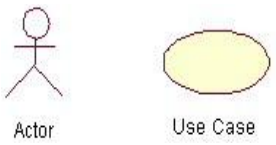
Program – 1:

Draw an Use Case Diagram for Airline Reservation system using Star UML Tool

Aim: To draw an Use Case diagram for Airline Reservation system using Star UML tool.

Description: A Use Case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases.

The two main components of a use case diagram are use cases and actors.



- An **Actor** is represents a user or another system that will interact with the system you are modelling.
- A **Use case** is an external view of the system that represents some action the user might perform in order to complete a task.
- Use case diagrams specify the events of a system and their flows. But use case diagram never describes how they are implemented.
- Use case diagram can be imagined as a black box where only the input, output, and the function of the black box is known.

USECASE NAME:- **AIRLINE TICKET RESERVATION SYSTEM**

PURPOSE:-

The purpose of developing this use case is to provide the customer with the facilities to enquire for a flight to know the availability of seats by using the flight number, flight date and flight time of journey and to reserve the tickets for the given flight at given date & time for specified number of persons. The customer should also have the possibility of canceling tickets for an booked flight ticket.

USERS:-

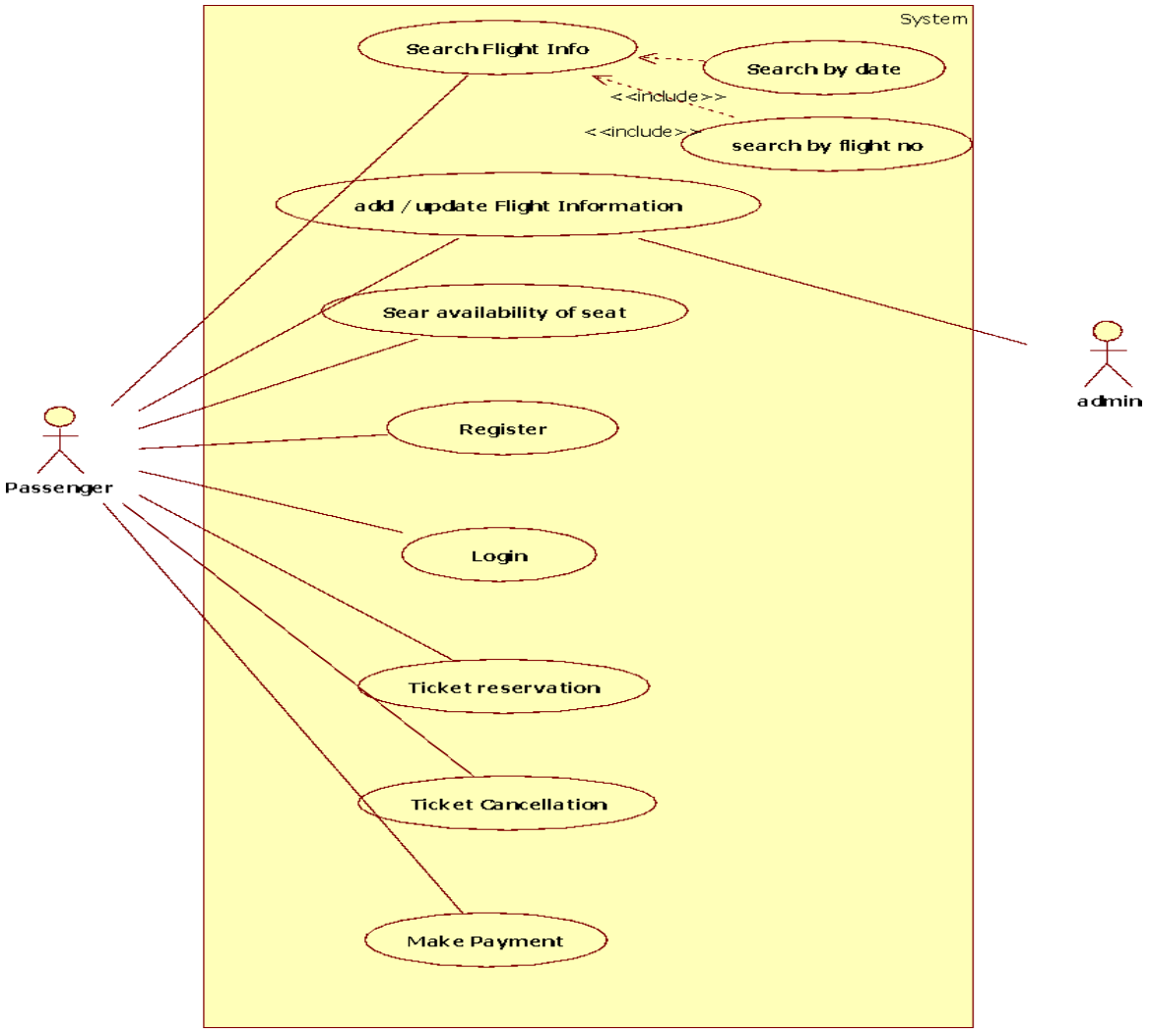
1) Primary Users ☮Customer
2) Secondary Users ☮
   1. Admin
   2. Database
   3. Interface

DESCRIPTION:-

1) The customer enters into the website of flight reservation system & should go to the search menu given the flight number as input & search for the availability of seats as input and search for availability of seats as input and search for availability of status on that specified airplane.
2) If there is an availability of seats as specified in the search criteria then the customer has to register himself with the airline reservation system by providing the necessary details like name, address, phone number, email, address proof, Identify proof.
3) After the reservation completed the customer has to login to the system by given the valid username and password.
4) One's login is accepted the system then customer can go for reservation by click on reservation option where has to provide to information like his name, data of journey, starting station, source & destination, number of males, females, children's, total passengers, category etc….,
5) Upon the completion of the reservation the customer should preview the reservation details & confirm to reservation.
6) For any reason the customer wishers to cancel the reservation seats then he has to choose the cancellation option. In which he has to mention the Reservation ID (or) Ticket ID.
7) For reserved the payment can be made through online either using credit, debit or online payment.
8) If the step1 is not satisfied then the customer can choose for different flight or different date.
9) Finally after completing required transactions the customer can logout the system.

Pre Conditions:
1. Any Customer to reserve a ticket he should be the registered user of the system. 2. Booking of seats are allowed if only seats are vacant. 3. The flight details and reservations should be stored in the database and should be made accessible to the passengers at any point of time.
Post Conditions:
1. The passenger should get the acknowledgement of the receipt of payment and should get the ticket for the reservation he/she made. 2. All the transactions of the system be easily accessible by the users.

Output:



Program -2

Draw an Activity Diagram for Online Book Store System

Aim: To draw an Activity diagram for Online Books Stores System using Star UML

Description:

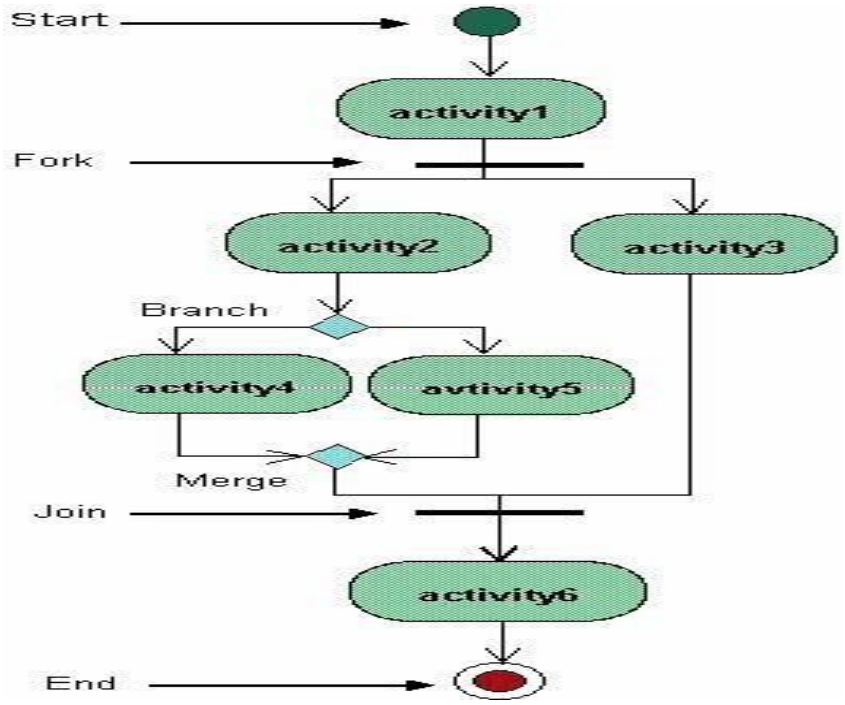
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

Purpose of Activity Diagrams

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques**.**



- Start Symbol Indicates the Starting point of theActivity
- End Symbol indicates end of the Process
- Fork is used when multiple activities are occurring atsame time.
- Branch indicates that based on the condition whichactivity has to be chosen.
- Join is used to combine all the parallel activities beforetransitioning into final state.

When to use Activity Diagram?

