MINI PROJECT

on

Simple 2D Graph Plotting

Submitted in partial fulfilment for the completion of

BE-III Semester

In

INFORMATION TECHNOLOGY

By

Preethivardhan Anusri Ega (160117737102)
Hemanth Naga Babu Chanda (160117737095)

Under the guidance of

Ms. Surya Samantha
Assistant Professor,
Dept. of IT, CBIT.

DEPARTMENT OF INFORMATION TECHNOLOGY
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)
(Affiliated to Osmania University; Accredited by NBA(AICTE) and NAAC(UGC), ISO Certified 9001:2015)
GANDIPET, HYDERABAD – 500 075
Website: www.cbit.ac.in

2018-2019

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)

DEPARTMENT OF INFORMATION TECHNOLOGY

(Affiliated to Osmania University)

GANDIPET, HYDERABAD – 500 075

CERTIFICATE

This is to certify that the project work entitled "**Simple 2D Graph Plotting**" submitted to **CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY,** in partial fulfilment of the requirements for the award of the completion of 3rd semester of B.E in Information Technology, during the academic year 2018-2019, is a record of original work done by **Preethivardhan Anusri Ega (160117737102), Hemanth Naga Babu Chanda (160117737095)** during the period of study in Department of IT, CBIT, HYDERABAD, under our supervision and guidance.

Project Guide
Ms . Surya Smanatha
Asst. Professor, Dept. of IT,
CBIT, Hyderabad.

Head of the Department
Dr.Suresh Pabboju
Professor, Dept. of IT,
CBIT, Hyderabad.

DECLARATION

I hereby declare that the mini project which we have done was under the supervision of the faculty of our college.

No part of our project has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.

Date :

Signature :

Preethivardhan Anusri Ega (160117737102)

Hemanth Naga Babu Chanda (160117737095)

Abstract

Simple 2D graph plotting is a mini-project implemented on C++ platform to plot two dimensional graphs. The program takes mathematical data from the user as input and generates a corresponding graph on the Cartesian plane in DOS window. The main module used in the program is <**graphics.h**> which include various predefined functions to achieve drawing lines, circles, mouse applications, etc..

The program flow continues in a hierarchical manner and after selecting a choice a new DOS window is opened and the graph is drawn in that window using GUI. After choosing which graph to be drawn we also provided an option to choose between different colours with which to be drawn. The axes are automatically constrained to the range of the graph. Graph can be analysed easily by getting the coordinates of any point in the graph by moving the mouse to that respective point.

ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude to Ms.Surya Samantha, our project guide, for her invaluable guidance and constant support, along with her capable instruction and persistent encouragement.

We are grateful to our Head of Department, Dr.Suresh Pabboju, for his steady support and for the provision of every resource required for the completion of this project.

We would like to take this opportunity to thank our Principal, Dr.P.Ravinder Reddy, as well as the management of the institute, for having designed an excellent learning atmosphere.

Our thanks are due to all member of the staff and our lab assistants for providing me with the help required to carry out the groundwork of this project.

CONTENTS

1. Introduction

1.1 Motivation

Mathematics is a common subject which we keep learning right from our kindergarten till higher studies which include many equations which can represented graphically. The graphs of the equations are used to analyze and solve the equations. The main idea behind the application had taken birth when we ourselves had faced the problem to draw the graphs of any given mathematical equations to solve them. So we have designed an application which takes the mathematical equation as input and displays the graph in Cartesian plane .

1.2 Objective of the Project

The main Objective while implementing this application was to simplify drawing graphs of any mathematical equations and there by the analysis of those mathematical equations is made easy by judging their respective graphs.

1.3 Tools

This application has been totally developed on C++ platform by using 'graphics.h' exclusively for the implementation of Graphical User Interface. The IDE we used was Dev C++. The application can be run on any computer with basic specifications.

2. Software and Tools

2.1.1 C++

C++ is a statically-typed, free-form, compiled, multi-paradigm, intermediate-level general-purpose middle-level programming language." In simple terms, C++ is a sophisticated, efficient and a general-purpose programming language based on C. It was developed by Bjarne Stroustrup in 1979.

Many of today's operating systems, system drivers, browsers and games use C++ as their core language. This makes C++ one of the most popular languages today. Since it is an enhanced/extended version of C programming language, C and C++ are often denoted together as C/C++.

2.1.2 Features of C++

- **C++ is an open ISO-standardized language.**
  For a time, C++ had no official standard and was maintained by a de-facto standard, however since 1998, C++ is standardized by a committee of the ISO. Their page may be accessed here.

- **C++ is a compiled language.**
  C++ compiles directly to a machine's native code, allowing it to be one of the fastest languages in the world, if optimized.

- **C++ is a strongly-typed unsafe language.**
  C++ is a language that expects the programmer to know what he or she is doing, but allows for incredible amounts of control as a result.

- **C++ supports both manifest and inferred typing.**
  As of the latest C++ standard, C++ supports both manifest and inferred typing, allowing flexibility and a means of avoiding verbosity where desired.

- **C++ supports both static and dynamic type checking.**
  C++ allows type conversions to be checked either at compile-time or at run-time, again offering another degree of flexibility. Most C++ type checking is, however, static.

- **C++ offers many paradigm choices.**
  C++ offers remarkable support for procedural, generic, and object-oriented programming paradigms, with many other paradigms being possible as well.

- **C++ is *portable.***
  As one of the most frequently used languages in the world and as an open language, C++ has a wide range of compilers that run on many different platforms that support it. Code that exclusively uses C++'s standard library will run on many platforms with few to no changes.

- **C++ is upwards compatible with C**
  C++, being a language that directly builds off C, is compatible with almost all C code. C++ can use C libraries with few to no modifications of the libraries' code.

- **C++ has incredible library support.**
  A search for "library" on the popular project-management website SourceForge will yield over 3000 results for C++ libraries. A link to the results of the search may be found here.

2.2 <graphics.h> in C++

C++ graphics using <graphics.h> functions can be used to draw different shapes, display text in different fonts, change colors and many more. Using functions of <graphics.h> in Dev C++ you can make graphics programs, animations, projects, and games. You can draw circles, lines, rectangles, bars and many other geometrical figures. You can change their colors using the available functions and fill them. Following is a list of functions of graphics.h header file.

