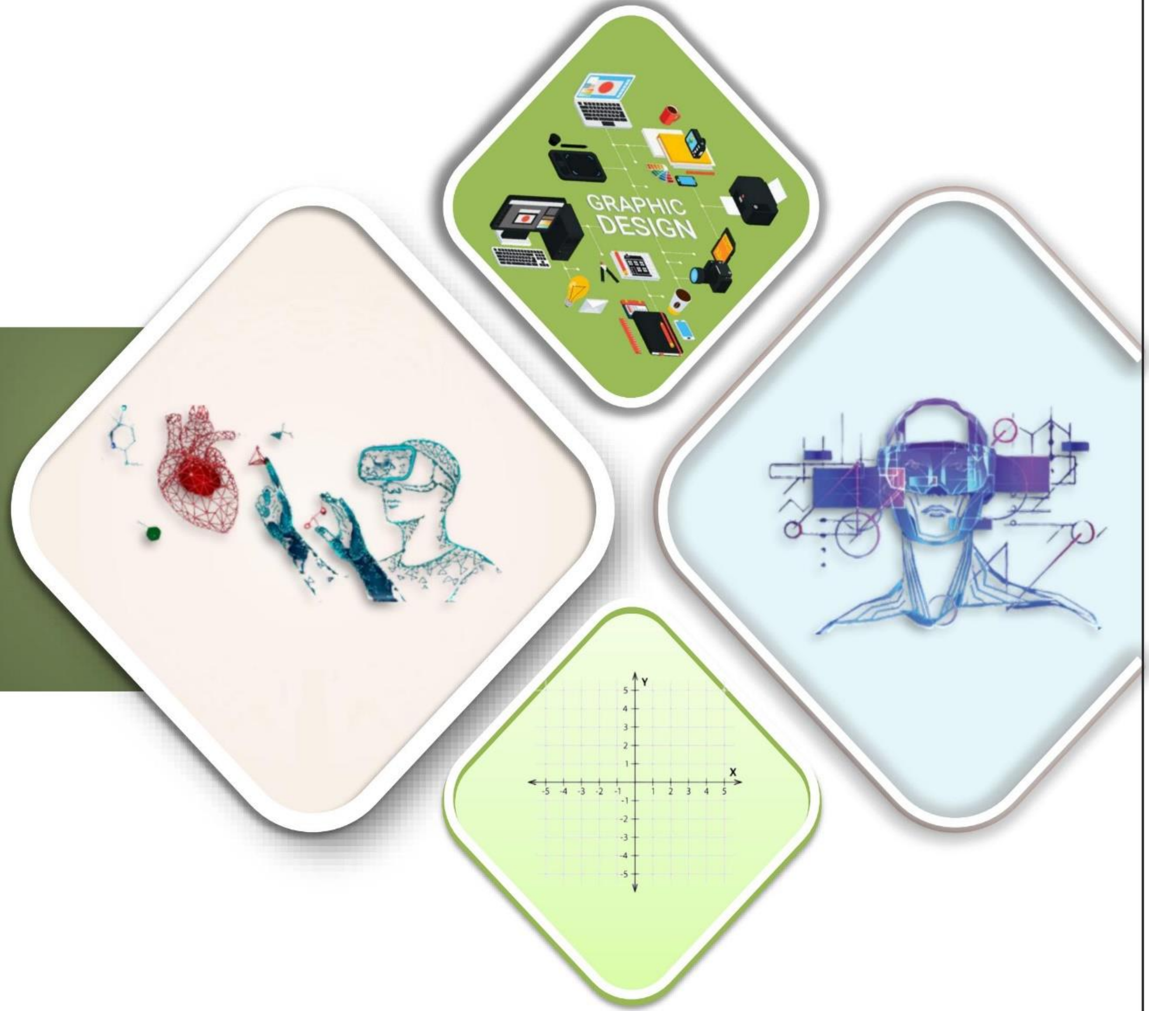
SCHEME : K

Name : ____________________________________

Roll No. : ________________ Year : 20___ 20___

Exam Seat No. : ____________________________

LABORATORY MANUAL FOR COMPUTER GRAPHICS (313001)



COMPUTER ENGINEERING GROUP

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI
(Autonomous) (ISO 9001: 2015) (ISO/IEC 27001:2013)

VISION

To ensure that the Diploma level Technical Education constantly matches the latest requirements of technology industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

\

MISSION

To provide high quality technical and managerial manpower, information and consultancy services to the industry and community to enable the industry and community to face the changing technological & environmental challenges.

QUALITY POLICY

We, at MSBTE are committed to offer the best in class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, Implementation, evaluation and monitoring system along with adequate faculty development programmes.

CORE VALUES

MSBTE believes in the followings:

- Education industry produces live products,
- Market requirements do not wait for curriculum changes.
- Question paper is the reflector of academic standards of educational organization.
- Well-designed curriculum needs effective implementation too.
- Competency based curriculum is the backbone of need based program.
- Technical skills do need support of life skills,
- Best teachers are the national assets.
- Effective teaching learning process is impossible without learning resources.

A Laboratory Manual
for

COMPUTER GRAPHICS (313001)

Semester-III

(CO/ CM/ CW/ HA)

Maharashtra State

Board of Technical Education, Mumbai

(Autonomous) (ISO 9001:2015) (ISO/IEC 27001:2013)

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr. / Ms ………………………………………………………..

Roll No………., of Third Semester of Diploma in……………..……………

of Institute, ……………………………….…………….. (Institute Code: ……………..)

has completed the term work satisfactorily in course **Computer Graphics (313001)** for

the academic year 20…….. to 20……..as prescribed in the curriculum.

Place: …………… Enrollment No:……………

Date: …………… Exam. Seat No: ……………

Subject Teacher **Head of the Department** **Principal**

Seal of

Institution

Preface

The primary focus of any engineering laboratory/ field work in the technical education system is to develop the much needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'K' Scheme curricula for engineering diploma programmed with outcome-base education as the focus and accordingly, relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work making each teacher; instructor and student to realize that every minute of the laboratory time need to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a 'vehicle' to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the 'K' scheme laboratory manual development team designed the practical's to focus on the outcomes, rather than the traditional age old practice of conducting practical's to 'verify the theory' (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the per-determined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read through the concerned practical procedure that they will do the next day and understand the minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. The students will then become aware about the skills they will achieve through procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student-centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

This course provides an introduction to the principles of Computer graphics. In particular, the course will consider methods for object design, transformation, scan conversion, visualization and modeling of real world. The emphasis of the course will be placed on understanding how the various elements that under-lie Computer graphics (algebra, geometry, algorithms) interact in the design of graphics software systems and also enables student to create impressive graphics easily and efficiently.

Although all care has been taken to check for mistakes in this laboratory manual, yet it is impossible to claim perfection especially as this is the first edition. Any such errors and suggestions for improvement can be brought to our notice and are highly welcome.

Program Outcomes (POs) an Program Specific Outcomes (PSOs) to be achieved through the Practical's of this course

PO1. Basic and Discipline specific knowledge: Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.

PO2.Problem analysis: Identify and analyse well-defined engineering problems using codified standard methods.

PO3.Design/ development of solutions: Design solutions for well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

PO4.Engineering Tools, Experimentation and Testing: Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.

PO5.Engineering practices for society, sustainability and environment: Apply appropriate technology in context of society, sustainability, environment and ethical practices.

PO6.Project Management: Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well- defined engineering activities.

PO7.Life-long learning: Ability to analyse individual needs and engage in updating in the context of technological changes.

Program Specific Outcomes (PSOs):

PSO1. Modern Information Technology: Use latest technology for operation and application of information.

PSO2. Information Technology Process: Maintain the information process using modern information and communication technologies

Practical- Course Outcome matrix

Course Outcomes (COs)

a. Manipulate Visual and Geometric information of Images.

b. Develop programs in C applying standard graphics algorithms.

c Perform and Demonstrate basic and composite graphical transformations on given object.

d. Implement various Clipping algorithms.

e. Develop programs to create Curves.

Sr. No	Title of the Practical	CO1	CO2	CO3	CO4	CO5
1	*Write a C program to draw various graphics objects (Pixel, Circle, Line, Ellipse, Rectangle, Triangle, Polygon) using graphics functions	√				
2	*Write a C program to draw line using DDA algorithm.		√			
3	Write a C program to draw line using Bresenham's algorithm.		√			
4	*Write a C program to draw circle using Bresenham's algorithm.		√			
5	*Write a C program for Flood fill algorithm of polygon filling.		√			
6	Write a C program for Boundary fill algorithm of polygon filling.		√			
7	*Write a C program for 2D Translation and Scaling.			√		
8	8 Write a C program for 2D Rotation.			√		
9	*Write a C program for 2D Reflection and Shear.			√		
10	*Write a C program for 3D Translation and Scaling .			√		
11	Write a C program for 3D Rotation.			√		
12	*Write a C program for Line Clipping using Cohen-Sutherland algorithm.				√	
13	Write a C program for Line Clipping using Midpoint Subdivision algorithm.				√	
14	Write a C program for Sutherland Hodgeman Polygon Clipping.				√	
15	Write a C program for Bezier Curve.					√

'*' Marked Practical (LLOs) Are mandatory.

Industry / Employer Expected Outcome

The aim of this course is to attain following Industry Identified Competency through various Teaching Learning Experiences: **'Develop programs using Graphics concepts'** are expected to be developed in you by undertaking the practical of this laboratory manual.

1. Draw various graphics objects.
2. Design CAD-CAM software.
3. Design Game.
4. Design Animation.
5. Perform 2D and 3D transformation.

Guidelines to Teachers

1. For incidental writing on the day of each practical session every student should maintain a dated logbook for the whole semester, apart from this laboratory manual, which s/he has to submit for assessment to the teacher in the next practical session.
2. Teachers should give opportunity to students for hands-on after the demonstration.
3. Assess the skill achievement of the students and COs of each unit.
4. Explain prior concepts to the students before starting of each experiment.
5. List of few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.
6. Teacher should ensure that the practical skill and competencies are developed in the students after the completion of the practical exercise.
7. Teacher may provide additional knowledge and skills to the students even though it’s not covered in the manual but are expected from the students by the industries.
8. Teacher may suggest the students to refer additional related literature of the Technical papers/ Reference books/ Seminar proceedings, etc.
9. Teacher shall assess the performance of students continuously as per norms prescribed by MSBTE.
10. During assessment teacher is expected to ask questions to the students to tap their Achievements grading related knowledge and skills so that student can prepare while submitting record of the practical focus should be given on development of enlisted skills rather than theoretical knowledge.

Instructions for Students

1. Understand the purpose of practical and its implementation.
2. Student shall develop practical skills as expected by the Industries.
3. Listen carefully to the instructions given by the teacher about importance of relevant program outcomes, relevant course outcomes, practical significance, competency and practical skills, practical outcome and the theoretical background during the practical session.
4. Write the answers of the questions allotted by the teacher during practical session.
5. Student should develop the habit of group discussion related to the practical, so that exchange of knowledge/skills could take place.
6. Student shall attempt to develop related hands-on-skills to gain confidence.
7. Student shall refer technical magazines, websites related to the scope of the course.
8. Student should develop habit to submit the practical, exercise continuously and progressively on the scheduled dates and should get the assessment done.
9. Student should be well prepared while submitting the write up of the exercise.
10. Student should not hesitate to ask any difficulty faced during conduct of practical.

