Quadratic Equation

Input:

```
import java.util.Scanner;
class Quad;
int a,b,c;
double root1, root2, d;
Scanner sc= new Scanner (System.in);
void input (){
 System.out.println(" Quadratic equation is in the
       form : ax^2 +bx +c ");
 System.out.print(" Enter a, b, c: ");
   a= sc.nextInt ();
   b= sc.nextInt ();
   c= sc.nextInt ();
}
void discriminant () {
    d= (b*b)-(4*a*c);
}
void calculateRoots () {
  If ( d>0)
{
 System.out.println ("Roots are real and unequal");
 root1 = (-b + Math.sqrt(d)) /(2*a);
 root2 = (-b + Math.sqrt(d)) / (2*a);
 System.out.println( "First root is : "+root1);
 System.out.println( "Second root is : "+root2);
} else if ( d==0)
{
```

```
System.out.println("Roots are equal and real");
root1 = (-b + Math.sqrt(d)) / (2*a);
System.out.println("Root: " + root1);
}
else
{
System.out.println("No real solutions. Roots are
    imaginary");
double real = -b/(2*a);

double imaginary = Math.sqrt(-d)/(2*a);

System.out.println("Eq has complex roots."
        + real + " + " + imaginary +
        "i and ");
}
}
}

class Quadratic-Eq {

public static void main(String[] args)
{
    Quad q = new Quad();
    q.input();
    q.discriminant();
    q.calculateRoots();
}
}
```

Output
Quadratic equation is in the form $ax^2 + bx + c$

a: 1
b: 5
c: 6
Roots are real and equal
First root: -2.0
Second root: 3.0.

Overload

```
class Overload {

    void print (int n) {
    int sum = 0;
    for ( int i=1 ; i<=n ; i++ ) {
        sum += i;
    }
    System.out.print( sum );
}

void print (int n, int m) {
    System.out.print ( "Prime numbers are ");

    for(int i=n; i<=m ; i++) {
        int flag = 0;
        for(int j = 2 ; j <= n/2 ; j++){
            if( i%j == 0) {
                flag = 1;
                break;
            }
        }
    }
```

```
if (flag == 0) {
System.out.print(" " +i);
}
}
}
}

class OverloadDemo {

    public static void main(String a[])
    {
        Overload O = new Overload ()
        O.part(5);
        O.print(7, 13);
    }
}
```

3) Grocery

```
Class Grocery {
    String c_name;
    String c_ph;
    double total;
    Grocery (String c_name, c_ph) {
        this.c_name = c_name
        this.c_ph = c_ph;
    }

    void cal (double q_dal, double q_plus,
              double q_sugar)
    {
        total = q_dal * 100 + q_plus * 80
              + q_sugar * 60;
    }

    System.out.println (total);
}

Class Demo {

    public static void main (String args[]) {

        Grocery g1 = new grocery ( );
        g1.cal (2,2,1);
    }
}
```

12/1/24

8 Book program

```
import java.util.Scanner;

class Books {
    Scanner sc = new Scanner (System.in);
    String name;
    String author;
    int price;
    int numPages;

Books ( String name, String author, int price,
    int numpages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
}

public String toString() {

    String bookName = "bookName : " +
                this.name + "\n";
    String authorName = "Author Name : " +
                this.author + "\n";
    String bookPrice = "price" : "+ this.price+"\n"

    String pages = "Number of pages : " +
                this.numPages + "\n";

    return bookname + authorname + bookprice
                + pages;
}
}
```

Class BookProgram {

public static void main (String args[]){

Scanner s= new Scanner(System.in);
int n;
System.out.println("Enter no. of books");
n= s.nextInt();
Books b[]= new Books[n];

for (int i=0; i<n ; i++) {

System.out.println("\nDetails of book" + (i+1) + ":");

System.out.println("Enter the name of Book:- ");
String name= s.next();
System.out.println("Enter the name of Author:- ");
String author = s.next();
System.out.println("Enter price:-");
int price = s.nextInt();
System.out.println("Enter the no. of pages:- ");
int numPages = s.nextInt();
b[i]= new books(name, author, price, numpages);