- void arc (int x, int y, int stangle, int endangle, int radius);
- void bar (int left, int top, int right, int bottom);
- void bar3d (int left, int top, int right, int bottom, int depth, int topflag);
- ostringstream bgiout;
- void circle (int x, int y, int radius);
- void cleardevice (void);
- void clearmouseclick(int kind);
- void clearviewport (void);
- void closegraph (int window=ALL_WINDOWS);
- int converttorgb (int color);
- void delay (int millisec);
- void detectgraph (int *graphdriver, int *graphmode);
- void drawpoly (int numpoints, int *polypoints);
- void ellipse (int x, int y, int stangle, int endangle, int xradius, int yradius);
- void fillellipse (int x, int y, int xradius, int yradius);
- void fillpoly (int numpoints, int *polypoints);
- void floodfill (int x, int y, int border);
- int getactivepage (void);
- void getarccoords (struct arccoordstype *arccoords);
- void getaspectratio (int *xasp, int *yasp);
- int getbkcolor (void);
- int getch (void);
- int getcolor (void);
- int getcurrentwindow (void);
- struct palettetype* getdefaultpalette (void);
- int getdisplaycolor (int color);
- char* getdrivername (void);
- void getfillpattern (char *pattern);
- void getfillsettings (struct fillsettingstype *fillinfo);
- int getgraphmode (void);
- void getimage (int left, int top, int right, int bottom, void *bitmap);
- void getlinesettings (struct linesettingstype *lineinfo);
- int getmaxcolor (void);

- int getmaxmode (void);
- int getmaxheight (void);
- int getmaxwidth (void);
- int getmaxx (void);
- int getmaxy (void);
- char* getmodename (int mode_number);
- void getmoderange (int graphdriver, int *lomode, int *himode);
- void getmouseclick(int kind, int& x, int& y);
- void getpalette (struct palettetype *palette);
- int getpalettesize (void);
- int getpixel (int x, int y);
- void gettextsettings (struct textsettingstype *texttypeinfo);
- void getviewsettings (struct viewporttype *viewport);
- int getvisualpage (void);
- int getwindowheight (void);
- int getwindowwidth (void);
- int getx (void);
- int gety (void);
- void graphdefaults (void);
- char* grapherrormsg (int errorcode);
- int graphresult(void);
- unsigned imagesize (int left, int top, int right, int bottom);
- void initgraph (int *graphdriver, int *graphmode, char *pathtodriver);
- int initwindow (int width, int height, const char* title="Windows BGI", int left=0, int top=0, bool dbflag=false, bool closeflag=true);
- int installuserdriver (char *name, int huge (*detect)(void));
- int installuserfont (char *name);
- bool ismouseclick(int kind);
- int kbhit (void);
- void line (int x1, int y1, int x2, int y2);
- void linerel (int dx, int dy);
- void lineto (int x, int y);
- int mousex (void);
- int mousey (void);
- void moverel (int dx, int dy);
- void moveto (int x, int y);
- void outtext (char *textstring);
- void outtextxy (int x, int y, char *textstring);
- void pieslice (int x, int y, int stangle, int endangle, int radius);
- void printimage (const char* title=NULL, double width_inches=7,double order_left_inches=0.75, double border_top_inches=0.75, int left=0, int right=0, int right=INT_MAX, int bottom=INT_MAX);
- void putimage (int left, int top, void *bitmap, int op);
- void putpixel (int x, int y, int color);

- void readimagefile (const char* filename=NULL,int left=0, int top=0, int right=INT_MAX, int bottom=INT_MAX);
- void rectangle (int left, int top, int right, int bottom);
- int registerbgidriver (void (*driver)(void));
- int registerbgifont (void (*font)(void));
- void registermousehandler (int kind, void h(int, int));
- void restorecrtmode (void);
- RGB functions:
- COLOR(r,g,b),
- RED_VALUE(v), GREEN_VALUE(v), BLUE_VALUE(v),
- IS_BGI_COLOR(v), IS_RGB_COLOR(v)
- void sector (int x, int y, int stangle, int endangle, int xradius, int yradius);
- void setactivepage (int page);
- void setallpalette (struct palettetype *palette);
- void setaspectratio (int xasp, int yasp);
- void setbkcolor (int color);
- void setcolor (int color);
- void setcurrentwindow (int window);
- void setmousequeuestatus(int kind, bool status=true);
- void setfillpattern (char *upattern, int color);
- void setfillstyle (int pattern, int color);
- unsigned setgraphbufsize (unsigned bufsize);
- void setgraphmode (int mode);
- void setlinestyle (int linestyle, unsigned upattern, int thickness);
- void setpalette (int colornum, int color);
- void setrgbpalette (int colornum, int red, int green, int blue);
- void settextjustify (int horiz, int vert);
- void settextstyle (int font, int direction, int charsize);
- void setusercharsize (int multx, int divx, int multy, int divy);
- void setviewport (int left, int top, int right, int bottom, int clip);
- void setvisualpage (int page);
- void setwritemode (int mode);
- int showerrorbox (const char *message);
- int swapbuffers (void);
- int textheight (char *textstring);
- int textwidth (char *textstring);
- void writeimagefile (const char* filename=NULL,double width_inches=7, double border_left_inches=0.75, double border_top_inches=0.75,int left=0, int top=0, int right=INT_MAX, int bottom=INT_MAX);