Content Page

List of Practical and Progressive Assessment Sheet

Sr. No.	Title of the Practical	Page No	Date of Performance	Date of submission	Assess-ment marks (25)	Dated sign. of teacher	Remark (if any)
1	*Write a C program to draw various graphics objects (Pixel, Circle, Line, Ellipse, Rectangle, Triangle, Polygon) using graphics functions	1					
2	*Write a C program to draw line using DDA algorithm.	9					
3	Write a C program to draw line using Bresenham's algorithm.	16					
4	*Write a C program to draw circle using Bresenham's algorithm.	23					
5	*Write a C program for Flood fill algorithm of polygon filling.	30					
6	Write a C program for Boundary fill algorithm of polygon filling.	37					
7	*Write a C program for 2D Translation and Scaling.	45					
8	8 Write a C program for 2D Rotation.	53					
9	*Write a C program for 2D Reflection and Shear.	61					
10	*Write a C program for 3D Translation and Scaling.	69					
11	Write a C program for 3D Rotation.	79					
12	*Write a C program for Line Clipping using Cohen-Sutherland	87					

	algorithm.						

Sr. No.	Title of the Practical	Page No	Date of Performance	Date of submission	Assess-ment marks (25)	Dated sign. of teacher	Remark (if any)
13	Write a C program for Line Clipping using Midpoint Subdivision algorithm.	98					
14	Write a C program for Sutherland Hodgeman Polygon Clipping.	106					
15	Write a C program for Bezier Curve.	115					
Total							

Practical No:01 Write a C program to draw various graphics objects (Pixel, Circle, Line, Ellipse, Rectangle,Triangle, Polygon) using graphics functions.

I. Practical Significance:

Understanding graphics programming in C is essential for developing applications that require visual representation of data. This includes video games, simulations, and GUI-based applications.

II. Industry, Employer Expected outcomes:

1. Proficiency in using graphics libraries in C.
2. Ability to implement visual elements in software applications.
3. Understanding of basic drawing algorithms.

III. Course Level Learning Outcomes:

Manipulate Visual and Geometric information of Images and **'Develop programs in C applying standard graphics algorithms.'**

1. Apply graphics functions to create various shapes and objects.
2. Understand and implement fundamental drawing algorithms.
3. Develop problem-solving skills related to computer graphics.

IV. Laboratory Learning outcomes:

LLO1.1Implement a C program using different graphics functions.

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

Output Primitives: Output Primitives are basic drawing elements provided by graphics libraries to create various shapes and text on the screen. These include points, lines, polygons, and text. In C programming, particularly with libraries like graphics.h, output primitives are essential for creating graphical applications.

Text Mode: Text Mode is a display mode that focuses on displaying text rather than graphics. It allows you to manipulate and output text using various functions. This mode is used primarily for text-based applications and console outputs. Below are some key text mode functions:

Function	Description
clrscr()	Display single character at cursor position.void clrscr();clrscr(); // Clears the screen
gotoxy(int x, int y)	Positions the cursor at the specified coordinates (x, y). void gotoxy(int x, int y);gotoxy(10, 5); // Moves the cursor to column 10, row 5
putch(char ch)	Display single character at cursor position.void putch(char ch); putch('A'); // Outputs the character 'A'
textcolor(int color)	Sets the text color for subsequent text output. void textcolor(int color);textcolor(RED);

Graphics Mode: Graphics Mode is a display mode that allows for the rendering of graphical elements such as shapes, lines, and images. This mode is used for creating visually rich applications like games and graphical user interfaces. Below are some key graphics mode functions:

#include <graphics.h>: The header file must be included in every graphics program.

initgraph(int *graphdriver, int graphmode, const char pathtodriver):Initializes the graphics system, setting up the graphics mode and specifying the graphics driver to be used.

closegraph(): Closes the graphics system and deallocates any resources used by it, such as memory and hardware resources.

putpixel(x, y, COLOR): Draws a single pixel at the specified (x, y) coordinates with the given color.
line(x1, y1, x2, y2): Draws a line from point (x1, y1) to point (x2, y2)
circle(x, y, radius): Draws a circle with a specified radius centered at (x, y).
ellipse(x,y,start_angle, end_angle, x_radius, y_radius): Draws an ellipse centered at (x, y) with specified radius and angles.
rectangle(left, top, right, bottom): Draws a rectangle with the top-left corner at (left, top) and the bottom-right corner at (right, bottom).
drawpoly(number_of_points, points_array): Draws a polygon using an array of points.

__Sample Code:__

```
#include <graphics.h>
#include <conio.h>
void main()
  {
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\Turboc3\\BGI");
   putpixel(100, 100, WHITE);          // Draw Pixel
   line(150, 150, 250, 250);           // Draw Line
   circle(300, 300, 50);               // Draw Circle
   ellipse(400, 400, 0, 360, 50, 25);  // Draw Ellipse
   rectangle(450, 150, 600, 300);      // Draw Rectangle
   // Draw Triangle
   line(200, 200, 250, 100);
   line(250, 100, 300, 200);
   line(300, 200, 200, 200);
   // Draw Polygon (example: pentagon)
   int poly[] = {320, 150, 370, 200, 350, 250, 290, 250, 270, 200, 320, 150};
   drawpoly(6, poly);
  getch();
   closegraph();
  }
```

VII. Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-if preferable), RAM minimum 2 Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

IX. Procedure :

1. Setup Development Environment:
   Install a C compiler and graphics library.
   Configure the development environment to include the graphics library.
2. Initialize Graphics Mode in C:
   ```
   #include <graphics.h>
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "path");
   ```
3. Draw Graphics Objects:
   Pixel: `putpixel(x, y, COLOR);`
4. Close Graphics Mode: closegraph();
5. Compile and Run the Program.

X. Resources used

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

XI. Result :

………………………………………………………………………………………

………………………………………………………………………………………

XII. Conclusions and recommendation:

………………………………………………………………………………………

………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. What are the advantages of using a graphics library in C?
2. How can you modify the code to draw a filled shape?
3. Explain the differences between vector and raster graphics.

(Space for Answer)

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

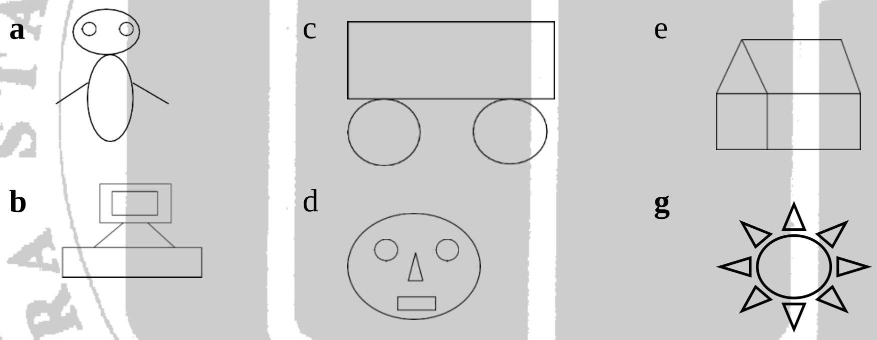
…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XIV. Excersize:

1. Write a program to draw following output



2. Write a program to draw a line from pixel (50, 50) to pixel (200, 200).
3. Write a program to draw a circle with a radius of 100 pixels centered at (300, 300).
4. Write a program to draw an ellipse centered at (250, 250) with x-radius of 150 and y-radius of 100.

(Space for Answer)

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….…………..

2. ……….……….…………..

3. ……….……….…………..

4. ……….……….…………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:02 Write a C program to draw line using DDA algorithm.

I. Practical Significance:

The Digital Differential Analyzer (DDA) algorithm is a fundamental method in computer graphics used for drawing lines between two points. Understanding and implementing the DDA algorithm is crucial for various applications such as drawing shapes, rendering images, and creating graphical user interfaces

II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms, particularly focusing on line drawing algorithms.

IV. Laboratory Learning outcomes:

LLO 2.1 Implement a C program todraw line using DDA algorithm

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

DDA (Digital Differential Analyzer) is a line drawing algorithm used in computer graphics to generate a line segment between two specified endpoints. It is a basic element in computer graphics. To draw a line, you need two points between which you can draw a line. DDA is used for interpolation of variables over an interval between start and end point. DDA are used for rasterization of lines, triangle and polygons. They can be extended to nonlinear functions, such as perspective correct texture mapping, quadratic curves, and traversing pixel

In its simplest implementation for linear cases such as lines, the DDA algorithms interpolates value in interval by computing for each xi the equation xi=xi-1+1, yi=yi-1+1.

VII. Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-if preferable), RAM minimum 2 Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

IX. Procedure :

```
Step1: Start Algorithm
Step2: Read the input of the 2 end points of the line as ( x1, y1),(x2, y2) such that
x1 !=x2 and y1!=y2
Step3: Calculate dx = x2-x1 and  Calculate dy = y2-y1
Step4: if abs (dx) >= abs (dy)
          then step = abs (dx)
          else
          step  abs(dy)
Step5: xinc=dx/step and  yinc=dy/step
Step 6: x=x1=0.5 and y=y1=0.5
Step 7:for(k0;k<step;k++)
          {
          x=x+xinc
          y=y+yinc
          putpixel(x,y,color)
}
```

X. Resources used:

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

‘C’ program code

XI. Result :

……………………………………………………………………………………………

……………………………………………………………………………………………

XII. Conclusions and recommendation:

……………………………………………………………………………………………

……………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. Define the term Rasterization.
2. Write slope intercept form of a line
3. Write advantage and disadvantage of DDA algorithm

(Space for Answer)

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XIV. Exercise:

1. Give following values for every iteration of DDA algorithm to draw a line from (3,4) to(6,8)

dx	dy	step	Xinc	Yinc

2. Give following values for every iteration of DDA algorithm to draw a line from (-5,-5) to(-12,-12)

dx	dy	step	Xinc	Yinc

3. Implement DDA algorithm to Draw line using For loop

(Space for Answer)

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..

2. ……….……….………..

3. ……….……….………..

4. ……….……….………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No: 03 Write a C program to draw a line using Bresenham’s algorithm.

I. Practical Significance:

Bresenham's line algorithm determines the points of an n-dimensional raster that should be selected to form a close approximation to a straight line between two points. It is commonly used to draw line primitives in bitmap images. Bresenham's line drawing algorithm is fundamental in computer graphics for drawing lines efficiently. It minimizes computational overhead by using integer arithmetic, making it suitable for raster displays where lines need to be rendered quickly.

II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Write and save a simple C program.
3. Set up graphics driver, mode, and directory to run graphics programs.
4. Compile C programs using Turbo C.
5. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms, particularly focusing on line drawing algorithms.

1. Develop programs in C using standard graphics algorithms.
2. Apply Bresenham's algorithm to create lines in graphics applications.
3. Understand and implement fundamental drawing algorithms.

IV. Laboratory Learning outcomes:

LLO 3.1: Implement a C program to draw a line using Bresenham’s algorithm.

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

Bresenham's line algorithm determines the points of an n-dimensional raster that should be selected to form a close approximation to a straight line between two points. It is used to draw straight line primitives in bitmap images, utilizing only integer addition, subtraction, and bit shifting. It is an Incremental error algorithm and one of the earliest algorithms developed in the field of computer graphics.

Consider a line with an initial point (x1, y1) and a terminal point (x2, y2) in device space. Let $\Delta x = x2 - x1$ and $\Delta y = y2 - y1$. We define the driving axis (DA) to be the x-axis if $|\Delta x| >= |\Delta y|$, and the y-axis if $|\Delta y| > |\Delta x|$. The DA is used as the "axis of control" for the algorithm and is the axis of maximum movement. Within the main loop of the algorithm, the coordinate corresponding to the DA is incremented by one unit. The coordinate corresponding to the other axis is only incremented as needed.

VII.Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-i5preferable), RAM minimum 2 Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
2.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

Ensure the graphics library is properly installed and configured.
Check for compatibility of the graphics library with the compiler.
Handle errors and exceptions to avoid crashes.

