VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A Computer Graphics & Visualization Mini Project Report
on

"TURING MACHINE IMPLEMENTATION"

Submitted in Partial fulfillment of the Requirements for VI Semester of the Degree of

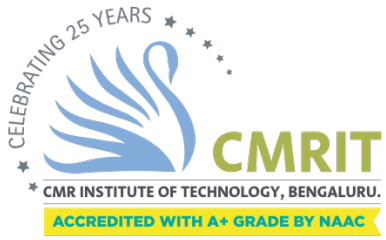
Bachelor of Engineering
In
Computer Science & Engineering
By

DAGA TARUN PAVAN
(1CR15CS047)

PRAJWAL DHATWALIA
(1CR15CS111)

Under the Guidance of

Mr. Kartheek GCR
Asst. Professor, Dept. of CSE



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CMR INSTITUTE OF TECHNOLOGY
#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI,

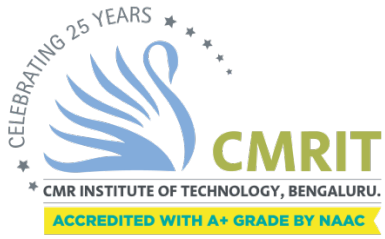
BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI,

BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Computer Graphics & visualization project work entitled **"Turing Machine Implementation"** has been carried out by **Daga Tarun Pavan (1CR15CS047)**and **Prajwal Dhatwalia (1CR15CS111)** bonafide students of CMR Institute of Technology in partial fulfillment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year **2017-2018**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. This CG project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

----------	**----------**
Signature of Guide	**Signature of HOD**
Mr. Kartheek GCR **Assst. Professor** **Dept. of CSE, CMRIT**	**Dr. Jhansi Rani P** **Professor & Head** **Dept. of CSE, CMRIT**

External Viva

Name of the examiners	Signature with date
1.	
2.	

ABSTRACT

OpenGL provides a set of commands to render a three dimensional scene. That means you provide them in an Open GL-useable form and Open GL will show this data on the screen (render it). It is developed by many companies and it is free to use. You can develop Open GL-applications without licensing.

Open GL is a hardware- and system dependent interface. An Open GL-application will work on every platform, as long as there is an installed implementation, because it is system independent, there are no functions to create windows etc., but there are helper functions for each platform. A very useful thing is GLUT.

This project includes the concepts of transformation, motion in objects. We have implemented Palindrome using Turing Machine concepts.

ACKNOWLEDGEMENT

Behind every success there is a master hand. A master hand will create unperturbed concentration, dedication and encouragement in everything good and bad, without whose blessing this would have never come into existence.

Firstly, we thank God for showering the blessings on us. We are grateful to our institution CMRIT for providing us a congenial atmosphere to carry out the project successfully.

We would like to express our heartfelt gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for extending his support.

We are highly thankful to **Mrs. Jhansi Rani,** HOD of Computer Science and Engineering, CMRIT, Bangalore for her support and encouragement given to carry out the project.

We are very grateful to our guide, **Mr. Kartheek G C R,** Assistant Professor, Department of Computer Science, for his able guidance and valuable advice at early stage of our project which helped us in successful completion of our project.

Finally, we would like to thank our parents and friends who helped us with the content of this report, without which the project would not have become a reality.

DAGA TARUN PAVAN (1CR15CS047)
PRAJWAL DHATWALIA (1CR15CS111)

LIST OF FIGURES

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction to Computer Graphics

- Graphics provides one of the most natural means of communicating with a computer, since our highly developed 2D or 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly.
- Computers have become a powerful medium for the rapid and economical production of pictures.
- Graphics provide a so natural means of communicating with the computer that they have become widespread.
- Interactive graphics is the most important means of producing pictures since the invention of photography and television.
- We can make pictures of not only the real world objects but also of abstract objects such as mathematical surfaces on 4D and of data that have no inherent geometry.
- A computer graphics system is a computer system with all the components of the general purpose computer system. There are five major elements in system: input devices, processor, memory, frame buffer, output devices.

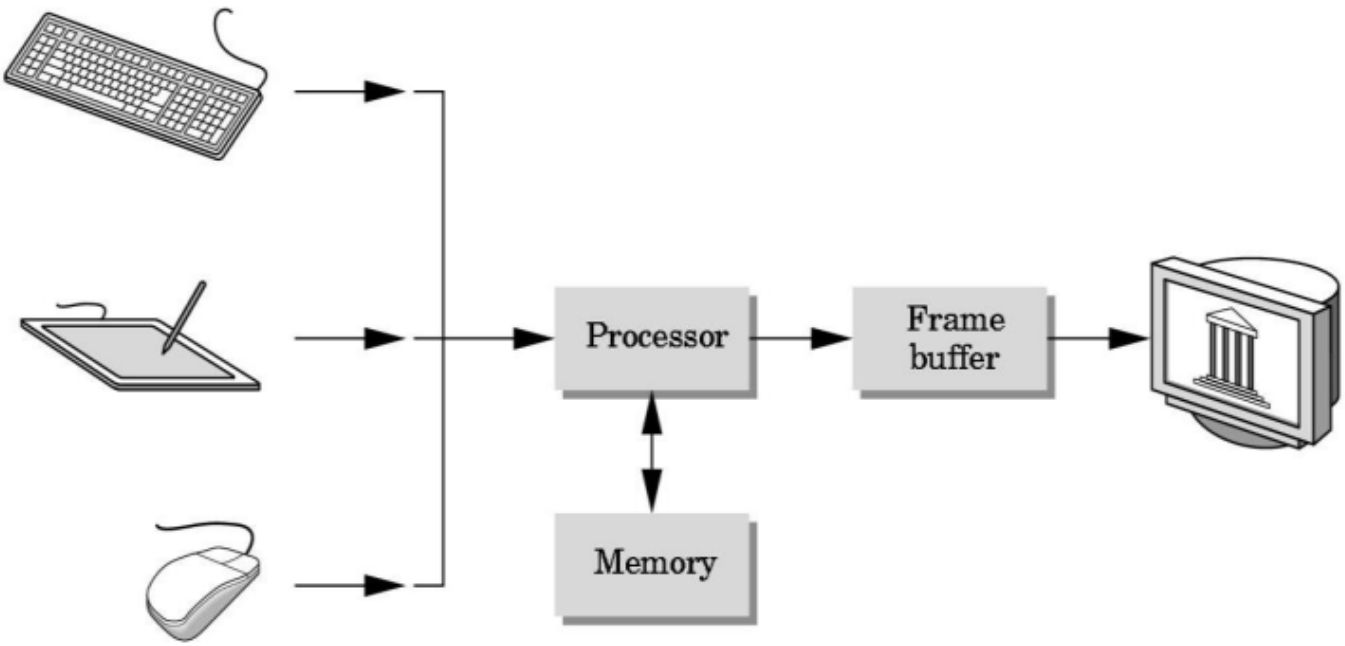


Fig 1.1: Graphic System

1.2 Areas of Application of Computer Graphics

➢ User interfaces and Process control

➢ Cartography

➢ Office automation and Desktop publishing

➢ Plotting of graphs and charts

➢ Computer aided Drafting and designs

➢ Simulation and Animation

1.3 Introduction to OpenGL

OpenGL is the premier environment for developing portable, interactive 2D and 3D graphics applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API), bringing thousands of applications to a wide variety of computer platforms.