3. System Design

3.1 Flow Chart

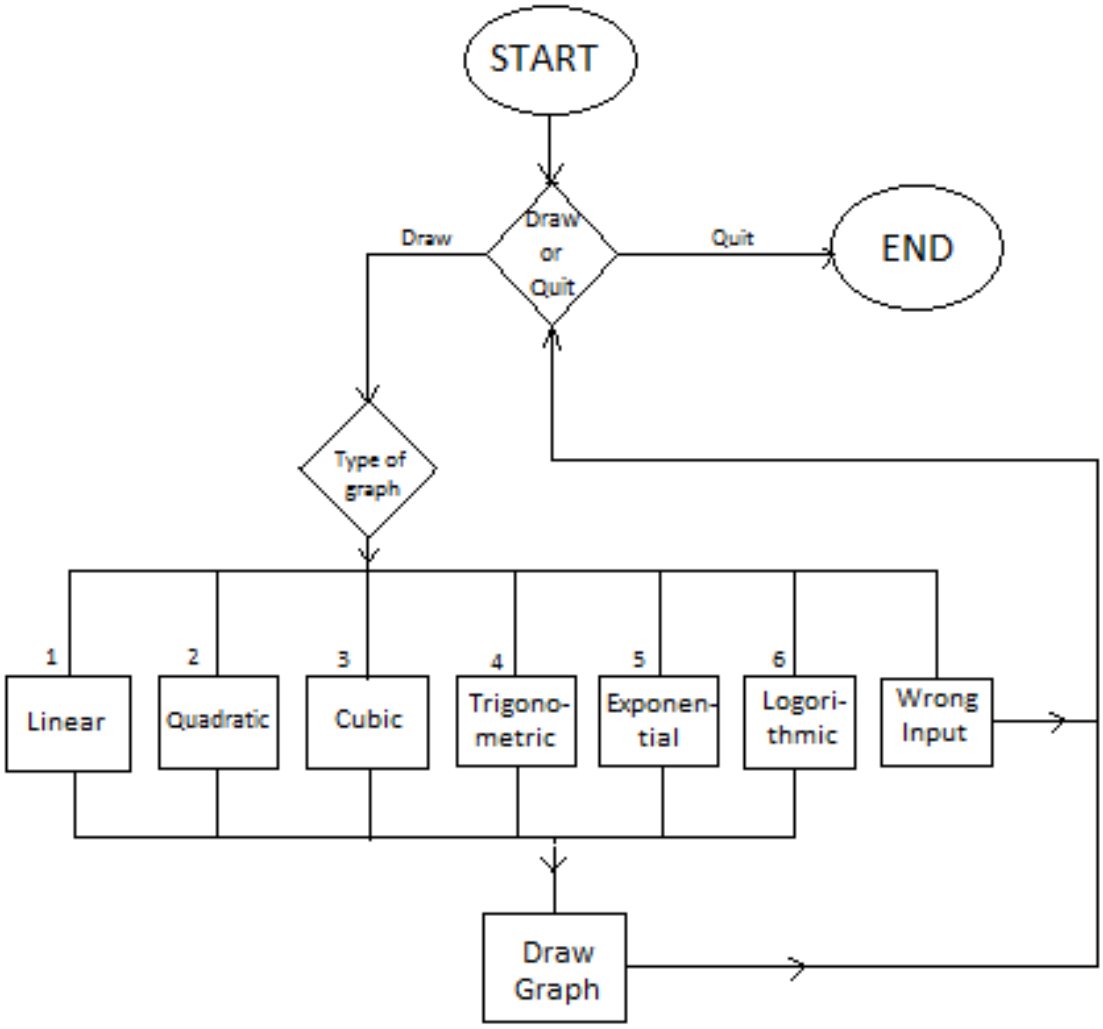

Fig 3.1 . Flowchart representing the operations carried out

4. Implementation

4.1 Code Structure

The first thing required was the <graphics.h>. So we have downloaded <graphics.h> header file from the internet and included in the libraries of our IDE i.e., Dev C++. Then this header file was also included into our program with the statement "#include<graphics.h>" in the code. To obtain Graphical User Interface our Console Application's linker options were changed as per the requirements. Then we have written code to develop different types of graphs using different functions for each and every type of graph. The list of functions and their description is given below.

1. void axes() : Draws the axes.
2. void point(int xd,int yd) : Displays the coordinates of the graph using pointer device.
3. int choosecolor() : Displays menu to choose graph color and sets the color as per the input given
4. void sine(int color) : Draws Trigonometric sine graph.
5. void asine(int color) : Draws Trigonometric sine inverse graph.
6. void sineh(int color) : Draws Trigonometric sine hyperbolic graph.
7. void asineh(int color) : Draws Trigonometric sine inverse hyperbolic graph.
8. void cosine(int color) : Draws Trigonometric cosine graph.
9. void acosine(int color) : Draws Trigonometric cosine inverse graph.
10. void cosineh(int color) : Draws Trigonometric cosine hyperbolic graph.
11. void acosineh(int color) : Draws Trigonometric cosine inverse hyperbolic graph.
12. void tangent(int color) : Draws Trigonometric tangent graph.
13. void atangent(int color) : Draws Trigonometric tangent inverse graph.
14. void tangenth(int color) : Draws Trigonometric tangent hyperbolic graph.
15. void atangenth(int color) :Draws Trigonometric tangent inverse hyperbolic graph.
16. void Sqrt(int color) : Draws square root graph.
17. void Cbrt(int color) : Draws cube root graph.
18. void Log(int color) : Draws Logarithm base 10 graph.
19. void ln(int color) : Draws Logarithm base e graph.

20. void Exp(int color) : Draws e^x graph.
21. void Pow(int k,int color) : Draws x^k graph.
22. void lin(int color) : Draws Linear equation graph.
23. void quad(int color) : Draws Quadratic equation graph.
24. void cub(int color) : Draws Cubic equation graph.

The main() function contains the menu format in a hierarchical switch case statements with above defined function calling statements.