}

```
System.out.println("\n Details of all books: ");

for(Books book : b){

    System.out.println(book.toString());

}

}
```

Output

```
Enter the number of Books:
1

Details of Book-1:
Enter the name of Book:-
JungleBook
Enter the name of Author:-
xyz
Enter the Price:-
200
Enter the number of Pages:-
400

Details of all Books:
Book Name: JungleBook
Author Name: xyz
Price: 200
Number of pages: 500
```

8. Student Program

```
import java.util.Scanner;

class Student {
    String USN;
    String name;
    int[] marks = new int[6];

    void acceptDetails() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter USN: ");
        USN = scanner.next();

        System.out.println("Enter Name: ");
        name = scanner.next();

        System.out.println("Enter marks for 6
                subjects:");

        for(int i=0; i<6; i++) {
            System.out.print("Subject " + (i+1)
                    + ": ");
            marks[i] = sc.nextInt();
        }

}
```

```
double calculatePercentage() {
    int totalMarks = 0;
    for (int mark : marks)
    {
        totalMarks += mark;
    }
    return (double) totalmarks/6;
}
void displayDetails() {
    System.out.println("student details : ");
    System.out.println("USN : " + USN);
    System.out.println("Name : " + name);
    System.out.println("Percentage : " +
                calculatePercentage() + "%");
    }
}

public class Student_D {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("\n Enter details for
            student " + (i+1) + ";");
        Students[i].acceptDetails();
    }
```

```
System.out.println("\n Details of all
                    students :");
for (student student : students) {
        student.displayDetails();
        System.out.println();
    }
}
```

Output:

```
Enter the number of students: 1 :

Enter details for student 1:
Enter USN: 123
Enter Name : abc
Enter marks for 6 subjects:
Subject 1: 80
Subject 2: 36
Subject 3: 49
Subject 4: 55
Subject 5: 75
Subject 6: 60

Details of all student
Student Details:
  USN: 123
  Name : abc
  Percentage : 59.1666
```

19/1/24

① Calculating Area of Rectangle, Circle, Triangle

Program:

```
abstract class shape {

        int length;
        int width;

        public Shape(int length, int width){

        this.length = length;
        this.width = width;
        }

        public abstract void printArea();
}

class Rectangle extends Shape {
        public Rectangle(int length, int width){
                super(length, width);
        }

        public void printArea() {
                int area = length* width;
                System.out.println("Rectagle Area: "+
        }

}
```

```
class Triangle extends Shape {
    public Triangle(int length, int width) {
        super(length, width);
    }
    public void printArea() {
        double area = 0.5 * length * width;
        System.out.println("Triangle area: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
    public void printArea() {
        double area = Math.PI * radius * radius;
        System.out.println("Circle Area: " + area);
    }
}

public class Area {
    public static void main(String[] args) {
        Rectangle R = new Rectangle(5, 18);
        R.printArea();
        Triangle T = new Triangle(5, 10);
        T.printArea();
        Circle C = new Circle(6);
        C.printArea();
    }
}
```

Output :

Rectangle Area : 50.
Triangle Area : 16.0
Circle Area : 113.0973355.

(2) Bank program :

```
import java.util.*;

class Amount {
    String customerName;
    int accountNumber;
    String accountType;
    Double balance;

    public Account (String customerName, int
    accountNumber, String accountType, double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }

    public void displayBalance() {
        System.out.println("Account Balance : $" + this.balance);
    }

    public void deposit (double amount) {
        balance += amount;
        System.out.println("Deposit is successfull");
    }
```

```
public void withdrawal ( double amount ) {
    if ( balance >= amount ) {
        balance -= amount ;
        System.out.println("Withdrawal is
                successfull ");
    } else {
        System.out.println ("Insufficient balance");
    }
}

class CurrAcct extends Account {
    double minBalance;
    double serviceCharge;

public CurrAcct ( String CustomerName, int AccountNumber,
    double balance, double minBalance, double serviceCharge)
{
    super ( customerName, accountNumber, "Current Account",
            balance) ;
    this.minBalance = minBalance ;
    this.serviceCharge = serviceCharge ;
}

public void withdrawal ( double Amount ) {
    if ( balance - amount >= minBalance)
        super.withdrawal (amount) ;
    } else {
        System.out.println ("Insufficient Balance");
    }
}
```