OpenGL fosters innovation and speeds application development by incorporating a broad set of rendering, texture mapping, special effects, and other powerful visualization functions. Developers can leverage the power of OpenGL across all popular desktop and workstation platforms, ensuring wide application deployment.

OpenGL available Everywhere: Supported on all UNIX® workstations, and shipped standard with every Windows 95/98/2000/NTand MacOS PC, no other graphics API operates on a wider range of hardware platforms and software environments.

OpenGL runs on every major operating system including Mac OS, OS/2, UNIX, Windows 95/98, Windows 2000, Windows NT, Linux, Open Step, and BeOS; it also works with every major windowing system, including Win32, MacOS, Presentation Manager, and X-Window System. OpenGL is callable from Ada, C, C++, Fortran, Python, Perl and Java and offers complete independence from network protocols and topologies.

1.3.1 The OpenGL interface

Our application will be designed to access OpenGL directly through functions in three libraries namely: gl,glu,glut.

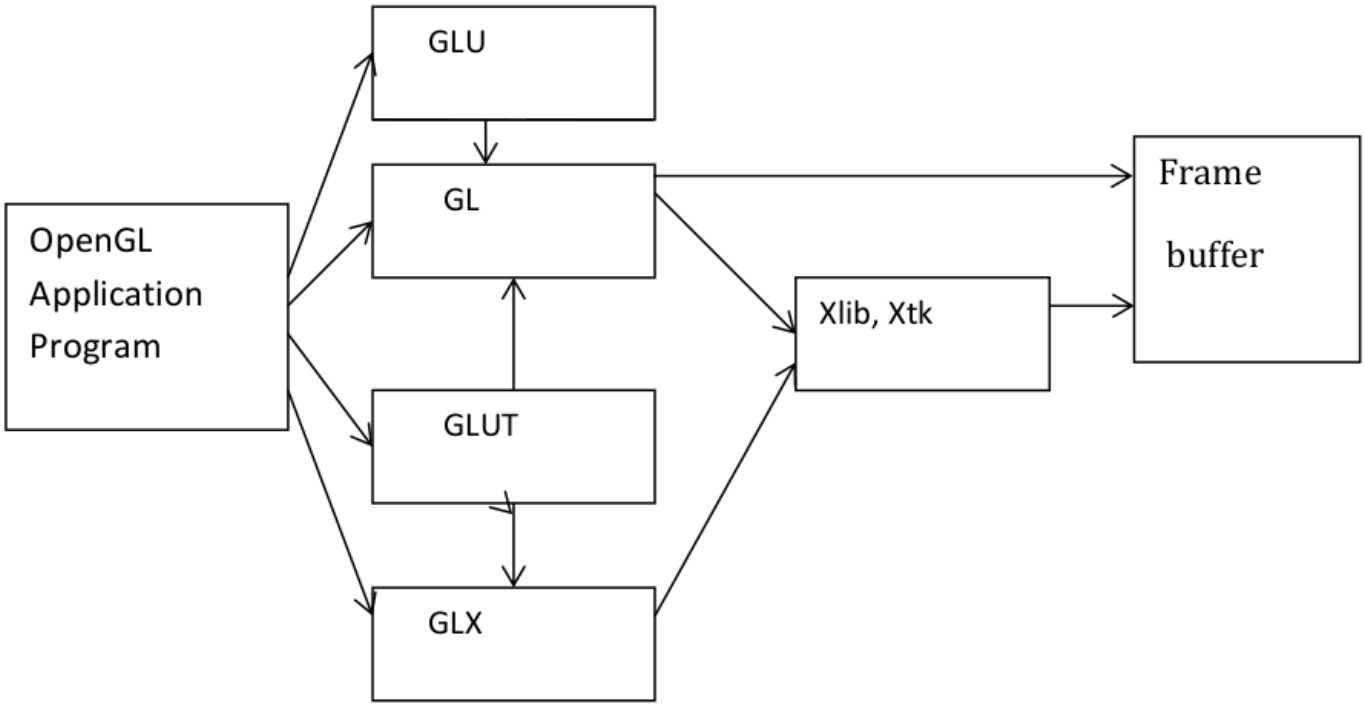


Fig 1.2: Library organization of OpenGL

1.3.2 Graphics Functions

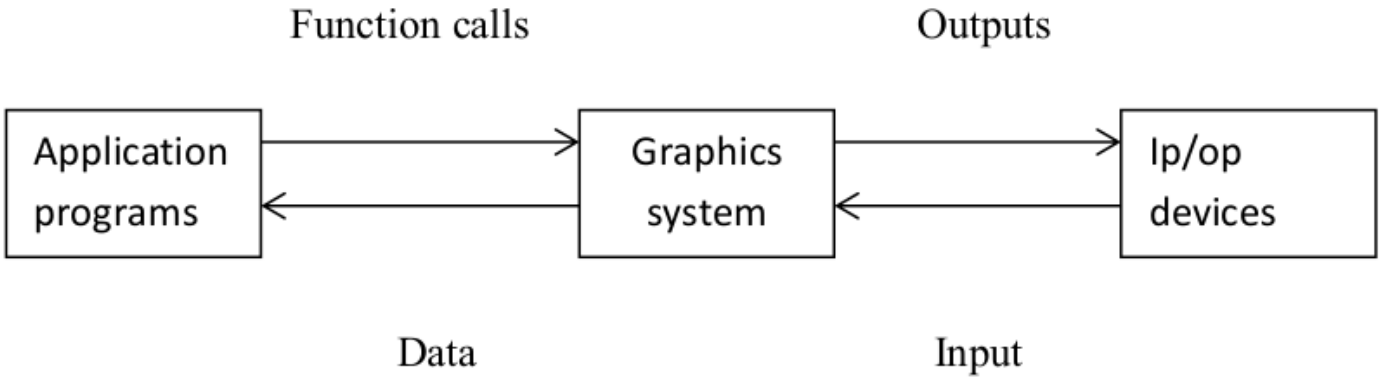


Fig 1.3: Graphics system as a black box.