4.2 Execution

1. The execution of the program starts by asking the user either to Draw or Exit. Choosing Exit terminates the program.

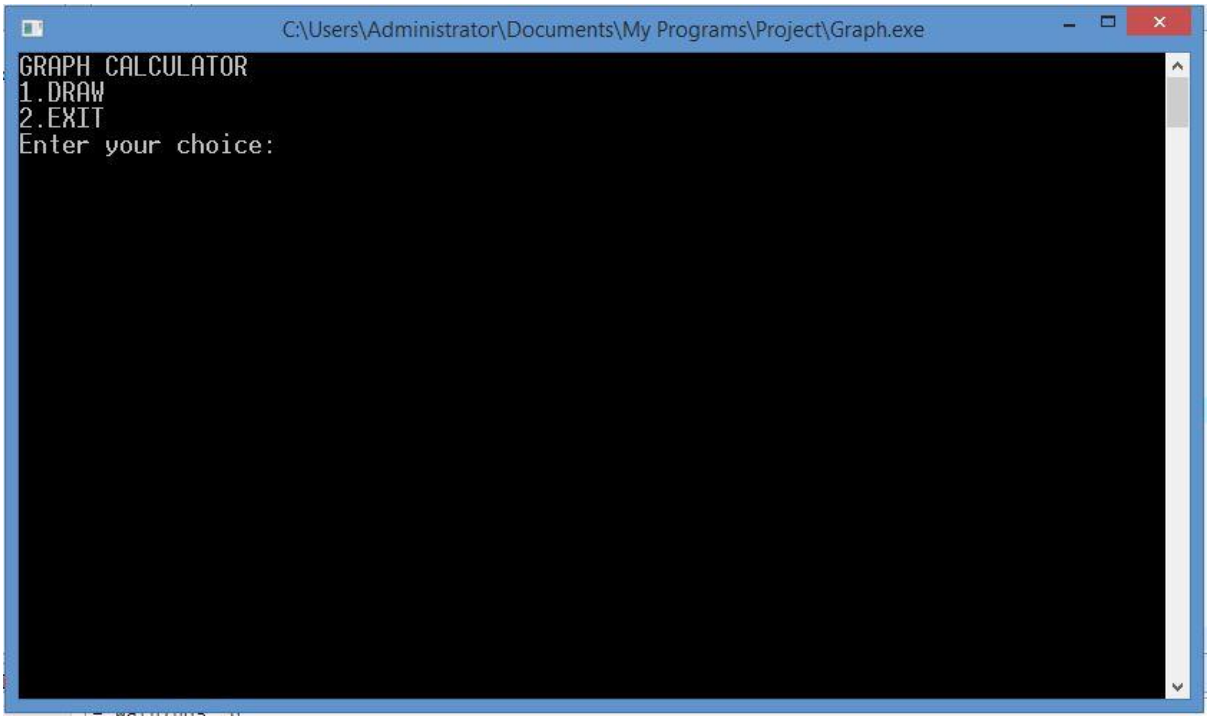

Fig 4.1. Main Menu

2. After choosing Draw option the user is asked to choose the type of the graph he wants.

