PART-I: Lab for MCS-051 (Advanced Internet Technologies)

Question 1: Write a Program using Servlet and JDBC for developing online application for displaying the details of Cars owned by the residents in XYZ society. Make necessary assumptions and create appropriate databases.

Refer:

```
CREATE TABLE Student

(
 student_id INT NOT NULL AUTO_INCREMENT,
 student_name VARCHAR(25),
 student_enrol VARCHAR(10),
 student_add VARCHAR(70),
 student_prog VARCHAR(10),
 student_prog_start VARCHAR(5),
 student_prog_end VARCHAR(5),
 PRIMARY KEY(student_id)
);


CREATE TABLE Course (
 course_id INT NOT NULL AUTO_INCREMENT,
 course_code VARCHAR(10),
 course_name VARCHAR(20),
 course_prog VARCHAR(10),
 PRIMARY KEY(course_id)
);




CREATE TABLE Result (
 ID INT NOT NULL AUTO_INCREMENT,
 course_code VARCHAR(10),
 student_id VARCHAR(10),
 marks_assign_got VARCHAR(5),
 marks_pract_got VARCHAR(5),
 marks_theory_got VARCHAR(5),
 result_status VARCHAR(10),
 PRIMARY KEY(ID),
FOREIGN KEY(course_code,student_id,marks_assign_got,marks_pract_got,marks_theory_got)
REFERENCES Course (course_code)
,Student(student_id),Marks(marks_assign_got,marks_pract_got,marks_pract_got)
);
```

Result Servlet
CODE

```
package ignou;
import java.io.*;
import java.sql.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.net.URL;
public class Result extends HttpServlet {
public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException,
IOException {
res.setContentType("text/html");
  PrintWriter out = res.getWriter();
out.println("<HTML><HEAD><TITLE>IGNOU MCA TEE Result:::</TITLE>");
  out.println("</HEAD>");
  out.println("<BODY >");
       out.println("<P align=center>MCA TEE Result</P>");
       out.println("<TABLE align=center border=1  width=\"65%\">");
Connection con=null;
Statement stm = null;
try{
           Class.forName("org.gjt.mm.mysql.Driver");
       String dbURL="jdbc:mysql://localhost/MCA";
       String username="root";
       String password="";
  con=DriverManager.getConnection(dbURL,username,password);
stm=con.createStatement();
      ResultSet rs=stm.executeQuery("SELECT
student_name,course_code,marks_assign_got,marks_theory_got,marks_pract_got,result_status
FROM Student,Result WHERE Result.student_id =Student.student_id");
 while(rs.next()) //retreive all the records into the table
 {
      out.println("<TR>");
      out.println("<TD>Name</TD><TD>" + rs.getString("student_name") + "</TD></TR>");
      out.println("<TR><TD>Course code</TD><TD>" + rs.getString("course_code") +
"</TD></TR>");
      out.println("<TR><TD>Assignments marks</TD><TD>" + rs.getString("marks_assign_got") +
"</TD></TR>");
      out.println("<TR><TD>Theory marks</TD><TD>" + rs.getString("marks_theory_got") +
"</TD></TR>");
   out.println("<TR><TD>Practical marks</TD><TD>" + rs.getString("marks_pract_got") +
"</TD></TR>");
```

```
   out.println("<TR><TD>Result</TD><TD>" + rs.getString("result_status") + "</TD>");
               out.println("</TR>");
            rs.close();
            stm.close();
            con.close();

     }}catch(Exception e){
 out.println(e.getMessage());
}
      out.println("<P> </P></FONT></BODY></HTML>");
}
}
```

Question 2: Write a JSP Program, which displays a web page containing detailed Bio-Data of yours including your educational qualifications, photograph and photo album etc..

