COMPUTER SCIENCE PROJECT

केन्द्रीय माध्यमिक शिक्षा बोर्ड
दिल्ली
असतो मा सद्गमय

NAME : Kunal Gupta

CLASS : 12 - C

TOPIC : Plot FunC++

Index

Index

Certificate

Certificate

This is to certify that Kunal Gupta of class twelve, Tagore International School, East of Kailash has successfully completed his project in computer practicals for the AISSCE as prescribed by CBSE in the year 2014-15 under my guidance.

Date :

Registration No. :

_____

Signature of Internal Examiner

Acknowledgement

Acknowledgement

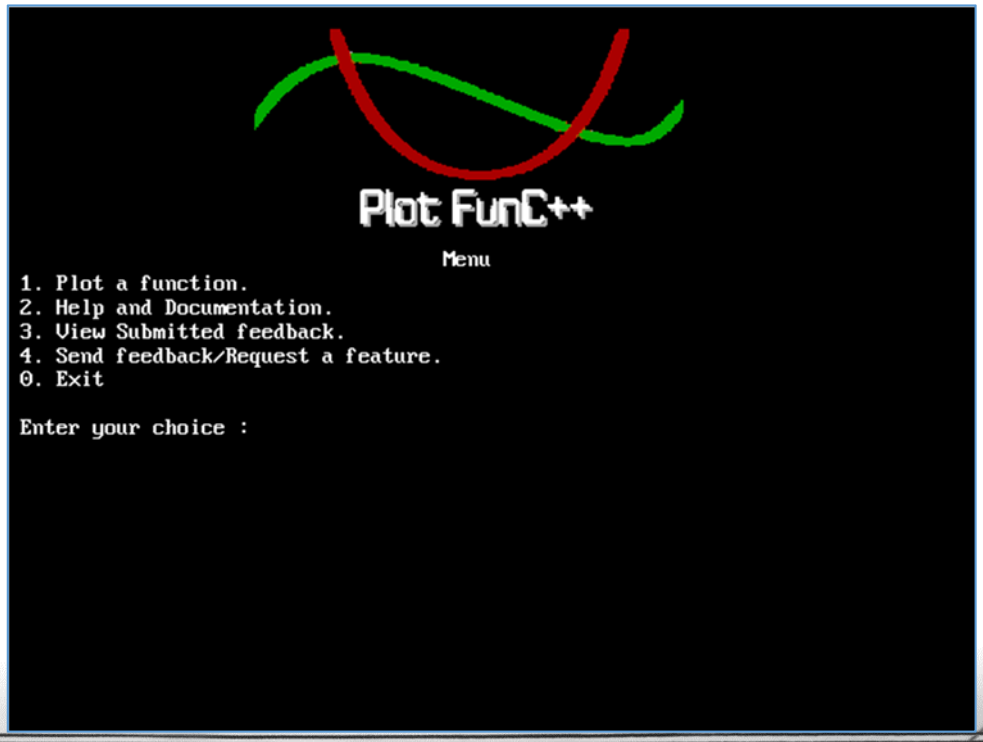
I Kunal Gupta along with my partner Sitakanta Mohapatra have made project “Plot FunC++” as a part of CBSE Project. I thank my Computer Science teacher Mrs Shikha Shah for guidance and support. Finally I would like to thank CBSE for giving me this opportunity to undertake this project.

Synopsis

Plot FunC++ is a program to take mathematical functions (of x) as input from the user and plots the graph of the inputted function on the DOS Screen using the Turbo C++ Graphics Library.

UI Design

The UI of the program has a text based interface (except for the plotting screen, which incorporates graphics). The main menu has options for Plotting viewing the Help & Documentation content, Submitting and viewing the submitted feedback along with the program logo at the top.
While plotting the function, the user can enter the Scale (number of pixels for a single unit on the screen), and precision (number of calculations per pixel). Setting the precision option greater than 1 helps in plotting graphs with steep slopes more accurately.
The graph is plotted with a green colour on a black screen with white axes (along with integral markings) with grey gridlines.