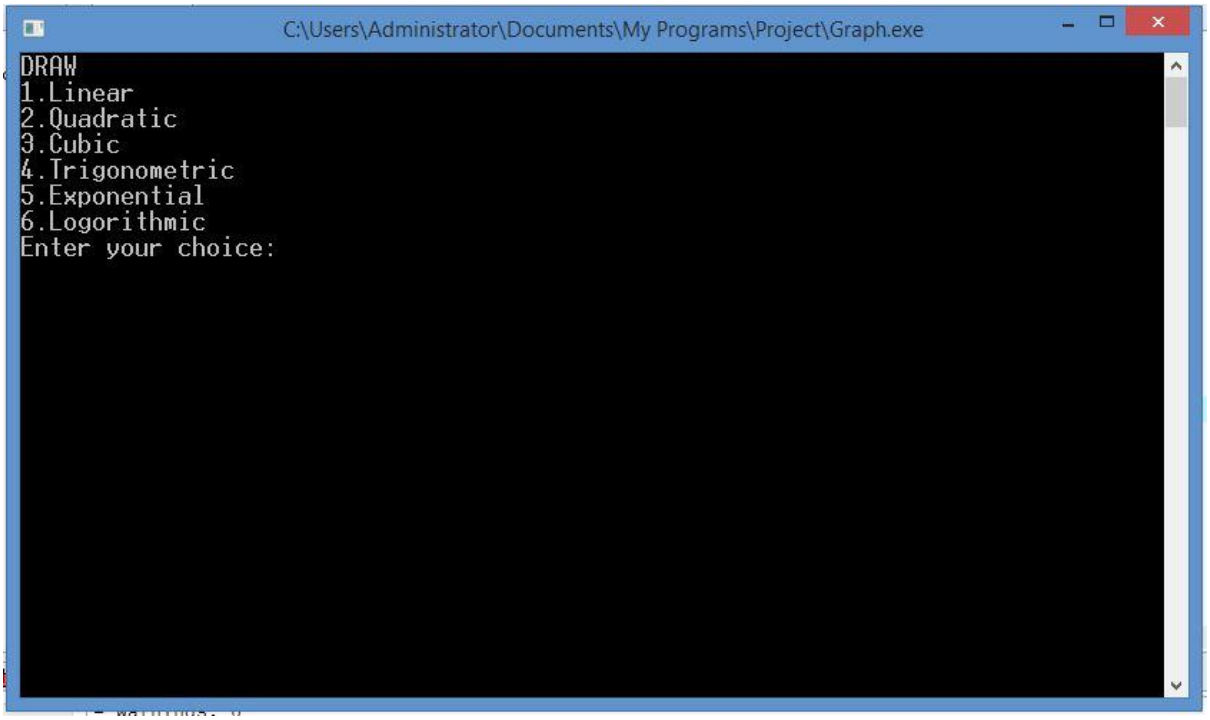

Fig 4.2. Menu of different types of graphs

3. There are sub menus in Trigonometric, Exponential, Logarithmic types of graphs.

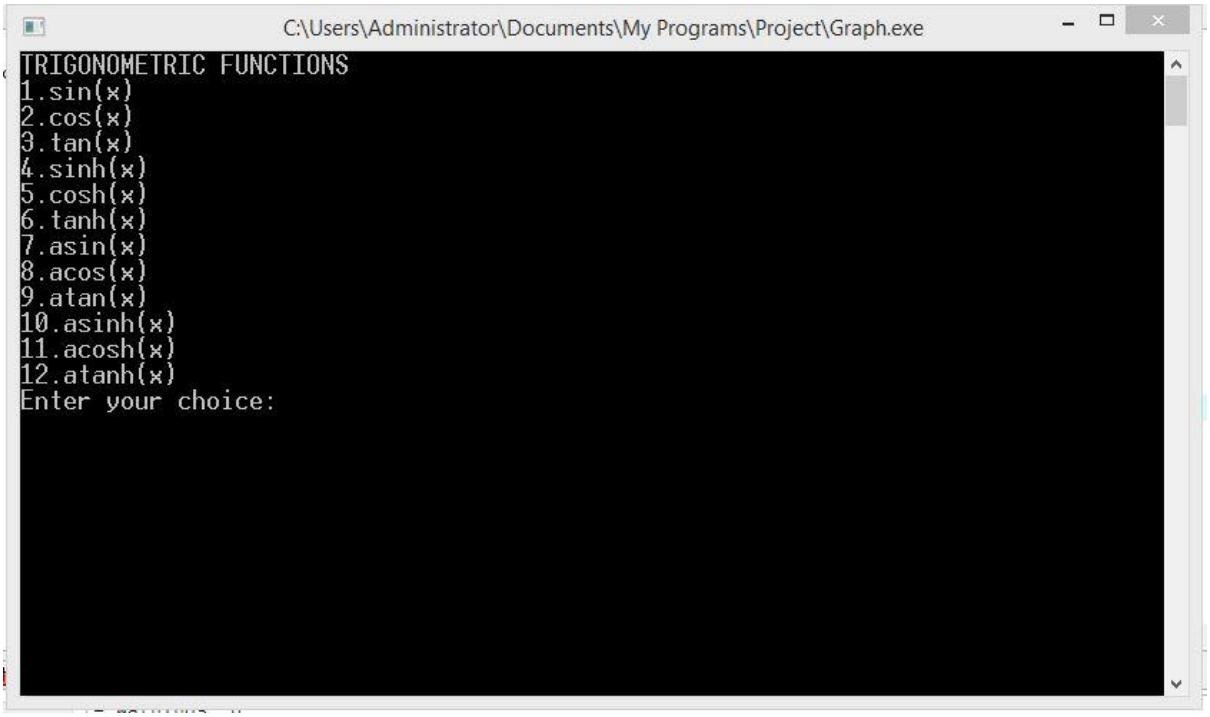

Fig 4.3. Trigonometric Functions Menu

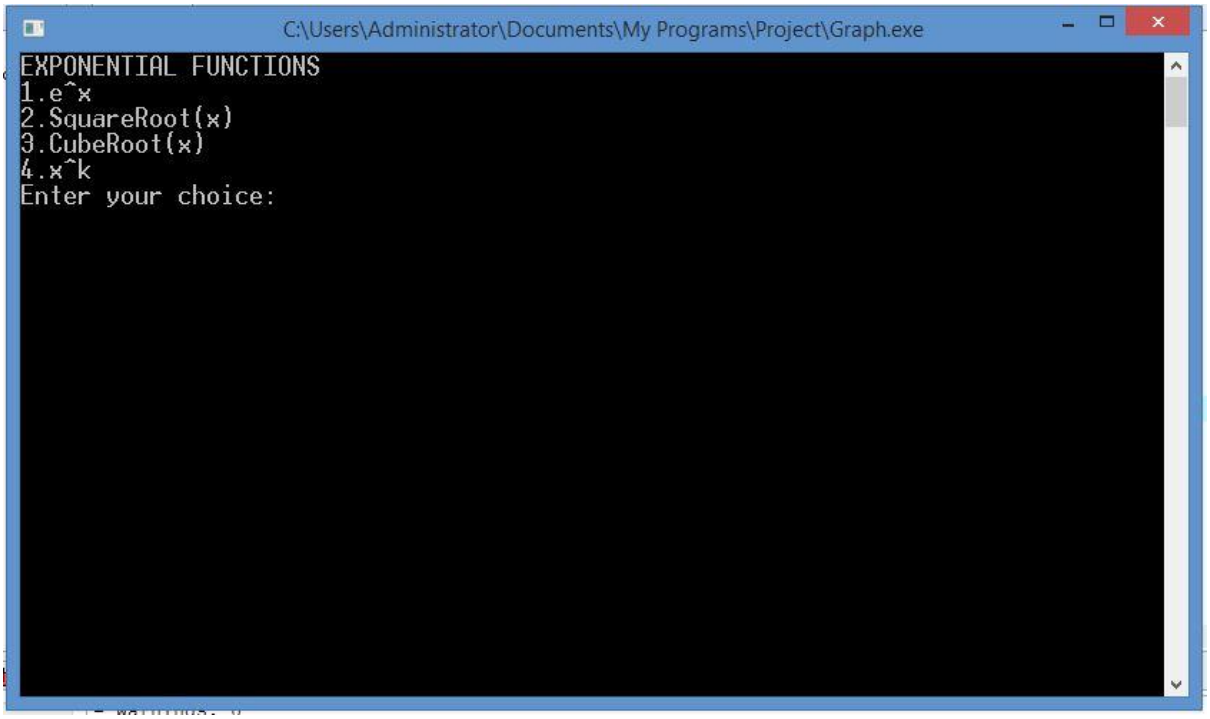