Ans: 1. info.JSP

<%@page language="java" import="java.io.*,java.util.*"%>

<html>

<head>

<title>Information</title>

</head>

<body>

<form action='Personal'method='post'>

 click here to view academic Information

<input type='submit'value='click here to view Personal Information'>

</form>

</body>

</html>

academic.JSP

3. Personal.java

```
import java.io.*;
import java.text.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Personal extends HttpServlet {
public void doPost(HttpServletRequest req,HttpServletResponse res)
throws ServletException,IOException{
String name = req.getParameter("user");
PrintWriter out = res.getWriter();
out.println("<html>");
out.println("<head>");
out.println("<title>Personal Info</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Personal Information </h1>");
out.println("<h1>Name: N Y Abhay </h1>");
out.println("<h1>Clss: Pre-Nursery </h1>");
out.println("<h1>Address: Bangalore </h1>");
out.println("</body>");
out.println("</html>");
}
}
```

academic.jsp

```
<%@page language="java" import="java.io.*,java.util.*"%>
```

```
<html><head><title>ASCII Conversion</title></head>
<body>
<form>
<input type='submit'value='Infoormation About Abhay N Y'>
<br>
Name :<input type='text'name='OutputChar'value='Abhay'>
<br>
<br>
Qualification :<input type='text'name='OutputChar'value='MTech, MBA'>
<br>
</form></body></html>
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
version="2.4">
<servlet>
<servlet-name>hello</servlet-name>
<servlet-class>Personal</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>hello</servlet-name>
<url-pattern>/Personal</url-pattern>
</servlet-mapping>
```

</web-app>

Question 3: Write a program using JDBC and JSP to display the names and addresses of all those saving account holders No 1002

Refer:

```
DATABASE

Database name:Company

CREATE TABLE Student (
StudentID VARCHAR(10)NOT NULL ,
Student_name VARCHAR(30),
Student_enrol VARCHAR(9),
Student_address VARCHAR(70),
Student_course VARCHAR(10),
CentreID VARCHAR(4),
PRIMARY KEY(StudentID)
);

CREATE TABLE Soft_company (
comp_id INT NOT NULL AUTO_INCREMENT,
comp_name VARCHAR(30),
comp_address VARCHAR(70),
StudentID VARCHAR(10),

PRIMARY KEY(comp_id),
FOREIGN KEY(StudentID) REFERENCES Student (StudentID)
);
```

JSP page for displaying students who are working in soft dev company

CODE

```
<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>Working MCA students</title>
</head>

<body bgcolor=#ffccdd text=#111ccc>
```

```
<h3>Information of students who are currently working in software development Company</h3>

<table cellspacing="5" cellpadding="5" border="1" align="center"width="70%">
<tr>
<td align="right">Name:</td>

<td align="right">Address:</td>

<td align="right">Software Company:</td>

</tr>
<%@page language="java"%>
<%@page import="java.sql.*"%>
<%

Connection con = null;

Statement stm = null;

try{
Class.forName("org.gjt.mm.mysql.Driver");
String dbURL="jdbc:mysql://localhost/Company";
String username="root";
String password="";

con=DriverManager.getConnection(dbURL,username,password);
stm=con.createStatement();

ResultSet rs=stm.executeQuery("SELECT
Student.Student_name,Student.Student_address,Soft_company.comp_name FROM
Student,Soft_company WHERE Student.StudentID=Soft_company.StudentID");

while(rs.next()){%>
<TR>
<TD><%out.println(rs.getString("Student_name"));%></TD>
<TD><%out.println(rs.getString("Student_address"));%></TD>
<TD><%out.println(rs.getString("comp_name"));%></TD>

</TR>
<%}%>
<%

rs.close();
stm.close();
```

```
con.close();

}catch(Exception e){
e.printStackTrace();
}

%>

</table>
</body>
</html>
```

Question 4: Write an XML document to represent the Items (at least three items) in a stationary shop.

23. e23.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<students-subjects>
<student-subject>
<subject>Visual Basic Programming</subject>
<qualification>B.Tech</qualification>
<student1>Mohan</student1>
<student2>Naveen</student2>
</student-subject>
<student-subject>
<subject>Java Programming</subject>
<qualification>M.Tech</qualification>
<student1>Robert John</student1>
<student2>Sudhansh</student2>
</student-subject>
<student-subject>
<subject>ASP Programming</subject>
<qualification>M.Sc</qualification>
<student1>Neeta</student1>
<student2>Ravi</student2>
</student-subject>
</students-subjects>
```

PART-II: Lab for MCS-053 (Computer Graphics and Multimedia)

Question 1: Write a program in C/C++ using OpenGL to draw a circle of red color inside of a rectangle of blue color on a background of green colors.

