Nepal College of Information Technology
Faculty of Science and Technology

Bachelor of Software Engineering

Lab Report

File Handling in C

Candidate name

Nischal Lal Shrestha

Supervised by

Asst. P.

January 21, 2019

1 Introduction

1.1 What is file?

File is a collection of bytes that is stored on secondary storage devices like disk. There are two kinds of files in a system.

Types of Files

- Text files (ASCII)
- Binary files

1.2 Basic file operations in C programming

In C, we can perform four major operations on the file, either text or binary:

- Creating a new file
- Opening an existing file
- Closing a file
- Reading from and writing information to a file

2 Working with files

When working with files, we need to declare a pointer of type file. This declaration is needed for communication between the file and program.

$$FILE * fptr;$$

2.1 Opening a file - for creation and edit

Opening a file is performed using the library function in the "stdio.h" header file: fopen(). The syntax for opening a file in standard I/O is:

$$ptr = fopen(\text{“}fileopen\text{”}, \text{“}mode\text{”})$$

For Example

$$fopen(\text{“}E : cprogramnewprogram.txt\text{”}, \text{”}w\text{”});$$

2.2 Opening Modes in Standard I/O

File Mode	Meaning of Mode	During Inexisten
r	Open for reading	If the file does not exist, fop
w	Open for writing	If the file exists, its contents are overwritten. If
a	Open for append	If the file does not exists,
r+	Open for both reading and writing	If the file does not exist, fop
w+	Open for both reading and writing	If the file exists, its contents are overwritten. If

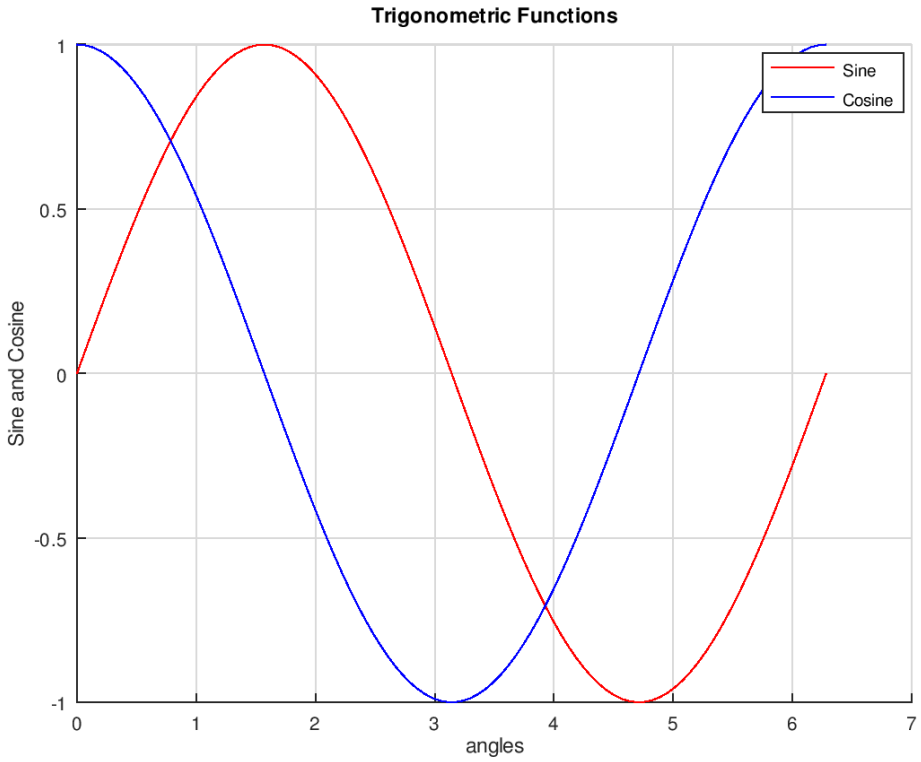
3 Plotting Mathematical Models

We can use tools like **Octave** or **Matlab** to plot various mathematical models. By plotting a mathematical models we can understand the system thoroughly.

3.1 Plotting Trigonometric Functions

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int main(){
    char item_name[33];
    char item_database[33];
    char enter_again;
    float item_price;
    int item_found = 0;
    int c;
    do{
        printf("Enter the Item Name: \n");
        scanf("%s", &item_name);
        printf("Enter Price of %s: \n", item_name);
        scanf("%f", &item_price);

    FILE *fp;
    fp = fopen("check.txt", "a+");
    if (fp){
            do{
               fscanf(fp, "%[^\n]", &item_database);
               if(strcmp(item_name, item_database) == 0){
                    printf("%s already in database.\n",
                       item_name);
                    item_found = 1;
                    break;
               }

            }while(c = fgetc(fp) != EOF);

            if (item_found == 0){
                    printf("%s doesn't exist.\n", item_name);
                    printf("Adding %s in database.\n",
                       item_name);
                    fprintf(fp, "%s\n", item_name);
            }

