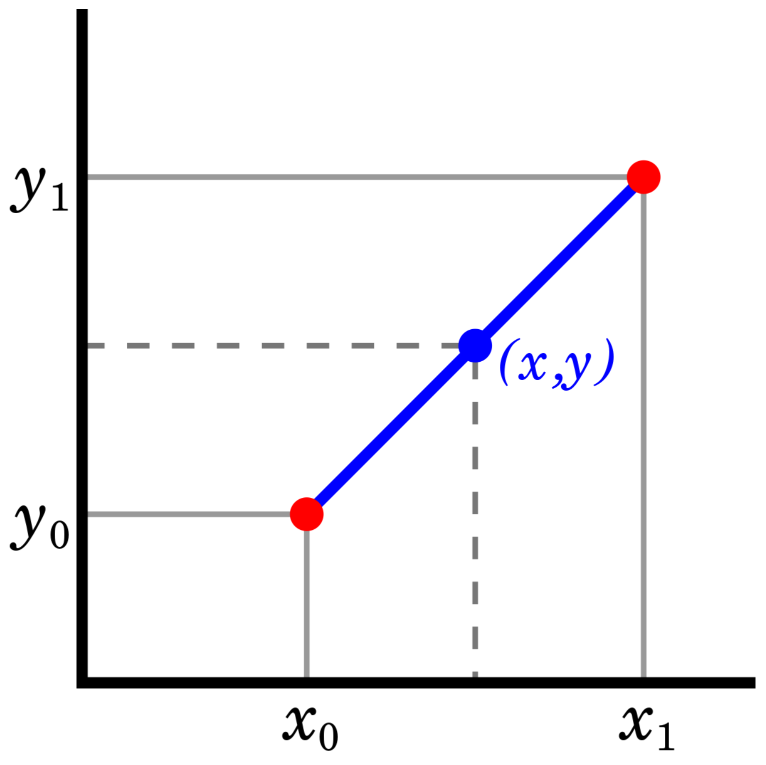
1-D Array Manipulations and Functions

GE 1502
Cornerstone of Engineering 2
CRN: 35280
Monday, Wednesday, Thursday from 10:30 AM - 11:35 AM
308 Hurtig Hall
Thursday, January 25, 2018
Jordan Lian

Table of Contents



Interpolation Nation

Pseudocode/Flowchart

```
//library files, iostream, fstream, cstdlib, iomanip
//constant MAX = 15
//sub-function order check
//declare variables, FPA[], COL[MAX], slope variables, for loop variables, order variables
//get program to read txt file using ifstream
//read entire array, use while loop, i=0, and keep reading, use i++ to keep going
//call order check to ensure that the function is in ascending order
//entering new FPA value, use while loop (big while loop where a == 0), a is initialized to a=0
//if FPA is greater than the final value (17), then break from loop
//else keep going (no need to write)
//print output, use while loop, j=0, if the entered FPA value is greater than FPA[j] (array from above, starting at j=0), then j++, while(FPA value > FPA[j])
//while loop will keep going until FPA value is less than one of the FPA numbers in the data table, then code can progress
//set to three decimal places
//calculate slope, then calculate corresponding COL value
//print value, then coordinate
//print two data points used
//print slope
//ask if user wants to test more values, enter 0, if not, enter 1
//if a == 1, then break, if a = 0, then program goes back to the top of the big while loop

//sub function
//for loop, i=1, i < number of points, i++
//if array[i] < array[i+1], then return 0
Then return 1 (outside for loop)
```

Test by hand-calculation

(COL, FPA)
FPA values tested: 3, 11, 15.5, 19

FPA = 3, values used are 2, 4
(0.240, 2) (0.480, 4) > slope = (4-2) / (0.480-0.240) = 8.3333 = 25/3
25/3 = (3 - 2) / (COL - 0.240)
25/3 = 1 / (COL - 0.240)

COL - 0.240 = 3/25
COL = 0.360 > (0.360, 3)

FPA = 11
(0.792, 10) (0.924, 12) > slope = (12-10) / (0.924-0.792) = 15.15151515 = 500/33
500/33 = (11-10) / (COL - 0.792)
500/33 = 1 / (COL - 0.792)
COL - 0.792 = 33/500
COL = 0.858 > (0.858, 11)