```
for rectangle
```

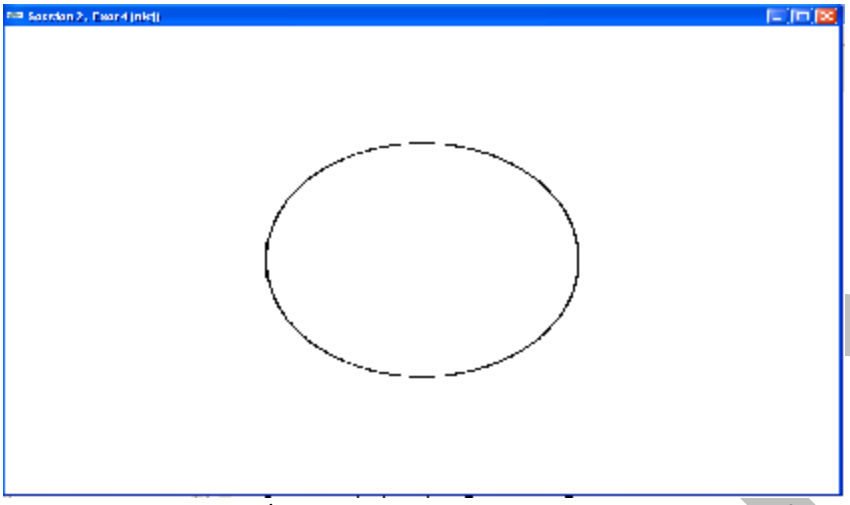
```
#include<windows.h>
#include<gl/gl.h>
#include<gl/glu.h>
#include<gl/glut.h>

void init()
{
	glClearColor(0.0,0.0,0.0,0.0); // background Color
	glColor3f(1.0,0.0,0.0);//Drawing Color
	glPointSize(4);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluOrtho2D(0.0,800,0.0,600.0);
}
void display()
{
	glClear(GL_COLOR_BUFFER_BIT);
	glBegin(GL_POLYGON);
		glVertex2i(100,100);
		glVertex2i(100,500);
		glVertex2i(700,500);
		glVertex2i(700,100);
	glEnd();
	glFlush();
}
void main(int argc, char **argv)
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
	glutInitWindowSize(800,600);
	glutCreateWindow("Session2, Exer4(Niraj)");
	glutDisplayFunc(display);
```

```
	init();
	glutMainLoop();
}
circle
/*
 * Problem definition:To impement Bresenham Circle-generation algorithm
 * By: Prasun Pal(prsn)
 * Date: 21/04/2009
 */

#include<windows.h>
#include<gl/gl.h>
#include<gl/glu.h>
#include<gl/glut.h>
#include<math.h>
#include<stdio.h>

void nkjSwap(float *,float *);

void nkjInit()
{
	glClearColor(1.0,1.0,1.0,0.0); //Black background Color
	glColor3f(0.0,0.0,0.0);//Drawing Color yellow
	glPointSize(2);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluOrtho2D(0.0,800,0.0,600.0);
}

void nkjBresenhamCircleGeneration3f(float x1, float y1, float radious)
{
	float dcsnPrmtr=5/4-radious;//decision Parameter
```

```
	int k=0;
	float x=x1, y=y1;
	x1=0;
	y1=radious;
	while(x1<y1)
	{
		if(dcsnPrmtr<0)
		{
			dcsnPrmtr += 2 * x1 + 2 + 1;
			x1++;
		}
		else //if(dcsnPrmtr
		{
			dcsnPrmtr += 2 * x1 + 2 - 2 * y1 - 2 + 1;
			x1++;
			y1--;
		}
		//generate symmetry points

		glVertex2i((int)(x+x1),(int)(y+y1));
		glVertex2i((int)(x-x1),(int)(y+y1));
		glVertex2i((int)(x+x1),(int)(y-y1));
		glVertex2i((int)(x-x1),(int)(y-y1));

		glVertex2i((int)(x+y1),(int)(y+x1));
		glVertex2i((int)(x-y1),(int)(y+x1));
		glVertex2i((int)(x+y1),(int)(y-x1));
		glVertex2i((int)(x-y1),(int)(y-x1));
	}
}
void nkjDisplay()
{
```