The main reason to use activity diagrams is to

- **Model the workflow behind the system being designed.**

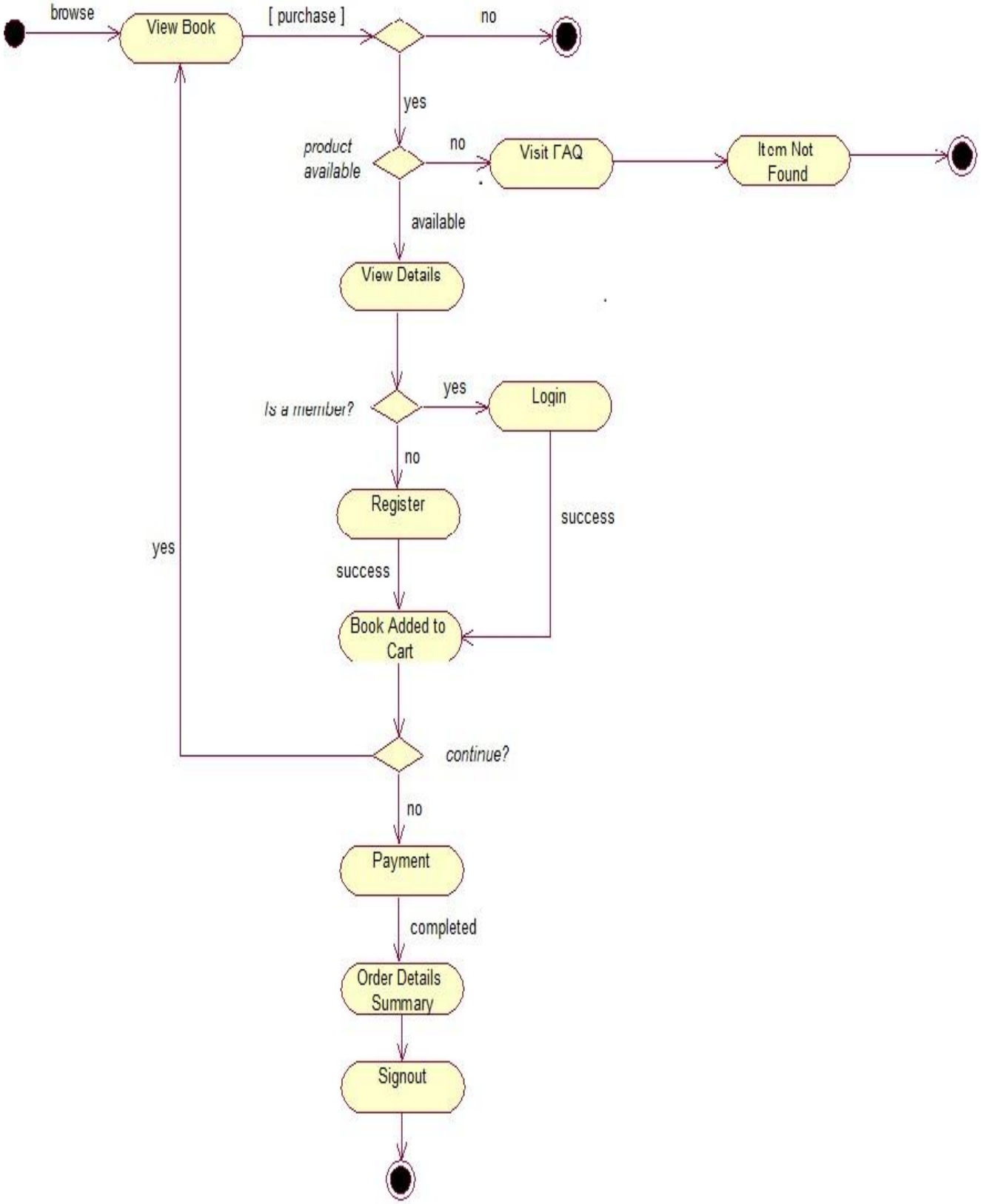
Activity Diagrams are also useful for:

(i) analyzing a use case by describing what actions needto take place and when they should occur.

(ii) describing a complicated sequential algorithm.

(iii) and modelling applications with parallel processes.

Output:


browse
View Book
[purchase]
no
yes
product available
no
Visit FAQ
Item Not Found
available
View Details
Is a member?
yes
Login
no
Register
success
yes
success
Book Added to Cart
continue?
no
Payment
completed
Order Details Summary
Signout

Program-3

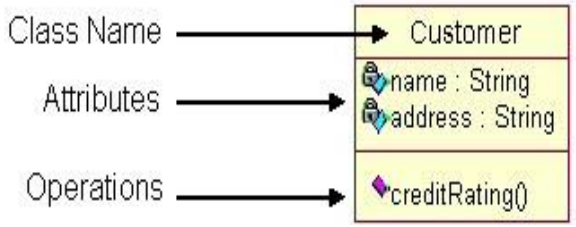
Draw a Class Diagram for Library Management System using STAR ULM

Aim: To draw a Class Diagram for Library Management System using STAR UML

Description:

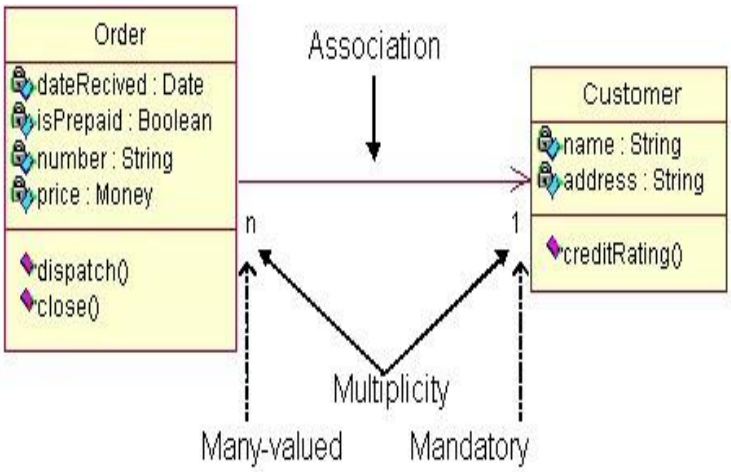
Class diagrams are widely used to describe the types of objects in a system and their relationships. Class diagrams model class structure and contents using design elements such as classes, packages and objects.

Classes are composed of three things: a name, attributes, and operations.
Below is an example of a class:



Class diagrams also display relationships such as containment, inheritance, associations Generalization etc.

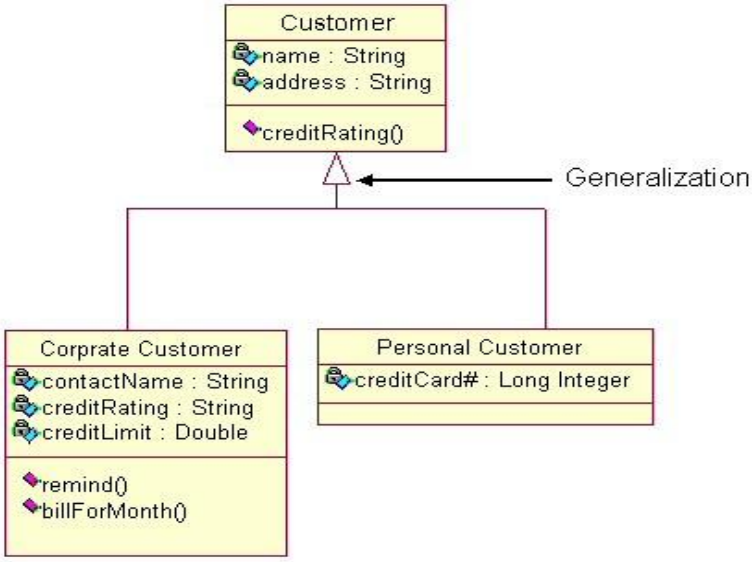
Below is an example of an **associative relationship**:



The class Order is associated with the class Customer. The multiplicity of the association denotes the number of objects that can participate in the relationship. Another common relationship in class diagrams is a generalization.

A generalization is used when two classes are similar, but have some differences.

Below is an example of an **Generalization relationship**:



Classes Corporate Customer and Personal Customer have some similarities such as name and address, but each class has some of its own attributes and operations.

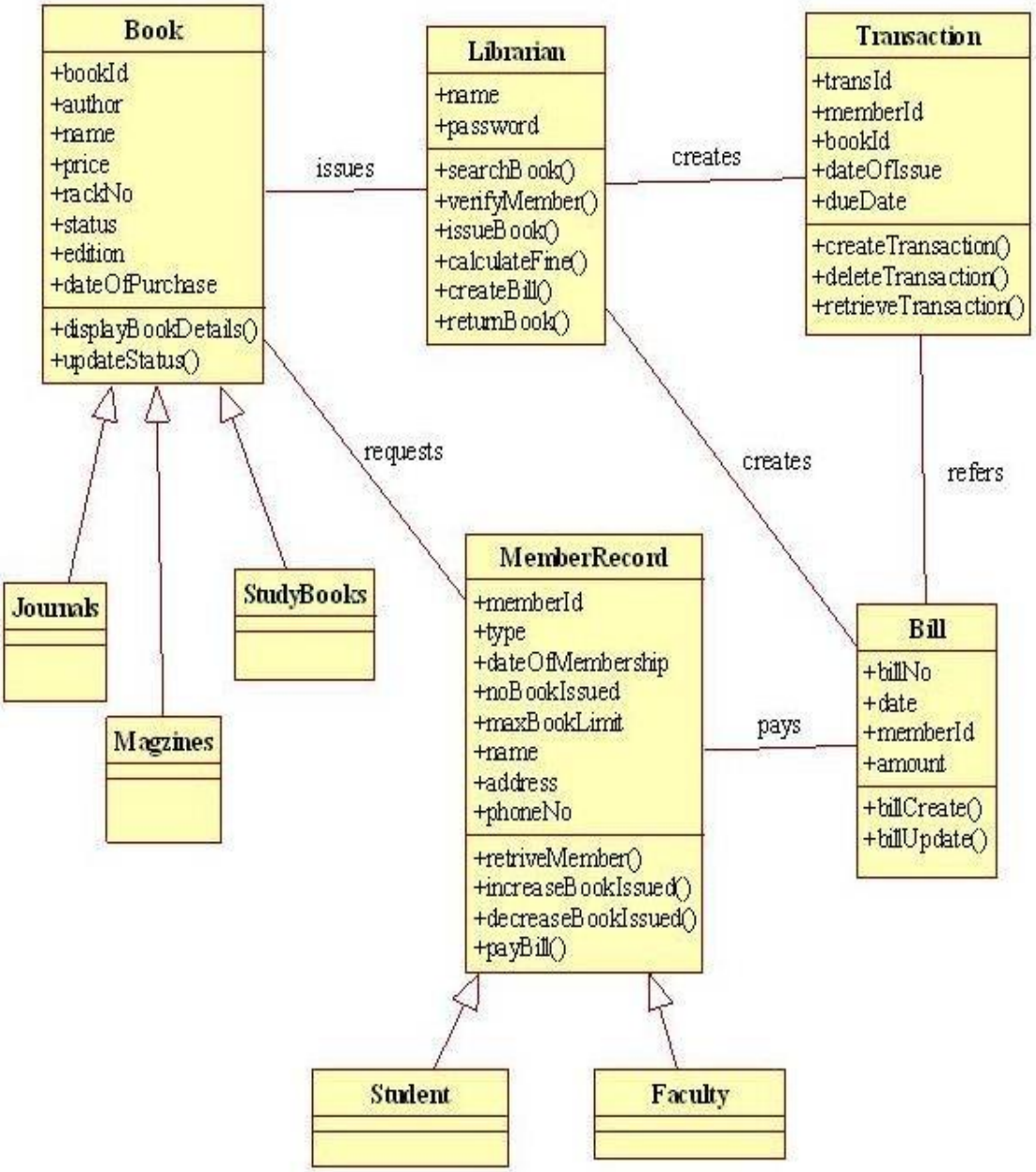
The class Customer is a general form of both the Corporate Customer and Personal Customer classes.

This allows the designers to just use the Customer class for modules and do not require in-depth representation of each type of customer.

When to Use: Class Diagrams

Class diagrams are used in nearly all Object Oriented software designs. Use them to describe the Classes of the system and their relationships to each other.