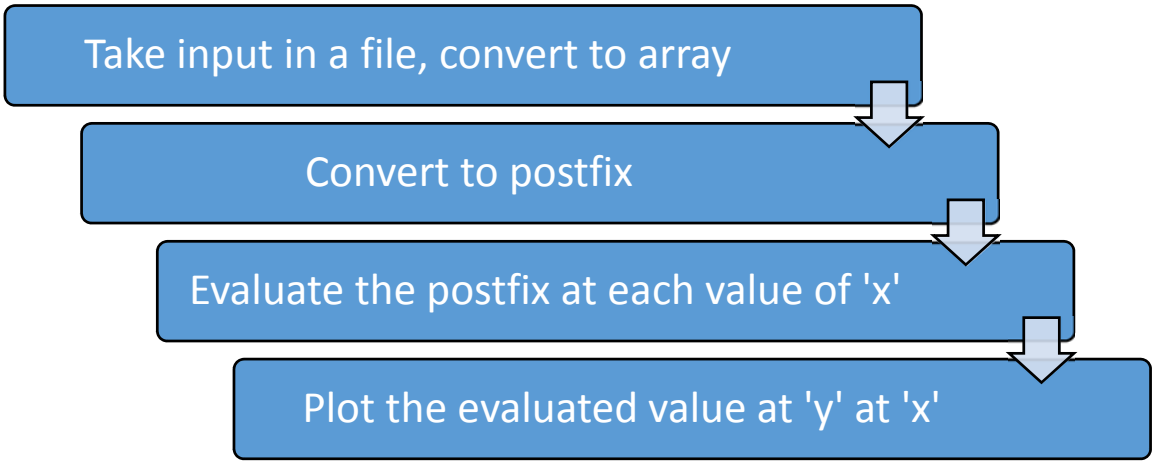


Synopsis

Design Architecture

Plotting

The main plotting program works in 4 steps:



Here's how the program works

After the user chooses to "Plot a function":

1. The program asks for the function. The function is stored directly in a file character by character, for a dynamic text input. Scale and Precision are stored as integers and validated. The stored input is then stored in dynamic memory-allotted array for easy manipulation.
2. The input is then converted into postfix (Reverse Polish Notation) while validating the input. The functions (listed under Documentation) are treated as operators.
3. A loop is triggered which increments values of 'x' from left to right (at steps of $\frac{1}{Scale \times Precision}$ and the postfix is evaluated for each value of x taking centre of the screen as origin.

Synopsis

Design Architecture

The evaluation takes place in a function which returns 0 for 'discontinuities' if evaluation is out of bounds or if there's a range or domain error and the `'continue'` is triggered.

4. If there is no discontinuity, the point is plotted at the appropriate point at corresponding 'y' of 'x' using `'putpixel()'`.

One of the first things were done was conversion of the Cartesian Coordinates to the DOS coordinates.

The DOS Coordinates have the origin at top right, x-axis towards right and y-axis downwards. The coordinates were converted so as to have origin at the centre of the screen (+ve) x-axis rightwards and (+ve) y axis downwards by:

$$x_{dos} = \frac{\text{screen height}}{2} + x_{cartesinan} \quad \text{and} \quad y_{dos} = \frac{\text{screen width}}{2} - y_{cartesinan}$$

Feedback

The feedback is stored in a binary file `'FEED.DAT'`.

The `Feed` class consists of the name, email and message of the user along with a `time_t` variable for storing the time of the feedback.

The feedback option was decided to be sending feedback online but since, Turbo C++ could not issue the new Command Prompt commands to open up a webpage in the browser, the online feedback sending could not be implemented and had to be dropped.

Synopsis

Design Architecture

Help and Documentation

There will be a `doc.txt` Text file present in the working directory which contains all the help and documentation details. The file contents can also be seen using the main menu.

`doc.txt` contains the following text currently:

```
Available functionalities in Plot FunC++ (with decreasing precedence):

Function/Operator/Constant    Syntax
--------------------------    ---------------

fabs() or the mod function    m()
floor()                       f()
ln() or loge()                n()
log() or log10()              l()
tan()                         t()
sin()                         s()
cos()                         c()
Exponentation                 ^
Division                      /
Multiplication/Product        *
Subtraction                   -
Addition/Sum                  +
e  ( = 2.71...)               e or E
pi ( = 3.1415...)             p or P

Example :  e^(sin(ln(2x)))    should be written as :    e^(s(n(2*x)))

NOTE : 2x is incorrect, 2*x is correct.
```

Requirements

The Requirements

Hardware Requirements:

1. Printer, to print the required documents of the project.
2. Compact Disk Drive.
3. Processor: Pentium III or higher.
4. RAM : 64MB
5. Hard Disk : 2GB

Software Requirements

1. Operating System: Windows XP 32-bit (preferable) or below. DOSBox required for OS higher than XP (and for 64-bit).
2. Turbo C++ for Compilation.
3. Turbo C++ BGI driver files (`'BGI'` folder to be present in the program directory).
4. Graphic Monitor Supporting VGA, EGA or CGA (or equivalent).
5. Ms Word for presentation of output.

Header Files

Header Files included and their purpose

The following header files were included in the project:

- `iostream.h` – For console I/O.
- `stdio.h` – For standard I/O like `gets()`.
- `string.h` – For `strcpy()`and `strlen()`.
- `conio.h` – For `getch()` (`clrscr()`not to be used for clearing graphics screen.)
- `fstream.h` – For file handling.
- `math.h` – For all the functions and constants to plot custom functions as described in the Help and Documentation.
- `ctype.h` – For `isdigit()` for carrying out postfix conversion and evaluation.
- `graphics.h` – For the graphics screen and functions to plot the inputted functions. Functions and constants required include `DETECT`, `initgraph()`, `putpixel()`, `setbkcolor()`, `cleardevice()`, `getmaxx()`, `getmaxy()`, `line()` and others.
- `time.h` – For the basic time routines (used in submitting feedback).
- `img.h` – The custom header file coded for the `getimg()` function which prints the Program logo on the screen by reading the bitmap `logo.bmp`, coversion into Turbo colors and printing the logo on the screen. The suitable functions were tediously searched and researched on the internet and modified for compilation of this header file.