Fig 4.4. Exponential Functions Menu

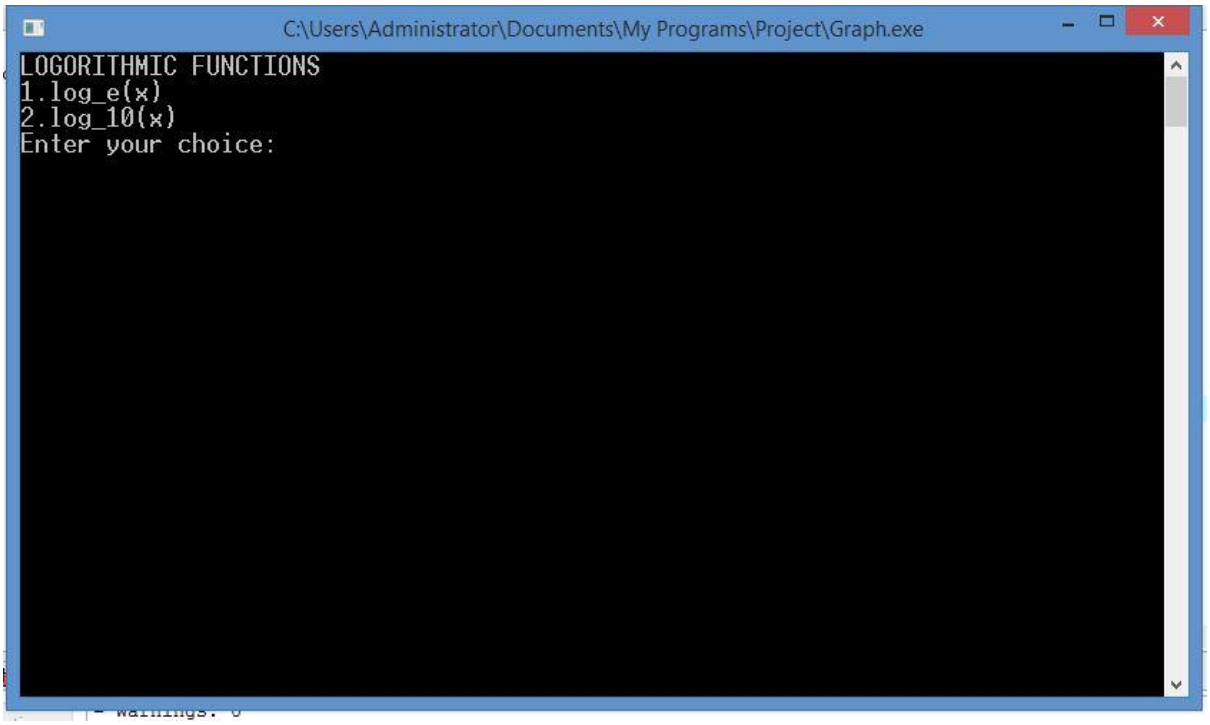

Fig 4.5. Logarithmic Functions Menu

4. If wrong option is entered in any menu, program is redirected to main menu.

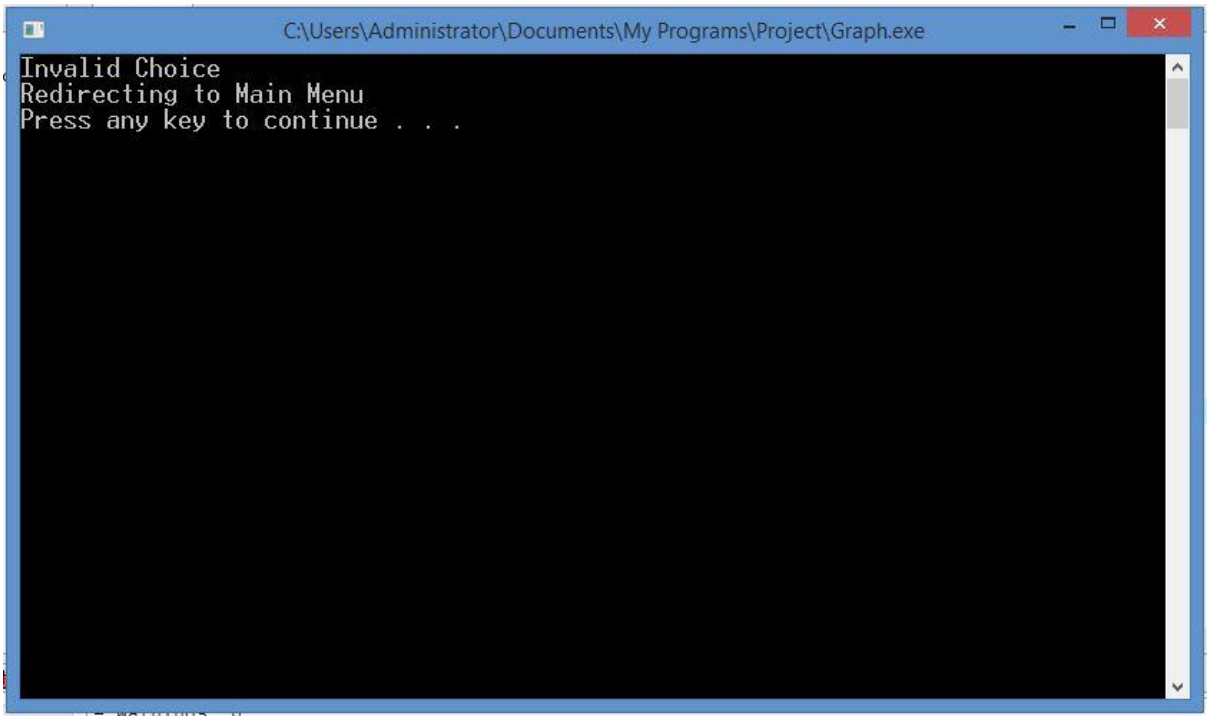

Fig 4.6. Entry of invalid choice