FPA = 15.5
(1.070, 15) (1.110, 16) > slope = (16-15) / (1.110 - 1.070) = 25
25 = (15.5 - 15) / (COL - 1.070)
25 = 0.5 / (COL - 1.070)
COL - 1.070 = 0.02
COL = 1.090 > (1.09, 15.5)

FPA = 19
invalid

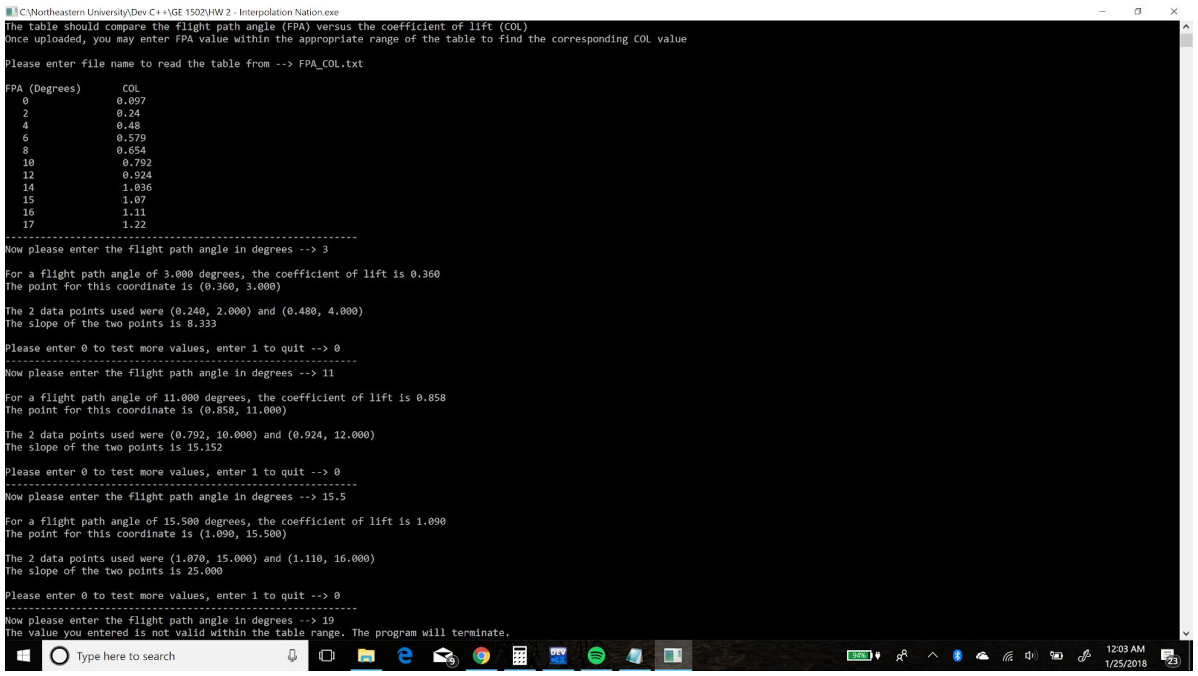
Program source code cpp. File

```
/* Jordan Lian, 1/25/2018, GE 1502, HW 2, Interpolation Nation

This program will solve a linear interpolation problem using arrays
The function given is a chart of flight path angle versus the coefficient of lift

*/

#include <iostream>
#include <fstream> //read from txt file
#include <cstdlib>
#include <iomanip>
using namespace std;

//constant
#define MAX 15

int order_check(double array_input[], int num_points); //return 1 or 0 for true or false

int main(void)
{
        //local variables
```

```
	double FPA[MAX], COL[MAX]; //arrays from txt file
	double slope, FPA_inp, COL_out; //slope calculation variables
	int i=0, j=0, a=0, order; //for loop variables
	int num_points;

	cout << "The table should compare the flight path angle (FPA) versus the coefficient of
lift (COL)" << endl;
	cout << "Once uploaded, you may enter FPA value within the appropriate range of the
table to find the corresponding COL value"
	<< endl << endl;

	//reading the txt file
	string filename = "FPA_COL.txt";
	cout << "Please enter file name to read the table from --> ";
	cin >> filename; //save user input as string

	//input file variables
	ifstream infile;
	infile.open(filename.c_str());

	//test to see if file exists
	if(!infile)
	{
		cout << "\nCan't find the file - terminating program\n";
		return 0;		//end program
	}

	//intro
	cout << "\nFPA (Degrees)" << "	COL" << endl;

	//reading file
	while (infile >> FPA[i] >> COL[i])
	{
		cout << "   " << FPA[i] <<"			"<< COL[i] <<endl;
		i++;
	}

	//checking order of points -- make sure it's in ascending order
	num_points = i;
	order = order_check(FPA,num_points);
	if(order == 0)
	{
		cout<< "Data is not in ascending order. Exiting...";
		exit(0);
	}

	//entering FPA and more
	while(a == 0)
	{
		cout << "----------------------------------------------------------\nNow please enter the
flight path angle in degrees --> ";
		cin >> FPA_inp; //store value(s) in array

		if( (FPA_inp > 17) || (FPA_inp < 0) )
		{
			cout << "The value you entered is not valid within the table range. The
program will terminate." << endl;
			break;
		}

		//print output
		while(FPA_inp > FPA[j])
		{
			j++;
		}

		//calculations
		slope = (FPA[j] - FPA[j-1]) / (COL[j] - COL[j-1]); //slope formula
		COL_out = ( (FPA_inp - FPA[j-1]) / slope ) + COL[j-1]; //recalculated COL value

		cout << setprecision(3) << fixed;
		cout << "\nFor a flight path angle of " << FPA_inp << " degrees, the coefficient of
lift is " << COL_out << endl;
		cout << "The point for this coordinate is (" << COL_out << ", " << FPA_inp << ")"
<< endl;

		cout << "\nThe 2 data points used were (" << COL[j-1] << ", " << FPA[j-1] << ")";
		cout << " and (" << COL[j] << ", " << FPA[j] << ")" << endl;;

		cout << "The slope of the two points is " << slope << endl;

		//stay in program or quit
		cout << "\nPlease enter 0 to test more values, enter 1 to quit --> ";
		cin >> a;

		if(a == 1)
```