Output:


Book
+bookId
+author
+name
+price
+rackNo
+status
+edition
+dateOfPurchase
+displayBookDetails()
+updateStatus()
Librarian
+name
+password
+searchBook()
+verifyMember()
+issueBook()
+calculateFine()
+createBill()
+returnBook()
Transaction
+transId
+memberId
+bookId
+dateOfIssue
+dueDate
+createTransaction()
+deleteTransaction()
+retrieveTransaction()
issues
creates
requests
creates
refers
Journals
StudyBooks
Magzines
MemberRecord
+memberId
+type
+dateOfMembership
+noBookIssued
+maxBookLimit
+name
+address
+phoneNo
+retriveMember()
+increaseBookIssued()
+decreaseBookIssued()
+payBill()
Bill
+billNo
+date
+memberId
+amount
+billCreate()
+billUpdate()
pays
Student
Faculty

Program-4

Draw a State Chart diagram for Railway Reservation System using STAR ULM

Aim: To draw a State Chart diagram for Railway Reservation System using STAR UML.

Description:

- A State-chart diagram describes a state machine.
- A State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.
- Activity diagram is a special kind of a State-chart diagram. As State-chart diagram defines the states, it is used to model the lifetime of an object.
- State-chart diagram describes the flow of control from one state to another state.
- States are defined as a condition in which an object exists and it changes when some event is triggered.
- The most important purpose of State-chart diagram is to model lifetime of an object from creation to termination.

Following are the main purposes of using State-chart diagrams:

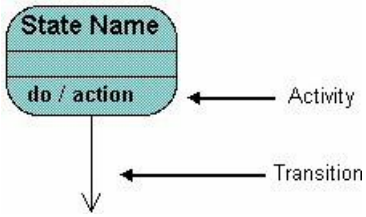
- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

Before drawing a State-chart diagram we should clarify the following points :

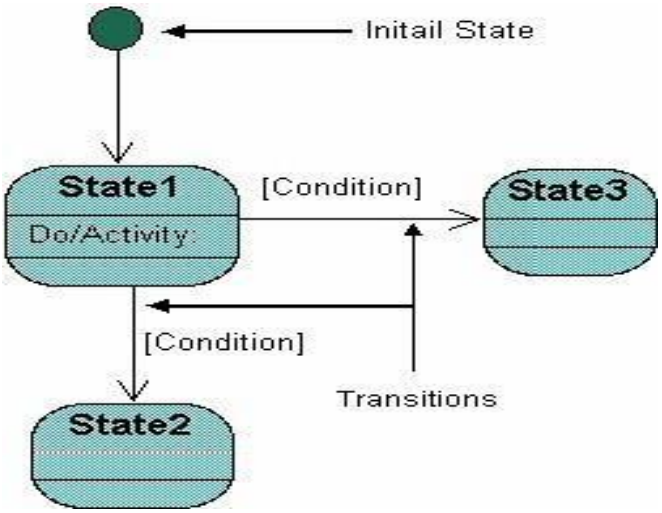
- Identify the important objects to be analysed.
- Identify the states.
- Identify the events.

Basic Elements on State Chart Diagram are :

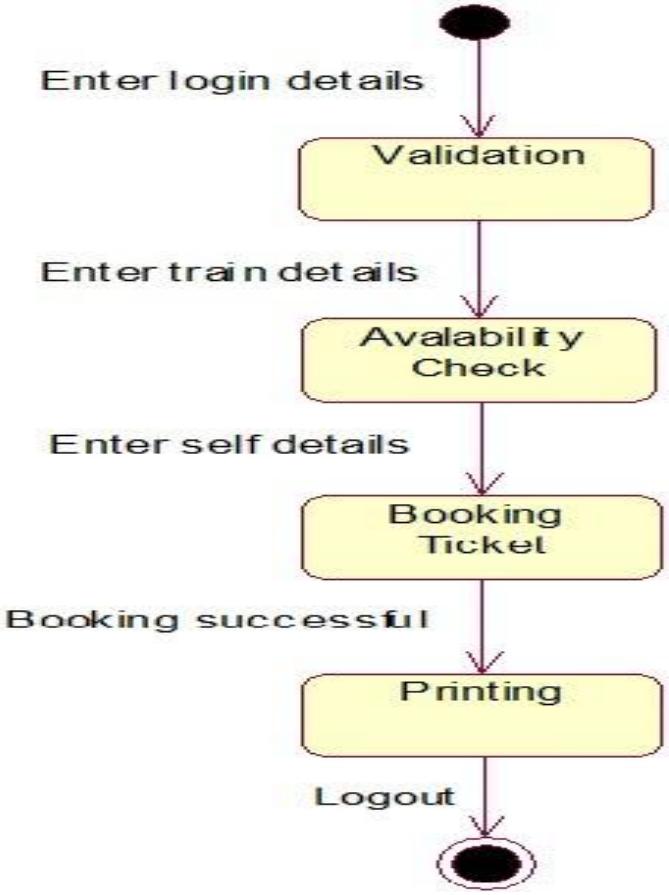
1. Rounded Boxes representing the state of the objects.
2. Arrow indicating the Transition of Next State.



- All state diagrams being with an initial state of the object. This is the state of the object when it is created. After the initial state the object begins changing states.
- Conditions based on the activities can determine what the next state the object transitions to.



Output


Enter login details
Validation
Enter train details
Avalability Check
Enter self details
Booking Tickel
Booking successful
Printing
Logout

Program-5

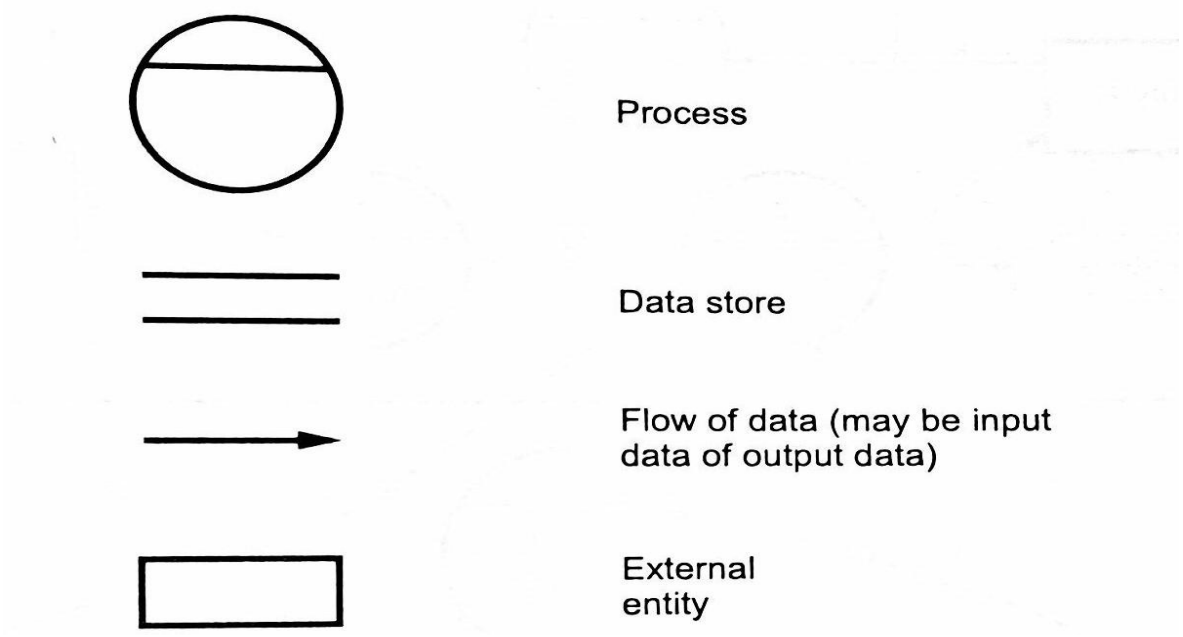
Draw a Zero & First Level Data flow Diagram's for Bank Loan System

Aim: To draw Zero & First Level DFD's for Bank Loan Process

Description:

The Data Flow Model depicts the information flow and the transforms that are applied on the data as it moves from input to output.

The symbols that are used in Data Flow Diagrams are:



Guidelines to Design DFD's:

1. Level Zero DFD should depict the system as a single Bubble.

2. Primary Input & Primary Output should be carefully identified.

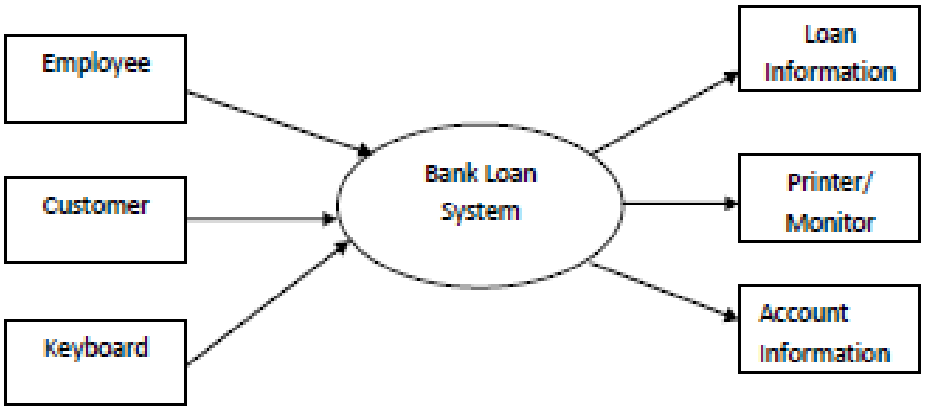
3. All the Bubbles (Processes) should be appropriately named.

4. One Bubble at a time should be refined.

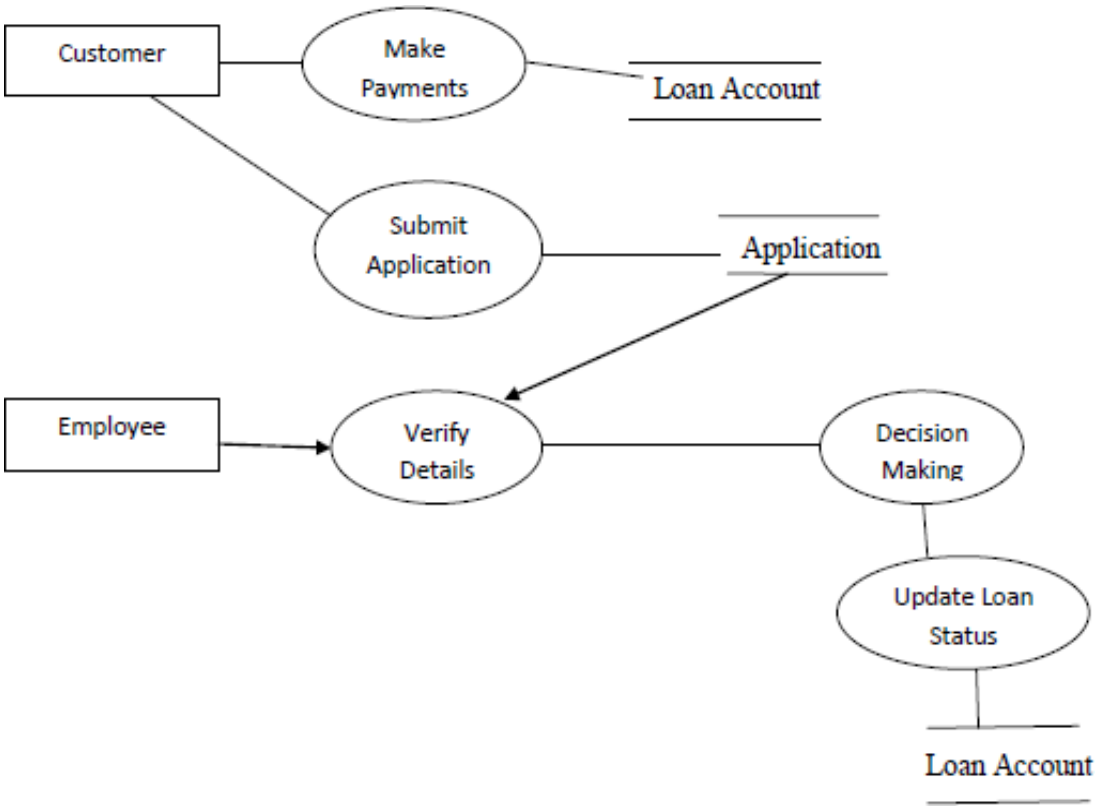
5. Information flow continuity must be maintained from level to level.