Our basic model of a graphics package is a black box, a term that engineers use to denote a system whose properties are described only by its inputs and outputs. We describe an API through the functions in its library. Some of the functions are:

- The primitive functions define the low-level objects or atomic entities that our system can display.

- Attribute functions allow us to perform operations ranging from choosing the color with which we display a line segment, to picking a pattern with which to fill the nside of a polygon, to selecting a typeface for the titles of a graph.
- Transformation function allows carrying out transformations of objects, such as rotation, translation, and scaling.
- A set of input functions allow us to deal with the diverse forms of input that characterize modern graphics systems.
- The control functions enable us to communicate with the window systems, to initialize our programs, and to deal with any errors that take place during the execution of programs.

CHAPTER 2

REQUIREMENTS SPECIFICATION

2.1 Purpose of the requirements document

- The software requirement specification is the official statement of what is required for development of particular project. It includes both user requirements and system requirements. This requirement document is utilized by variety of users starting from project manager who gives project to the engineer responsible for development of project.

- It should give details of how to maintain, test, verify and what all the actions to be carried out through life cycle of project.

2.1.1 Scope of the project

- The scope is to use the basic primitives defined in openGL library creating complex objects. We make use of different concepts such as glColor() , gluOrtho2D(), timer function.

2.1.2 Definition

- This project is created to demonstrate OpenGL's concepts. It encompasses some of the skills learnt in our OpenGL classes such as glColor(), gluOrtho2D(), timer function.

2.1.3 Acronyms & Abbreviations

- OpenGL provides a powerful but primitive set of rendering command, and all higher-level design must be done in terms of these commands. OpenGL Utility Toolkit(GLUT): -windows-system-independent toolkit.

2.1.4 References

- OpenGL tutorials
- Interactive Computer Graphics (Edward Angel)

2.2 Specific requirements

2.2.1 User Requirement:

- Easy to understand and should be simple.
- The built-in functions should be utilized to maximum extent.
- OpenGL library facilities should be used.

2.2.2 Software Requirements:

- Platform used: UBUNTU
- Technology used: OpenGL Libraries such has OpenGL Utility library, OpenGL Utility toolkit
- Language: C

2.2.3 Hardware Requirements:

- Processor-Intel or AMD(Advanced Micro Devices)
- RAM-512MB(minimum)
- Hard disk-1MB(minimum)
- Mouse
- Keyboard
- Monitor
-

CHAPTER 3

IMPLEMENTATION

3.1 OpenGL Function details

➢ **GlutInitDisplayMode** — sets the initial display mode.

▪ Declaration: void glutInitDisplayMode (unsigned int mode);

▪ Remarks: The initial display mode is used when creating top- level windows, sub windows, and overlays to determine the OpenGL display mode for the to-be-created window or overlay.

➢ **glutInitWindowposition** --- set the initial window position.

▪ Declaration: void glutInitWindowPosition(int x, int y);

x: Window X location in pixels.

y: Window Y location in pixels.

➢ **glutInitWindowSize** --- set the initial window size.

▪ Declaration: void glutInitWindowSize(int width,int height);

width: Width in pixels

height: Height in pixels.

➢ **glutCreateWindow** --- set the title to graphics window.

▪ Declaration: Int glutCreateWindow(char *title);

▪ Remarks: This function creates a window on the display. The string title can be used to label the window.The integer value returned can be used to set the current window when multiple windows are created.

➢ **glutDisplayFunc**

▪ Declaration: void glutDisplayFunc(void(*func)void));

▪ Remarks: This function registers the display function that is executed when the window needs to be redrawn.

➢ **glClear**:

▪ Declaration: void glClear();

▪ Remarks: This function clears the particular buffer.

➢ **glClearColor**:

▪ Declaration: void glClearColor(GLfloat red, GLfloat green, Glfloat blue, Glfloat alpha);

▪ Remarks: This function sets the color value that is used when clearing the color buffer.

➢ **glEnd**

▪ Declaration: void glEnd();

▪ Remarks: This function is used in conjunction with glBegin to delimit the vertices of an opengl primitive.

➢ **glMatrixMode**

▪ Declaration: void glMatrixMode(GLenum mode);

▪ Remarks: This function specifies which matrix will be affected by subsequent transformations mode can be GL_MODELVIEW,GL_PROJECTION or GL_TEXTURE..

➢ **gluOrtho2D**

▪ Declaration: void glOrtho(GLdouble left, GLdouble right, Gldouble bottom,GLdouble top);

▪ Remarks: It defines an orthographic viewing volume with all parameters measured from the center of the projection plane.