IX. Procedure :

For lines |m| ≤ 1, the Bresenham's line drawing algorithm

1. Read the end points of the line and store left point in (x_0, y_0).
2. Plot (x_0, y_0), the first point.
3. Calculate constants Δx, Δy, 2Δy and (2Δy - 2Δx), and obtain a decision parameter p_0.
4. Perform the following test for each x_k, starting at k = 0
5. if p_k< 0, then next plotting point is (x_{k+1}, y_k) and P_{k+1}=P_k+2Δy and Otherwise, the next point to plot is (x_{k+1}, y_{k+1}) and P_{k+1}=P_k+2Δy-2Δx
6. Repeat step 4 Δx times. For a line with positive slope

X. Resources used:

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm

Flowchart:

‘C’ program code

XI. Result :

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII. Conclusions and recommendation:

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. What are the advantages of Bresenham's algorithm over the DDA algorithm?
2. How can Bresenham's algorithm be modified to draw lines with different slopes?
3. Explain the role of the decision parameter in Bresenham's algorithm.

(Space for Answer)

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

XIV. Exercise:

1. Give following values for every iteration of Bresenham's algorithm to draw a line from (3,4) to(6,8)

dx	dy	step	Xinc	Yinc

2. Give following values for every iteration of Bresenham's **algorithm to draw a line from (-6,-6) to(-14,-14)**

dx	dy	step	Xinc	Yinc

3. Implement Bresenham's algorithm to Draw line using While loop

(Space for Answer)

……………………………………………………………………………………………………

……………………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..

2. ……….……….………..

3. ……….……….………..

4. ……….……….………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:04 - Write a C program to draw a circle using Bresenham’s algorithm.

I. Practical Significance:

Bresenham's circle drawing algorithm is a fundamental algorithm in computer graphics that allows for the efficient and accurate rendering of circles. This algorithm uses integer arithmetic to plot points of a circle, which enhances performance on raster displays by avoiding floating-point calculations.

II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms.

1. Develop and implement programs in C using advanced graphics algorithms.
2. Apply Bresenham's algorithm to draw circles in graphics applications.
3. Gain a deeper understanding of fundamental drawing algorithms in computer graphics.

IV. Laboratory Learning outcomes:

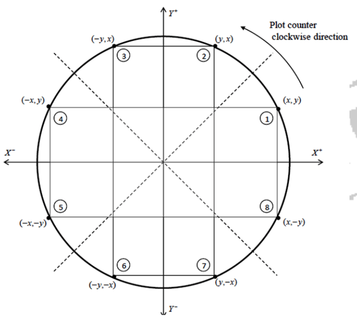
LLO 4.1 Implement a C program to draw circle using Bresenham's algorithm.

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

The equation of a circle $x^2 + y^2 = r^2$, where (x, y) are the coordinates of the center and (r) is the radius. To draw a circle on a computer, we can utilize algorithms like the Bresenham's circle drawing algorithm or the midpoint circle drawing algorithm.

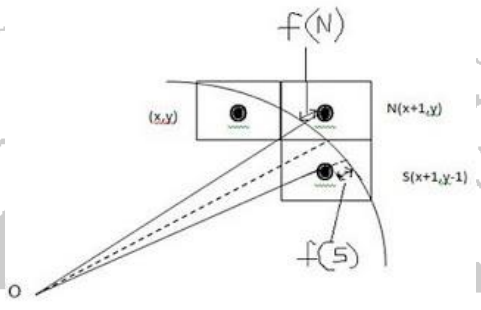


Due to the discrete nature of raster displays, displaying a continuous arc is challenging. Instead, we must select the nearest pixel position to complete the arc, resulting in an approximation of the desired curve. These algorithms efficiently plot points along the circumference of the circle, ensuring that it is recognizable despite the limitations of raster displays.

Bresenham's Circle Algorithm

Bresenham's circle algorithm is based on the idea of determining the subsequent pixel required to draw a circle. (x+1, y) is used to find the next pixel to draw the circle.

In the context of Bresenham's circle algorithm, **f(n**)typically refers to the decision variable used to determine the next pixel position along the circle.Similarly, **f(s)** represents the value of the decision variable at the symmetric position.As for **(x+1, y) and (x+1, y-1)** , they are the candidate pixels that are evaluated to determine the next pixel position along the circle's circumference. The decision is based on choosing the midpoint that is closer to the true circle



VII.Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-if preferable), RAM minimum 2 Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
2.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

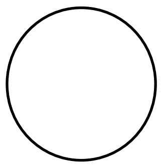
IX. Procedure :

1. Start
2. Declare x, y, r, D, xc, yc where (xc, yc) are center coordinates, x and y points to be plotted, r
is the radius, D is decision factor.
3. Calculate decision parameter D as: D = 3–(2 * r)
4. Initialize x = 0, y = r
5. Compute next pixels on circle based upon decision parameter

```
While x ≤ y
Plot (x, y)
if D<0,
then D = D + 4x + 6
Else
 D = D + 4(x - y) + 10
y = y – 1
end if
```

```
        x = x + 1
6. End
```

Sample Output of The Source Example:



X. Resources used

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

‘C’ program code

XI. Result :

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII. Conclusions and recommendation:

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. Write equation for Circle.
2. Write algorithm to draw 8-way symmetry of circle.
3. How the Value of Decision Parameter (d) is calculated. Explain with Example.

(Space for Answer)

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XIV. Excersize:

1. **Calculate pixels for a circle with radius 10 using Bresenham's Circle Algorithm**

d	x	y	Plot(x,y)

2. **Draw a circle with center(50,50) and radius 20 by using Bresenham's Circle Algorithm**

(Space for Answer)

……………………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..
2. ……….……….………..
3. ……….……….………..
4. ……….……….………..

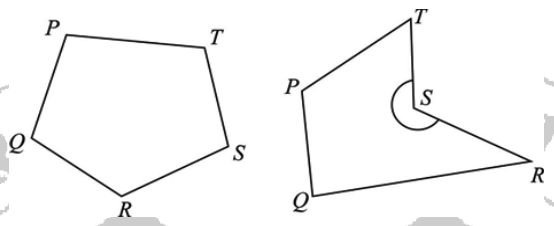
Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:05 - Write a C program for Flood Fill Algorithm of Polygon Filling.

I. Practical Significance:

A polygon consists of a series of connected line segments forming a closed shape. To fill polygons with specific colors, you need to identify which pixels lie on the border of the polygon and which pixels fall inside it.

The Flood Fill algorithm is a fundamental technique in computer graphics used for filling bounded areas with a specific color. This algorithm is widely used in various applications such as paint programs, games, and image processing to color enclosed areas efficiently.



II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:

Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms.

1. Develop and implement C programs using standard graphics algorithms.
2. Apply flood fill algorithm for polygon filling in graphical applications.
3. Understand the principles of area filling algorithms in computer graphics.

IV. Laboratory Learning outcomes:

LLO 5.1: Implement a C program for the Flood Fill algorithm.

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

The Flood Fill algorithm begins at a specified seed point and examines neighboring pixels. Instead of checking for a boundary color, it checks for a specified interior color and replaces it with a new color. The algorithm can be implemented using a stack-based approach (depth-first search) or a queue-based approach (breadth-first search). It can be applied using either the 4-connected or 8-connected region method.

VII. Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-i5preferable), RAM minimum 2 Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

IX. Procedure :

```
Flood-fill(node,target-color,replacement-color):
    1. If target-color is equal to replacement-color, return
    2. If the color of node is not equal to target-color,return.
    3. Set color  of node to replacement-color.
    4. Perform flood-fill(one step to south node, target-color,replacement-color).
    5. Perform flood-fill(one step to north node, target-color,replacement-color).
    6. Perform flood-fill(one step to west node, target-color,replacement-color).
    7. Perform flood-fill(one step to east node, target-color,replacement-color).
    8. Return
```

X. Resources used

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

‘C’ program code

XI. Result :

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XII.Conclusions and recommendation:

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. Define Polygon.
2. Explain types of polygon.
3. List coordinates of neighboring pixel in 8-connected method for seed pixel with coordinates(x,y).
4. List coordinates of neighboring pixel in 4-connected method for seed pixel with coordinates(x,y)
5. Explain inside-outside test of polygon.

(Space for Answer)

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XIV. Exercise:

i. WAP to draw hexagon and fill hexagon with pink color using flood fill algorithm with 8-connected method.
ii. WAP to draw triangle(USING LINE FUNCTION) and fill hexagon with red color using flood fill algorithm with 4-connected method.

(Space for Answer)

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……………………………..

2. ……………………………..

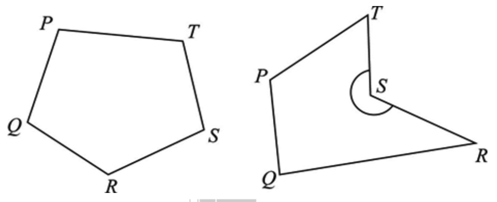
3. ……………………………..

4. ……………………………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:06 - Write a C program for Boundary fill algorithm of polygon filling.

I. Practical Significance:



II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms.

1. Develop and implement C programs using standard graphics algorithms.
2. Apply Boundary fill algorithm for polygon filling in graphical applications.
3. Understand the principles of area filling algorithms in computer graphics.

IV. Laboratory Learning outcomes:

LLO 6.1 Implement a C program forBoundary fill algorithm.

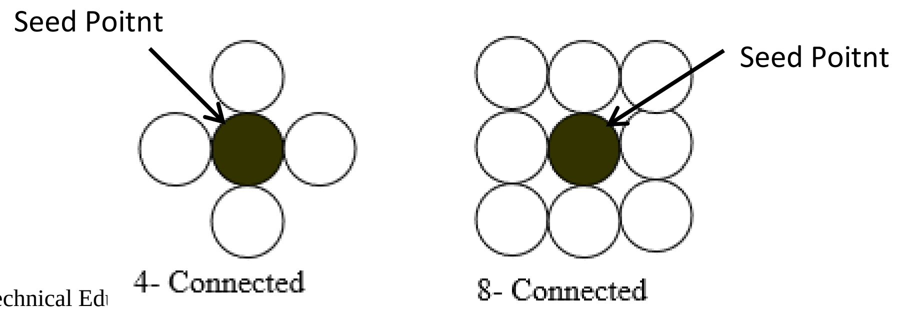
V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

Boundary fill algorithm information

Boundary fill algorithm picks a point inside that is a seed point of an object and starts to fill until it hits the boundary of the object. If boundary pixels are not reached, pixels are highlighted and the process is continued until boundary pixel are reached.
The color of the boundary and the color that we fill should be different for this algorithm work.
In this algorithm, we assume that color of the boundary is the same for the entire object.
The Boundary Fill Algorithm can be executed using two different connectivity methods:
4 - Connected boundary fill, 8 - Connected boundary fill



4- Connected boundary fill

This method fills pixels that are directly above, below, left, and right of the current pixel.

```
void boundary_fill_4(int x, int y, int boundary_color, int fill_color)
{
 int current = getpixel(x, y);
if(current != boundary_color && current != fill_color)
 {
 putpixel(x, y, fill_color);
 boundary_fill_4(x + 1, y, boundary_color, fill_color);
boundary_fill_4(x, y + 1, boundary_color, fill_color);
boundary_fill_4(x - 1, y, boundary_color, fill_color);
boundary_fill_4(x, y - 1, boundary_color, fill_color);
 }
}
```

8- Connected boundary fill

This method fills pixels that are directly above, below, left, right, and the four diagonal pixels around the current pixel.

```
void boundary_fill_4(int x, int y, int boundary_color, int fill_color)
{
 int current = getpixel(x, y);
if(current != boundary_color && current != fill_color)
 {
 putpixel(x, y, fill_color);
 boundary_fill_4(x + 1, y, boundary_color, fill_color);
boundary_fill_4(x - 1, y, boundary_color, fill_color);
boundary_fill_4(x, y + 1, boundary_color, fill_color);
boundary_fill_4(x, y - 1, boundary_color, fill_color);
boundary_fill_4(x + 1, y + 1, boundary_color, fill_color);
boundary_fill_4(x - 1, y - 1, boundary_color, fill_color);
boundary_fill_4(x +1, y - 1, boundary_color, fill_color);
boundary_fill_4(x - 1, y + 1, boundary_color, fill_color);
 }
}
```

VII.Required Resources/apparatus/equipment with specification:

Sr.	**Name of Resource**	**Specification**	**Qty.**	**Remarks**
1.	Hardware: Computer System	Computer(i3-i5preferable), RAM minimum 2Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

IX. Procedure :

Step 1- Initialize the value of seed point seedx, seedy, fcolor, bcolor.
Step 2- Define the boundary values of polygon.
Step 3- check if the current seed point is of default color, then repeat the step3 and 5 til boundary pixels reached.
Step 4- Change the default color with fill color at the seed point
Step 5 - Recursively follow the procedure with four neighborhood points.
Step 6- Exit.