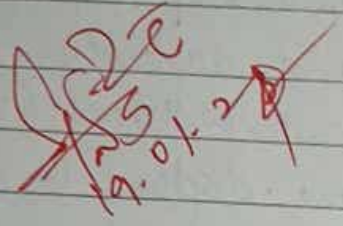
```
public void ServiceCharge() {
        if(balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Service Charge imposed");
        }
    }
}

Class SavingAcct extends Account {
    double interestRate;
    public SavingAcct(String customerName, int
    accountNumber, double balance, double
    interestRate) {
        super(customerName, accountNumber, "Saving
            account", balance);
        this.interestRate = interestRate;
    }

    public void compInterest() {
        double interest = balance * interestRate/100;
        balance += interest;
        System.out.println("Interest computed and
                            deposited");
    }
}
```

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

        CurrAcct currentAccount = new CurrAcct
            ("Sidhan", 12345, 1000, 500, 10);
        currentAccount.deposit(200);
        currentAccount.displayBalance();
        currentAccount.withdrawal(300);
        currentAccount.displayBalance();
        currentAccount.withdrawal(800);
        currentAccount.displayBalance();
        currentAccount.imposeServiceCharge();
        currentAccount.displayBalance();

        System.out.println();

        SavingAcct S = new SavingAcct("Sidd",
            9765, 5000, 5);
        S.computeInterest();
        S.displayBalance();
        S.withdrawal(1000);
        S.displayBalance();
    }
}
```

Output :

Deposit of 200 successfull
Account Balance = 1200.
Withdrawal of 300 successful.
Account Balance: 900
Withdrawal Insufficient Balance

Amount Balance : 5000
Interest computed and deposited : 250
Account Balance : 5250
Withdrawal of 1000 successful
Account Balance : 4250



① Exception Handling

```
Class WrongAge extends Exception {
    public WrongAge() {
        super("Age cannot be negative");
    }
    public WrongAge(String message) {
        super(message);
    }
}

class Father {
    int age;
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge();
        }
        this.age = age
    }
}
```

```
class Son extends Father {
    int son_age;
    public Son(int fatherage, int son_age)
                throws WrongAge {
        super(FatherAge);
        if(son_age >= FatherAge){
            throw new WrongAge("son's age
                cannot be greater than father's");
        this.sonAge = son_age;
    }

}

public class ExceptionHandling {

    public static void main(String args[]){

        try {
            Father father = new Father(50);
            Son son = new Son(50, 25);
        }
        catch(WrongAge e) {
            System.out.println("Ex caught:" + e.getMessage
                age());
        }
    }
}
```

Output: FatherAge = -50

Exception caught: Age cannot be negative

FatherAge = 50 & SonAge = 55

Exception caught: Son Age cannot be greater than
father Age.

(2)

Threads

```
class BMS extends Thread {
    public void run() {
        while (true) {
            try {
                System.out.println("BMSCE");
                Thread.sleep(10000);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

class CSE extends Thread {
    public void run() {
        while (true) {
            try {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
            catch (InterruptedException e) {
                e.PrintStackTrace();
            }
        }
    }
}
```

(2)

Threads

```
class BMS extends Thread {
    public void run() {
        while (true) {
            try {
                System.out.println("BMS CE");
                Thread.sleep(10000);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

class CSE extends Thread {
    public void run() {
        while (true) {
            try {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
            catch (InterruptedException e) {
                e.PrintStackTrace();
            }
        }
    }
}
```

```
public class Thread1 {

    public static void main (String args[]){
        Thread t1 = new BMS();
        Thread t2 = new CSE();
        t1.start();
        t2.start();
    }
}
```

Output:
CSE
CSE
~~BMS~~ CSE
CSE
~~BMSCE~~

(3) Packages:

CIE:

Internals.java