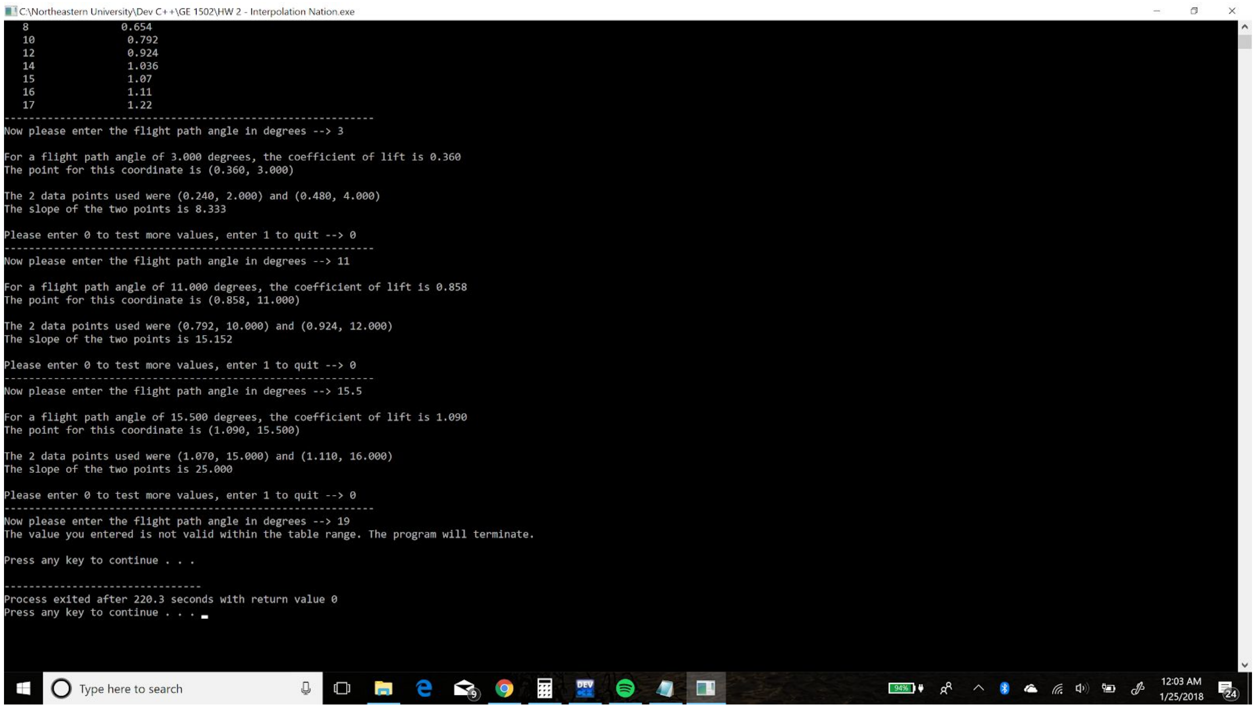
```
                        break; //break from while loop

        } //end of big while loop

        cout << endl;
        system("pause");
        return 0;
} //end of main

//sub-function

int order_check(double array_input[], int num_points)
{
        for(int i=1; i < num_points; i++)
        {
          if(array_input[i] < array_input[i-1])
                        return 0;
        }
        return 1;
} //end of sub-function
```