The Code

SOFTWARE.CPP

```
#include<iostream.h>
#include<string.h>
#include<conio.h>
#include<fstream.h>
#include<math.h>
#include<ctype.h>
#include<graphics.h>
#include<time.h>
#include"img.h"
//img.h required for getimg()
float prec = 20;       //Precision
float scale=50;            //Not greater than 100
void pause(){          // system("pause") for graphics
  cout<<"Press any key to continue...";
  getch();
}
void clr(){           //Clearing screen in Graphics
  setbkcolor(0);
  cleardevice();
  gotoxy(1,1);
}
int height(){                 //Text mode screen size
  int i=1;
  int x = wherex(), y = wherey(); //Save current cursor position
  while(wherey()!=(i-1)){
      gotoxy(1,++i);
  }
  gotoxy(x,y);        //Return back to original position
  return (i-1);
}
int width(){                  //Text mode screen size
  int i=1;
  int x = wherex(), y = wherey(); //Save current cursor position
  gotoxy(1,1);
  while(wherex()!=(i-1)){
      gotoxy(++i,1);
  }
  gotoxy(x,y);        //Return back to original position
  return (i-1);
}

int maxx;
int maxy;
      //For graph
void Line(float slope, float intercept){
  float x,y;
```

The Code

```
  for (x=-maxx/2;x<maxx;x++){
      y=slope*x+intercept;
      putpixel(maxx/2+x,maxy/2-y,9);
      }
  }
void Line(int x1,int y1, int x2, int y2)
  {
  line(maxx/2+x1,maxy/2-y1,maxx/2+x2,maxy/2-y2);
  }
void Axes(){
  line(maxx/2,0,maxx/2,maxy);
  line(0,maxy/2,maxx,maxy/2);
  }
void mark(){
  int x,y,color;
  color=WHITE;
  setcolor(8);
  for (x=scale;x/scale<=maxx/2;x+=scale){
      Line(x,maxy/2,x,-maxy/2);
      Line(-x,maxy/2,-x,-maxy/2);
  }
  for (y=scale;y/scale<=maxy/2;y+=scale){
      Line(maxx/2,y,-maxx/2,y);
      Line(maxx/2,-y,-maxx/2,-y);
  }
  setcolor(color);
  for (x=scale;x/scale<=maxx/2;x+=scale){
      Line(x,2,x,-2);
      Line(-x,2,-x,-2);
  }
  for (y=scale;y/scale<=maxy/2;y+=scale){
      Line(1,y,-1,y);
      Line(1,-y,-1,-y);
  }
}
int size = 0; //Size of the stack;
int top = -1;
fstream fl("FL.TXT",ios::out|ios::in|ios::trunc);
char *infix, *postfix, *Stack;
double *eval;
      //Calculation
/*--------Function prototypes-------*/
void getinput();
int precedence(char ch);     //Function to get precedence of operator
char Pop();              //Function to pop an element
char Topelement();           //Function to return top element without popping
void Push(char ch);          //Function to push an element in the stack
int braces(char *s);         //Function to match number of braces
void intopost();
```

The Code

```
int posteval(double x);
void sendfeedback();
int showfeedback();


/*----End of Function Prototypes----*/

/*--------Function Definitions-------*/
void getinput(){
  char ch = 0;
  size=0;
  while(ch!='\r'){
      fl.seekp(size, ios::beg);
      ch = getch();
      if(ch==' ')
          continue;
      else if(ch=='\b'){
          if(size)
              size--;
          cout<<"\b \b";
      }
      else{
          fl<<ch;
          cout<<ch;
          size++;
      }
  }
  cout<<'\n';
  infix = new char[2 * size];
  int i = 0;
  fl.seekg(0, ios::beg);
  while(i < size-1){
      fl>>ch;
      infix[i++] = ch;
  }
  infix[i] = '\0';
  postfix = new char [2 * size];
  Stack = new char [2 * size];
}
int precedence(char ch){    //Function to get precedence of operator and/or the math functions
  switch(ch){
      case 'm': return 12;        //fabs()
      case 'f': return 11;        //floor()
      case 'n': return 10;        //ln()
      case 'l': return 9;     //log()
      case 't': return 8;     //tan()
      case 's': return 7;     //sin()
```