3.2. Code in C Language

```
void grid()                                    // To print the grid
{
        int size=70;
        int x_pos=-375;
        int y_pos=135;
        for(int i=0; i<size*stringLength(string)/2;i+=size)
{
                glColor3f(0,0.3+0.5*(i%140),1);
glBegin(GL_POLYGON);
glVertex2i(x_pos+i,             y_pos);
glVertex2i(x_pos+size+i,y_pos);
glVertex2i(x_pos+size+i,y_pos+size);
glVertex2i(x_pos+i,             y_pos+size);
glEnd();
}
}
void writer()                                  // To print the string contents
{
        unsigned int i, j;
        unsigned int count=0;

        GLfloat x = -350;
        GLfloat y = 155;

        // Draw the strings, according to the current font.
        glColor4f(0,0,0,0);

        // Displaying the message
glRasterPos2f(x, y);
for(i=0;i<stringLength(string)/2;i++)
        {
                glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,string[i]);
                for(j=0;j<9;j++)
                        glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,");
}
```

```
}

void needle(int position)                    // To tell the position of the pointer needle
{
        //Display the pointer needle triangle
        glColor3f(0,0,0);
        int size2=30;
        int x_pos2=-350+position;
        int y_pos2=135;
        glBegin(GL_TRIANGLES);
        glVertex2f(x_pos2+size2/2,y_pos2);
        glVertex2f(x_pos2,y_pos2-size2);
        glVertex2f(x_pos2+size2,y_pos2-size2);
        glEnd();

        //Displaying the pointer needle rectangle
        glColor3f(0,0,0);
        int size3=15;
        int x_pos3=-350+size2/4+position;
        int y_pos3=135-size2;
        glBegin(GL_QUADS);
        glVertex2f(x_pos3,y_pos3);
        glVertex2f(x_pos3+size3,y_pos3);
        glVertex2f(x_pos3+size3,y_pos3-size3*3);
        glVertex2f(x_pos3,y_pos3-size3*3);
        glEnd();
}

void showTape(int arrow)                     // Displays a state for Turing Machine
{
        glClearColor(1,1,1,1);
        glClear(GL_COLOR_BUFFER_BIT);

        //Displaying the grid
        grid();

        //Displaying the content of grid
        writer();

        //Display the pointer needle
        needle(arrow*70);

        glutSwapBuffers();

        //Delay
        sleep(1);
}

void accept()
{
        char text[7]="ACCEPT";
        glClearColor(1,1,1,1);
        GLfloat x=-350;
        GLfloat y=0;
        glRasterPos2f(x, y);
        for(int i=0;i<6;i++)
                glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,text[i]);
```

```
	glutSwapBuffers();
}

void reject()
{
	char text[7]="REJECT";
	glClearColor(1,1,1,1);
	GLfloat x=-350;
	GLfloat y=0;
	glRasterPos2f(x, y);
	for(int i=0;i<6;i++)
		glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,text[i]);

	glutSwapBuffers();
}

int forward(int m)
{
	for(k=m;k<n*2;)
		if(string[k]==B)
			return k;
		else
			k++;
}

int backward(int l)
{
	for(k=l;k>=0;)
		if(string[k]==B)
			return k;
		else
			k--;
}

void checkpal(int i)                                    // Checks for Palindrome
{
	j=i;
	var1=string[i];
	i=forward(i);
	i--;
	if(var1==string[i])
	{
		string[i]=B;
		showTape(i);
		string[j]=B;
		showTape(j);

		i--;
		i=backward(i);
		i++;
		for(x=0;x<n*2;x++)
		{
			if(string[x]==B)
			{
				flag++;
			}
```

```
                }
                if(i==(n/2))
                {
                        accept();
                }
                else
                {
                        checkpal(i);
                }
        }
        else
        {
                reject();
        }
}
```

CHAPTER 4

DESCRIPTION AND SNAPSHOTS

4.1 Description

- The objective of this project is to demonstrate the working of Turing Machine.
- Turing macine is a mathematical model of a hypothetical computing machine which can use a predefined set of rules to determine a result from a set of input variables.
- Despite the model's simplicity, given any computer algorithm, a Turing machine capable of simulating that algorithm's logic can be constructed.
- The machine operates on an infinite memory tape divided into discrete cells. The machine positions its head over a cell and "reads" (scans) the symbol there. Then, as per the symbol and its present place in a finite table of user-specified instructions, the machine

  (i) writes a symbol (e.g., a digit or a letter from a finite alphabet) in the cell (some models allowing symbol erasure or no writing), then

  (ii) either moves the tape one cell left or right (some models allow no motion, some models move the head), then

  (iii) (as determined by the observed symbol and the machine's place in the table) either proceeds to a subsequent instruction or halts the computation.
- The Turing machine was invented in 1936 by Alan Turing, who called it an a-machine (automatic machine). With this model, Turing was able to answer two questions in the negative:

  (1) Does a machine exist that can determine whether any arbitrary machine on its tape is "circular" (e.g., freezes, or fails to continue its computational task); similarly,

  (2) does a machine exist that can determine whether any arbitrary machine on its tape ever prints a given symbol. Thus by providing a mathematical description of a very simple device capable of arbitrary computations, he was able to prove properties of computation in general—and in particular, the uncomputability of the Entscheidungsproblem ("decision problem").
- Thus, Turing machines prove fundamental limitations on the power of mechanical computation. While they can express arbitrary computations, their minimalistic design makes them unsuitable for computation in practice: real-world computers are based on different designs that, unlike Turing machines, use random-access memory.