Outputs:

1. 0 Level DFD



2. 1'st Level DFD



Program-6

Write the Test Cases for Login & Book Entry Form in Library Management System?

Aim: To write the Test Cases for Login & Book Entry Form in Library management System.

Description:

What is the Test Case?

A Test Case is a set of actions executed to verify a particular feature or functionality of your software application. The Test Case has a set test data, precondition, certain expected and actual results developed for specific test scenario to verify any requirement.

A test case includes specific variables or conditions, using which a test engineer can determine as to whether a software product is functioning as per the requirements of the client or the customer.

Why do we write Test Cases?

Here, are some important reasons to create a Test Case-

- Test cases help to verify conformance to applicable standards, guidelines and customer requirements.
- Helps you to validate expectations and customer requirements.
- Increased control, logic, and data flow coverage.
- You can simulate 'real' end user scenarios.
- Exposes errors or defects.
- When test cases are written for test execution, the test engineer's work will be organized better and simplified.

Example:-Test cases for the **Test Scenario:** "Check the Login Functionality" would be

1. Check system behaviour when **valid email id and Valid password** is entered.

2. Check system behaviour when **invalid email id and valid password** is entered.
3. Check system behaviour when **valid email id and invalid password** is entered.
4. Check system behaviour when **invalid email id and invalid password** is entered.
5. Check system behaviour when **email id and password are left blank and Sign in** entered.
6. Check **Forgot your password** is working as expected
7. Check system behaviour when **valid/invalid phone number and password** is entered.
8. Check system behaviour when "**Keep me signed**" is checked.

Test Cases for Library Management System

LOGIN FORM:

SL.No	Test Case	Excepted Result	Test Result
1.	Enter valid user name and password & click on login button.	System should display main Window	**Pass**
2.	Enter invalid name & password & click on Login Button.	System should display an error message and should not display Main window.	**Pass**

BOOK ENTRY FORM:

SL. No	Test Case	Excepted Result	Test Result
1.	On the click of ADD	At first user have to fill all fields with proper data , if any Error like entering the Text data instead of numbers or entering numbers instead of Text is found then give proper error message, otherwise Add Record to the Database.	Successful
2.	On the Click of Delete Button	This Deletes the details of the book using Access number.	Successful
3.	On Click of Update Button	Modified Records are Updated in Database	Successful
4.	On Clicking Search Button	Display the details of the Books entered along with Access Number. Otherwise Display a proper error message	Successful

5.	On Clicking Clear Button	Clear all the field's in the Form	Successful
6.	On Clicking EXIT Button	Exit From Current Book Details Form	Successful
7.	On Clicking NEXT Button	Display corresponding Next Form	Successful

Program-7

Write the Test Scenarios for Hospital Application System.

Aim: To write Test Scenarios fro Hospital Application System

Description:

What is a Test Scenario?

A Test Scenario is defined as any functionality that can be tested. It is a collective set of test cases which helps the testing team to determine the positive and negative characteristics of the project.

Test Scenario gives a high-level idea of what we need to test.

Why do we write Test Scenario?

Here, are important reasons to create a Test Scenario:

- The main reason to write a test scenario is to verify the complete functionality of the software application
- It also helps you to ensure that the business processes and flows are as per the functional requirements
- Test Scenarios can be approved by various stakeholders like Business Analyst, Developers, Customers to ensure the Application Under Test is thoroughly tested. It ensures that the software is working for the most common use cases.
- They serve as a quick tool to determine the testing work effort and accordingly create a proposal for the client or organize the workforce.
- They help determine the most critical end-to-end transactions or the real use of the software applications.
- Once these Test Scenarios are finalized, test cases can be easily derived from the Test Scenarios.

Example of Test Scenario

For an E-Commerce Application, a few test scenarios would be:

Test Scenario 1: Check the Search Functionality.

Test Scenario 2: Check the Payments Functionality.

Test Scenario 3: Check the Login Functionality

Test Scenarios For HOSPITAL APPLICATION SYSTEM

Name of the Scenario	**Add Patient Entry**
Description	This function get details of a patient and add record to the patient file and generate a patient registration number
Actors	Data Entry Operator, Receptionist
Pre-Condition	The operation must Login with their User Account
Main Flow of events	1. Use Selects “Add Patient Entry” at Home page. 2. Patient entry form is Displayed. 3.User enters the data to the required fields 4. Then User selects “Add Entry Button.” 5. If all the fields are properly entered then a message is Displayed as “Record Added Successfully.” 6. System now generates a Patient ID and display.
Extensions	In 3(a) If any necessary field is left blank by the user then prompt the user to enter all the required fields.
Post Condition	Patient record has to be Added to Patient File

Name of the Scenario	Issue Clinical Number to the Patient / User
Description	This function assign number of Patients assigned to relevant Channelling.
Actors	Receptionist
Pre-Condition	Patient must First Register through System
Main Flow of events	1. User Selects “Generate a Number “at OPD module 2. System prompts to select the clinical type. 3. If OPD generates next available number to the available Doctor and display the number then 4. User confirms the Number and take a print of the Ticket.
Extensions	In 3(a) if channelling a counsellor, system prompts the counselling field and doctor. 3) b) user enter counselling field and doctor. 3) C system generates next available number or required Counsellor.
Post Condition	Patient channelling record should be updated with patient details.

Name of the Scenario	Add Prescription Entry
Description	This function records patients prescription details
Actors	Data entry operator/ Patient
Pre-Condition	Doctor's prescription chit must be issued.
Main Flow of events	1. User Selects the "Perception form" the Patient Module. 2. System Prompt's to enter Patient Registration Number. 3. User enter system registration Number. 4. Prescription form displayed with relevant patient details. 5. User navigates to 'tests' field and selects prescribed test Details. 6. User navigates to 'Vaccine' field and selects prescribed Vaccine details. 7. User Navigates to 'Medicine' field and enters Medicine details. 8. User enter Re-Consultation date and selects 'ADD' Button and add Prescription details. 9.User Selects 'PRINT' Button and Prints the Prescription details.

Name of the Scenario	Patient Diagnosis History
Description	This function adds patient's diagnosis details into the System.
Actors	Patient / Data Entry operator
Pre-Condition	Patient must register to the System
Main Flow of events	1. User Selects Patient Diagnosis Card. 2. System prompts for Patient Id. 3. After entering User Id system display's user details in the Form.

	4. User now enters the Diagnosis details. 5. User Selects 'Add Diagnosis record' Button
Post Condition	The Diagnosis Record must be added to the Diagnosis File.

Name of the Scenario	**Calculate Bill After Consultation**
Description	This Function Total Bill charges for the Patient after the Consultation from the Doctor side is completed.
Actors	Receptionist / Cashier
Pre-Condition	The Receptionist / Cashier must Register to the System.
Main Flow of events	1. Cashier selects Patient Receipt card. 2. System Prompts to Enter Patient registration number. 3. After entering the Registration number by the cashier the system prompts for date & Time for which the Bill has to be Prepared. 5. After the Cashier enters Date & Time system displays patient details, lab tests details, X-ray details, ECG details etc. and total fee in number and in word. 6. Cashier Selects Print Receipt and prints the Bill. 7. The Patient Receipt is Printed Successfully.
Post Conditions	Payment Details has to be Updated in Payment File.

Program-8

Prepare a Data Dictionary for Online Shopping Portal

Aim: To prepare a Data Dictionary for Online Shopping Portal.

Description:

DATA DICTIONARY

A data dictionary contains metadata i.e data about the database.

The data dictionary is very important as it contains information such as what is in the database, who is allowed to access it, where is the database physically stored etc.

The users of the database normally don't interact with the data dictionary; it is only handled by the database administrators.

The data dictionary in general contains information about the following:

1. Names of all the database tables and their schemas.
2. Details about all the tables in the database, such as their owners, their security constraints, when they were created etc.
3. Physical information about the tables such as where they are stored and how.
4. Table constraints such as primary key attributes, foreign key information etc.
5. Information about the database views that is visible.

Different types of data dictionary are:

Active Data Dictionary

If the structure of the database or its specifications changes at any point of time, it should be reflected in the data dictionary. This is the responsibility of the database management system in which the data dictionary resides.

So, the data dictionary is automatically updated by the database management system when any changes are made in the database. This is known as an active data dictionary as it is self-updating.

Passive Data Dictionary

This is not as useful or easy to handle as an active data dictionary. A passive data dictionary is maintained separately to the database whose contents are stored in the dictionary. That means that if the database is modified the database dictionary is not automatically updated as in the case of Active Data Dictionary.

So, the passive data dictionary has to be manually updated to match the database. This needs careful handling or else the database and data dictionary are out of synchronization.

Data Dictionary For Online Stores(Shopping Portal)

Table1: - User Description:

This table store information of User like Name, Address, Contact No, and Email Address. Each User has associated reference in user which stores products belongs to User Product Information belong to User, Products which stores Product Information belonging to User.