```
	glClear(GL_COLOR_BUFFER_BIT);
	glBegin(GL_POINTS);
		nkjBresenhamCircleGeneration3f(400.0,300.0,150.0);
	glEnd();
	glFlush();
}
void main(int argc, char **argv)
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
	glutInitWindowSize(800,600);
	glutCreateWindow("Session2, Exer4(nkj)");
	glutDisplayFunc(nkjDisplay);
	nkjInit();
	glutMainLoop();
}
```

Output:



Question 2: Write a C/C++ program to implement DDA algorithm for line generation. Use this algorithm to draw a line with endpoints (2, 3) and (9, 8).

Hint: Bresenham algorithm is accurate and efficient raster line generation algorithm. This algorithm scan converts lines using only incremental integer calculations and thes calculations can also be adopted to display circles and other curves. **Algorithm m < 1:**
(i) Input two line end points and store left end point in (x_0,y_0)
(ii) Load (x_0,y_0) on frame buffer i.e., plot the first point .
(iii) Calculate Δx, Δy, $2\Delta y$, $2\Delta y - 2.x$ and obtain the starting value of decision parameter as $p_0 = 2\Delta y - \Delta x$
(iv) At each x_k along the line, starting at k=0, perform following test:

If $p_k < 0$, the next plot is (x_{k+1}, y_k) and $p_{k+1} = p_k + 2\Delta y$ Else next plot is (x_{k+1}, y_{k+1}) and $p_{k+1} = p_k + 2(\Delta y - \Delta x)$

(a) Repeat step (d) Δx times.
$(x_1, y_1) = (2, 1)$, $(x_2, y_2) = (6, 4)$
$m = y_2 - y_1 = \Delta y = 4\text{-}1 = 3 < 1$

$$m = \frac{y2\text{-}y1}{x2\text{-}x1} = \frac{\Delta y}{\Delta x}$$

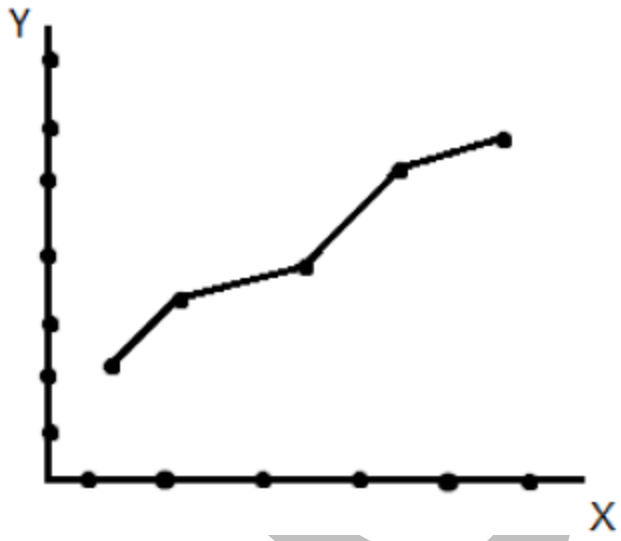
$\Delta y = 3$, $\Delta x = 4$ Initial decision parameter $P_0 = 2\Delta y - \Delta x = 6 - 4 = 2$ Increments for computing successive decision parameters are: $2\Delta y = 6$ $2\Delta y - 2\Delta x = -2$ Plot initial point (2, 1) in frame buffer. Determine successive pixel position along line path using the rule:
(i) If $P_k > 0$, next point is $(x_k + 1, y_k + 1)$

$P_{k+1} = P_k + 2\Delta y - 2\Delta x$
(ii) If $P_k \leq 0$, next point is $(x_k + 1, y_k)$

$P_{k+1} = P_k + 2\Delta y$ Thus, we have:

k	P_k	(x_{k+1}, y_{k+1})
0	2	(3, 2)
1	0	(4, 2)
2	6	(5, 3)
3	4	(6, 4)

Thus, we get the line joining (2, 1) to (6, 4).



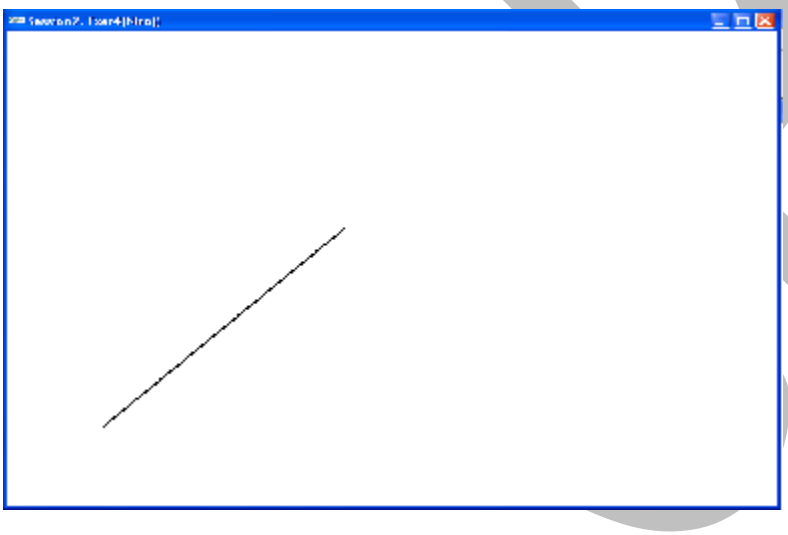