4.2 Screen snapshots

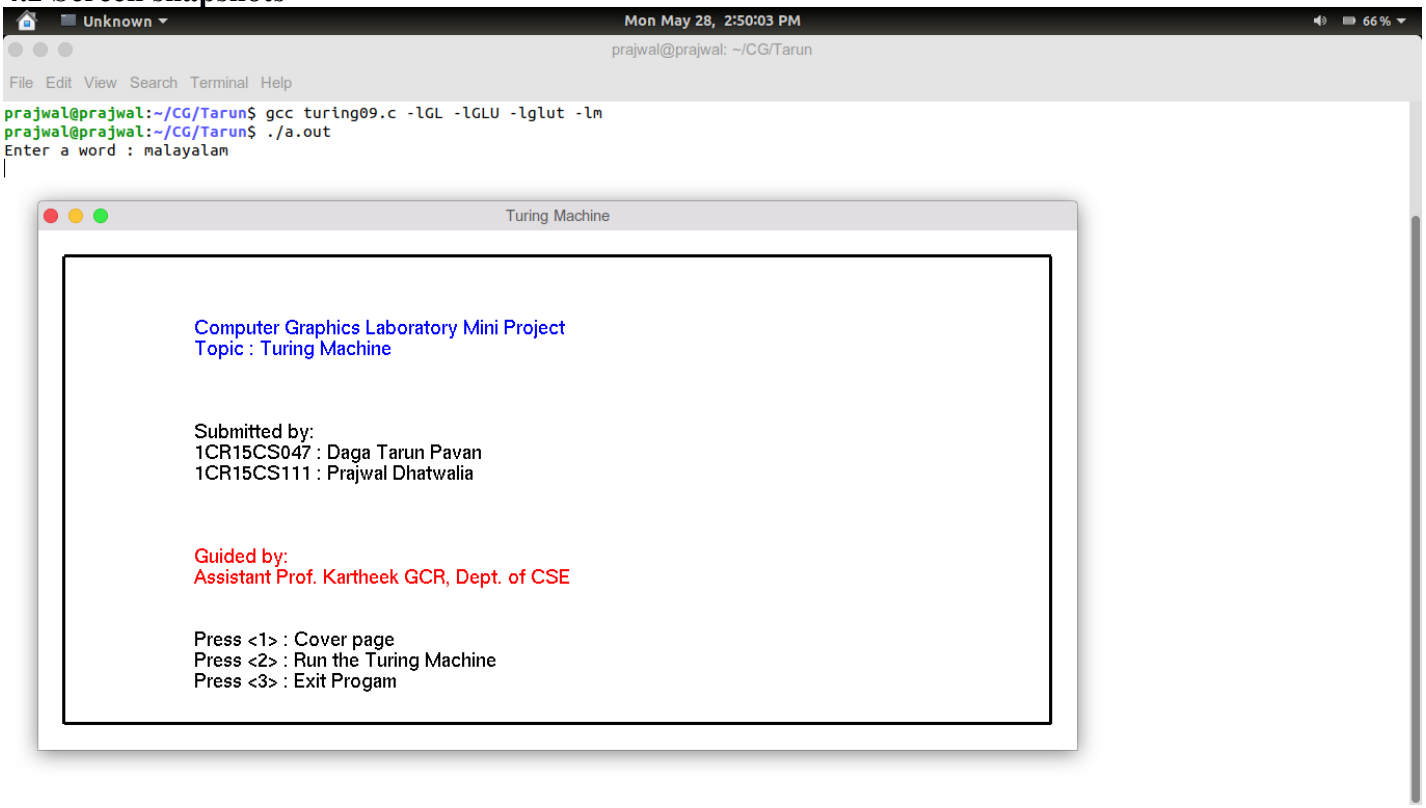


Fig 4.1: Cover page

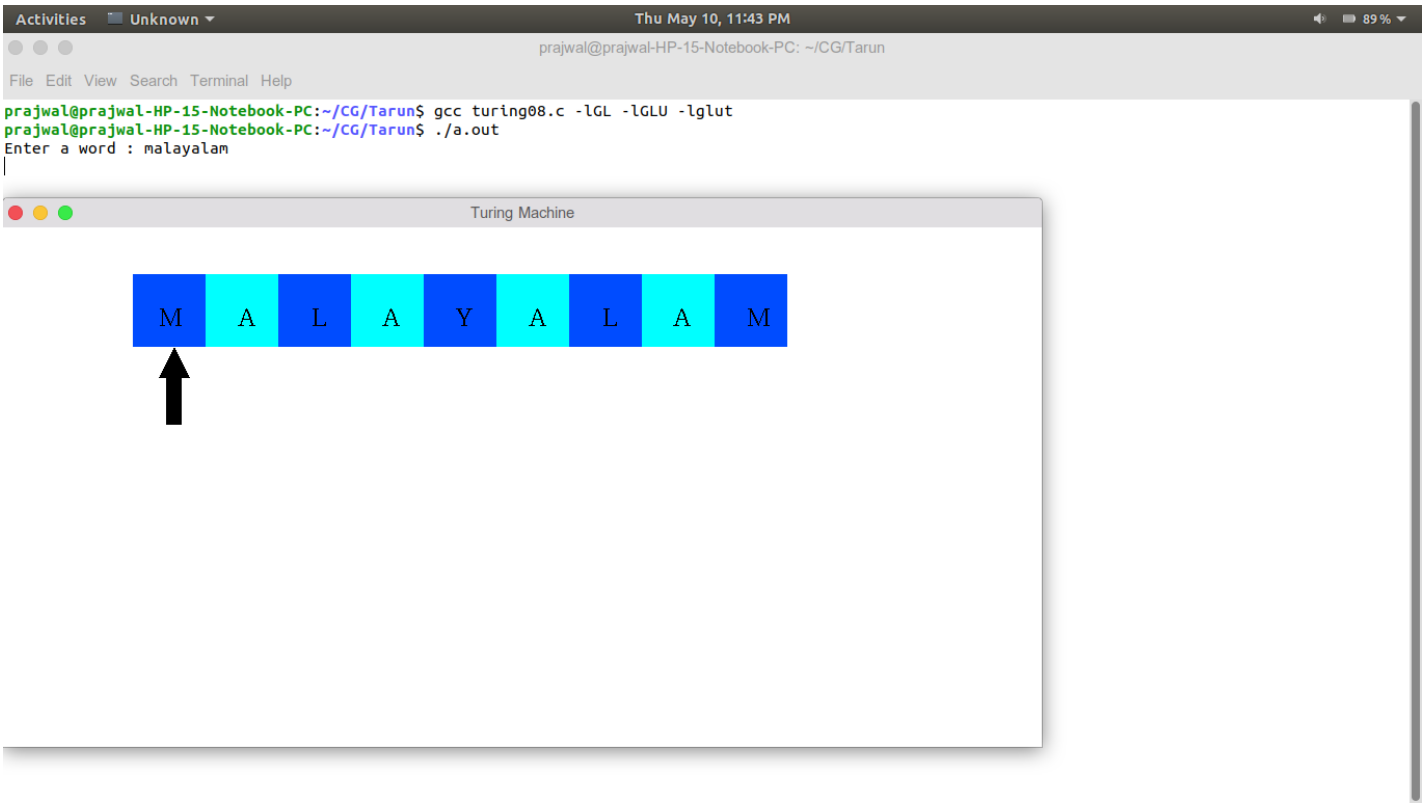


Fig 4.2: String malayalam has been entered