```
package CIE;

public class Internals extends Student {
    protected int[] internalmarks;

    public Internals( String usn, String name, int sem,
                      int[] internalmarks ){
        super(usn, name, sem);
        this.internalmarks = internalmarks;
    }
}
```

Student.java

```
package CIE;
public class student {
```

```
protected String usn;
protected string name;
protected int sem;

public student ( string usn, string name, int sem
    this.usn = usn;
    this.name = name;
    this.sem = sem;
}
}
```

SEE

```
package SEE;

import CIE.student;

public class External extends Student {
    public External (String usn, String name, int sem
        super (usn, name, sem);
    {
    this.externalMarks = externalMarks;
}
```

Main.java

```
import CIE.Internals;
import SEE.Externals;
```

```
public class Main {

    public static void main (String [] a ){

        int [] internalMarks = { 80, 75, 85, 90, 70};
        Internals internals = new Internals ("123", "xyz",
                                3, 3, internalMarks);

        int [] externalMarks = { 75, 70, 80, 85, 65};
        Externals external = new External ("567",
                                "abc", 3,
                                externalMarks);

    int finalMarks [] = calculateFM (internals.internalMarks,
                                external.externalMarks);

    int [] finalMarks
        display (FinalMarks) ;

    }
    public static int [] calculateFM (int [] internals,
                                int [] externals ) {
        int [] finalMarks = new int[internals.length];

        for ( i=0; i < 5; i++) {
            finalMarks[i] = internal[i] + externals[i];
        }
        return finalMarks;

    }
```

```
public static void display (int[] finalMarks)
    System.out.println("Final Marks : ");

    for(i=0 ; i<5 ; i++){
        System.out.println("Course "+(i+1)+" : "+
                finalMarks[i]/2);
    }
}
}
```

Output

Final Marks

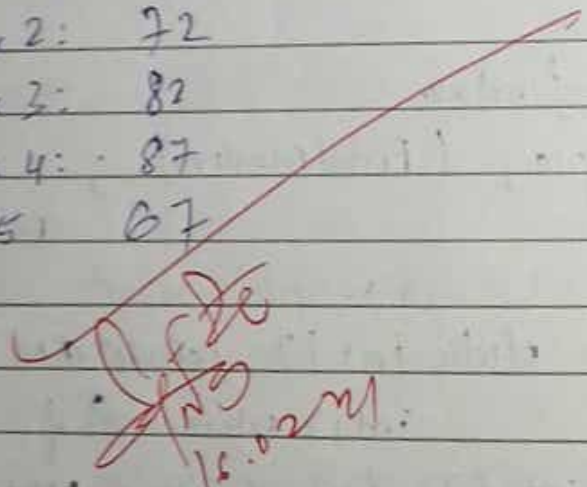
Course 1: 77

Course 2: 72

Course 3: 82

Course 4: 87

Course 5: 67

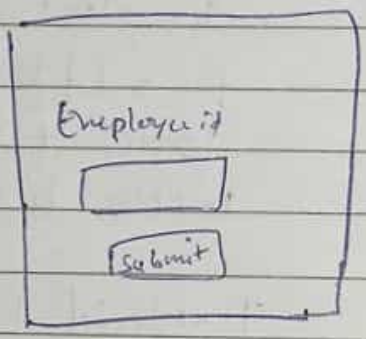


1 Create label, button and text field

```
import java.awt.*;
import java.awt.event.*;

public class AWTExample extends WindowAdapter {

    Frame f;
    AWTExample (){

        f = new Frame ();
        f.addWindowListener (this);
        Label l = new Label ("Employee id:");
        Button b = new Button ("submit");

        TextField t = new TextField ();

        l.setBounds (20, 80, 80, 30);
        t.setBounds (20, 100, 80, 30);
        b.setBounds (100, 100, 80, 30);

        f.add (b);
        f.add (l);
        f.add (t);
        f.setSize (400, 300);

        f.setTitle ("Employee info");
        f.setTitle (
        f.setLayout (null);
        f.setVisible (true);
    }
```