```
/*
 * Problem definition:To impement DDA Line-generation algorithm
 * By: Prasun Pal(prsn)
 * Date: 21/04/2009
 */

#include<windows.h>
```

```
#include<gl/gl.h>
#include<gl/glu.h>
#include<gl/glut.h>
#include<math.h>
void nkjSwap(float *,float *);

void nkjInit()
{
	glClearColor(1.0,1.0,1.0,0.0); //Black background Color
	glColor3f(0.0,0.0,0.0);//Drawing Color yellow
	glPointSize(2);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluOrtho2D(0.0,800,0.0,600.0);
}

void nkjDDA4f(float x1, float y1, float x2, float y2)
{
	if(x1>x2)
	{
		nkjSwap(&x1,&x2);
		nkjSwap(&y1,&y2);
	}

	float slope=(y2-y1)/(float)(x2-x1);
	if(slope>0 && slope<1)
	{
		while(x1<=x2)
		{
			glVertex2i((int)x1,int(y1));
			x1++;
			y1+=slope;
```

```
		}
	}
	else if(slope>=1)
	{
		float slope1=1/slope;
		while(y1<=y2)
		{
			glVertex2i(int(x1),int(y1));
			y1++;
			x1+=slope1;
		}

	}
}

void nkjSwap(float *x, float *y)
{
	float temp=*x;
	*x=*y;
	*y=temp;
}

void nkjDisplay()
{
	glClear(GL_COLOR_BUFFER_BIT);
	glBegin(GL_POINTS);
		nkjDDA4f(100.0,100.0,350.0,350.0);//two points p1(100,100), p2(350,350)
	glEnd();
	glFlush();
}
```

```
void main(int argc, char **argv)
{
      glutInit(&argc, argv);
      glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
      glutInitWindowSize(800,600);
      glutCreateWindow("Session2, Exer4(Niraj)");
      glutDisplayFunc(nkjDisplay);
      nkjInit();
      glutMainLoop();