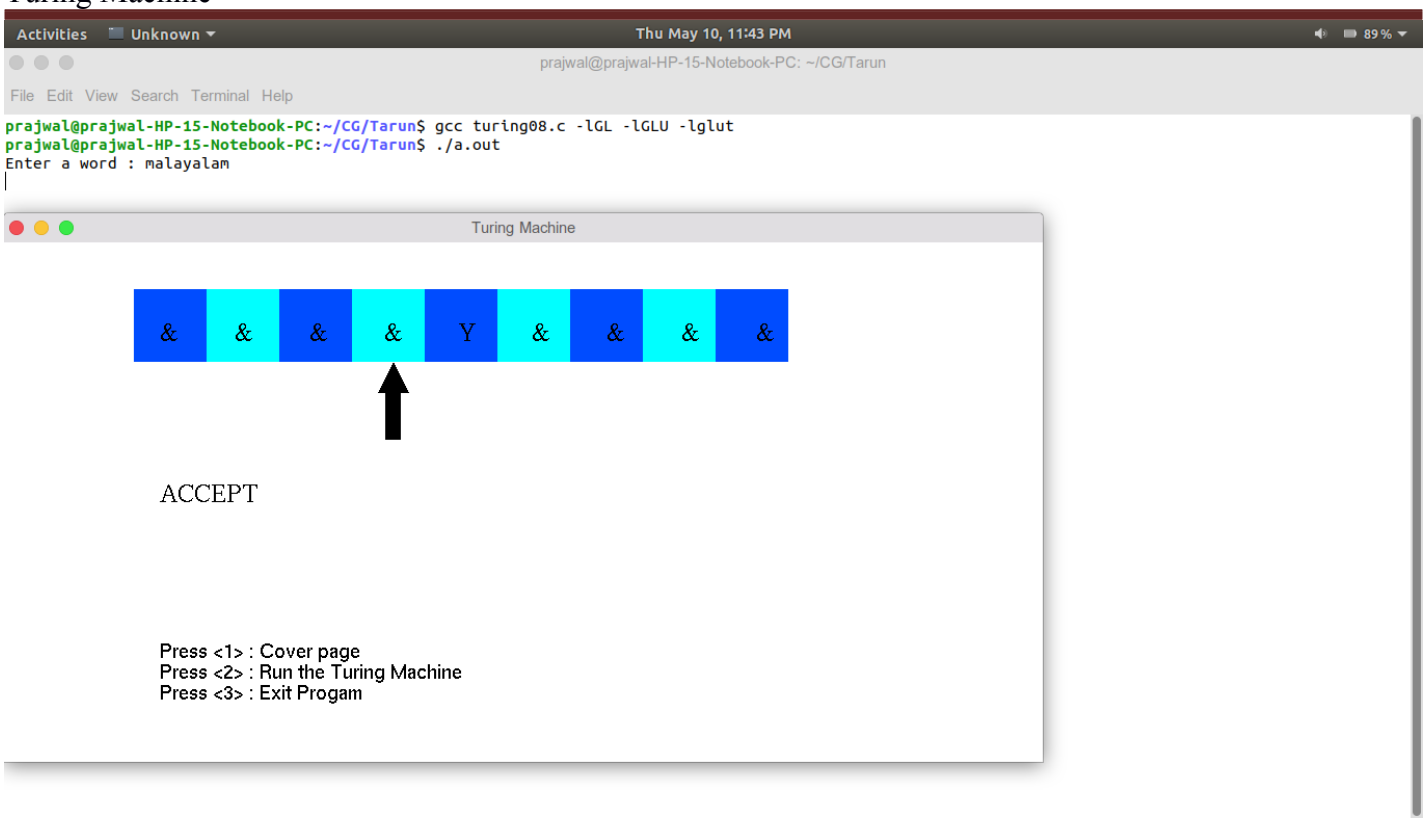


Fig 4.3: String malayalam has been accepted

This concludes that our project will work for all strings of odd length.