Screenshot of Outputs



C:\Northeastern University\Dev C++\GE 1502\HW 2 - Interpolation Nation.exe
Type here to search
12:03 AM
1/25/2018

```
The table should compare the flight path angle (FPA) versus the coefficient of lift (COL)
Once uploaded, you may enter FPA value within the appropriate range of the table to find the corresponding COL value

Please enter file name to read the table from --> FPA_COL.txt

FPA (Degrees)       COL
   0               0.097
   2               0.24
   4               0.48
   6               0.579
   8               0.654
   10               0.792
   12               0.924
   14               1.036
   15               1.07
   16               1.11
   17               1.22
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 3

For a flight path angle of 3.000 degrees, the coefficient of lift is 0.360
The point for this coordinate is (0.360, 3.000)

The 2 data points used were (0.240, 2.000) and (0.480, 4.000)
The slope of the two points is 8.333

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 11

For a flight path angle of 11.000 degrees, the coefficient of lift is 0.858
The point for this coordinate is (0.858, 11.000)

The 2 data points used were (0.792, 10.000) and (0.924, 12.000)
The slope of the two points is 15.152

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 15.5

For a flight path angle of 15.500 degrees, the coefficient of lift is 1.090
The point for this coordinate is (1.090, 15.500)

The 2 data points used were (1.070, 15.000) and (1.110, 16.000)
The slope of the two points is 25.000

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 19
The value you entered is not valid within the table range. The program will terminate.
```



```
C:\Northeastern University\Dev C++\GE 1502\HW 2 - Interpolation Nation.exe
   8              0.654
   10              0.792
   12              0.924
   14              1.036
   15              1.07
   16              1.11
   17              1.22
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 3

For a flight path angle of 3.000 degrees, the coefficient of lift is 0.360
The point for this coordinate is (0.360, 3.000)

The 2 data points used were (0.240, 2.000) and (0.480, 4.000)
The slope of the two points is 8.333

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 11

For a flight path angle of 11.000 degrees, the coefficient of lift is 0.858
The point for this coordinate is (0.858, 11.000)

The 2 data points used were (0.792, 10.000) and (0.924, 12.000)
The slope of the two points is 15.152

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 15.5

For a flight path angle of 15.500 degrees, the coefficient of lift is 1.090
The point for this coordinate is (1.090, 15.500)

The 2 data points used were (1.070, 15.000) and (1.110, 16.000)
The slope of the two points is 25.000

Please enter 0 to test more values, enter 1 to quit --> 0
------------------------------------------------------------
Now please enter the flight path angle in degrees --> 19
The value you entered is not valid within the table range. The program will terminate.

Press any key to continue . . .

--------------------------------
Process exited after 220.3 seconds with return value 0
Press any key to continue . . .
```

Let’s hang ten

Pseudocode/Flowchart

```
//library files, iostream, fstream, iomanip
//constant max 1000 (a lot of data)
//variables int year, month, day, hour, minute (all arrays)
//double wave height, wave length, wave steepness (wave height/ wave length) (all arrays)
//int i=0, counter for while loop
//int b=0, total number of waves exceeding the average
//double average wave steepness, sum = 0, perc;
//sum is sum of wave steepness values (to calculate average)
//perc is the percentage of wave steepness values above the average value

//read the txt file using while loop, store the 7 columns as year, month, day, hour, minute, wave height, wave length
//print arrays
//calculate wave steepness, then print
//i++, so array keeps printing until done

//sum up the wave steepness values
//sum = sum + wave steepness or sum += wave steepness
//waves = i, total number of values/rows
```

```
//average wave steepness = sum / waves

//counting wave steepness values, use for loop
//(i=0, i < waves, i++) use this to keep counting until the end
//if wave steepness[i] (use i to test each value in the for loop) > avg wave steepness
//then b++ (b=0 is initialized)
//percentage is 1.0 * b / waves * 100, 1.0 is there to make the value a double

//print the total waves, sum, average, percentage of values exceeding the average, and the
number of values exceeding the average
```