The Code

```
        case 'c': return 6;             //cos()
        case '^': return 5;
        case '/': return 4;
        case '*': return 3;
        case '-': return 2;
        case '+': return 1;
        default : return 0;
    }
}
char Pop(){             //Function to pop an element
    char ret;
    if(top!=-1){
        ret = Stack[top--];
        return ret;
    }
    else
        return '#';
}
char Topelement(){          //Function to return top element without popping
    char ch;
    if(top!=-1)
        ch = Stack[top];
    else
        ch = '#';
    return ch;
}
void Push(char ch){         //Function to push an element in the stack
    if (top!=size-1)
        Stack[++top] = ch;
}
int braces(char *s){        //Function to match number of braces
    int leftbr,rightbr;
    leftbr = rightbr = 0;
    for (int i = 0; s[i]; i++){
        if (s[i]=='(')
            leftbr++;
        else if(s[i]==')')
            rightbr++;
    }
    if(leftbr==rightbr)
        return 0;
    else if(leftbr<rightbr)
        return 1;
    else return -1;
}
void intopost(){
    char ele,elem,st[2];
    int prep,pre,popped,j=0,chk=1;
    strcpy(postfix, " ");
```

The Code

```
  while(chk!=0){
      getinput();
      chk = braces(infix);
      if(chk!=0){
          cout<<"Unbalanced no. of braces.\nExtra ";
          cout<<(chk==1?"right braces":"left braces")<<'\n';
      }
  }
  for (int i=0;infix[i]!='\0';i++){
      //cout<<i;   <--For debugging
      if  (precedence(infix[i])==0 && infix[i]!='(' && infix[i]!=')')
          postfix[j++]=infix[i];
      else if(infix[i]=='('){
          elem = infix[i];
          Push(elem);
      }
      else if(infix[i]==')'){
          while((popped = Pop())!='(')
              postfix[j++] = popped;
      }
      else {
          elem = infix[i];
          pre = precedence(elem); //Precedence of operator from infix
          ele = Topelement();
          prep = precedence(ele); //Precedence of operator from top of stack
          if(pre > prep)
              Push(elem);
          else{
              while(prep >= pre){
                  if(ele=='#')
                      break;
                  popped = Pop();
                  ele = Topelement();
                  postfix[j++] = popped;
                  postfix[j++] = ' ';
                  prep = precedence(ele);
              }
              Push(elem);
          }
      }
      if  (!(precedence(infix[i+1])==0 && infix[i+1]!='(' && infix[i+1]!=')'))
          postfix[j++] = ' ';
  }
  while((popped = Pop()) != '#'){
      postfix[j++] = ' ';
      postfix[j++] = popped;
  }
  postfix[j] = '\0';
}
```

The Code

```
int posteval(double x){
  eval = new double(strlen(postfix));
  double num=0,result=0;
  int v=0,dec=0,top=-1,i=0;
  char ch;
  int success=1;
  while(postfix[i]!='\0' && i<strlen(postfix)){
      v=dec=0;
      num=0;
      ch = postfix[i];
      if (ch==' '){
          i++;
          continue;
      }
      if (isdigit(ch)){
          while(ch!=' ' && postfix[i]!=0){
              if(v)
                  dec++;
              if(ch=='.')
                  v=1;
              else
                  num = (ch-'0') + num*10;
              ch = postfix[++i];
          }
          num/=pow(10,dec);
          eval[++top] = num;
          i++;
      }
      else if (ch=='x'||ch=='X'){
          eval[++top] = x;
          i++;
      }
      else if (ch=='e'||ch=='E'){
          eval[++top] = M_E;
          i++;
      }
      else if (ch=='p'||ch=='P'){
          eval[++top] = M_PI;
          i++;
      }
      else {
          result = 0;
          switch(ch){
              case 'm':
                  result = fabs(eval[top]);
                  top++;
                  break;
              case 'f':
                  result = floor(eval[top]);
```

The Code

```
                    top++;
                    break;
                case 'n':
                    if(eval[top]>0){                    //Error checking
                        result = log(eval[top]);
                        }
                    else
                        success = 0;
                    top++;
                    break;
                case 'l':
                    if(eval[top]>0){
                        result = log10(eval[top]);
                        }
                    else
                        success = 0;
                    top++;
                    break;
                case 't':
                    for (int i=0;(2*i+1)*M_PI_2<=scale*maxx;i++){
                        if
(fabs(eval[top])+(0.1/scale)>=(2*i+1)*M_PI_2&&fabs(eval[top])-
(0.1/scale)<=(2*i+1)*M_PI_2)
                            success = 0;
                        else if (scale*fabs(tan(eval[top]))>=10e3)
                            success = 0;
                        else
                            result = tan(eval[top]);
                        }
                    top++;
                    break;
                case 's':
                    result = sin(eval[top]);
                    top++;
                    break;
                case 'c':
                    result = cos(eval[top]);
                    top++;
                    break;

                case '^':
                    if (eval[top-1]==0){
                        if (eval[top]<=0)
                            success = 0;
                        }
                    else if (eval[top-1]<0 && ceil(eval[top])!=eval[top])
                        success = 0;

                    else
```