Field Name	Data Type	Description	Allow Null
vcID	Varchar(50)	User Name	Primary key
vcPass	Varchar(20)	Password	Not Null
vcFsNm	Varchar(50)	First Name	Not Null
vcLsNm	Varchar(50)	Last Name	Not Null
vcGender	Varchar(6)	Gender	Not Null
vcAdd	Varchar(50)	Address	Not Null
vcCity	Varchar(50)	City	Not Null
vcState	Varchar(50)	State	Not Null
vcZipCode	Varchar(10)	Zip Code	Not Null
vcCnctNo	Varchar(20)	Contact No	Not Null
vcEmailId	Varchar(50)	Email ID	Not Null
vcConPass	Varchar(20)	Conform Password	Not Null

Table2: - Category:

This Table stores Information about Category of each Use

Field Name	Data Type	Description	Allow Null
nmCtgryID	Numeric(3,0)	Category ID	Primary key
vcNm	Varchar(50)	Category Name	Not Null
txtDes	Text	Description	Not Null

Table3: - Product:

This Table stores Information about Product of each User

Field Name	Data Type	Description	Allow Null
nmID	Numeric(4,0)	Product ID	Primary key
vcNm	Varchar(50)	Product Name	Not Null
txtDes	Text	Description	Not Null
nmPrice	Numeric(18,0)	Price	Not Null
nmQuan	Numeric(18,0)	Quantity	Not Null
nmDis	Numeric(18,0)	Discount	Not Null
Image	Image	Product image	Not Null
nmCtgryID	Numeric(3,0)	Category ID	Foreign key

Table4: - Shopping Chart:

This Table Stores Information about Shopping Chart

Field Name	Data Type	Description	Allow Null
nmID	Numeric(4,0)	Product ID	primary key
vcNm	Varchar(50)	Product Name	Not Null
nmQuan	Numeric(18,0)	Quantity	Not Null

Table5: - Admin Table

This Table stores Information about Admin

Field Name	Data Type	Description	Allow Null
vcID	Varchar(50)	Admin ID	Primary key
vcPass	Varchar(20)	Admin Password	Not Null
vcEmailID	Varchar(50)	Admin Email ID	Not Null

Table6: - Order Details:

This Table stores Information about Order of each User

Field Name	Data Type	Description	Allow Null
nmNo	Numeric(4,0)	Order No	Primary key
nmID	Numeric(4,0)	Product ID	Foreign key
nmQuan	Numeric(18,0)	Quantity	Not Null
nmPrice	Numeric(18,0)	Order Price	Not Null
nmAmt	Numeric(18,0)	Total Amount	Not Null

9.Write a java program to demonstrate LINE DDA Algorithm.

Aim : To demonstrate the line DDA Algorithm.

Description: The Digital Differential Analyzer helps us to interpolate the variables on an interval from one point to another point. We can use the digital Differential Analyzer algorithm to perform rasterization on polygons, lines, and triangles.

Procedure:

It stands for digital differential analyzer

Algorithm:

Step 1 :if m<=1

$\Delta x=1$

y-values=?

$m=\Delta y/\Delta x$

$\Delta y=y_2-y_1,\Delta x=x_2-x_1$

$m=y_2-y_1|x_2-x_1$

$m=y_2-y_1|1$

$m=y_2-y_1$

$m=y_{k+1}-y_k$

$y_{k+1}=m+y_k$ ……(1)

Step 2 : if m>1

$\Delta Y=1$

x-values=?

$m=\Delta y|\Delta x$

$m=1|\Delta x$

$\Delta x=1|m$

$X_k+1-x_k = 1|m$

$X_{k+1}=1|m+x_k$ ……..(2)

Step 3 : if m = $\Delta y|\Delta x$ <from –ve slope>

$m =\Delta y|-1$

$m= -\Delta y$

$m=-(y_{k+1}-y_k)$

$m=-y_{k+1}+y_k$

$y_{k+1}=y_k-m$…….(3)

Step 4 : Here $\Delta y=-1$ then

$m=\Delta y|\Delta x$

$m=-1|\Delta x$

$\Delta x=-1|m$

$X_{k+1}=x_k-1|m$……(4)

From eq 1 & 2 we can calculate the pixels for negative slope.

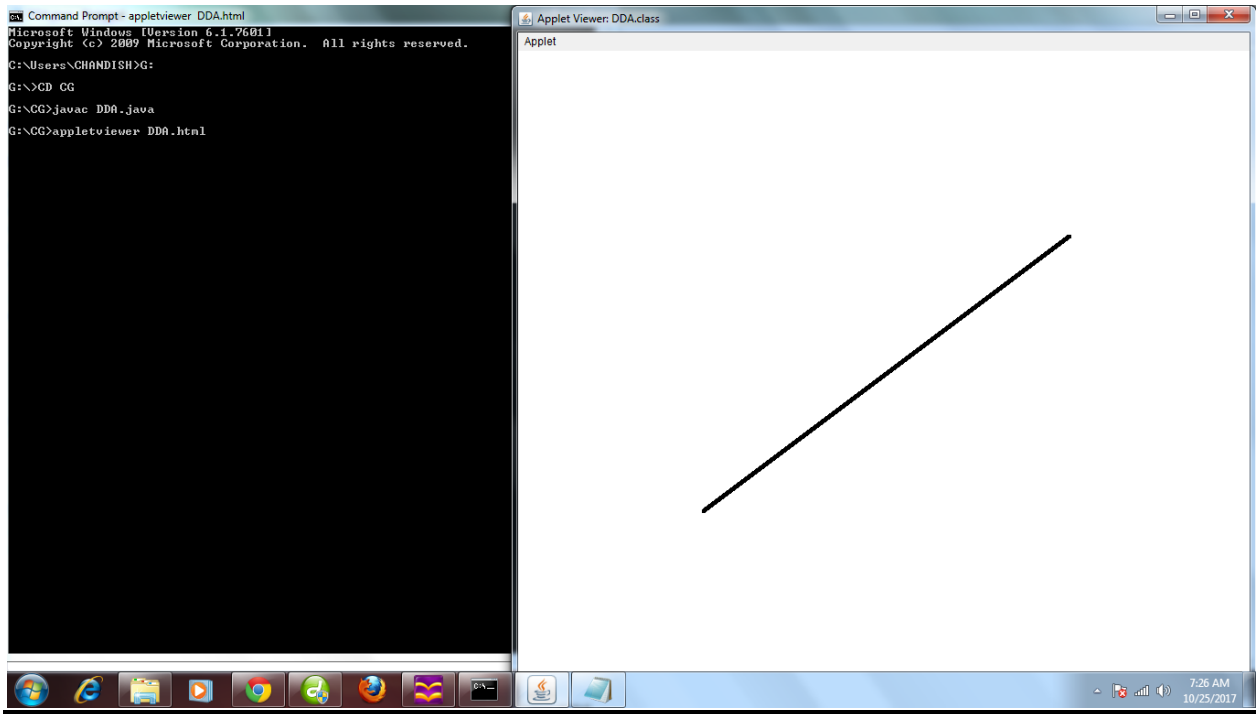
Digital Differential Analyzer (DDA) Algorithm

```
import java.io.*;
import java.awt.*;
import java.applet.*;
import java.util.*;
public class DDA extends Applet
{
public void paint(Graphics g)
{
        double dx,dy,steps,x,y,k;
        double xc,yc;
        double x1=200,y1=500,x2=600,y2=200;
        dx=x2-x1;
        dy=y2-y1;
        if(Math.abs(dx)>Math.abs(dy))
        steps=Math.abs(dx);
        else
        steps=Math.abs(dy);
        xc=(dx/steps);
        yc=(dy/steps);
        x=x1;
        y=y1;
        g.fillOval(200,500,5,5);
        for(k=1;k<=steps;k++)
        {
                x=x+xc;
                y=y+yc;
                g.fillOval((int)x,(int)y,5,5);
        }
}
}
```

DDA.html

```
<html>
<body>
<applet code="DDA.class" width="800" height="800">
</applet>
</body>
</html>
```

OUTPUT:-



10. Write a java program to demonstrate Bresenham's Line Drawing Algorithm

Aim : To demonstrate the Bresenham's Line Drawing Algorithm

Description: This algorithm is used for scan converting a line. It was developed by Bresenham. It is an efficient method because it involves only integer addition, subtractions, and multiplication operations. These operations can be performed very rapidly so lines can be generated quickly.

Procedure:

Algorithm:

Step 1 : Input the two lines end points and store the left end point is (x0,y0)

Step 2 : Load(x0,y0) into the frame Buffer i.e plotted the 1^{st} point

Step 3 : Calculate constants $\Delta x, \Delta y, 2\Delta y$ and $2\Delta y - 2\Delta x$ and obtain the starting values for the decision parameter as $p0 = 2\Delta y - \Delta x$

Step 4 : At each xk along the line,starting at k=0.perform the following test if pn<0 the next point to plot is (xk+1,yk) and $pk+1 = pk+2\Delta y-2\Delta x$

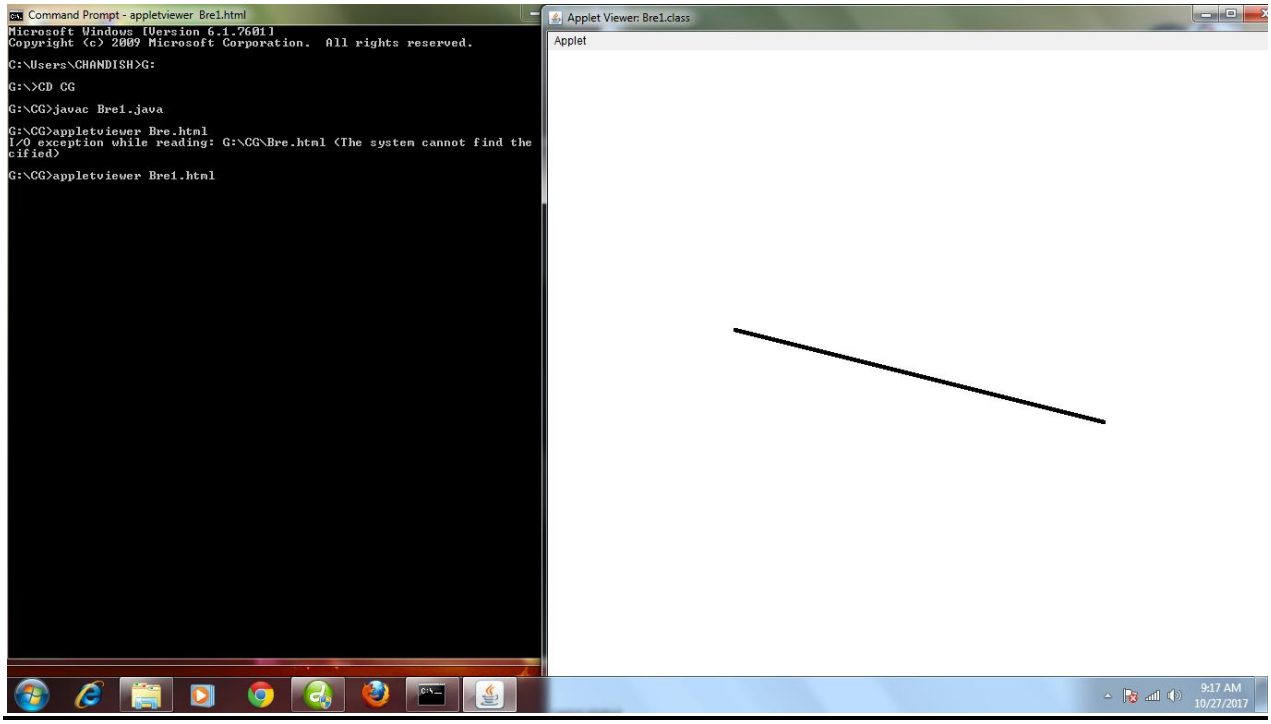
Step 5 : Repeat step 4 'Δx' times

Bresenham's Line Algorithm

```
import java.io.*;
import java.applet.*;
import java.util.*;
import java.awt.*;
public class Bre1 extends Applet
{
public void paint(Graphics g)
{
int x,y,k;
double dx,dy,p;
int x1=200,y1=300,x2=600,y2=400;
dx=Math.abs(x2-x1);
dy=Math.abs(y2-y1);
x=x1;
y=y1;
p=2*dy-dx;
g.fillOval(200,300,5,5);
for(k=0;k<dx;k++)
{
if(p<0)
{
g.fillOval(x++,y,5,5);
p=p+(2*dy);
}
else
{
g.fillOval(x++,y++,5,5);
p=p+(2*(dy-dx));
}
}
```

```
}
}
/*<applet code="Bre1.class" width="800" height="800">
 </applet>*/
```

OUTPUT:-



11. Write a java program to demonstrate MIDPOINT CIRCLE

Aim:To demonstrate the mid point circle algorithm

Description: The **mid-point** circle drawing algorithm is an algorithm used to determine the points needed for rasterizing a circle.

