssignPkg.Assignment_7.pkg2;
import AssignPkg.Assignment_7.pkg1.Interface1;
public class Class1
{
    protected class ClassX implements Interface1
    {
        @Override
        public void show ()
        {
            System.out.println("this is show method which is implemented by a class”
            System.out.println(“which is a inner class of class1");
        }
    }
}

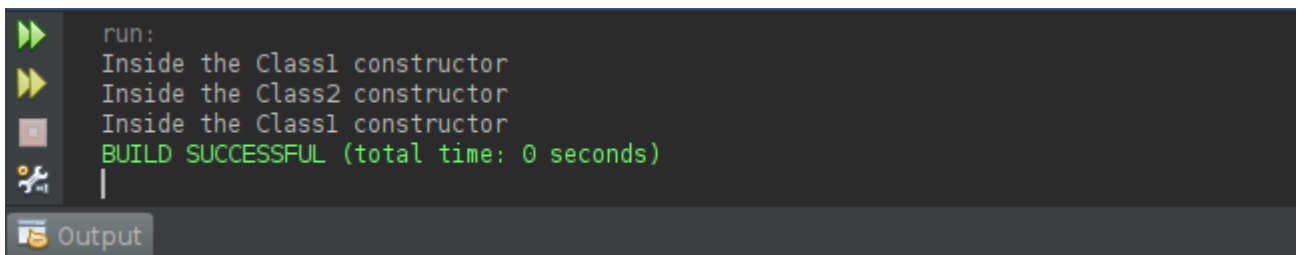
```

```
package AssignPkg.Assignment_7.pkg3;
import AssignPkg.Assignment_7.pkg2.Class1;
```

```
public class Class2 extends Class1
{
    public Class1 method ()
    {
        Class1 abc = new Class1();
        return (abc);
    }
}
```

```
package MainClass;
import AssignPkg.Assignment_7.pkg1.Interface1;
import AssignPkg.Assignment_7.pkg3.Class2;
```

```
public class Main
{
    public static void main (String [] args)
    {
        Interface1 objInt;
        Class2 ab = new Class2 ();
        objInt = ab.getObj();
        objInt = ab.method ();
    }
}
```



```
run:
Inside the Class1 constructor
Inside the Class2 constructor
Inside the Class1 constructor
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name : Suman Byapari
Roll : DBPCCSTS5
No : 10004556
Registration No : D192004263
Assignment : Java Lab Assignment

B.P.C Institute Of Technology

Name : Suman Byapari

Roll : DBPCCSTS5

No : 10004556

Registration No : D192004263

Assignment : Java Lab Assignment