The Code

```
                    result = pow(eval[top-1],eval[top]);
                break;
            case '/':
                if (eval[top] == 0)
                    success = 0;
                else
                    result = eval[top-1]/eval[top];
                if (scale*fabs(result)>=10e3)
                    success = 0;
                break;
            case '*':
                result = eval[top-1]*eval[top];
                break;
            case '+':
                result = eval[top-1]+eval[top];
                break;
            case '-':
                result = eval[top-1]-eval[top];
                break;
        }
        if (scale*fabs(result)>=10e3)
            success = 0;
        eval[--top] = result;
        ++i;
    }
  }
  return success;
}
class Feed{
  char name[30];
  char email[30];
  char feed[500]; //User's message
  time_t t;
  public:
  Feed(){
  t = time(0);
  }
  void getd(){
      cout<<"\n Enter Name  : ";
      gets(name);
      cout<<" Enter Email : ";
      gets(email);
      cout<<" Enter Message (max 500 chars) :\n";
      gets(feed);
  }
  void putd(){
      int wt=width();
      cout<<"\n Name  : "<<name;
      gotoxy(wt-30,wherey());
```

The Code

```
386         cout<<"At : ";
387         cout<<ctime(&t);
388         cout<<" Email : "<<email;
389         cout<<"\n Message : "<<feed;
390     }
391 };
392
393 int showfeedback(){
394     clr();
395     Feed post;
396     int i=1;
397     int n=4;          //Max entries per page
398     fstream rev("FEED.DAT",ios::in|ios::binary);
399     if (!rev)
400         return 0;
401     clr();
402     while(!rev.eof()){
403         rev.read((char*)&post,sizeof(post));
404         if(rev.eof()){
405             if(i%n!=1){
406             cout<<"\n\n";
407             pause();
408             }
409             break;
410         }
411         post.putd();
412         cout<<"\n\n";
413         if(i%n==0){
414             pause();
415             clr();
416         }
417         i++;
418         }
419     rev.close();
420     return 1;
421 }
422 void sendfeedback(){
423
424     Feed post;
425     post.getd();
426     post.putd();
427     getch();
428     fstream rev("FEED.DAT",ios::out|ios::app|ios::binary);
429     rev.write((char*)&post,sizeof(post));
430     rev.close();
431 }
432
433 /*----End of Function Definitions----*/
434
```

The Code

```
void main(){
clrscr();
int choice=1;
int ht = height();
while(choice){
getimg();
cout<<"\t\t\t\t    Menu\n";
cout<<" 1. Plot a function.\n";
cout<<" 2. Help and Documentation.\n";
cout<<" 3. View Submitted feedback.\n";
cout<<" 4. Send feedback/Request a feature.\n";
cout<<" 0. Exit\n\n";
cout<<" Enter your choice : ";
cin>>choice;
switch(choice){
case 1:
  cout<<" Enter the function below :\n> ";
  intopost();
  cout<<" Enter the scale for the graph (between 5 and 100) : ";
  cin>>scale;
  if (scale>100 || scale <5)
      scale = 50;//Default
  cout<<" Enter the precision (1 to 50) :";
  cout<<"\n [Warning : tan() will be slow with high precision, unless the CPU is fast]\n> ";
  cin>>prec;
  if(prec<1||prec>50)
      prec = 20; //Default
  pause();
  int gd=DETECT,gm;
  initgraph(&gd, &gm, "c:/tc/bgi ");
  maxx = getmaxx();
  maxy = getmaxy();
  double x, y;
  mark();
  Axes();
  for (x=(-maxx/2)/scale;x<maxx;x+=(1.0/((scale)*prec))){
      if(!posteval(x))
          continue;
      y=scale*(eval[0]);
      delete eval;
      if (x*scale>=maxx/2)
          break;
      x=x*scale;
      if (y>(maxy)*scale||y<(-maxy)*scale)
          continue;
      else
          putpixel(maxx/2+x,maxy/2-y,LIGHTGREEN);
      x=x/scale;
```

The Code

```
483   //  delay(0);
484   }
485   getch();
486   break; //End of choice 1
487 case 2:
488   clr();
489   fstream doc("doc.txt", ios::in);
490   if(!doc){
491       cout<<" Documentation unavailable!";
492       getch();
493   }
494   else{
495       char ch;
496       while(!fl.eof()){
497           ch = doc.get();
498           if(doc.eof()){
499               cout<<"\n";
500               pause();
501               break;
502               }
503           if(ch=='\n' && wherey()==(ht-2)){
504               cout<<ch;
505               pause();
506               clr();
507           }
508           else{
509               if(wherey()==(ht-2)){
510                   cout<<'\n';
511                   pause();
512                   clr();
513               }
514               cout<<ch;
515           }
516       }
517   }
518   doc.close();
519   break;
520 case 3:
521   clr();
522   if(!showfeedback()){
523       cout<<" No feedback history available!";
524       getch();
525   }
526   break;
527 case 4:
528   sendfeedback();
529   getch();
530   break;
531 }
```