}
```

Output:



Question 3: Write a C/C++ program to draw a Bezier curve having the control points as p0 (0, 0), P1 (2, 5), P2 (5, 9), P3 (10, 20). Calculate the coordinates of the points on the curve corresponding to the parameter u = 0.2, 0.4, 0.6.

Hint : Bezier curves are commonly found in painting and drawing packages, as well as the CAD systems, since they are easy to implement and the reasonably powerful in curve design. Efficient methods for determining coordinate position along a Bezier curve can be set up using recursive calculations. For example, successive binomial coefficients can be calculated as

C(n,k) = (n-k+1)/k*C(n,k-1), for n>=k.

Bezier curve with control point P1(0,0), P2(5,10), P3(30,9), P4(40,10).
We know cubic Bezier curve is

$$P(u) = \sum P_i \; B_{3,i}\{u\}$$

$(0,0)(1-u)^3+3(2,5)u(1-u)^2+3(5,9)u^2(1-u)+(10,20)u^3$

With u=0.2

$P(0.2)=(0,0)(0.8)^3+(6,10)(0.2)(0.8)^2+(15,27)(0.2)^2(0.8)+(10,20)(0.2)^3$
= (0,0)(0.512)+(6,10)(0,128)+(15,27)(0.032)+(10,20)(0.008)
= (0,0)+(0.768,1.28)+(0.48,0.864)+(0.08,0.16)
= (1.328,2.304)

With u=0.4

$P(0.4)=(0,0)(0.6)^3+(6,10)(0.4)(0.6)^2+(15,27)(0.4)2(0.6)+(10,20)(0.4)^3$
= (0,0)+(6,10)(0.144)+(15,27)(0.096)+(10,20)(0.064)
= (0.864,1.44)+(1.44,2.592)+(0.64,1.28)
= (2.944,5.312)

$P(0.6)= (0,0)(0.4)^3+(6,10)(0.6)(0.4)^2+(15,27)(0.6)^2(0.4)+(10,20)(0.6)^3$
= (6,10)(0.096)+(15,27)(0.144)+(10,20)(0.216)
= (0.576,0.96)+(2.16,3.888)+(2.16,4.32)
= (4.896,9.168)

Question 4: Write a program in C/C++ using OpenGL to perform a 3-Dimensional transformation, such as translation ,rotation and reflection, on a given triangle.

Solution:

```
#include <stdio.h>
#include <stdlib.h>
#include<graphics.h>
```

```
#include<conio.h>
void draw3d(int fs,int x[20],int y[20],int tx,int ty,int d);
void draw3d(int fs,int x[20],int y[20],int tx,int ty,int d)
{
int i,j,k=0;
for(j=0;j<2;j++)
{
for(i=0;i<fs;i++)
{
if(i!=fs-1)
line(x[i]+tx+k,y[i]+ty-k,x[i+1]+tx+k,y[i+1]+ty-k);
else
line(x[i]+tx+k,y[i]+ty-k,x[0]+tx+k,y[0]+ty-k);
}
k=d;
}
for(i=0;i<fs;i++)
{
line(x[i]+tx,y[i]+ty,x[i]+tx+d,y[i]+ty-d);
}
}

void main()
{
int gd=DETECT,gm;
int x[20],y[20],tx=0,ty=0,i,fs,d;
initgraph(&gd,&gm,"");
printf("no of sides (front view only) : ");
scanf("%d",&fs);
printf("co-ordinates : ");
for(i=0;i<fs;i++)
{
printf("(x%d,y%d)",i,i);
scanf("%d%d",&x[i],&y[i]);
}
printf("Depth :");
scanf("%d",&d);
draw3d(fs,x,y,tx,ty,d);
```