5. After selecting which graph to be drawn, it takes all the required information to draw the graph from the user.

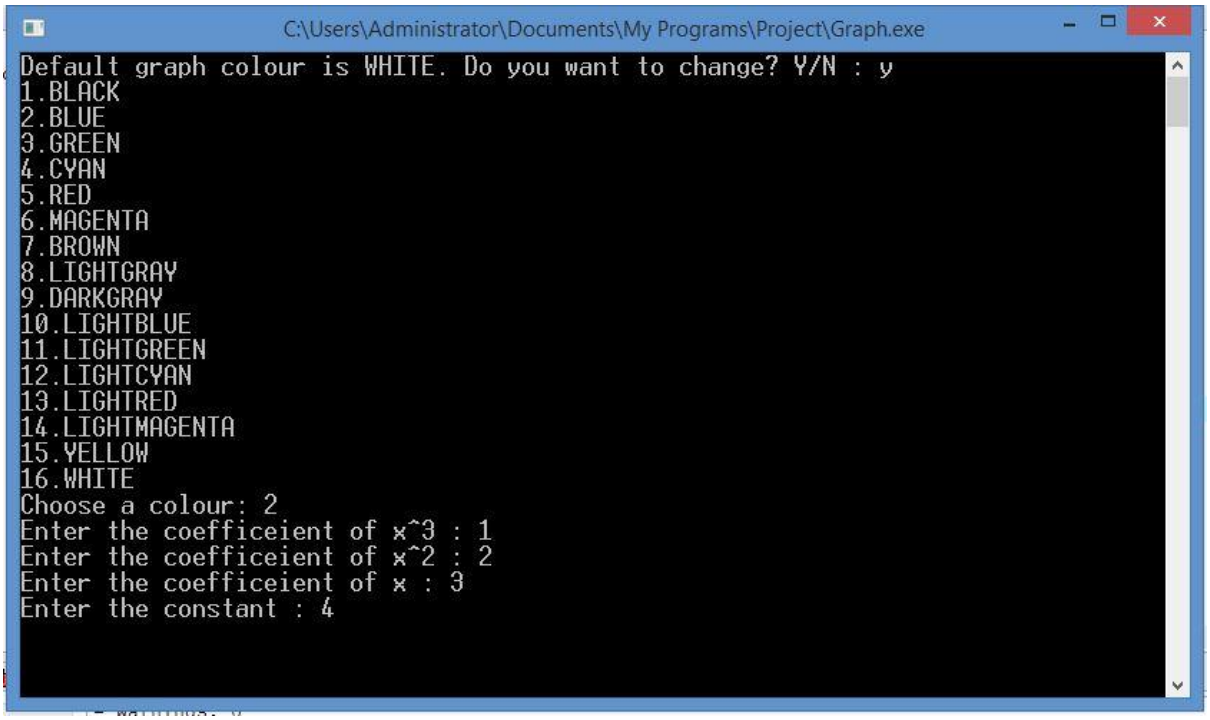

Fig 4.7. Input of required information

6. The Graph is Drawn. Some sample Graphs are shown below.

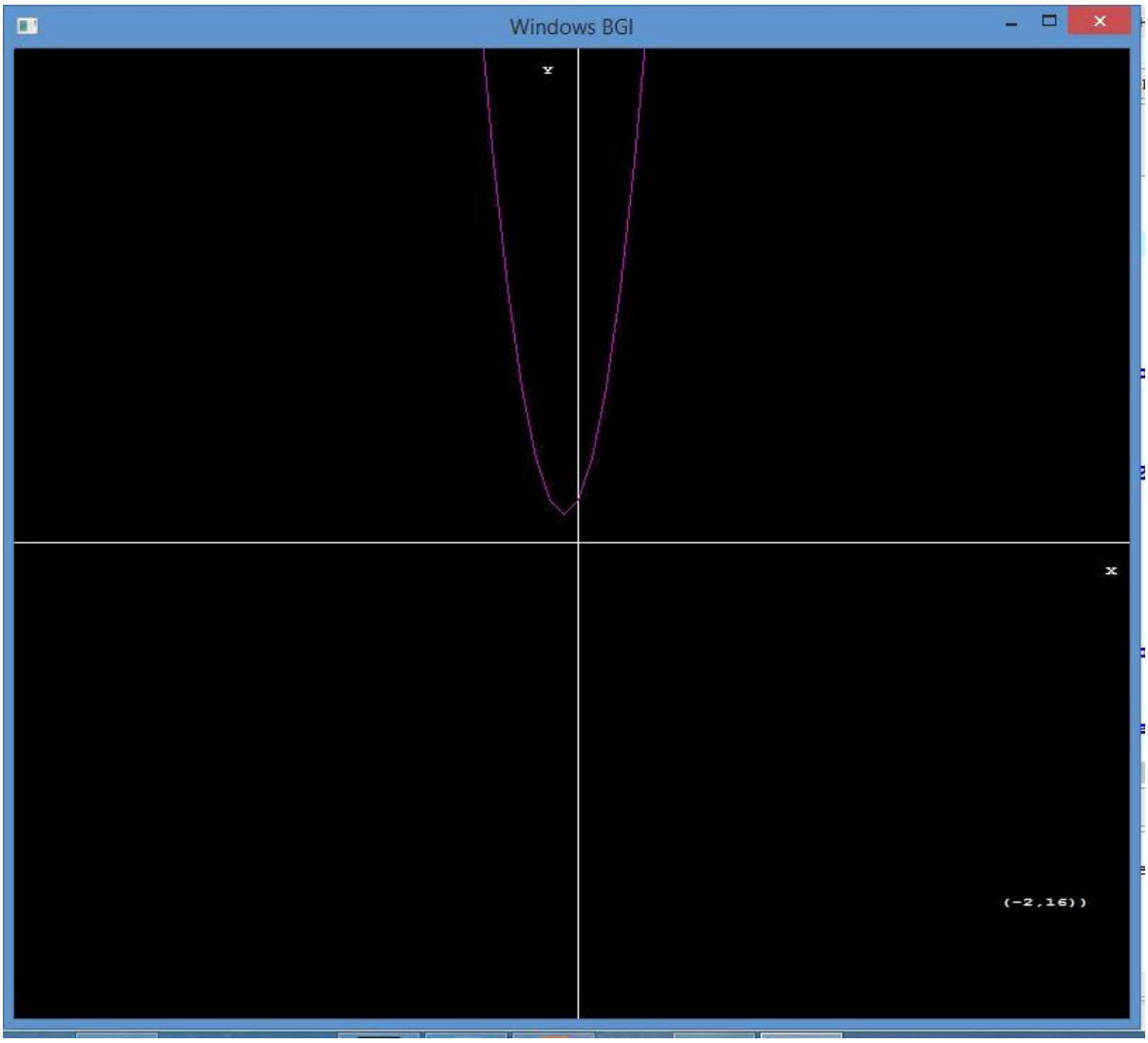

Fig 4.8. Graph of a Quadratic Equation.