X. Resources used

Sr. No.	**Name of Resource**	**Specification**
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

'C' program code

XI. Result :

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XII. Conclusions and recommendation:

…………………………………………………………………………………………

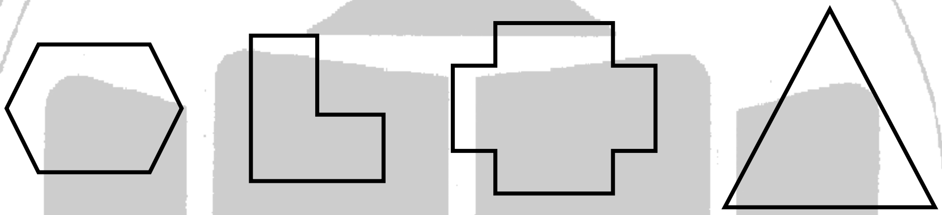
…………………………………………………………………………………………

…………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. Compare 4 - connected and 8 - connected method to fill polygon.
2. Give Difference between flood fill and boundary fill.
3. Identify type of polygon in following diagram.



(Space for Answer)

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XIV. Exercise:

iii. WAP to draw Pentagon and fill it with red color using boundary fill algorithm with 8-connected method.

iv. WAP to draw trapezoid and fill it with blue color using boundary fill algorithm with 4 connected method

(Space for Answer)

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……………………………………..

2. ……………………………………..

3. ……………………………………..

4. ……………………………………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:07 - Write a C program for 2D Translation and Scaling.

I. Practical Significance:

One of the most common and important tasks in Computer Graphics is to transformation the coordinates (positions, orientation, and size) of objects. Transformation are one of the primary vehicles used in computer graphics to manipulate object in two or three dimensional space. There are three different types of transformation such as translation,scaling up or down , rotation,shearing,etc. When a transformation takes place on 2D plane, it is called 2D transformation.

II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms.

1. Develop and implement C programs using standard graphics algorithms.
2. Apply translation, scaling algorithm to draw graphical image.
3. Understand the principles of 2D translation, scaling in computer graphics.

IV. Laboratory Learning outcomes:

LLO 7.1 Implement a C program for 2D Translation and Scaling

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

Translation:

Giving linear displacement to an object along X and Y axis in a particular direction in a plane is translation. A translation is a process of changing the position of an object in a straight path from one coordinate location to another. We can translate a point 2D by adding translation coordinates(t_x, t_y) to original coordinates(X,Y) to get the new coordinates(x',y').

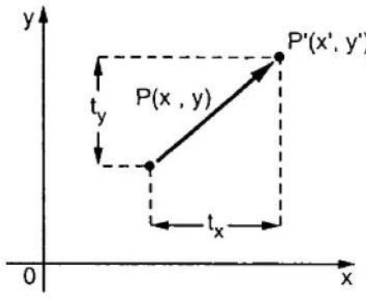


Figure 1: 2D Translation

By adding translation vector,

$$\mathbf{x'} = \mathbf{x} + \mathbf{t_x}$$
$$\mathbf{y'} = \mathbf{y} + \mathbf{t_y}$$

The pair (t_x, t_y) is called the translation vector or shift vector. The above equations can also be represented using the column vectors.

$$P = \frac{[X]}{[Y]} \qquad P' = \frac{[X']}{[Y']} \qquad T = \frac{[t_x]}{[t_y]}$$

We can write it as-

$$P' = P + T$$

Scaling:

Scaling is a process that increases and decreases original size of an object by virtue of it the object can be made big or small. This transformation makes necessary changes in the size of an object keeping its original shape in tact. The operation can be carried out for an object by multiplying the coordinates value(x, y) of each vertex by scaling factors sx and sy to produce the transformed coordinates (x', y'), only condition is to ensure that the base point is remained unaltered. Changing the size of an compress the dimension of the object. Scaling can be archived by multiplying the original coordinates of the object with the scaling factor to get desired result.

Let us assume that the original coordinates are (X, Y) the scaling factors are(Sx, Sy) and the product coordinates are(X',Y'). This can be mathematically represented as shown below-

$$\mathbf{X' = X * Sx \quad and \quad Y' = Y * Sy}$$

The scaling factor Sx, Sy scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below.

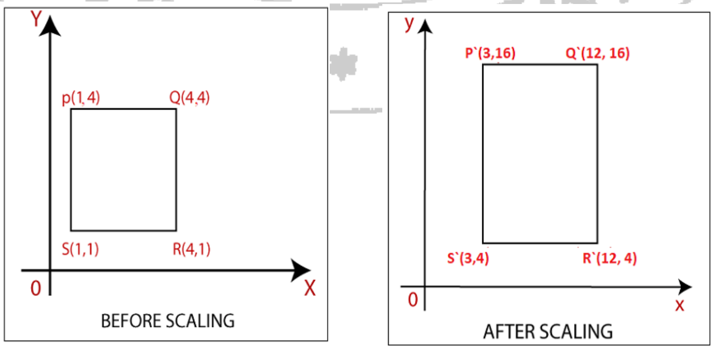
$$\begin{pmatrix} X' \\ Y' \end{pmatrix} = \begin{pmatrix} X \\ Y \end{pmatrix} * \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

OR

$$\mathbf{P' = P * S}$$

Where S is scaling Matrix.

The Scaling s shown in following figure.



VII.Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-i5preferable), RAM minimum 2Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.
- Handle errors and exceptions to avoid crashes.

IX. Procedure :

Step 1: Start the program
Step 2: Input the object coordinates
Step 3: For translation
a) Enter the translation factor tx,ty.
b) Move original coordinate position(x, y) to new position(x1, y1) ie. x=x+x1, y=y+y1
c) Display the object after translation
Step 4: For scaling
a) Enter the scaling factor sx,sy.
b) Move transform coordinate (x1, y1) ie. X1=x*sx, y1=y+sy
c) Display the object after scaling
Step 5: Stop Program

X. Resources used

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm

Flowchart:

'C' program code

XI. Result :

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII.Conclusions and recommendation:

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

1. Write a transformation matrix for 2D scaling.
2. Write a transformation matrix for 2D Translation.
3. What does scaling transformation do?
4. Whether size of object remains same or changed in case of translation?

(Space for Answer)

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XIV. Exercise:

1. Translate the polygon with co-ordinates A(2,5), B(7,10), C(10,2) by 3 units in x direction and 5 units in y direction.
2. Scale the polygon with co-ordinates A(2,5), B(7,10), C(10,2) by 2 units in x direction and 2 units in y direction.
3. Give a 3X3 homogeneous co-ordinates transformation matrix for each of the following translation
   a) Shift image to right 3 units
   b) Shift image up 2 units
   c) Move the image down 0.5 units and right 1 unit
   d) Move the image down 1.5 units and left 5 units.

(Space for Answer)

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….………..………..

2. ……….………..………..

3. ……….………..………..

4. ……….………..………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No:08 - Write a C program for 2D Rotation.

I. Practical Significance:

One of the most common and important tasks in Computer Graphics is to transformation the coordinates (positions, orientation, and size) of objects. It is a process by virtue of which the object can rotate to specific angle. A 2-D rotation is applied to an object by re-positioning it along a circular path in the x, y plane. To generate a rotation, we give a rotation angle ‘θ’ and the position rotation point about which the object is to be rotated.

II. Industry, Employer Expected outcomes:

This practical aims to develop the following skills:
Develop C programs to draw basic graphics objects.

1. Write syntax for graphics functions.
2. Set up graphics driver, mode, and directory to run graphics programs.
3. Compile C programs using Turbo C.
4. Debug and execute programs effectively.

III. Course Level Learning Outcomes:

Develop programs in C applying standard graphics algorithms.

1. Develop and implement C programs using standard graphics algorithms.
2. Apply Rotation algorithm to draw graphical image.
3. Understand the principles of 2D Rotation in computer graphics.

IV. Laboratory Learning outcomes:

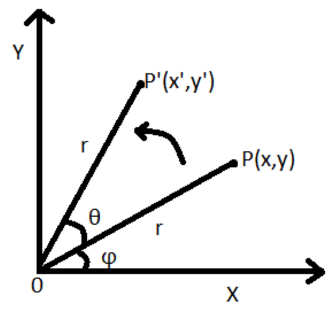
LLO 8.1 Implement a C program for 2D Rotation.

V. Relevant Affective Domain Related Outcomes:

- Demonstrate patience and attention to detail when debugging graphics code.
- Show creativity in designing visual representations.

VI. Relevant Theoretical Domain Related outcomes(with Diagram if required):

Giving angular displacement to an object is a rotation. In rotation, we rotate object at particular angle θ (theta) from its origin. From the following figure, we can see that the point P(x, y) is located at angle Φ from the horizontal X coordinate with distance r from the origin.
Consider, you want to rotate it at angle θ. after rotating it ti a new location, you will get a new point p’(x’ y’).



The original coordinate of point P(X, Y) can be represented as-

$$X=r\cos\Phi …….(1)$$
$$Y= r\sin\Phi …….(2)$$

Same way we can represent the point P'(X' Y') As-

$$\mathbf{x' = r\cos(\emptyset + \theta) = r\cos\emptyset\cos\theta - r\sin\emptyset\sin\theta …….(3)}$$
$$\mathbf{y' = r\sin(\emptyset + \theta) = r\cos\emptyset\sin\theta + r\sin\emptyset\cos\theta ……..(4)}$$

Substituting equation(1) &(2) in (3)&(4) respectively, we will get transformation equations for rotating point P(x,y) through an angle θ abut the origin:

$$\mathbf{x' = x\cos\theta - y\sin\theta}$$
$$\mathbf{y' = x\sin\theta + y\cos\theta}$$

Representing the above equation in the matrix form.

$$[X'Y'] = [XY]\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \text{ or}$$

$$\mathbf{P' = P.R}$$

Where R is rotation matrix is represented as

$$\mathbf{R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}}$$

The rotation angle can be positive and negative(I.e. clockwise and counterclockwise) For positive rotation angle we use above equation matrix. However, for negative angle rotation, the matrix will change as shown below-

$$\mathbf{R = \begin{bmatrix} \cos(-\theta) & \sin(-\theta) \\ -\sin(-\theta) & \cos(-\theta) \end{bmatrix}}$$

$$\mathbf{= R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} (\therefore cps(-\theta) = \cos\theta \text{ and } \sin(-\theta) = -\sin\theta)}$$

VII.Required Resources/apparatus/equipment with specification:

Sr.	Name of Resource	Specification	Qty.	Remarks
1.	Hardware: Computer System	Computer(i3-i5preferable), RAM minimum 2Gb and onwards but not limited	As per batch Size	For All Experiments
2.	Operating System	Windows XP/ Windows 7/LINUX version 5.0 or later		
3.	Software	Turbo C/C++ Version 3.0 or later with DOSBOX		

VIII. Precautions to be followed :

- Ensure the graphics library is properly installed and configured.
- Check for compatibility of the graphics library with the compiler.