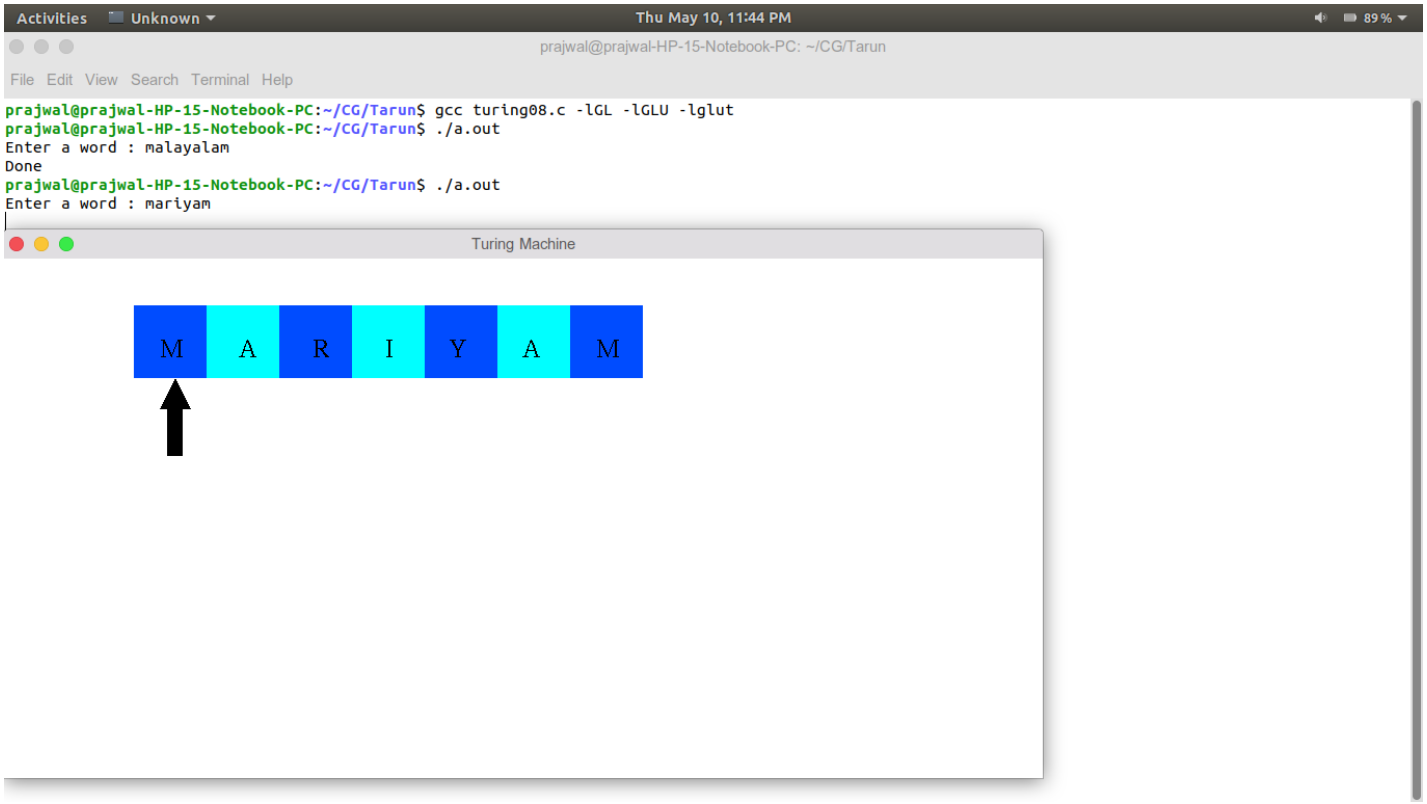


Fig 4.4: String mariyam has been entered

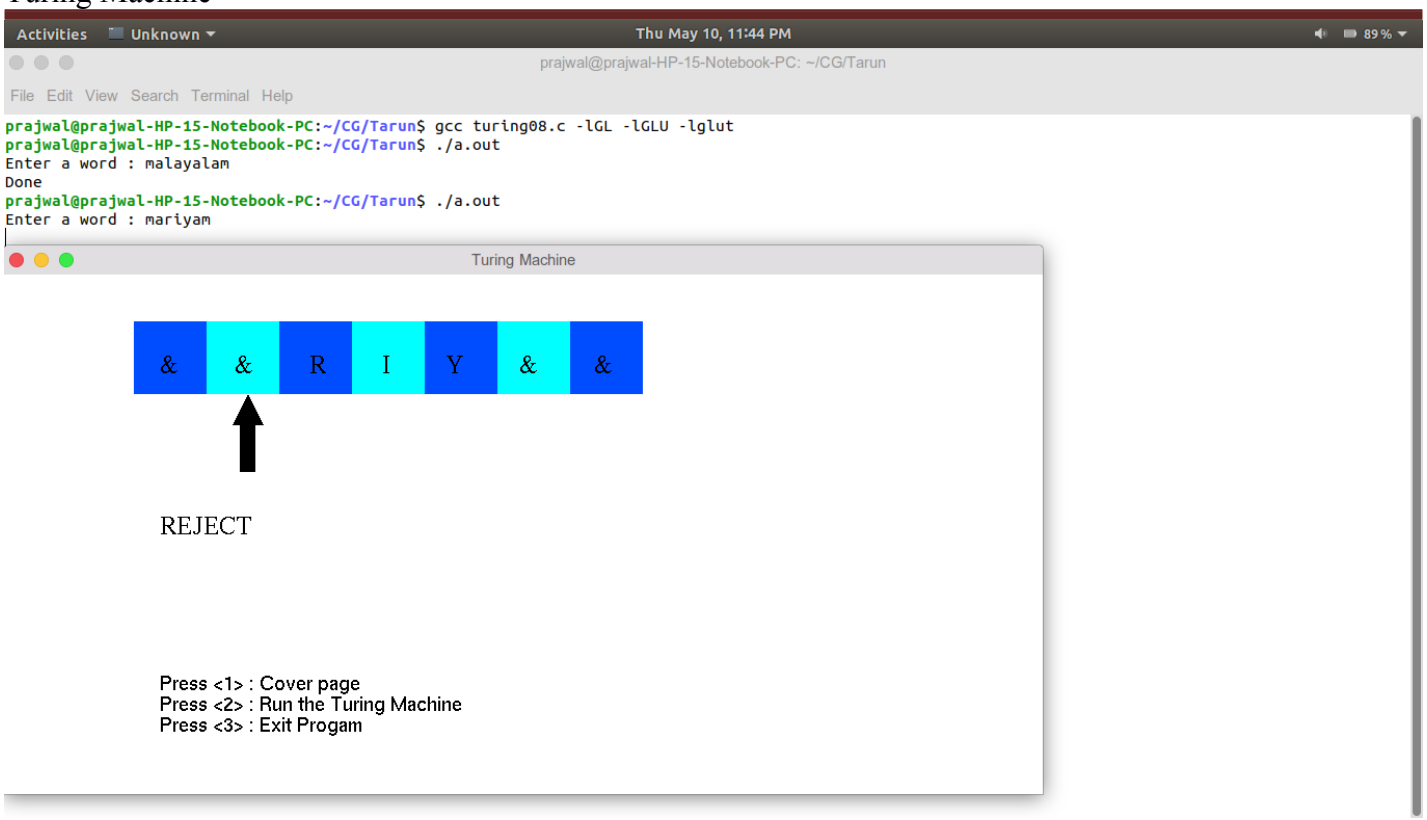


Fig 4.5: String mariyam has been rejected

This shows that the string will be rejected if its not a palindrome.