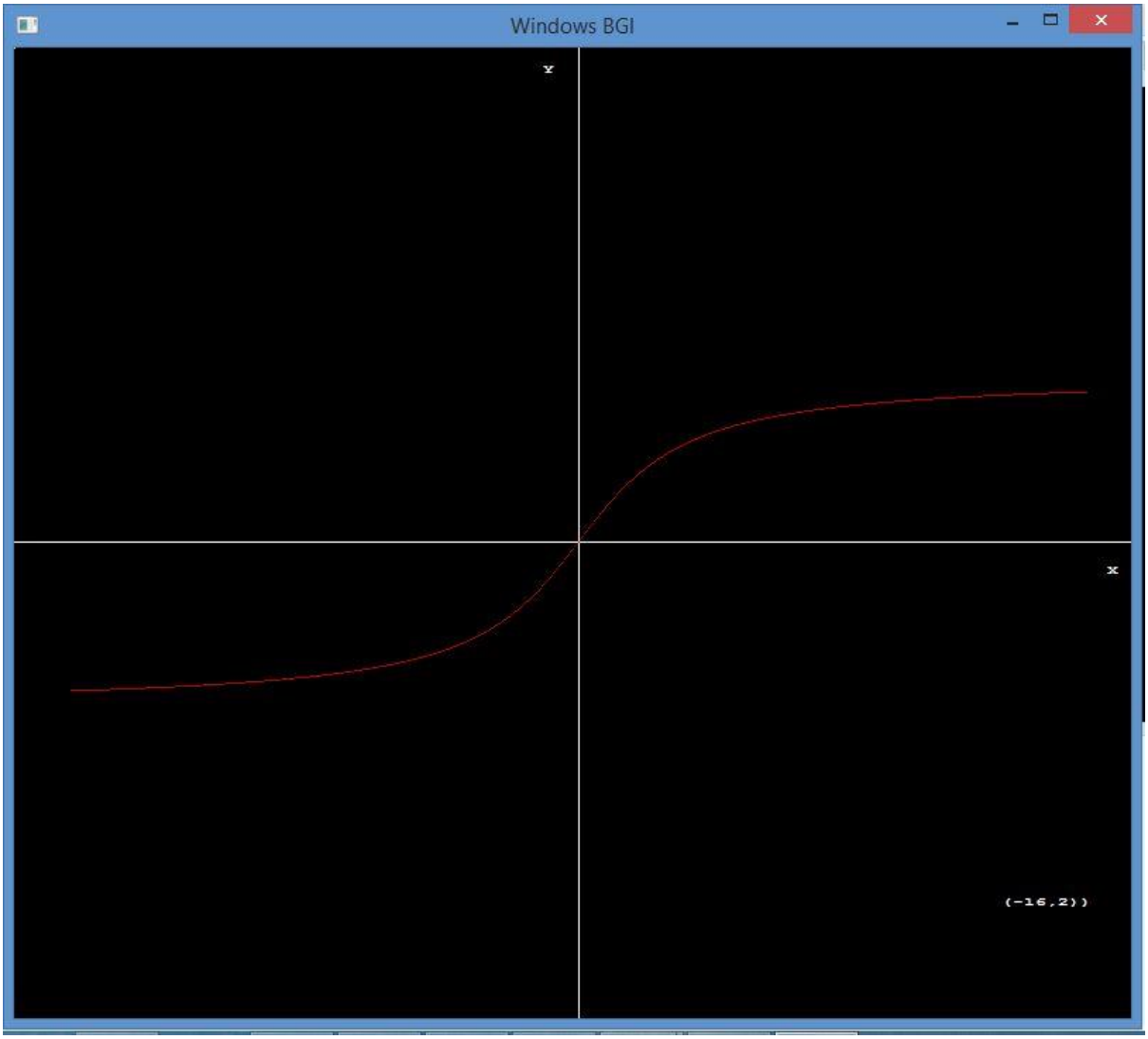

Fig 4.9. Graph of Inverse Tangent Function

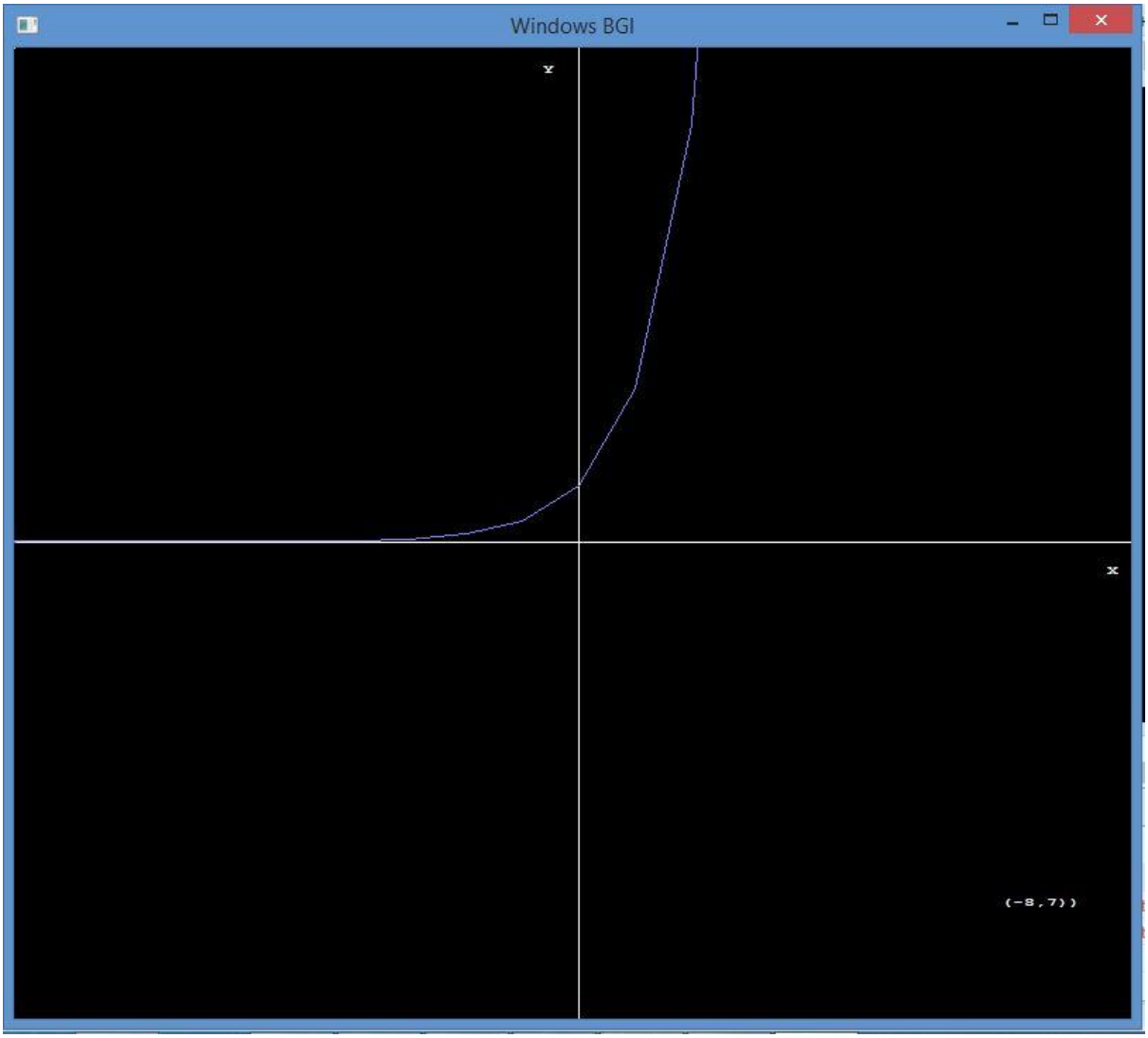

Fig 4.10 Graph of e^x

5. Conclusion

Using this application will make life very simple for students who study Mathematics mainly for higher studies of Mathematics, Physics and Chemistry. Various concepts and phenomena in Physics and Chemistry are explained mathematically and represented graphically. In each and every such concept or phenomenon this application is useful which displays the graph by taking the mathematical equation as input.

The main Objective while implementing this application was to simplify drawing graphs of any mathematical equations and there by the analysis of those mathematical equations is made easy by judging their respective graphs. We designed the program in such a way that it perfectly matched our objectives.

6. Future Scope

The next version of this application will have many more features like :

- Two different graphs can be drawn and compared at a time.
- The front end of the application can be designed in a web page.
- The application can be made usable online.
- The application can be released in Android version.

7. Bibliography

1. The C++ Programming Language by Bjarne Stroustrup
2. www.koolplot.com
3. en.wikipedia.org
4. www.cplusplus.com
5. www.geeksforgeeks.com
6. www.quora.com