- Handle errors and exceptions to avoid crashes.

IX. Procedure :

Step 1: Start the program
Step 2: Input the object coordinates
Step 3: For Rotation

d) Enter the radian for rotation angle θ.
e) Rotate point at position (x, y, z) through an angle θ about the origin
$x1 = x\cos\theta - y\sin\theta$
$y1 = x\sin\theta + y\cos\theta$
f) Display the object after translation

Step 4: Stop Program

X. Resources used

Sr. No.	Name of Resource	Specification
1.	Computer System with broad specifications	
2.	Software	
3.	Any other resource used	

Algorithm:

Flowchart:

‘C’ program code

XI. Result :

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII.Conclusions and recommendation:

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XIII. Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design more such questions so as to ensure the achievement of identified CO.

4. Write a Rotation matrix for 2D Rotation.
5. Write a Rotation .matrix for 2D Rotation 30°

(Space for Answer)

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………

XIV. Exercise:

1. A point is rotated counterclockwise by an angle of 45°. Find the rotation matrix and the resultant point.
2. Perform a counterclockwise 45° rotation of triangle A(2,3) B(5,5) C(4,4) about point(1,1)

(Space for Answer)

…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………
…………………………………………………………………………………………

XV. References/Suggestions for further reading: include Website. Link:

1. https://books.google.co.in/books?isbn=8184317379
2. https://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
3. https://books.google.com/books/about/Computer_Graphics.html?id=XgAeEAAAQBAJ
4. https://www.freebookcentre.net/CompuScience/Free-Computer-Graphics-Books-Download.html#google_vignette

XVI. Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……………………………..
2. ……………………………..
3. ……………………………..
4. ……………………………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.9: Write a C program for 2D Reflection and Shear.

I Practical Significance

One of the most common and important tasks in Computer graphics is to transform the coordinates (position, orientation, and size) of objects. It is a transformation which slants or bends an object to specified direction. There are two types of shearing transformation available in Computer graphics. One which slants x coordinate values is known as X shearing and one that slants y coordinate values is known as Y shearing. Irrespective of shearing only one co-ordinate is change its coordinate and other values are same. In Reflection the mirror copy of an object is generated based on the axis to which reflection is to be made.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Perform and Demonstrate basic and composite graphical transformations on given object.

IV Laboratory Learning Outcome

Implement a C program for 2D Reflection and Shear.

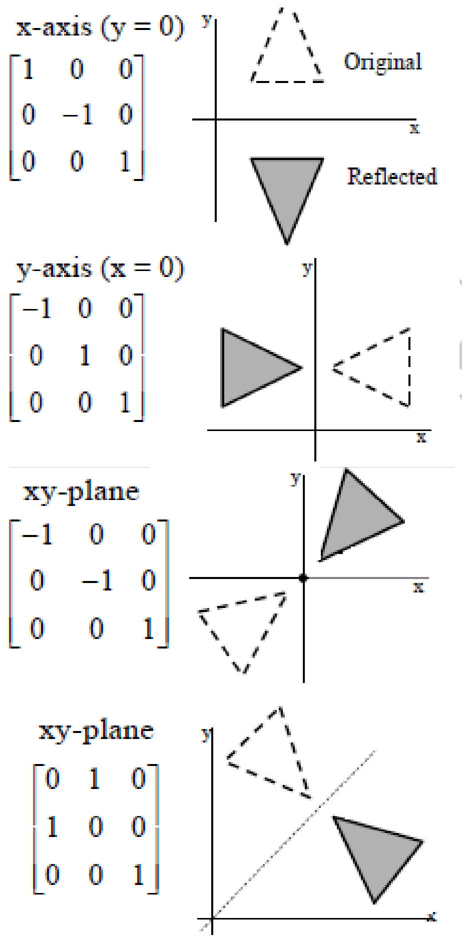
V Relevant Affective domain related Outcome(s)

1. Follow safety practices.
2. Maintain tools and equipment.
3. Follow ethical practices.

VI Relevant Theoretical Background

1. Reflection:-

Reflection is a transformation that results into a mirror image of original object. In reflection transformation, the size of the object does not change. The mirror image of any image for 2D reflection is generated with respect to the "Axis of Reflection". For that we need to rotate main object 180 Degrees about the reflection axis. Reflection transformation is generally implemented with respect to the coordinate axes or its coordinate origin as the scaling transformation with t minus (negative) scaling factors.



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

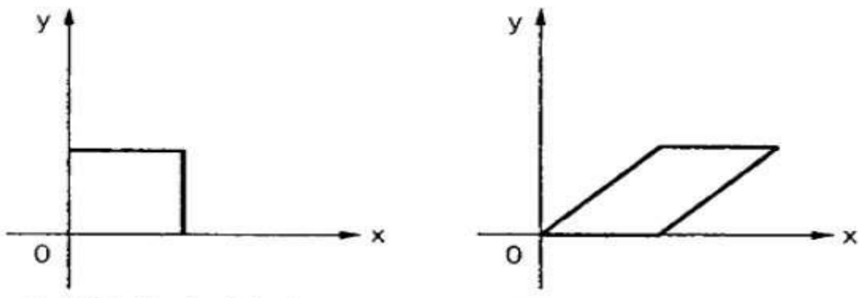
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2. Shear:

Slanting the shape of an object is shearing transformation. There are two shear transformations X-Shear and Y-Shear. One shifts X coordinates values and other shifts Y coordinate values. In fact, in both the cases only one coordinate changes its coordinates and other preserves its values. Shearing is also called as Skewing.

X-Shear

The X-Shear preserves the Y coordinate and changes are made to X coordinates, which causes the vertical lines to tilt right or left as shown in below figure.



(a) Original object (b) Object after x shear

The transformation matrix for X-Shear can be represented as –

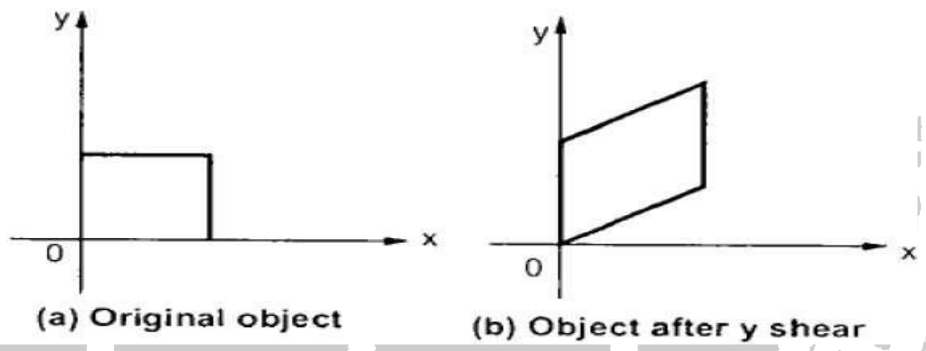
$$\begin{pmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$X' = X + Sh_x \,.\, Y$$

$$Y' = Y$$

Y-Shear

The Y-Shear preserves the X coordinates and changes the Y coordinates which causes the horizontal lines to transform into lines which slopes up or down as shown in the following figure.



The Y-Shear can be represented in matrix from as –

$$\begin{pmatrix} 1 & 0 & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$Y' = Y + Sh_y \,.\, X$$

$$X' = X$$

Procedure:

Step 1: Start the program.
Step 2: Input the object coordinates
Step 3: For Shearing

a) Input the shearing factors shx and shy.
b) Shearing related to x axis: Transform coordinates x1=x+shx*y and y1=y.
c) Shearing related to y axis: Transform coordinates x1=x and y1=y+shy*x.

d) Input the xref and yref values.

e) X axis shear related to the reference line y-yref isx1=x+shx(y-yref) and y1=y.

f) Y axis shear related to the reference line x=xref isx1=x

g) Display the object after shearing

Step 4: For Reflection

Reflection can be performed about x axis and y axis.

a) Reflection about x axis: The transformed coordinates are x1=a and y1=-y.

b) Reflection about y axis: The transformed coordinates are x1=x and y1=y.

c) Display the object after reflection

Step 5: Stop the Program.

VII Algorithm

VIII Flow Chart

IX C Program Code

X Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

XI Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (**;**).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XII Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XIII Result (Output of Program)

XIV Conclusion

XV Practical Related Questions

1. Write the transformation matrix for 2D Reflection.
2. Write the transformation matrix for 2D Shear.
3. Differentiate between X-shear and Y-shear.
4. Define Reflection and Shearing.

XVI Exercise

1. A point (4, 3) is rotated counterclockwise by an angle of 45^0. Find the rotation matrix and the resultant point.Apply the Shearing transformation to square with A(0,0),B(1,0),C(1,1) and D(0,1) as given below :
   i. Shear parameter value of 0.5 relative to the line Yref= -1;
   ii. Shear parameter value of 0.5 relative to the line Xref= -1;
2. Apply shearing transformation to square with A(0,0),B(1,0),C(1,1) and D(0,1).If Sh_x =0.5 then find the resultant co-ordinates.

[Space for Answer]

XVII References / Suggestions for further Reading Software/Learning Websites

http://www.it.hiof.no/~borres/j3d/math/twod/p-twod.html

XVIII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..

2. ……….……….………..

3. ……….……….………..

4. ……….……….………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.10: Write a C program for 3D Translation and Scaling.

I Practical Significance

3D Computer graphics are graphics that use a three dimensional representation of geometric data that is stored in the Computer for the purposes of performing calculations and rendering 2D images.3D transformations are extended from 2D methods by including considerations for the Z coordinate.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Perform and Demonstrate basic and composite graphical transformations on given object.

IV Laboratory Learning Outcome

Implement a C program for 3D Reflection and Shear.

V Relevant Affective domain related Outcome(s)

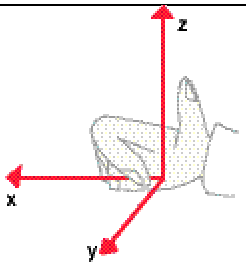
1. Plan, construct, compile, debug and test programs.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

Transformations in 3D:-

As, 2D transformations, but has a coordinate system with three axes as a basis. In this material all reasoning in space is done in a right hand system.

This means that if I put my right hand vertically down, like in Karate, with my fingers along the positive x-axis, and bend the hand towards the y-axis, the thumb will point up along the positive z-axis.



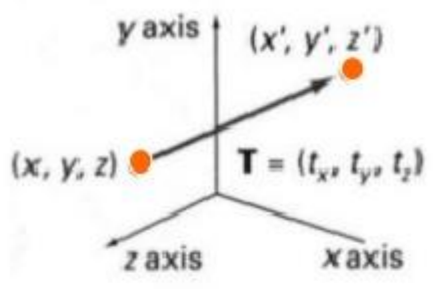
We use homogeneous coordinates from the beginning. This means that the general transformation matrix is a 4x4 matrix, and that the general vector form is a column vector with four rows.

$P_2 = M \cdot P_1$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

1. Translation:

Moving of object is called translation.



A translation in space is described by t_x, t_y and t_z. It is easy to see that this matrix realizes the equations:

$P' = T.P$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

This matrix representation is equivalent to following three equations.