Procedure:

Algorithm:

Step1:Input radius 'r' and circle center(x_c,y_c)and obtain the 1st point on the circumference of the circle centered on the origin as

(x_0,y_0)=(0,r)

Step2: Calculate the initial value of the decision parameter as

P_0=5/4-r

Step3:At each x_k position,starting at k=0,perform the following test if p_k<0 then next point along the circle centered on (0,0) is (x_k+1,y_k) and p_{k+1}=p_{k+2} y_{k+1}+1 otherwise the next point along the circle is (x_{k+1},y_{k-1}) and p_{k+1}=p_{k+2} x_{k+1}+1-2y_{k+1}

Where 2xk+1=2xk+2 and

2y_{k+1}=2y_{k-2}

Step4:Determine symmetry points in the other seven octants

Step5:Move each calculated pixel position (x,y) on to the circular path centered at (x_c,y_c) and plot the coordinate values

X=x+x_c

Y=y+y_c

Step6=Repeat step3 through steps until x>=y

MID POINT CLRCLE ALGORITHM

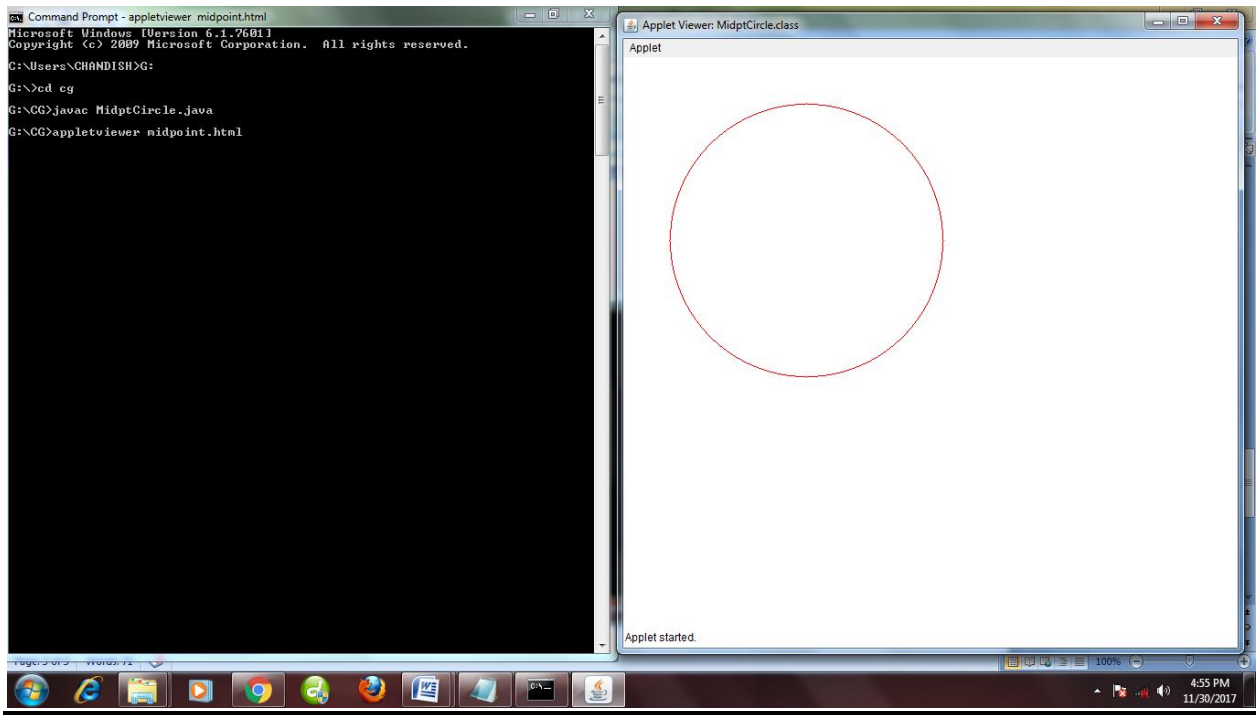
```
import java.io.*;
import java.util.*;
import java.math.*;
import java.applet.*;
import java.awt.*;
public class MidptCircle extends Applet
{
 public void paint(Graphics g)
 {
  int r=150;
  int d=(5/4)*r;
  int x=0;
  int y=r;
 do
  {
   g.setColor(Color.red);
   g.drawLine(y+200,x+200,y+200,x+200);
   g.drawLine(x+200,y+200,x+200,y+200);
   g.drawLine(x+200,-y+200,x+200,-y+200);
   g.drawLine(y+200,-x+200,y+200,-x+200);
   g.drawLine(-y+200,-x+200,-y+200,-x+200);
   g.drawLine(-x+200,-y+200,-x+200,-y+200);
   g.drawLine(-x+200,y+200,-x+200,y+200);
   g.drawLine(-y+200,x+200,-y+200,x+200);
```

```
   if(d<0)
{
  d=d+2*x+3;
}
 else
{
 d=d+2*(x-y)+5;
 y=y-1;
}
 x=x+1;
}
while(x<y);
}
}
```

midpoint.html

```
<html>
<body>
<applet code="MidptCircle.class" width="400" height="400">
</applet>
</body>
</html>
```

OUTPUT:


Command Prompt - appletviewer midpoint.html
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\CHANDISH>G:
G:\>cd cg
G:\CG>javac MidptCircle.java
G:\CG>appletviewer midpoint.html
Applet Viewer: MidptCircle.class
Applet
Applet started.
4:55 PM
11/30/2017

12. Write a java program to demonstrate Ellipse

Aim:To demonstrate a Ellipse algorithm

Description: Midpoint ellipse algorithm plots(finds) points of an ellipse on the first quadrant by dividing the quadrant into two regions.
Each point(x, y) is then projected into other three quadrants (-x, y), (x, -y), (-x, -y) i.e. it uses 4-way symmetry.

Procedure:

STEP1:IN put r^1 $_{x,r_y}$ ^{1} and ellipse center (x0,y0) and obtain the 1st point on an ellipse centered on the origin1 as

(x0,y0)=(0,ry)

STEP2a:calculate the initial value of the decision parameter e in region 1 as $P^10=r^2y-$

$r^2xry+1/4r^2x$

STEP3:At each xk, position in region 1,Starting at K=0,Perform the following test , if Pk,<0,the next point along the ellipse

$P^1=P^1k+2r^2yXk+1+r^2$ with

$2r^2yXk+1=2r^2yXk+2r^2y$,

$2r^2yk+1=2r^2y^k-2r^2x$ and continuous until $2r^2y^x>$

$2r^2xy$ -

STEP4:Calculate the initial value of the decision parameter in region 2 using the point (X0,Y0) calculate in region1

$P0^2=r^2y(x0+y0)+r^2x(y0-0^2-r^2xr^2y)$

STEP5:At each yk position in region 2,Starting at k=0 perform the following test if p2k>0,the next point along the ellipse centered on(0,0)is (Xk,yk-1) and

$P2k+1=P2k-2r^2xyk+1+r^2k$

0therwise,the next point along the circle is (xk+1,yk+1)and

$p2k+1=p2k+2r^2yxk+1-2r^2xyk+1+r^2x$

Using the same incremental calculate for x and y as in region1

STEP6:Determine Symmetry point in the other three quadrants

STEP7:More each calculate pixel position (x,y) onto the elliptical path centered on (x0,y0) and plot the coordinate values