The Code

```
closegraph();
restorecrtmode();
}
closegraph();
restorecrtmode();
fl.close();
}
```

The Code

IMG.H

```
#ifndef IMG_H_INCLUDED
#define    IMG_H_INCLUDED
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<fstream.h>
#include<stdio.h>
int loadbitmap(char*);
int getcol(int);
int getimg (void)
{
int gd=DETECT,gm;
initgraph(&gd, &gm, "bgi");
cout<<"\n\n\n\n\n\n\n\n\n\n";
int h = loadbitmap("logo.bmp");
return h;
}
int loadbitmap(char *filename)
{
FILE *ptr=NULL; //file handle to open bitmap file
int width,height; //width and height of the bitmap
unsigned long temp=0,i=0,j=0; //some variables i need
unsigned long ww;
ptr=fopen(filename,"rb"); //open the bitmap file
if(!ptr) return 0; //if its not there return
width=0;height=0; //initialise wisth and height to zero
fseek(ptr,18,SEEK_SET); //got to offset 18 in file
for(i=0x1;i<=0x10000;i*=0x100) //read the width
{
temp=fgetc(ptr);
width+=(temp*i);
}
fseek(ptr,22,SEEK_SET); //go to offset 22 in file
for(i=0x1;i<=0x10000;i*=0x100) //read the height
{
temp=fgetc(ptr);
height+=(temp*i);
}
ww=width; //ww is the number of reads to make for each horizontal line
if(ww%2) //ww has to be half of width - since each pixel is only 4 bits of information
ww++;ww/=2; //just getting the correct value
if(ww%4) //now - ww is stored as sets of 4 pixels each so this is the adjustment made
ww=(ww/4)*4+4; //if width is less than ww*2 we ignore what we read

fseek(ptr,119,SEEK_SET); //Ok! offset 119 - lets read the pixels -
```

The Code

```
//remember the bitmap is stored up - side - down
int ch,ch1,ch2;
for(i=0;i<height;i++)
for(j=0;j<ww;j++)
{
ch=fgetc(ptr); //each character read is 2 pixels - yes 4 bits per pixel - so 16 colors
ch1=ch;ch2=ch; //find those colors using bitwise ops
ch1=ch1&(0xf0);ch1=ch1>>4; // ~~
ch2=ch2&(0x0f); // ~~
if(j*2<width) //ok so put the first pixel read on screen
putpixel((j)*2,(height-1-i),getcol(ch1));
if(j*2+1<width) //put the second pixel read on screen
putpixel((j)*2+1,(height-1-i),getcol(ch2));
}
fclose(ptr); //close the file handle
return height;
}
int getcol(int col)
{
switch(col)
{
case 0: return 0; //BLACK;
case 1: return 4; //RED;
case 2: return 2; //GREEN
case 3: return 6; //BROWN
case 4: return 1; //BLUE;
case 5: return 5; //MAGENTA;
case 6: return 3; //CYAN;
case 7: return 7; //LIGHTGRAY;
case 8: return 8; //DARKGRAY;
case 9: return 12; //LIGHTRED;
case 10:return 10; //LIGHTGREEN
case 11:return 14; //YELLOW;
case 12:return 9; //LIGHTBLUE;
case 13:return 13; //LIGHTMAGENTA
case 14:return 11; //LIGHTCYAN;
case 15:return 15; //WHITE;
}
}
#endif
```