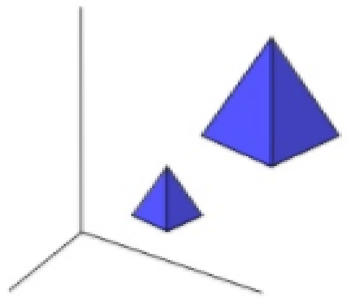
$$X' = X + t_x$$

$$Y' = Y + t_y$$

$$Z' = Z + t_z$$

2. Scaling:

Changing the size of an object is called scaling transformation. In the scaling process, you either expand or compress the dimensions of the object. Scaling can be achieved by multiplying the original coordinates of the object with the scaling factor to get the desired result. The following figure shows the effect of 3D scaling −



In 3D scaling operation, three coordinates are used. Consider that, the original coordinates are (X, Y, Z), scaling factors are (S_X,S_Y,S_Z) respectively, and the produced coordinates are (X', Y', Z'). This can be mathematically represented in matrix form as below –

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P' = P \cdot S$$

$$[X' \;\; Y' \;\; Z' \;\; 1] = [X \;\; Y \;\; Z \;\; 1] \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= [X.S_x \;\; Y.S_y \;\; Z.S_z \;\; 1]$$

Scaling in space is described by S_x, S_y and S_z. We see that this matrix realizes the following equations:

$$X' = X \cdot S_x$$
$$Y' = Y \cdot S_y$$
$$Z' = Z \cdot S_z$$

Procedure:

Step 1: Start the program.
Step 2: Input the object coordinates
Step 3: For Translation
    a) Enter the translation factors t_x, t_y and t_z.
    b) Move the original coordinate position (x,y,z) to a new position (x1,y1,y1).ie. x=x+x1, y=y+y1 and z=z+z1.
    c) Display the object after translation
Step 4: For Scaling
    a) Input the scaled factors sx,sy and sz.
    b) The transformed coordinates (x1,y1,z1) , x1=x.s_x , y1=y.s_y and z1=z.s_x.
    c) Display the object after scaling
Step 5: Stop the Program.

VII Algorithm

VIII Flow Chart

IX C Program Code

X Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

XI Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XII Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XIII Result (Output of Program)

XIV Conclusion

XV Practical Related Questions

1. Write the transformation matrix for 3D Translation.
2. Write the transformation matrix for 3D Scaling.
3. What Z- Coordinate indicates in 3D transformations.
4. Explain Homogeneous coordinates.

XVI Exercise

1. Draw a cube in C by using bar3d function. Translate the cube by 50 units in X,50 units in Y and 50 units in Z direction.
2. Draw a cube in C by using bar3d function. Scale it to double of its original size.

[Space for Answer]

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

XVII References / Suggestions for further Reading Software/Learning Websites

http://www.it.hiof.no/~borres/j3d/math/twod/p-twod.html

XVIII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……………………………..
2. ……………………………..
3. ……………………………..
4. ……………………………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.11: Write a C program for 3D Rotation.

I Practical Significance

3D Computer graphics are graphics that use a three dimensional representation of geometric data that is stored in the Computer for the purposes of performing calculations and rendering 2D images.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Perform and Demonstrate basic and composite graphical transformations on given object.

IV Laboratory Learning Outcome

Implement a C program for 3D Rotation.

V Relevant Affective domain related Outcome(s)

1. Plan, construct, compile, debug and test programs.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

Rotation:-

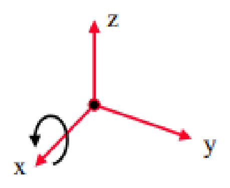
3D rotation is not same as 2D rotation. In 3D rotation, we have to specify the angle of rotation along with the axis of rotation. We can perform 3D rotation about X, Y, and Z axes. They are represented in the matrix form as below –

$$R_x(\Theta)=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\Theta & -\sin\Theta & 0 \\ 0 & \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad R_y(\Theta)=\begin{pmatrix} \cos\Theta & 0 & \sin\Theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\Theta & 0 & \cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

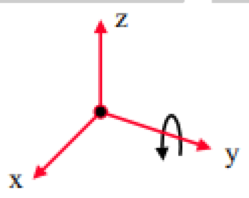
$$Rz(\Theta)=\begin{pmatrix} \text{Cos}\Theta & -\sin\Theta & 0 & 0 \\ \sin\Theta & \cos\Theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The following figure explains the rotation about various axes –

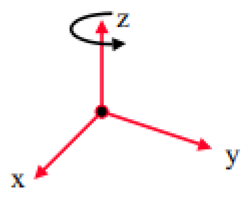
X-axis rotation-



Y-axis rotation-



Z-axis rotation-



Procedure:

Step 1: Start the program.
Step 2: Input the object coordinates
Step 3: For Rotation

a) Enter the radian for rotation angle θ.
b) Perform rotation about each axis.

i) Z-Axis Rotation

Z-axis rotation is identical to the 2D case:

x' = x*cos q - y*sin q
y' = x*sin q + y*cos q
z' = z

ii) X-Axis Rotation

X-axis rotation looks like Z-axis rotation if replace:

y' = y*cos q - z*sin q
z' = y*sin q + z*cos q
x' = x

iii) Y-Axis Rotation

Y-axis rotation looks like Z-axis rotation if replace:

z' = z*cos q - x*sin q

x' = z*sin q + x*cos q

y' = y

c) Display the object after rotation

Step 4: Stop the Program.

VII Algorithm

VIII Flow Chart

IX C Program Code

X Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

XI Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XII Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XIII Result (Output of Program)

XIV Conclusion

XV Practical Related Questions

1. Write the transformation matrix for 3D Rotation about Z-axis.
2. Write the transformation matrix for 3D Rotation about X-axis.
3. Write the transformation matrix for 3D Rotation about Y-axis.

XVI Exercise

1. Draw a cube in C by using bar3d function. Rotate the cube by 45^0 around X-axis.
2. A triangle is defined by three vertices A(0,2,1),B(2,3,0),C(1,2,1).Rotate the given triangle around Y-axis.

[Space for Answer]

XVII References / Suggestions for further Reading Software/Learning Websites

http://www.it.hiof.no/~borres/j3d/math/twod/p-twod.html

XVIII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. **……….…………..………..**

2. **……….…………..………..**

3. **……….…………..………..**

4. **……….…………..………..**

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.12: Write a C program for Line Clipping using Cohen-Sutherland algorithm.

I Practical Significance

The process that identifies which portion is inside and which portion is outside the specified region of space is called as a clipping. The Cohen–Sutherland algorithm is a line clipping algorithm used for line clipping. This algorithm uses four digit code to point out which of nine regions contain the end point of the line. The four bit code is called as region code.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Implement various Clipping algorithms.

IV Laboratory Learning Outcome

Implement a C program for Line Clipping using Cohen-Sutherland algorithm

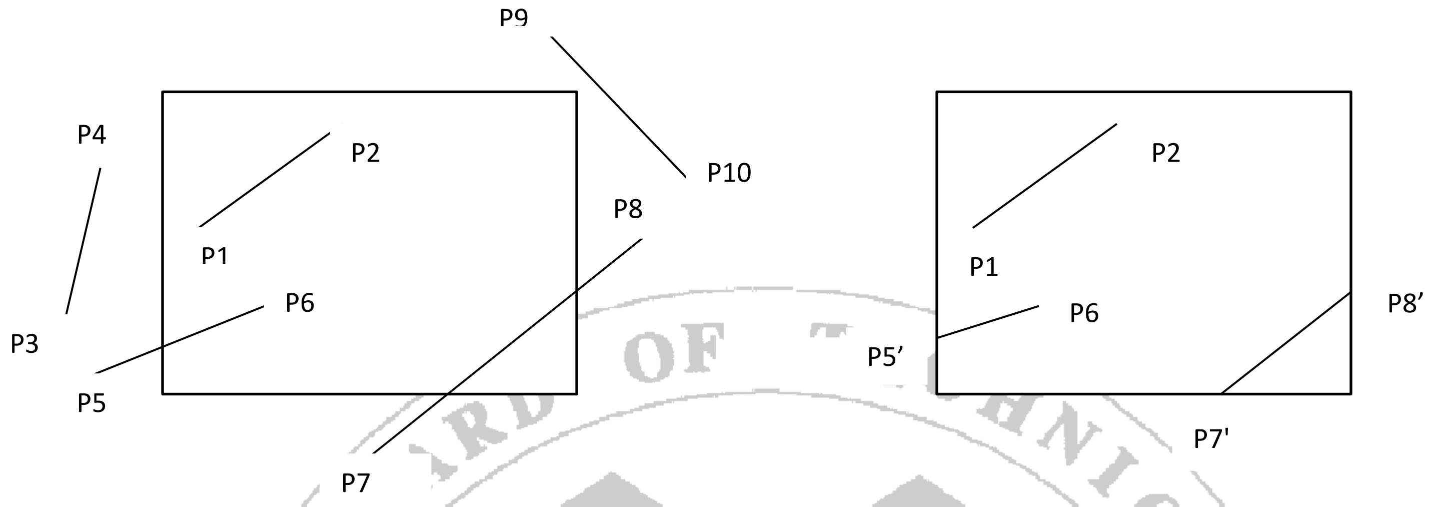
V Relevant Affective domain related Outcome(s)

1. Handle command prompt environment.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

Line clipping algorithm:

It is the process of removing lines or portions of lines outside of an area of interest called clipping region or window. Any line or the part of any line which is outside the clipping window is removed. Following figure shows the lines before and after clipping the window.



Before clipping **After Clipping**

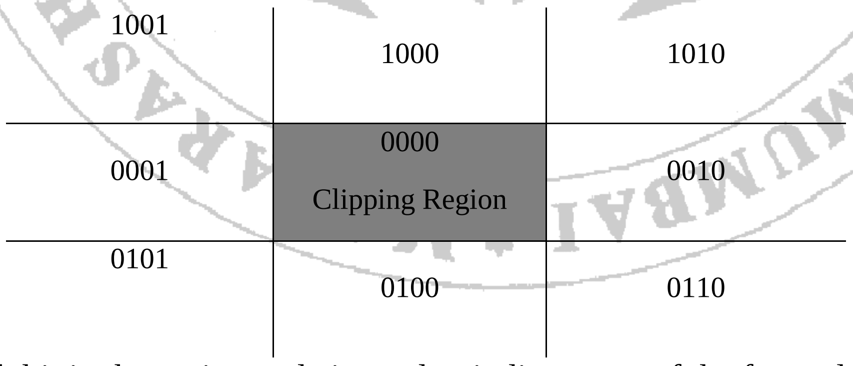
Lines fall into one of the following clipping categories:

a. Completely Visible: These lines are said to be interior to the clipping window. Both end points of these lines are interior to window. In the above figure line P1P2 is completely visible.

b. Completely Invisible: If both end points of a line are outside the window and not crossing the boundaries the line is completely invisible in window. In above figure P3P4, P9P10 are completely invisible.

c. Partially Visible: If the lines cross the clipping boundaries the lines are partially visible. Line P5P6, P7P8 is partially visible.

Four bit codes (Region Codes):
To find the clipping category of the line, four bit code is used. The codes for the end points of the lines are assigned by considering the region in which the end points resides.



Each bit in the region code is used to indicate one of the four relative co-ordinates positions of point with respect to clipping window to left, right, top and bottom.

Four bit is assigned to points as,

Top	Bottom	Right	Left
4	3	2	1

Bit 1 is set to 1 if point lies to the left of the window.