    } //end of if(file)
fclose(fp);
fflush(stdin);
    printf("Enter another item?\n");
```

```
40      scanf("%c", &enter_again);
41      } while(enter_again == 'Y' || enter_again == 'y');
42
43      printf("Exiting the system.\n");
44      getch();
45      return 0;
46
47  }
```



3.2 Convergence

If we suppose ϵ to be the error quantity of approximated root determined using Newton-Raphson, then by the use of Taylor Series we obtain the following relationship

$$\epsilon_{n+1} = k\epsilon_n^2$$

Above equation shows that the error is proportional to the square of the previous error. This means that the number of correct decimal places approximately doubles with each iteration. Such behavior is referred to as *quadratic convergence*.

4 Algorithm

An algorithm for the Newton-Raphson Method is obtained simply by using one initial guess as approximated root and by using above generalized formula to calculate the root.

The program must also be able to calculate first derivative of the equation provided, which can be done on MATLAB via `diff` command. An algorithm to solve the non-linear equations using Newton-Raphson Method is defined below:

1. Define $f(x)$, $f'(x)$ and Error (E).
2. Input the value of initial root value x_0.
3. Caculate
$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$
4. If $(f(x_1) == 0)$
   Display "The root lies at $x =$", x_0
   Goto Step 6
   End
5. If $(abs(x_1 - x_0) > E)$
   Set,
   $x_0 = x_1$
   $f(x_0) = f(x_1)$
   $f'(x_0) = f'(x_1)$
   Goto Step 3
   Else
   Display "The root lies at $x =$", x_1
   End
6. Stop

5 Flowchart

The flow of the False Position Method program is shown in the figure below

6 Program Code

The algorithm is implemented on MATLAB and the code is below

```
close all;
clear variables;
clc;

func  = input('Enter your desired function:            ');
funcd = input('Enter the derivative of that function   ');
x0    = input('Enter the initial root                  ');

f = inline(func);
fd = inline(funcd);
disp(f);
Error = 0.000005;

fx0 = f(x0);
fdx0 = fd(x0);
x1 = (x0 - (fx0 / fdx0));
fx1 = f(x1);
fdx1 = fd(x1);

if(fx1 == 0)
    result = strcat('The root lies at x = ', num2str(x1));
end

disp('-----------------------------------------------------');
disp('      x0       f(x0)     fd(x0)      x1      f(x1)    ');
disp('-----------------------------------------------------');
result = [x0, fx0, fdx0, x1, fx1];
disp(result);

while(abs(x1 - x0) > Error)
    x0 = x1;
    fx0 = fx1;
    fdx0 = fdx1;
    x1 = (x0 - (fx0/fdx0));
    fx1 = f(x1);
    result = [x0, fx0, fdx0, x1, fx1];
    disp(result);
end

disp('-----------------------------------------------------');
result = strcat('The root lies at x = ', num2str(x1));
disp(result);
```

7 Result and Discussion

The program is run on MATLAB and upon execution the console window displays the result and or solution of the function. One such output for the equation $f(x) = x * sin(x) + 4 * cos(x) - 2$ is shown below.

Listing 1: Output of the Implementation on MATLABS

```
Enter your desired function:            'x*sin(x)+4*cos(x)-2'
```

```
Enter the derivative of that function    'x*cos(x)+sin(x)-4*sin(x)'
Enter the initial root                   1.6

     Inline function:
     f(x) = x*sin(x)+4*cos(x)-2

------------------------------------------------------------
      x0        f(x0)      fd(x0)       x1       f(x1)
------------------------------------------------------------
    1.6000    -0.5175    -3.0454     1.4301    -0.0230

    1.4301    -0.0230    -2.7698     1.4218    -0.0001

    1.4218    -0.0001    -2.7698     1.4217    -0.0000

    1.4217    -0.0000    -2.7698     1.4217    -0.0000

------------------------------------------------------------
The root lies at x =1.4217
```

The program implemented above is not able to calculate first derivative of the function on own, but it rather asks the user to input the first derivative of the function he/she has entered. This can be solved by using a user defined function which is able to differentiate a function, which is beyond the scope of this report.

8 Conclusion

Newton-Raphson Method is perhaps one of the most widely used root-locating formulas. The main advantage of Newton-Raphson method over other methods is that it converges quadratically as we approach the root. Unlike the bisection and false position methods, the Newton-Raphson technique requires only one inital value x_0.

However, the convergence of the Newton-Raphson Method is not guranteed. The convergence depends on the nature of the function and on the accuracy of the initial guess. Bad initial guesses can lead to a chaotic series of iterates which may or may not converge at all. So, the only remedy is to have an initial guess that is 'sufficiently' close to the root.

References

[1] E Balagurusamy. *Numerical Methods*. McGraw Hill Education, 978-0-07-463311-3, 1999.

[2] Chapra S.C. and Canale R.P.. *Numerical Methods for Engineers, Sixth Edition*. McGraw Hill Education, 978-0-07-340106-5, 2010.