Test by hand-calculation

Average wave steepness = total sum of values / number of values

Number of values = 86
Total sum = 19.21
Avg wave steepness = 19.21 / 86 = 0.22

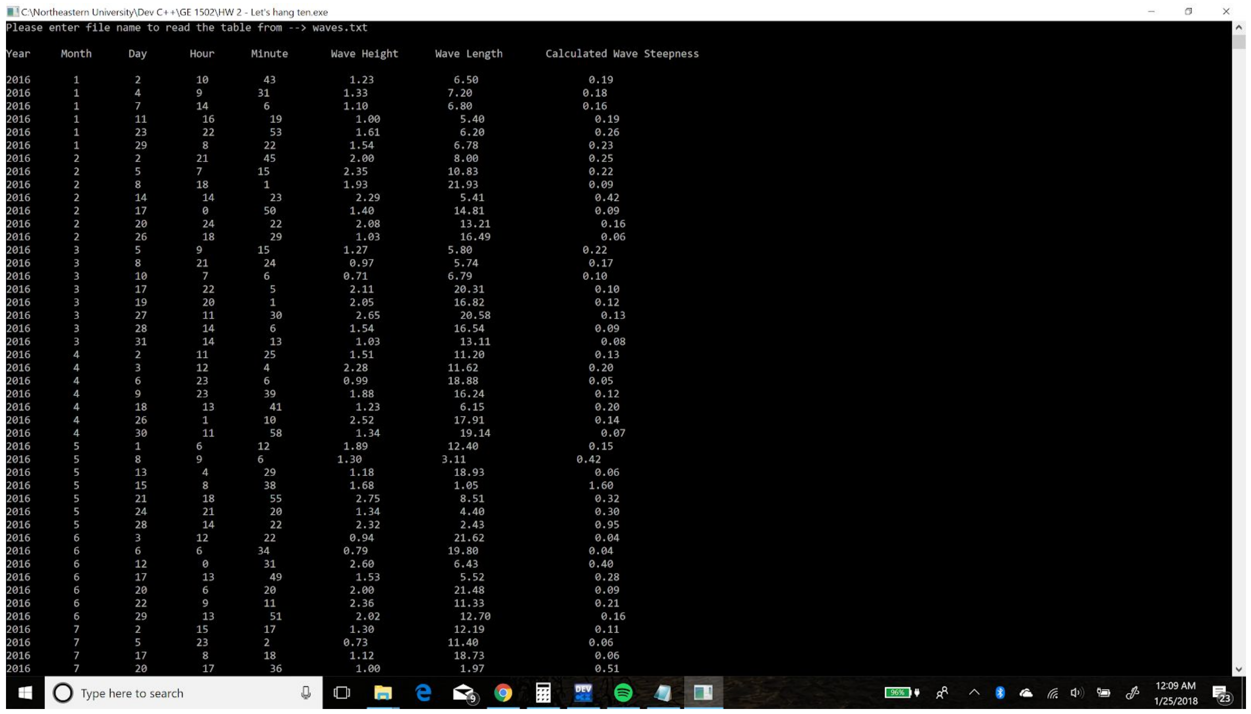
23 of those values exceed the average
23 / 86 = 26.74%

26.74% of the wave steepness values are greater than the average wave steepness of 0.22