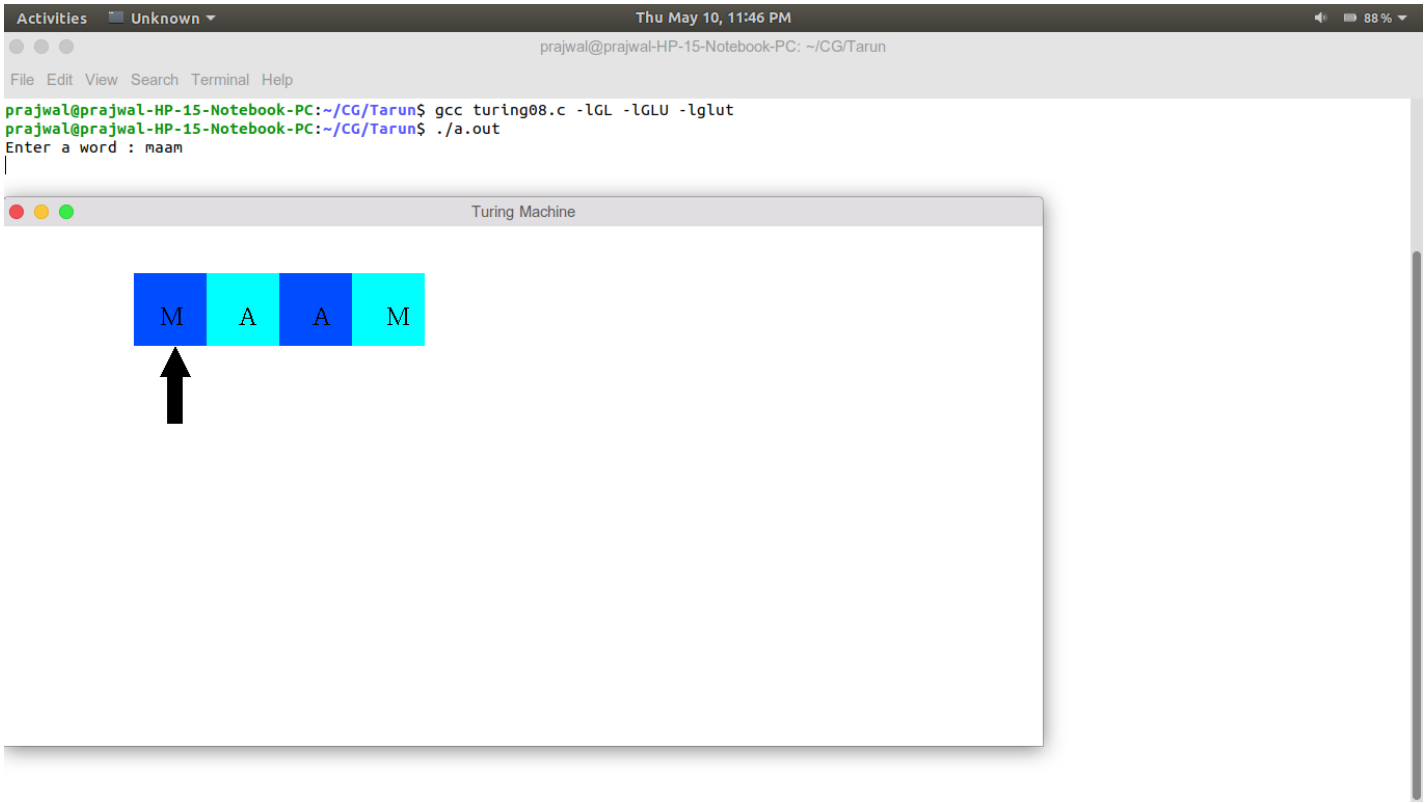


Fig 4.6: String maam has been entered

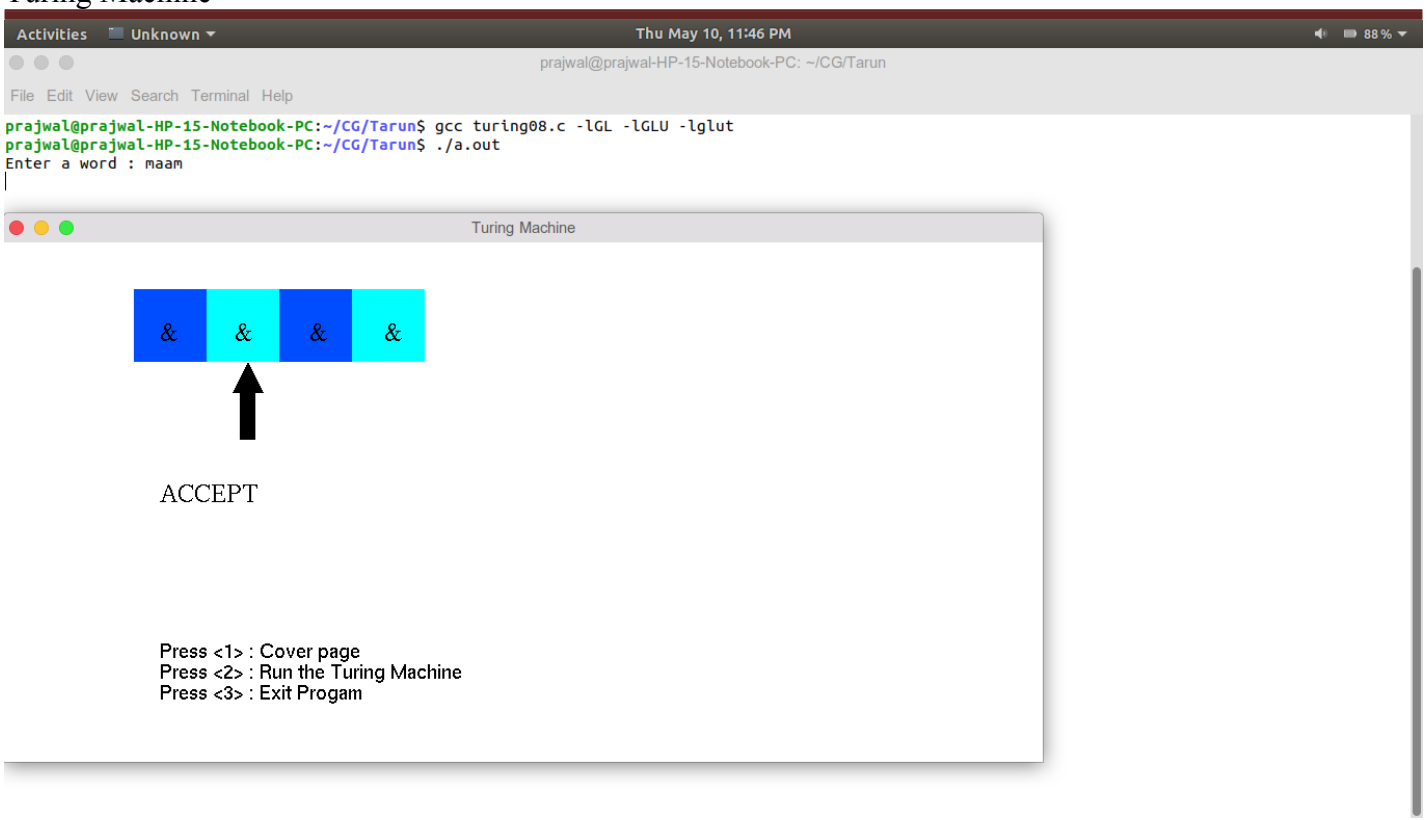


Fig 4.7: String maam has been accepted

This shows that Turing machine will accept strings which are palindromes for even length.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

On implementing this project we got to know how to use some of the built in functions effectively and how to interact efficiently and easily. We got a good exposure of how simulations are developed, by working on this project.

The OpenGL Utility Toolkit (GLUT) is a programming interface with ANSI C bindings for writing window system independent OpenGL programs.

The GLUT application-programming interface (API) requires very few routines to display a graphics scene rendered using OpenGL. The GLUT API (like the OpenGL API) is stateful. Most initial GLUT state is defined and the initial state is reasonable for simple programs. The GLUT routines also take relatively few parameters. No pointers are returned. The only pointers passed into GLUT are pointers to character strings (all strings passed to GLUT are copied, not referenced) and opaque font handles.

Turing machine prove fundamental limitations on the power of mechanical computation. While they can express arbitrary computation, there minimalistic design makes them unsuitable for computation in practise: real-world computers are based on different designs that, unlike Turing machine, use random access memory.

REFERENCES

[1] Edward Angel, "Interactive Computer Graphics", 5th edition, Pearson Education, 2005

[2] "Computer Graphics", Addison-Wesley 1997 James D Foley, Andries Van Dam, StevenK Feiner, John F Hughes.

[3] F.S.Hill and Stephen M.Kelly, "Computer Graphics using OpenGl ", 3rd edition

[4] http://www.opengl.org

[5] http:// www.openglprojects.in

[6] http://www.openglprogramming.com/