X=x+xc,y=y+yc

STEP8:Repeated the steps for _region 1 until $2x^2yx>2r^2x$

Ellipse

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.event.*;
 public class Ellipse extends Applet implements MouseListener
{
int width=500;
int height=500;
int pushX=400,pushY=200;
public void init()
{
resize(width,height);
addMouseListener(this);
}
public void putPixel(Graphics g,int x0,int y0)
{
g.drawLine(x0,y0,x0,y0);
}
public void ellipse(Graphics g,int CX,int CY,int XRadius, int YRadius)
{
int x,y;
int XChange,YChange,EllipseError,TwoASquare,TwoBSquare,StoppingX,StoppingY;
TwoASquare=2*XRadius*XRadius;
TwoBSquare=2*YRadius*YRadius;
x=XRadius;
y=0;
XChange=YRadius*YRadius*(1-2*XRadius);
YChange=XRadius*XRadius;
EllipseError=0;
StoppingX=TwoBSquare*XRadius;
StoppingY=0;
while(StoppingX>=StoppingY)
{
plot4EllipsePoints(g,CX,CY,x,y);
y++;
StoppingY+=TwoASquare;
EllipseError+=YChange;
YChange+=TwoASquare;
if((2*EllipseError+XChange)>0)
{
x--;
StoppingX-=TwoBSquare;
```

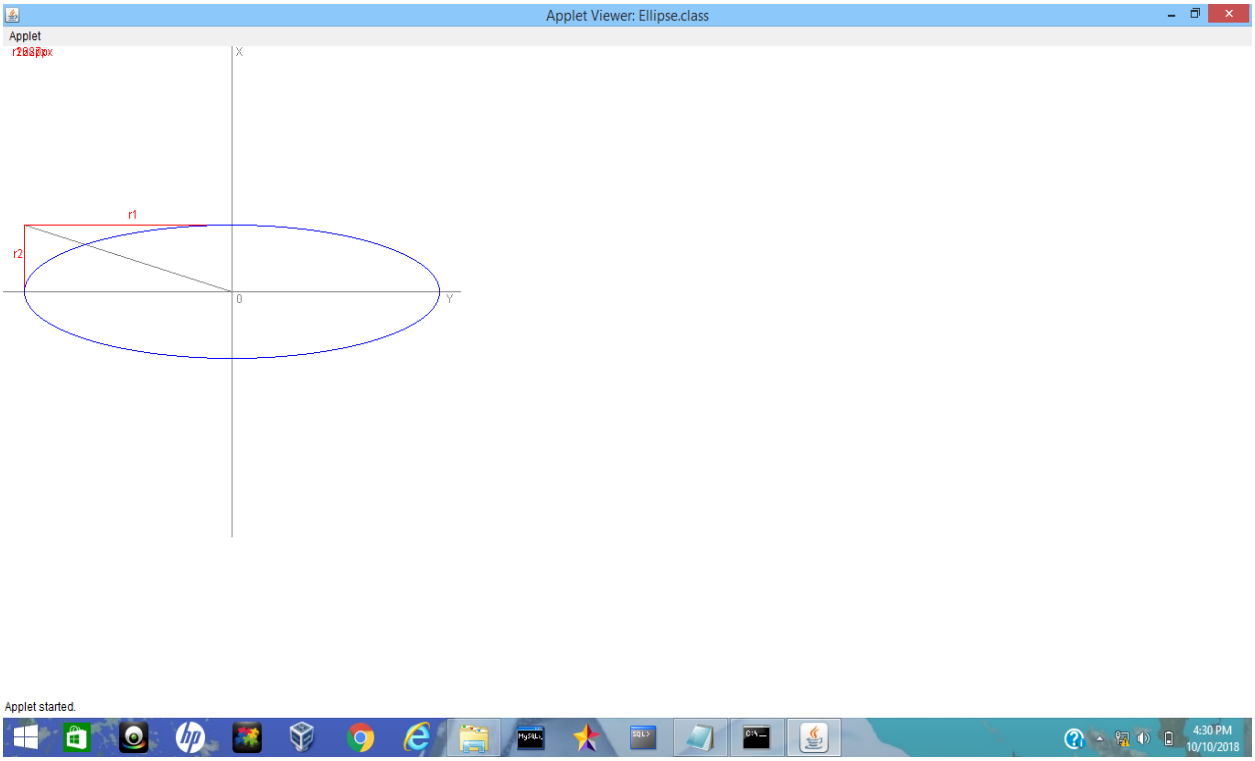
```
EllipseError+=XChange;
XChange+=TwoBSquare;
}
}
x=0;
y=YRadius;
XChange=YRadius*YRadius;
YChange=XRadius*XRadius*(1-2*YRadius);
EllipseError=0;
StoppingY=TwoASquare*YRadius;
StoppingX=0;
while(StoppingX<=StoppingY)
{
plot4EllipsePoints(g,CX,CY,x,y);
x++;
StoppingX+=TwoBSquare;
EllipseError+=XChange;
XChange+=TwoBSquare;
if((2*EllipseError+YChange)>0)
{
y--;
StoppingY-=TwoASquare;
EllipseError+=YChange;
YChange+=TwoASquare;
}
}
}

public void plot4EllipsePoints(Graphics g,int CX,int CY,int x,int y)
{
putPixel(g,CX+x,CY+y);
putPixel(g,CX-x,CY+y);
putPixel(g,CX-x,CY-y);
putPixel(g,CX+x,CY-y);
}
public void paint(Graphics g)
{
g.setColor(Color.gray);
g.drawLine(0,height/2,width,height/2);
g.drawLine(width/2,0,width/2,height);
g.drawString("X",width/2+5,10);
g.drawString("Y",width-15,height/2+12);
g.drawString("0",width/2+5,height/2+12);
g.drawLine(width/2,height/2,pushX,pushY);
```

```
g.setColor(Color.red);
g.drawLine(width/2,pushY,pushX,pushY);
g.drawLine(pushX,height/2,pushX,pushY);
g.drawString("r1",width/2-(width/2-pushX)/2,(pushY<height/2?pushY-6:pushY+12));
g.drawString("r2",(pushX<width/2?pushX-12:pushX+6),height/2-(height/2-pushY)/2);
g.drawString("r1"+Math.abs(width/2-pushX)+"px",10,10);
g.drawString("r2"+Math.abs(height/2-pushY)+"px",10,10);
g.setColor(Color.blue);
ellipse(g,width/2,height/2,Math.abs(width/2-pushX),Math.abs(height/2-pushY));
}
public void mouseClicked(MouseEvent e)
{
pushX=e.getX();
pushY=e.getY();
repaint();
}
public void mouseEntered(MouseEvent e){}
public void mouseExited(MouseEvent e){}
public void mousePressed(MouseEvent e){}
public void mouseReleased(MouseEvent e){}
}

/*<applet code='Ellipse.class' width=800 height=1000></applet>*/
```

Output:



13. Write a java program to demonstrate Cohen Sutherland

Aim:To demonstrate the cohen-sutherland line clipping diagrams.

Description: The **Cohen–Sutherland algorithm** is a computer-graphics algorithm used for line clipping. The algorithm divides a two-dimensional space into 9 regions and then efficiently determines the lines and portions of lines that are visible in the central region of interest (the viewport).

Procedure:

- It speeds up the processing of line segments by performing initial tests that reduce the number of intersections that must be calculated.
- Every line endpoint in a picture is assigned a four digit binary code called a region code that identifies the location of the point relative to the boundaries of the clipping rectangle

1001	1000	1010
0001	0000	0010
0101	0100	0110

- Each bit position in region code is used to indicate one of four relative coordinate positions of points with respect to clip window:to the left,right,top or bottom.
- By numbering the bit positions in the region code as 1 through 4 from right to left, the coordinate regions are corrected with bit position as

  Bit 1: left
  Bit 2: right
  Bit 3: below
  Bit 4: above
- A value of 1 in array bit position indicates that the point is in that relative position.
- Otherwise the bit position is set to 0.If a point is within the clipping rectangle the region code is 0000.
- A point i.e below and to the left of the rectangle has a region code of 0101.
- Bit values in the region code are determined by comparing endpoint coordinate values(x,y) to clip boundaries. Bit 1 is set to 1 if $x<xw_{min}$
- For programming language which bit manipulation is possible region-code bit values can be determined with following two steps.
- Calculate differences between endpoint coordinates and clipping boundaries.
- Use the resultant sign bit of each difference calculation to set the corresponding value in the region code.

  Bit 1 is the sign bit of $x-xw_{min}$

  Bit 2 is the sign bit of xw_{max-x}

  Bit 3 is the sign bit of $y-yw_{min}$

  Bit 4 is the sign bit of yw_{max-y}

- Once we have established region codes for all line endpoints,we can quickly determine which lines are completely inside the clip window and which are clearly outside.
- Any lines that are completely contained within the window boundaries have a region code of 0000 for both endpoints,and we accept these lines.
- Any lines that have a 1 in the same bit position in the region codes for each endpoint are completely outside the clipping rectangle and we reject these lines.

Cohen-Sutherland

```
import java.applet.*;
import java.awt.*;
import java.util.*;
public class CohenSutherland extends Applet
{
	int xmax=90,ymax=80,xmin=40,ymin=40;
	public int[] set(int x,int y)
	{
		int a[]=new int[4];
		if(x<xmin)
			a[3]=1;
		else
			a[3]=0;
		if(x>xmax)
			a[2]=1;
		else
			a[2]=0;
		if(y<ymin)
			a[0]=1;
		else
			a[0]=0;
		if(y>ymax)
			a[1]=1;
		else
			a[1]=0;
		return a;
	}
	boolean check(int a[])
	{
		for(int i=0;i<a.length;i++)
			if(a[i]==1)
				return false;
		return true;
	}
	int[] produceXY(int i,int x1,int y1,float m)
	{
		int a[]=new int[2];
		float x=0,y=0;
		switch(i)
		{
		case 0:
			x=xmin;
			y=y1+m*(x-x1);
```

```
                break;
        case 1:
                x=xmax;
                y=y1+m*(x-x1);
                break;
        case 3:
                y=ymin;
                x=x1+(y-y1)/m;
                break;
        case 2:
                y=ymax;
                x=x1+(y-y1)/m;
                break;
        }
        a[0]=(int)x;
        a[1]=(int)y;
        return a;
}
boolean doAnd(int a[],int b[])
{
        for(int i=0;i<a.length;i++)
        {
                int k=a[i]&b[i];
                if(k==1)
                        return false;
        }
        return true;
}
public void paint(Graphics g)
{
        g.drawRect(xmin,ymin,xmax-xmin,ymax-ymin);
        g.drawRect(xmin+100,ymin,xmax-xmin,ymax-ymin);
        int a[][]=new int[2][4];
        int b[][]=new int[2][4];
        int c[]=new int[2];
        int c1=20;
        int x1=45,y1=45,x2=20,y2=90;
        float m=(y2-y1)*1.0f/(x2-x1);
        //g.drawString(m+" ",100,100);
        g.drawLine(x1,y1,x2,y2);
        a[0]=set(x1,y1);
        a[1]=set(x2,y2);
        //g.drawString(Arrays.toString(a[0]), 300, 300);
        //g.drawString(Arrays.toString(a[1]),400,400);
```

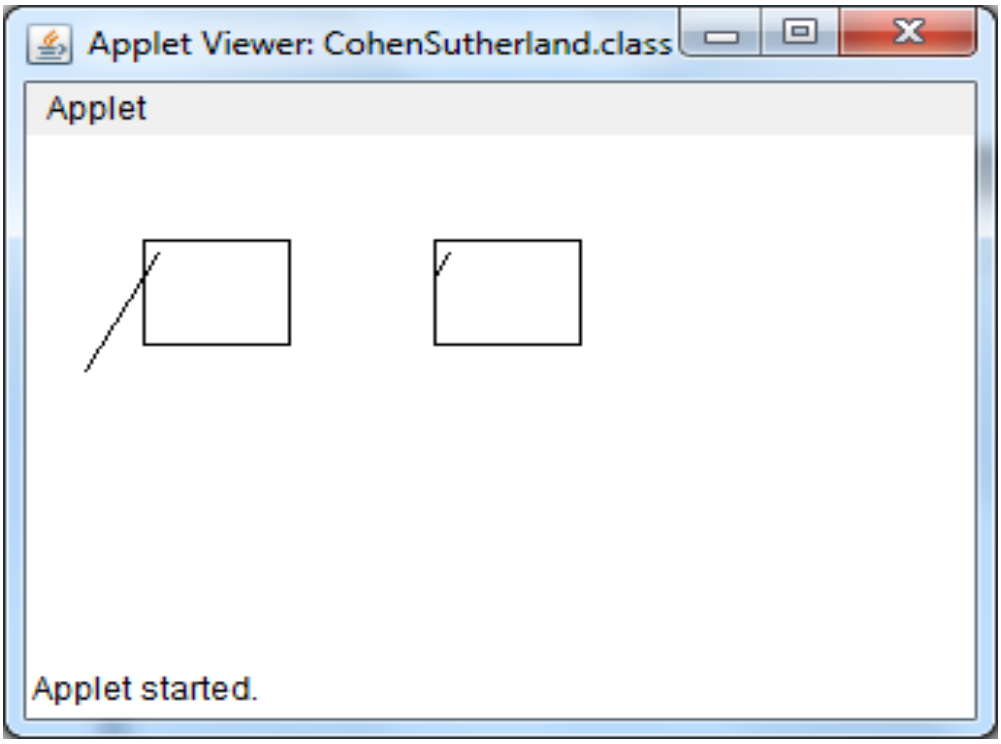
```
        if(check(a[0])&&check(a[1]))
        {
                g.drawLine(x1, y1, x2, y2);
        }
        else
        {
                if(doAnd(a[0],a[1]))
                {
                        for(int i=a[0].length-1;i>=0;i--)
                        {
                                if(a[0][i]==1)
                                {
                                        c=produceXY(a[0].length-1-i,x1,y1,m);
                                        b[0]=set(c[0],c[1]);

g.drawString("first"+Arrays.toString(b[0]),200,300+c1);
                                        if(check(b[0]))
                                        {
                                                x1=c[0];
                                                y1=c[1];
                                                break;
                                        }
                                        c1+=20;
                                }
                        }
                        for(int i=a[0].length-1;i>=0;i--)
                        {
                                if(a[1][i]==1)
                                {
                                        c=produceXY(a[0].length-1-i,x1,y1,m);
                                        b[1]=set(c[0],c[1]);