```
public void windowClosing(WindowEvent e){
        System.exit(0);
}

public static void main(String[] args){

        AWTExample awt obj= new AWTExample();
}

}
```

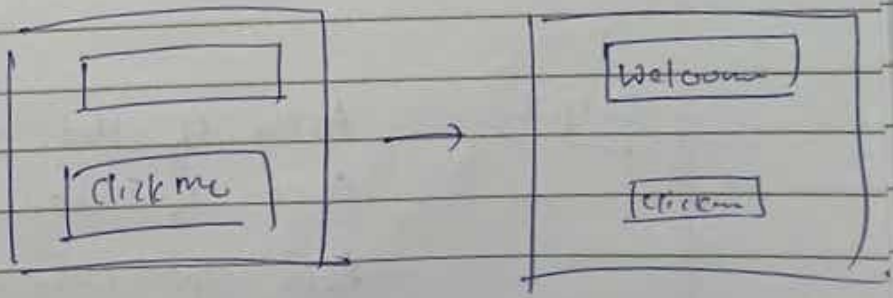
Output:



2. Create button & add action listener for Menu cli

```
import java.awt.*;
import java.awt.event.*;

public class EventHandling extends WindowAdapter
implements ActionListener {

    Frame f;
    TextField tf;
    EventHandling(){
        f= new Frame();
        f.addWindowListener(this);
```

```
tf = new Textfield ();
tf.setBounds (60,50, 170,20);
Button b= new button ("click me");
b.setBounds (100,120, 80, 30);

b.addActionListener (this);

f.add(b);
f.add(tf);
f.setLayout(null);
f.setVisible(true);

}
public void actionPerformed (ActionEvent d){
    tf.setText("welcome");
}

public void window(closing(WindowEvent e){
        System.exit(0);
}
public static void main (String args()){
    new OvalHandling();
}

}
```



Programs on IO

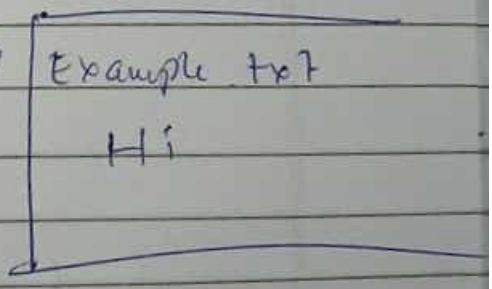
Example 1:

```
import java.io.*;
public class ByteArrayInput {
    public static void main(String args[]) throws
                                    IOException {

    byte[] buf = { 35, 36, 37, 38};

    ByteArrayInputStream byt= new ByteArrayInputStream
                                    (buf);
    int k = 0;

    while ((k = byt.read()) != -1) {

        char ch =(char) k;
        System.out.println("ASCII value of
        character is: " +k+ "; Special
        character is: " +ch);

    }

}
}
```

Output: Ascii of char : 35 : Special char #
Ascii of char : 36 Special char $
Ascii of char 37 special char %
Ascii of char 38 special char &

2. Example 2.

```
public class FileEx {
    public static void main (String a[]) throws
    IOException {

    FileInputStream fin = new FileInputStream ("example.txt");

    int content;
    System.out.println("Remaining bytes : " +
                        fin.available());
    content = fin.read();
            System.out.print ((char) content + " ");
            System.out.print (content + " ");
            System.out.println("Remaining bytes " +
                        fin.available());

    System.out.println("Remaining bytes that
        can be read : " + fin.available());
    }
}
```



* Output:

Remaining bytes that can be read: 2

h
104
i
105

Remaining bytes that can be read: 0

Example - 4:

```
import java.io.FileInputStream;
import java.io.IOException;

public class FileEx2 {
    public static void main(String a[]) throws
                                    IOException {

FileInputStream fin = new FileInputStream("Exam

byte [] bytes = new byte[20];
int i;
char c;

i = fin.read(bytes);
System.out.println("No. of bytes read; "+

for (byte b: bytes) {

        c = (char) b;

        System.out.print(c);

    }

}
```

Output :

Number of byte read : 2
Bytes read : hi