```
printf("translation (x,y)");
scanf("%d%d",&tx,&ty);
draw3d(fs,x,y,tx,ty,d);
getch();
}
```

Question 5: Write C/C++ program to implement the Sutherland Hodgman polygon clipping algorithm. Using this program clip the following polygon against the rectangular window as given below in figure 1. Make suitable assumptions.

Source Code:

Any polygon of any arbitrary shape can be described with the help of some set of vertices associated with it. When we try to clip the polygon under consideration with any rectangular window, then, we observe that the coordinates of the polygon vertices satisfies one of the four cases listed in the table shown below, and further it is to be noted that this procedure of clipping can be simplified by clipping the polygon edgewise and not the polygon as a whole. This decomposes the bigger problem into a set of subproblems, which can be handled separately as per the cases listed in the table below. Actually this table describes the cases of the Sutherland-Hodgman Polygon Clipping algorithm.

Thus, in order to clip polygon edges against a window edge we move from vertex V_i to the next vertex V_{i+1} and decide the output vertex according to four simple tests or rules or cases listed below:

Table: Cases of the Sutherland-Hodgman Polygon Clipping Algorithm

Case	V_i	V_{i+1}	Output Vertex
A	Inside window)	Inside	V_{i+1}
B	Inside	Outside	V'_i i.e intersection of polygon and window edge
C	Outside	Outside	None
D	Outside	Inside	V'_I ; V_{i+1}

In words, the 4 possible Tests listed above to clip any polygon states are as mentioned below:

1) If both Input vertices are inside the window boundary then only 2nd vertex is added to output vertex list.

st

90

2) If 1 vertex is inside the window boundary and the 2nd vertex is outside then, only the intersection edge with boundary is added to output vertex.

3) If both Input vertices are outside the window boundary then nothing is added to the output list.

4) If the 1st vertex is outside the window and the 2nd vertex is inside window, then both the intersection points of the polygon edge with window boundary and 2nd vertex are added to output vertex list.

So, we can use the rules cited above to clip a polygon correctly. The polygon must be tested against each edge of the clip rectangle; new edges must be added and existing edges must be discarded , retained or divided. Actually this algorithm decomposes

the problem of polygon clipping against a clip window into identical subproblems, where a subproblem is to clip all polygon edges (pair of vertices) in succession against a single infinite clip edge. The output is a set of clipped edges or pair of vertices that fall in the visible side with respect to clip edge. These set of clipped edges or output vertices are considered as input for the next sub problem of clipping against the second window edge. Thus, considering the output of the previous subproblem as the input, each of the subproblems are solved sequentially, finally yielding the vertices that fall on or within the window boundary. These vertices connected in order forms, the shape of the clipped polygon.

For better understanding of the application of the rules given above consider the Figure 14, where the shaded region shows the clipped polygon

Draw the picture in the question here.....

Pseudocode for Sutherland – Hodgman Algorithm

Define variables

inVertexArray is the array of input polygon vertices

outVerteArray is the array of output polygon vertices

Nin is the number of entries in inVertexArray

Nout is the number of entries in outVertexArray

n is the number of edges of the clip polygon

ClipEdge[x] is the xth edge of clip polygon defined by a pair of vertices

s, p are the start and end point respectively of current polygon edge

i is the intersection point with a clip boundary

j is the vertex loop counter

Define Functions

AddNewVertex(newVertex, Nout, outVertexArray)

: Adds newVertex to outVertexArray and then updates Nout

InsideTest(testVertex, clipEdge[x])

: Checks whether the vertex lies inside the clip edge or not; retures TRUE is inside else returns FALSE

Intersect(first, second, clipEdge[x])

: Clip polygon edge(first, second) against clipEdge[x], outputs the intersection point

{ : begin main

x = 1

while (x ≤ n) : Loop through all the n clip edges

{

Nout = 0 : Flush the outVertexArray

s = inVertexArray[Nin] : Start with the last vertex in inVertexArray

for j = 1 to Nin do : Loop through Nin number of polygon vertices (edges)

{

p = inVertexArrray[j]

ifInsideTest(p, clipEdge[x] = = TRUE then : Case A and D

ifInsideTest(s, clipEdge[x] = = TRUE then

AddNewVertex(p, Nout, outVertexArray) : Case A

else

i = Intersect(s, p, clipEdge[x]) : Case D

AddNewVertex(i, Nout, outVertexArray)

AddNewVertex(p, Nout, outVertexArray)

end if

else : i.e. if InsideTest(p, clipEdge[x] = = FALSE

(Cases 2 and 3)

ifInsideTest(s, clipEdge[x]) = =TRUE then : Case B

{

Intersect(s, p, clipEdge[x])

AddNewVertex(i, Nout, outVertexArray)

end if : No action for case C

s = p : Advance to next pair of vertices

j = j + 1

end if : end {for}

}

x = x + 1 : Proceed to the next ClipEdge[x +1]

Nin = Nout

inVertexArray = outVertexArray : The ouput vertex array for the current clip edge becomes the input vertex array for the next clip edge

} : end while

} : end main

Window

Polygon

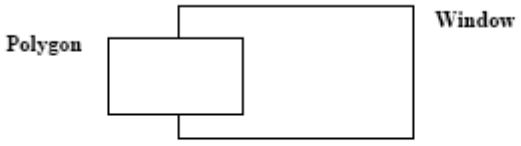


Figure 1.