//g.drawString("second"+Arrays.toString(b[1]),200,300+c1);
                                        if (check(b[1]))
                                        {
                                                x2=c[0];
                                                y2=c[1];
                                                break;
                                        }
                                        //c1+=20;
                                }
                        }
                        g.drawLine(x1+100,y1,x2+100,y2);
                }
```

```
            }
        } }
/*<applet code="CohenSutherland.class" width=500 height=500> </applet>*/
```

OUTPUT



14. Write a java program to demonstrate Scaling

Aim: To demonstrate scaling transformation diagrams.

Description: It is used to alter or change the size of objects. The change is done using scaling factors. There are two scaling factors, i.e. S_x in x direction S_y in y-direction. If the original position is x and y. Scaling factors are S_x and S_y then the value of coordinates after scaling will be x^1 and y_1.

Procedure:

- A scaling transformation alters the size of an object.
- This operation can be carried out for polygons by multiplying the coordinate values(x,y) to each vertex by scaling factor s_x& s_y to produce the transformed ***coordinates(x^1,y^1)***

  $X^1=x.s_x$, $y^1=y.s_y$ ->eq(1)

- Scaling factor s_x scales object in x-direction while s_y scales in y-direction.
- The transformation equation in matrix form

$$\begin{matrix} x_{=}sx & 0 \\ y & 0 \end{matrix} \begin{matrix} x \\ y \end{matrix} \text{ or } p^1=s.p$$

Where s is 2 by 2 scaling matrix

- The figure shows the tuming a square into a rectangle with scaling factor.
  S_x=2 and s_y=1
  Any positive numeric values are valid for scaling factors s_x and s_y
- Values less than one reduce the size of the objects and values greater than 1 produce an enlarged objects.
- There are two types of scaling.They are
  - Uniform scaling – s_x and s_y have same values
  - Non-uniform scaling – s_x and s_y have different value.

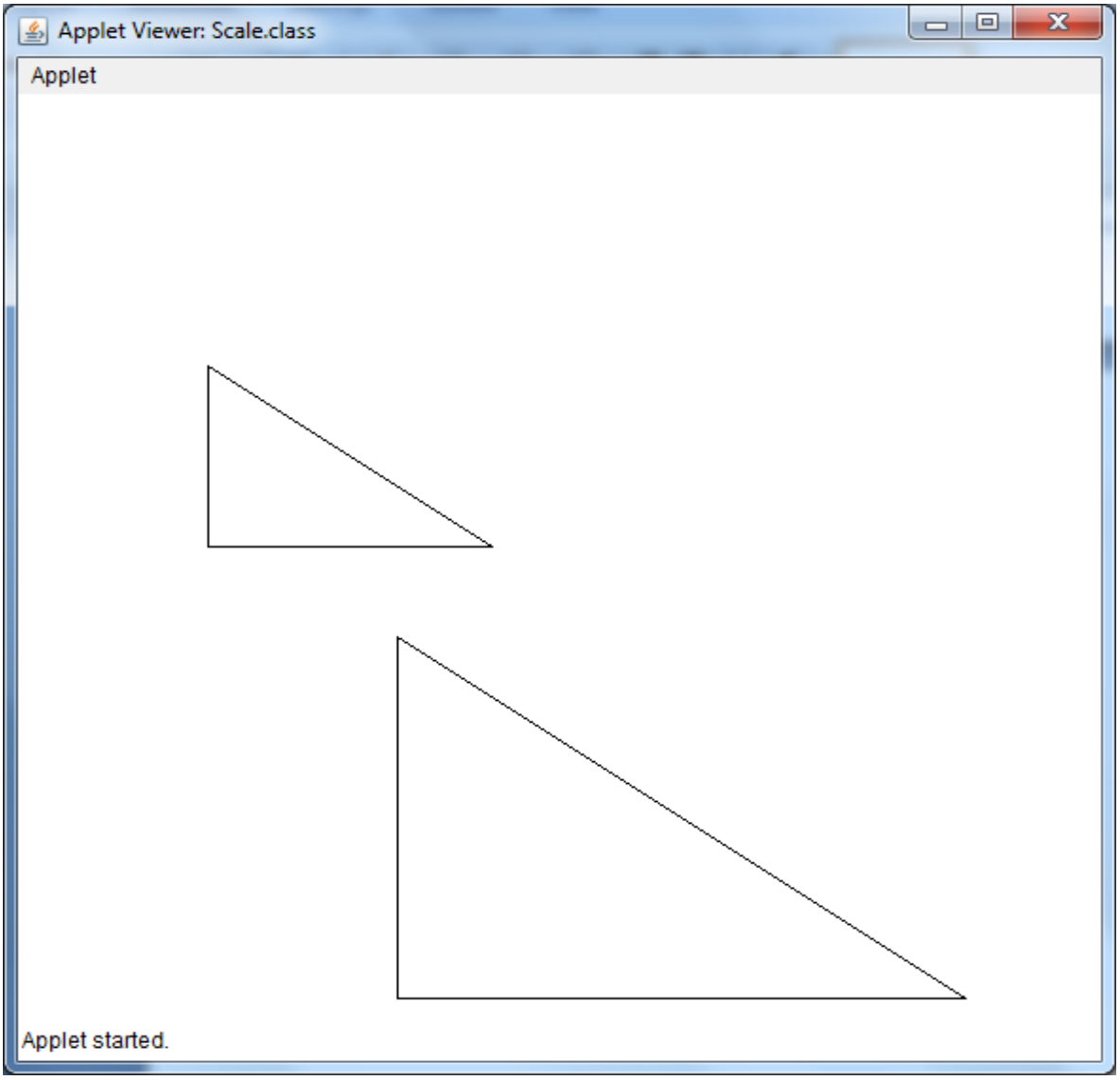
Scaling

```
import java.awt.*;
import java.applet.*;
import java.math.*;
public class Scale extends Applet
{
public void paint(Graphics g)
{
        g.drawLine(100,150,100,250);
        g.drawLine(100,150,250,250);
        g.drawLine(100,250,250,250);
        int a[][]=new int[3][3];
        int b[][]=new int[3][3];
        int c[][]=new int[3][3];
        int i,j,k;
        a[0][0]=2;
        a[0][1]=0;
        a[0][2]=0;
        a[1][0]=0;
        a[1][1]=2;
        a[1][2]=0;
        a[2][0]=0;
        a[2][1]=0;
        a[2][2]=1;
        b[0][0]=100;
        b[0][1]=100;
        b[0][2]=250;
        b[1][0]=150;
        b[1][1]=250;
        b[1][2]=250;
        b[2][0]=1;
        b[2][1]=1;
        b[2][2]=1;
        for(i=0;i<3;i++)
                for(j=0;j<3;j++)
                {
                        c[i][j]=0;
                        for(k=0;k<3;k++)
                        {
                                c[i][j]=(c[i][j]+(a[i][k]*b[k][j]));

                                }
                        System.out.println();
                }
        g.drawLine(c[0][0],c[1][0],c[0][1],c[1][1]);
        g.drawLine(c[0][1],c[1][1],c[0][2],c[1][2]);
        g.drawLine(c[0][0],c[1][0],c[0][2],c[1][2]);
        }

}
```

Output



15. Write a java program to demonstrate Translation

Aim : To demonstrate the Translation diagrams **Description**

:

A translation is applied to an object by representing it a long a straight line path from one coordinate location to another adding translation distances,tx, ty to original coordinate position(x,y) to move the point to a new position (x',y')

$$X'=x+tx\ ,\ Y'=y+ty$$

The translation distance point (tx ,ty) is called translation vector or shift vector.

Translation equation can be expressed as single matrix equation by using column vectors to represent the coordinate position and the translation vector

as $P=(X,Y)$ $T=(tx,ty)$

$X'=X+tx\ ,$ $Y'=Y+ty$

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$P'= P+T$$

Procedure:

1.Point Translation P(X, Y) : Here we only translate the x and y coordinates of given point as per given translation factor dx and dy respectively.
2.Line Translation: The idea to translate a line is to translate both of the end points of the line by the given translation factor(dx, dy) and then draw a new line with inbuilt graphics function.
3.Rectangle Translation : Here we translate the x and y coordinates of both given points A(top left) and B(bottom right) as per given translation factor dx and dy respectively and then draw a rectangle with inbuilt graphics function

Translation

```
package chinni;
import java.math.*;
```

```
import java.awt.*;
import java.applet.*;
public class Trans extends Applet
{
	public void paint(Graphics g)
	 {
		g.drawLine(100,150,100,250);
		g.drawLine(100,150,250,250);
		g.drawLine(100,250,250,250);
		int a[][]=new int[3][3];
		int b[][]=new int[3][3];
		int c[][]=new int[3][3];
		int i,j,k;
		a[0][0]=1;
		a[0][1]=0;
		a[0][2]=100;
		a[1][0]=0;
		a[1][1]=1;
		a[1][2]=100;
		a[2][0]=0;
		a[2][1]=0;
		a[2][2]=1;
		b[0][0]=100;
		b[0][1]=100;
		b[0][2]=250;
		b[1][0]=150;
		b[1][1]=250;
		b[1][2]=250;
		b[2][0]=1;
		b[2][1]=1;
		b[2][2]=1;
		for(i=0;i<3;i++)
			for(j=0;j<3;j++)
			{
				c[i][j]=0;
				for(k=0;k<3;k++)
				{
					c[i][j]=(c[i][j]+(a[i][k]*b[k][j]));
				}
				System.out.println();
				}
		g.drawLine(c[0][0],c[1][0],c[0][1],c[1][1]);
		g.drawLine(c[0][1],c[1][1],c[0][2],c[1][2]);
		g.drawLine(c[0][0],c[1][0],c[0][2],c[1][2]);

}
}
```

output