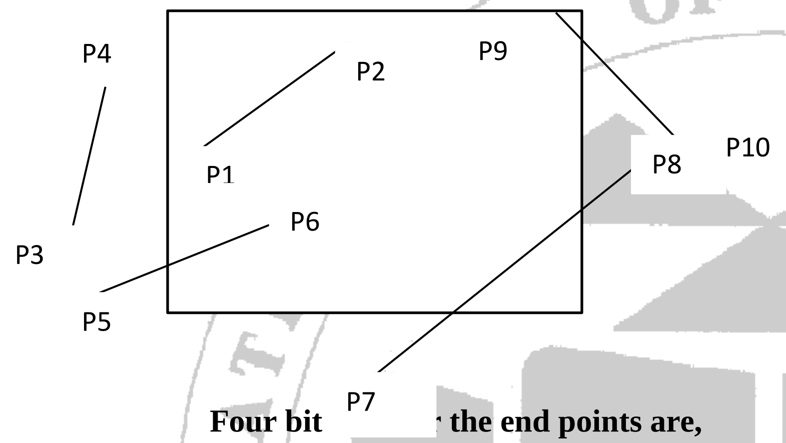
Bit 2 is set to 1 if point lies to the right of the window.

Bit 3 is set to 1 if point lies to the below of the window.

Bit 4 is set to 1 if point lies to the above of the window.

Otherwise the bit is set to zero.

Example:



Four bit the end points are,

P1	0000	P6	0000
P2	0000	P7	0100
P3	0001	P8	0010
P4	0001	P9	1000
P5	0001	P10	0010

Identifying category of line using four bit code:

From the four bit codes of all line end points clipping categories of lines are determined as below,

1) If four bit codes of both endpoints of the line is 0000 then the line is completely visible.

2) If bitwise logical AND of four bit codes of both endpoints is not 0000 then the line is completely invisible.

3) If bitwise logical AND of four bit codes of both endpoints is 0000 then the line is partially visible and needs clipping.

To find out intersection boundary:

Choose the end points whose region code is not 0000. Then depending upon the position of the 1 in region code following scheme decides with which boundary line intersects.

-If bit 1 is 1, intersects with line Y=Ymax (top boundary)

-if bit 2 is 1, intersects with line Y=Ymin (bottom boundary)

-if bit 3 is 1, intersects with line X=Xmax (right boundary)

-if bit 4 is 1, intersects with line X=Xmin (left boundary)

To find out co-ordinates of intersection point:

Once line of intersection is known then point of intersection can be calculated as,

- If boundary line is vertical, then

Xi=Xmin or Xmax

Yi=y1+m(Xi-X1)

- If boundary line is Horizontal, then

Xi=X1+(yi-y1)/m

Where, m is slope of line = $\frac{Y2-Y1}{X2-X1}$

(X1,y1) = end point of line

(Xi, yi)= intersection point

Now replace end points (X1, y1) with intersection point (Xi, yi) and display this new line joining two intersection points or a line joining one intersection point and other end point in 0000 region code.

Procedure:

1. Given a line segment with endpoint $P_1=(x_1,y_1)$ and $P_2=(x_2,y_2)$
2. Compute the 4-bit codes for each endpoint.
   If both codes are **0000**,(bitwise OR of the codes yields 0000) line lies completely **inside** the window: pass the endpoints to the draw routine.
   If both codes have a 1 in the same bit position (bitwise AND of the codes is **not** 0000), the line lies **outside** the window. It can be trivially rejected.
3. If a line cannot be trivially accepted or rejected, at least one of the two endpoints must lie outside the window and the line segment crosses a window edge. This line must be **clipped** at the window edge before being passed to the drawing routine.
4. Examine one of the endpoints, say $P_1=(x_1,y_1)$. Read P_1's 4-bit code in order: **Left**-to-**Right**, **Bottom**-to-**Top**.
5. When a set bit (1) is found, compute the intersection **I** of the corresponding window edge with the line from P_1 to P_2. Replace P_1 with **I** and repeat the procedure.

VII Algorithm

VIII Flow Chart

IX C Program Code

X Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

XI Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XII Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XIII Result (Output of Program)

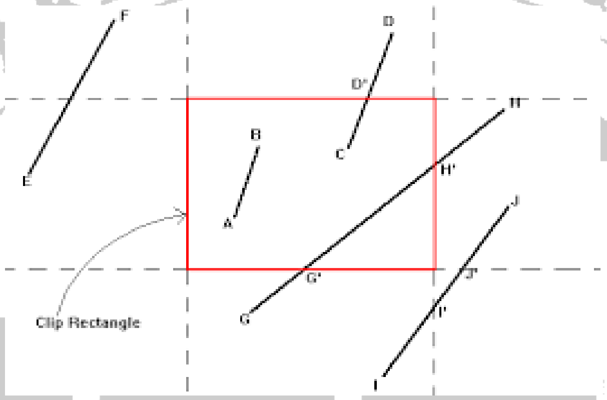
XIV Conclusion

XV Practical Related Questions

1. Define clipping.
2. How to calculate intersection points of a line if line is partially visible.
3. Give 3 possible conditions to clip line using cohen Sutherland line clipping algorithm.

XVI Exercise

1. Use the Cohen Sutherland algorithm to clip two lines P1(35,10)- P2(65,40) and P3(65,20)-P4(95,10) against a window A(50,10), B(80,10), C(80,40) and D(50,40).
2. Use Cohen-Sutherland algorithm to clip two lines PI (40,15) - P2 (75, 45) and P3 (70, 20)-P4 (100, 10) against a window A (50, 10), B (80, 10). C(80, 40) & D(50,40)
3. Write four bit code for following lines and determine clipping categories of each line.



[Space for Answer]

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XVII References / Suggestions for further Reading Software/Learning Websites

https://www.geeksforgeeks.org/line-clipping-set-1-cohen-sutherland-algorithm/

XVIII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……………………………..
2. ……………………………..
3. ……………………………..
4. ……………………………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.13: Write a C program for Line Clipping using Midpoint Subdivision algorithm.

I Practical Significance

Clipping process removes objects, lines, or parts of objects that are outside the viewing pane. The strength of this algorithm over the Cohen-Sutherland algorithm is that it requires no floating point arithmetic to find the point of intersection with the line and the clip boundary. The midpoint subdivision algorithm clips a line by finding the endpoints of the visible portion of the line segment. Each endpoint can be found by an identical process and given appropriate hardware; this can be done in parallel for both endpoints.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Implement various Clipping algorithms.

IV Laboratory Learning Outcome

Implement a C program for Line Clipping using Midpoint Subdivision algorithm.

V Relevant Affective domain related Outcome(s)

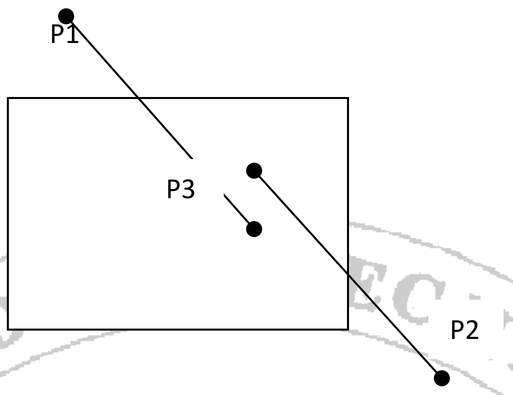
1. Handle command prompt environment.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

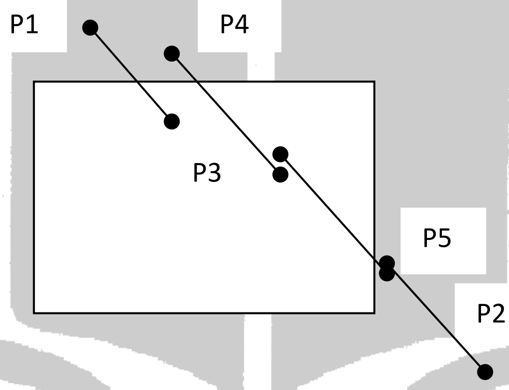
Midpoint Subdivision Line clipping algorithm

Midpoint subdivision algorithm works same as Cohen-Sutherland algorithm. It assigns four bit codes at end point of line to find out clipping category of line like completely visible, completely invisible and partially visible.

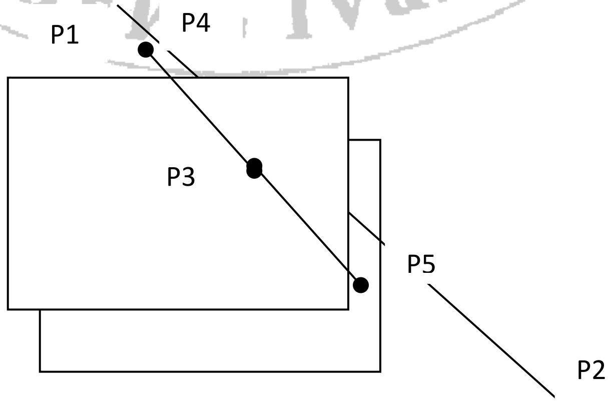
Lines with partially visible category send for clipping.
Demonstration of midpoint subdivision algorithm for clipping line P1P2:



1) Four digit for points P1 and P2 are 1000 and 0010 respectively.
2) As ANDing of four digit codes is 0000 line P1P2 is partially visible.
3) Divide line P1P2 at mid-point P3 forming two segments P1P3 and P3P2 , apply visibility test on both segments. Both lines are in partially visible category and hence we have to again subdivide it.
4) Divide line P1P3 at midpoint P4 and P3P2 at midpoint P5 check all line segments for their visibility.



5) line P1P5 and line P2P5 are completely invisible and hence removed. Line P4P3 and Line P3P5 are partially visible. Line P4P3 and P3P5 are subdivided again and this procedure continues until all the visible and invisible lines are found.



If P1(x1,y1) and P2(x2,y2) are two points then midpoint P3(x3,y3) is calculated as

x3=(x1+x2)/2

y3=(y1+y2)/2

VII Algorithm

VIII Flow Chart

IX C Program Code

X Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

XI Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XII Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XIII Result (Output of Program)

XIV Conclusion

XV Practical Related Questions

1. Calculate coordinates of midpoint between two points p1(-10,20)andp2(50,10)
2. Compare Cohen Sutherland and Midpoint subdivision line clipping algorithm.
3. Write disadvantages of Midpoint subdivision line clipping algorithm

XVI Exercise

1. Using Midpoint subdivision line clipping algorithm illustrate a clipping of a line segment joining two end points A(-1,5),B(3,8) by considering clipping window with left corner at(-3,1) and upper right corner at(2,6).

[Space for Answer]

XVII References / Suggestions for further Reading Software/Learning Websites

http://www.ques10.com/p/22054/explain-midpoint-subdivision-algorithm/

XVIII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..

2. ……….……….………..

3. ……….……….………..

4. ……….……….………..

Marks obtained			**Dated Sign of Teacher**
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.14: Write a C program for Line Clipping using Midpoint Subdivision algorithm.

I Practical Significance

The algorithm finds the polygon that is the intersection between an arbitrary polygon (subject polygon) and a convex polygon (clip polygon).
It is used in Computer graphics to reduce the complexity of a scene being displayed by eliminating parts of a polygon that do not need to be displayed.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Implement various Clipping algorithms.

IV Laboratory Learning Outcome

Implement C program for Sutherland Hodgeman Polygon Clipping.