The Code

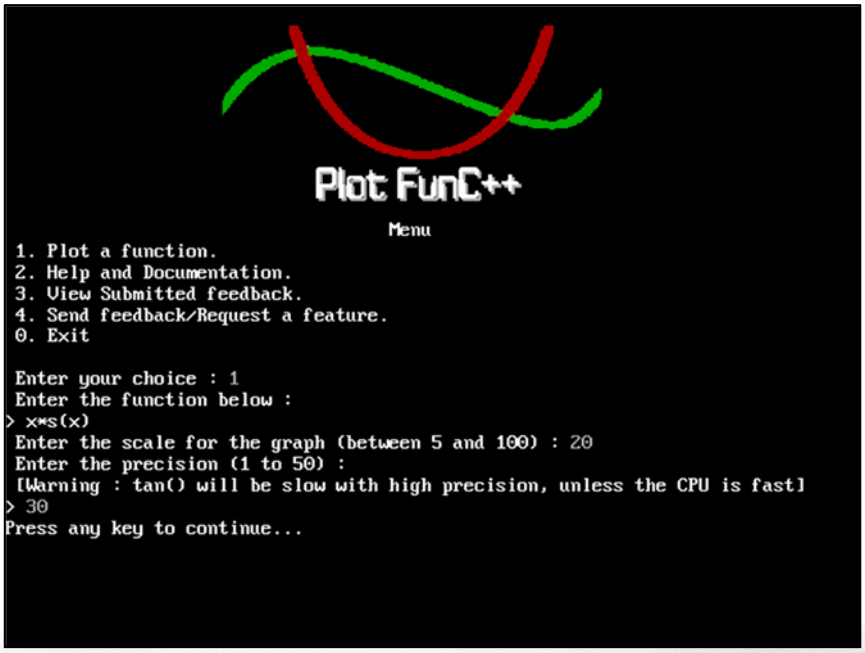
DOC.TXT

```
Available functionalities in Plot FunC++ (with decreasing precedence):

Function/Operator/Constant    Syntax
--------------------------    ---------------

fabs() or the mod function    m()
floor()                       f()
ln() or loge()                n()
log() or log10()              l()
tan()                         t()
sin()                         s()
cos()                         c()
Exponentation                 ^
Division                      /
Multiplication/Product        *
Subtraction                   -
Addition/Sum                  +
e  ( = 2.71...)               e or E
pi ( = 3.1415...)             p or P

Example :  e^(sin(ln(2x)))    should be written as :    e^(s(n(2*x)))

NOTE : 2x is incorrect, 2*x is correct.
```

Output

The Output

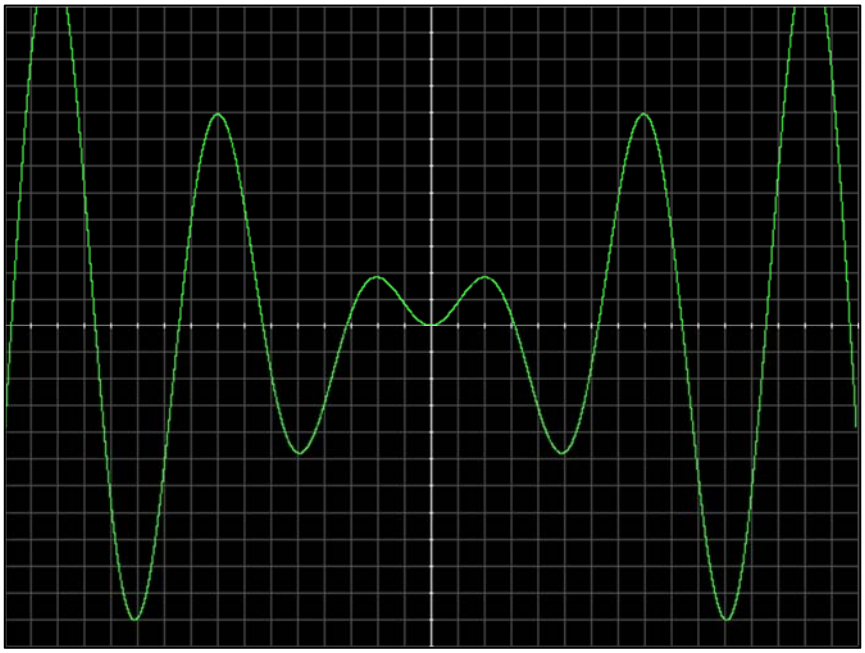
Here are the screenshots of the output while running the program.



Output

The Output

Output for `x*s(x)` that is, $\times\, sin(x)$:



Viewing Help and Documentation:

```
Available functionalities in Plot FunC++ (with decreasing precedence):

Function/Operator/Constant      Syntax
--------------------------      ---------------

fabs() or the mod function      m()
floor()                         f()
ln() or loge()                  n()
log() or log10()                l()
tan()                           t()
sin()                           s()
cos()                           c()
Exponentation                   ^
Division                        /
Multiplication/Product          *
Subtraction                     -
Addition/Sum                    +
e  ( = 2.71...)                 e or E
pi ( = 3.1415...)               p or P

Example :  e^(sin(ln(2x)))    should be written as :    e^(s(n(2*x)))


Press any key to continue...
```

Output

The Output

Submitting Feedback:

```
Plot FunC++
Menu
1. Plot a function.
2. Help and Documentation.
3. View Submitted feedback.
4. Send feedback/Request a feature.
0. Exit

Enter your choice : 4

Enter Name  : Admin
Enter Email : admin@plotfuncpp.tk
Enter Message (max 500 chars) :
Hello, This app is really nice.

Name  : Admin                                  At : Sun Dec 14 00:05:48 2014
Email : admin@plotfuncpp.tk
Message : Hello, This app is really nice.
```

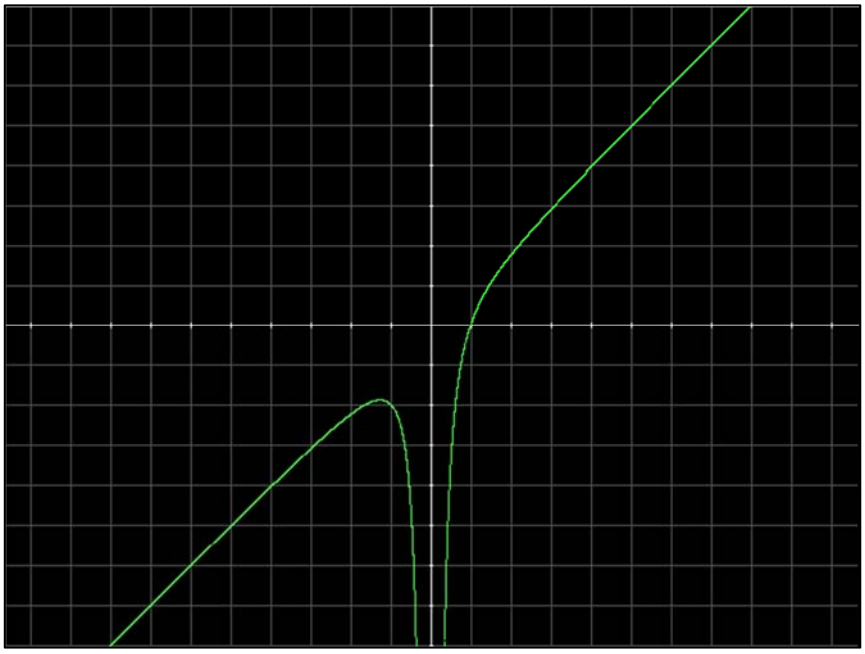
Viewing the Submitted Feedback.

```
Name  : Admin                                  At : Sun Dec 14 00:05:48 2014
Email : admin@plotfuncpp.tk
Message : Hello, This app is really nice.


Press any key to continue...
```

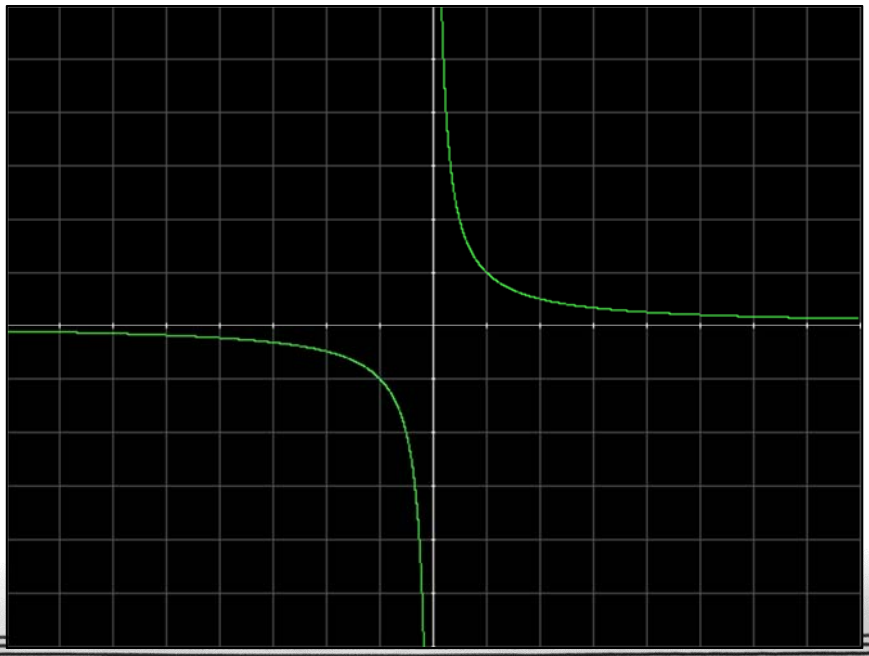
Output

The Output

Graph for `x-1/x^2` that is, $-\frac{1}{x^2}$:



Graph for `1/x` that is, $\frac{1}{x}$:



Limitations

Limitations

The limitations of this program currently include:

- Slow processing time for functions having `tan(x)`.
- The feedback stays local; it is not actually submitted to the developer via the internet. There was a piece of code written especially for sending the feedback online via a PHP GET request (by opening the webpage), but since DOS's native `COMMAND.COM` does not support Windows' `CMD` commands nor does it allow an native `internet` support. The unanswered question for such support can be found at:

  http://stackoverflow.com/questions/26867707/

  The online submission feature had to be dropped. The PHP webpage is still live and can be visited at:

  www.thepirategamer.tk/cpp/cpp.php?name=YourName&email=YourEmail&feed=YourMessage

- User interaction with the plotted graph is limited. The user cannot scroll through the graph or see the coordinates of the points other than the integral points on the axes.

Bibliography

Bibliography

- Computer Science with C++ : Class XII by Sumita Arora
- http://programmersheaven.com/discussion/361125/loading-a-bitmap-image-in-c - for inserting the BMP logo in the program.

Thank You !