V Relevant Affective domain related Outcome(s)

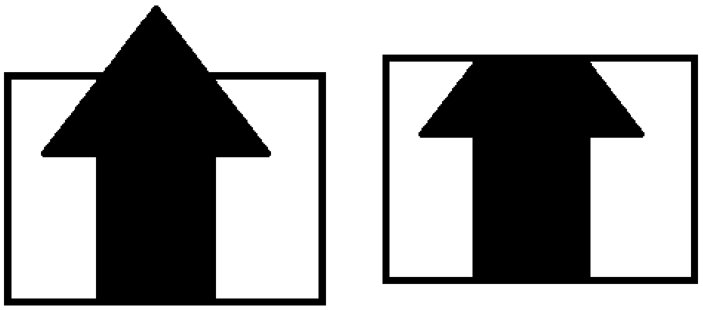
1. Handle command prompt environment.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

Polygon Clipping:

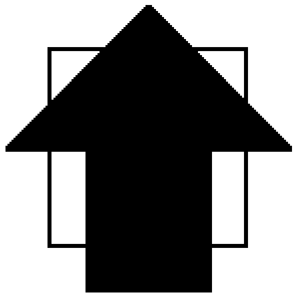
Polygon is specified by set of three or more co-ordinates positions, called vertices. These vertices are connected in sequence by straight line segment. These straight line segments are called as edges or side if the polygon.

Figure shows original polygon with clipping window and polygon after clipping.



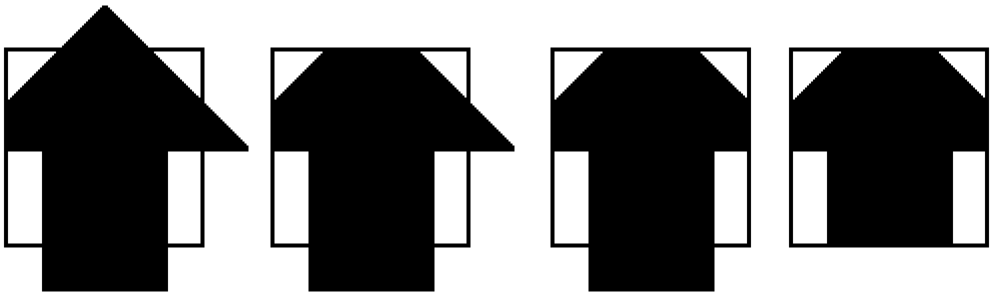
Sutherland - Hodgeman algorithm

The Sutherland - Hodgeman algorithm performs a clipping of a polygon against each window edge in turn. It accepts an ordered sequence of vertices v1, v2, v3, ..., vn and puts out a set of vertices defining the clipped polygon.
Following figure represents the polygon before clipping has occurred.



Before clipping

The following figures shows how this algorithm works at each edge, clipping the polygon.



[a][[b] [c] [d]

a. Clipping against the left side of the clip window.

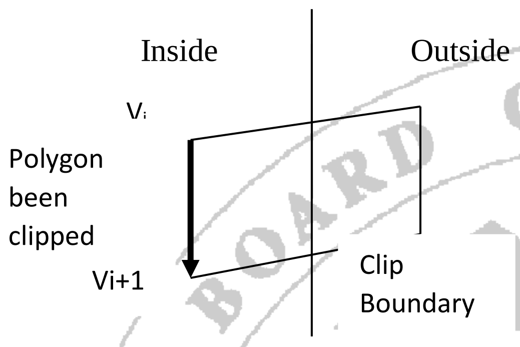
b. Clipping against the top side of the clip window.

c. Clipping against the right side of the clip window.

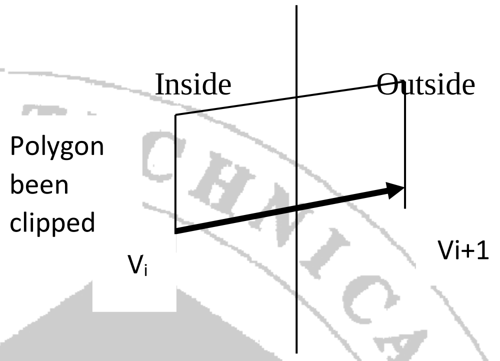
d. Clipping against the bottom side of the clip window.

Four test cases of edges:
Every edge of polygon is compared with clipping plane to find out the new vertices called output vertices. According to this there are fur relationships between the edge and clipping boundary.
Case1: If both input vertices are inside the clipping window boundary then add only second vertex to output vertex list.

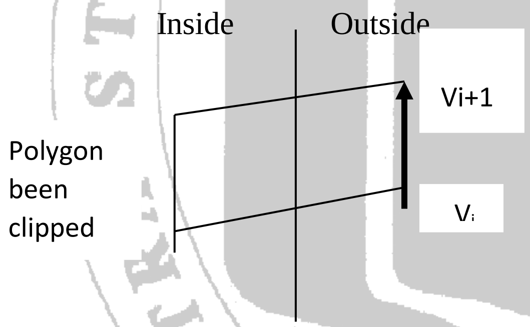
Case2: If the first vertex is inside the window boundary and second vertex is outside then only intersection with window boundary is added to output vertex list.
Case3: If both input vertices are outside the clipping window boundary then nothing is added to output vertex list.
Case4: If the first vertex is outside the clipping window the clipping window boundary and second vertex is inside it, then both the intersection point of the polygon edge with window boundary and second vertex are added to output vertex list.



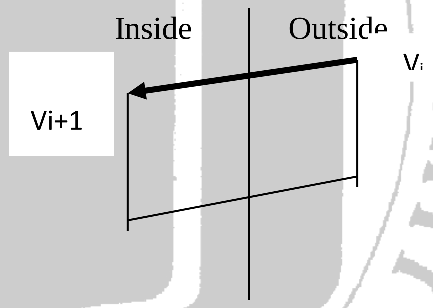
Case: 1



Case: 2



Case: 3



Case: 4

VII Algorithm

VIII Flow Chart

C Program Code

IX Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

X Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

XI Resources Used

S. No.	Name of Resource	Specification
1	Computer System with broad specifications	
2	Software	
3	Any other resource used	

XII Result (Output of Program)

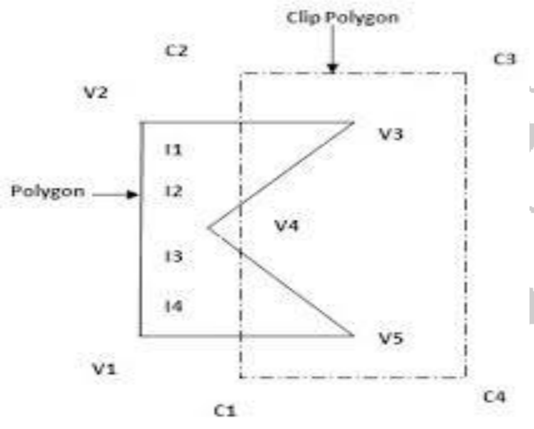
XIII Conclusion

XIV Practical Related Questions

1. If the first vertex is outside the clipping window and second point is inside the clipping window, then write which points are added to output vertex list.
2. Write the procedure to clip polygon using Sutherland Hodgeman Polygon Clipping algorithm

XV Exercise

1. Clip the following polygon using Sutherland Hodgeman Polygon Clipping algorithm.



[Space for Answer]

XVI References / Suggestions for further Reading Software/Learning Websites

https://www.geeksforgeeks.org/computer-graphics-curve-in-computer-graphics/

XVII Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….…………..………..
2. ……….…………..………..
3. ……….…………..………..
4. ……….…………..………..

Marks obtained			Dated Sign of Teacher
Process Related(15)	**Product Related(10)**	**Total(25)**	

Practical No.15:.Write a C program for Bezier Curve.

I Practical Significance

Objects in real world are not always made up of regular geometric shapes they may include curves. Drawing curves involves complex mathematical analysis in the form of various interpolation techniques.

II Industry/ Employer Expected Outcome

This practical is expected to develop the following skills in you

Develop 'C' programs to draw basic graphics objects:

1. Write syntax for graphics functions.
2. Write and Save a simple C program.
3. Setup graphics drivers, graphics mode and directory to run graphics program.
4. Compile the C program using Turbo C.
5. Debug and execute the program.

III Course Level Learning Outcomes

Develop programs to create Curves.

IV Laboratory Learning Outcome

Implement a C program for Bezier Curve.

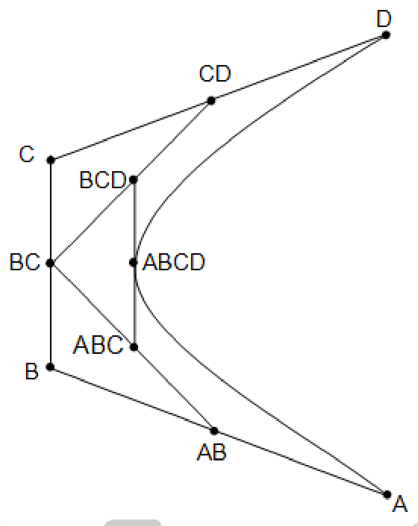
V Relevant Affective domain related Outcome(s)

1. Handle command prompt environment.
2. Experiment with graphics environment.
3. Follow ethical practices.

VI Relevant Theoretical Background

Bezier Curves

Bezier Curve is adequate for most graphical applications.
This curve requires four control points. These four control points completely specify the curve. Additional points cannot be added. We cannot extend Bezier curve but we can take four more points and we can construct a second Bezier curve which can be attached to second Bezier curve.



Procedure:

Bezier Curve:

To each set of four points P_0, P_1, P_2, P_3 we associate a curve with the following properties:

1. It starts at p0 and ends at p3.
2. When it starts from p0 it heads directly towards p1, and when it arrives at p3 it is coming from the direction of p2.
3. The entire curve is contained in the quadrilateral whose corners are the four given points (their **convex hull**).

Algorithm

Flow Chart

C Program Code

VII Resources required

Sr. No.	Name of Resource	Specification	Quantity	Remarks
1	Hardware: Computer System	Computer (i3-i5 preferable), RAM minimum 2 GB and onwards but not limited to	As per batch size	For all Experiments
2	Operating system	Windows XP/Windows 7/LINUX version 5.0 or later		
3	Software	Turbo C /C++ Version 3.0 or later with DOSBOX		

VIII Precautions to be Followed

1. Ensure that all C statements must end with a semicolon (;).
2. Use white spaces in c to describe blanks and tabs.
3. Ensure use of proper graphics function for relevant object.
4. Follow safety practices.

IX Resources Used

S. No.	Name of Resource	Specification
4	Computer System with broad specifications	
5	Software	
6	Any other resource used	

X Result (Output of Program)

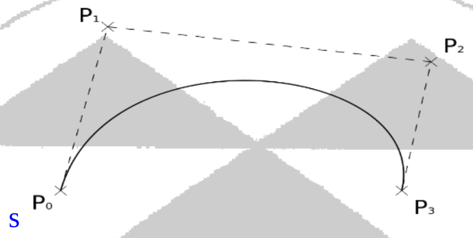
XI Conclusion

XII Practical Related Questions

1.Define fractal.
2.State types of Curves.
3.Write procedure to draw Bezier curve.
4.Define fractal dimension and topological dimension

XIII Exercise

1. What are the properties of Bezier Curve?
2. Draw following Bezier curve.



[Space for Answer]

XIV References / Suggestions for further Reading Software/Learning Websites

https://www.geeksforgeeks.org/computer-graphics-curve-in-computer-graphics/

XV Assessment Scheme

Performance indicators		Weightage
Process related: 15 Marks		**60%**
1.	Debugging ability	20%
2.	Correctness of Program codes	30%
3.	Quality of output achieved(LLO mapped)	10%
Product related: 10 Marks		**40%**
1.	Completion and submission of practical in time	20%
2.	Answer to sample questions	20%
Total 25 Marks		**100%**

List of Student /Team Members

1. ……….……….………..

2. ……….……….………..

3. ……….……….………..

4. ……….……….………..

Marks obtained			Dated Sign of Teacher
Process Related(15)	Product Related(10)	Total(25)	