Program source code cpp. File

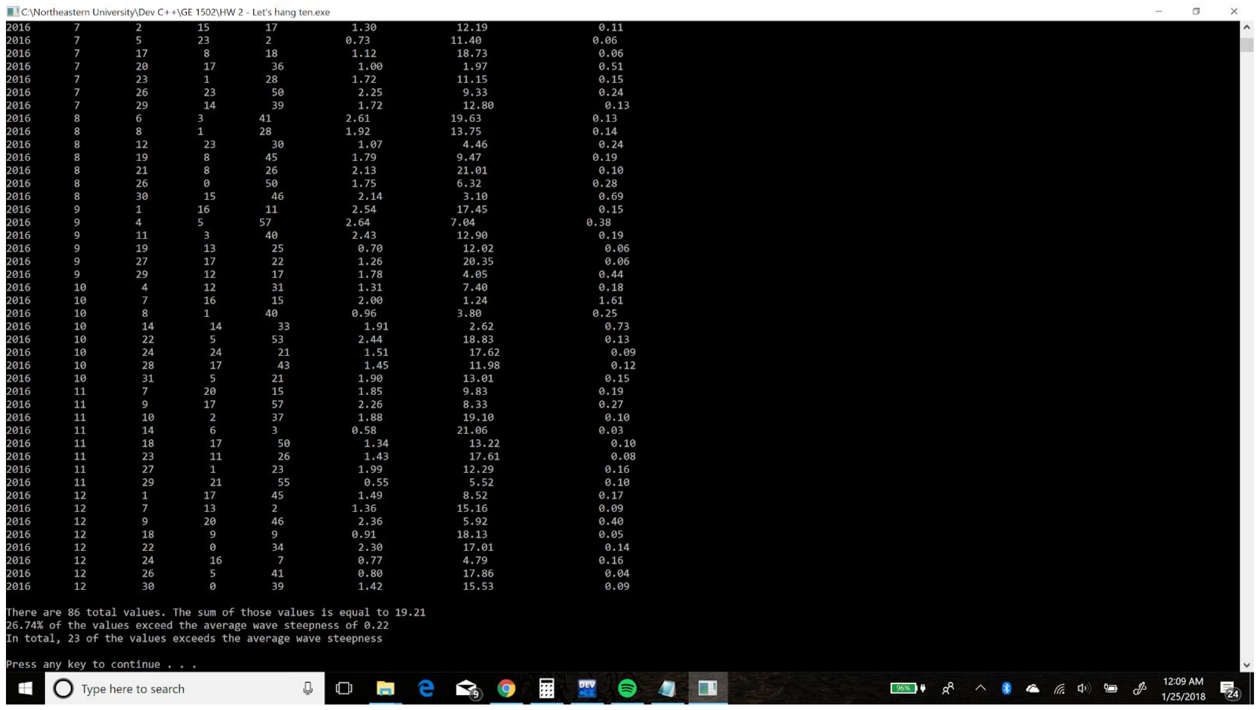
```
/* Jordan Lian, 1/25/2018, GE 1502, HW 2, Let's hang ten

This program reads in a txt file full of data and gets the wave steepness for each set of data.
It will calculate the average wave steepness and then find which sets of data exceeds the
average.

*/

#include <iostream>
#include <fstream> //read from txt file
using namespace std;
#include <iomanip> //decimal place, set precision function

#define MAX 1000 //test up to 100 chars

int main(void)
{
	//local variables
	int year[MAX], month[MAX], day[MAX], hour[MAX], minute[MAX], waves; //int arrays
	double wave_height[MAX], wave_length[MAX], wave_steepness[MAX]; //double arrays
	int i=0, b=0; //counter for while loop, number of waves exceeding the average
	double average_wave_steepness, sum=0, perc; //average, total sum of wave steepness
values, percentage

	//reading the txt file
	string filename = "waves.txt";
	cout << "Please enter file name to read the table from --> ";
	cin >> filename; //save user input as string

	//input file variables
	ifstream infile;
	infile.open(filename.c_str());

	//test to see if file exists
	if(!infile)
	{
		cout << "\nCan't find the file - terminating program\n";
		return 0; //end program
	}

	//reading file
	cout << endl;
	cout << "Year     Month     Day     Hour     Minute     Wave Height     Wave Length
Calculated Wave Steepness" << endl << endl;
		while ( infile >> year[i] >> month[i] >> day[i] >> hour[i] >> minute[i] >>
wave_height[i] >> wave_length[i] )
		{
			//formula for wave steepness
			wave_steepness[i] = (1.0 * wave_height[i] / wave_length[i]);

			cout << setprecision(2) << fixed; //set to 2 fixed decimal places
			cout << year[i] << "       " << month[i] << "         " << day[i] << "         " <<
hour[i] << "         " <<
			minute[i] << "            " << wave_height[i] << "             " << wave_length[i]
<< "                 " << wave_steepness[i]
			<< "       " << endl;

			//sum of the wave steepness values for average
			sum += wave_steepness[i];
			i++; //go through arrays
		} //end while loop

		//average wave steepness calculations
		waves = i; //total waves
		average_wave_steepness = sum/waves;

		//counting wave steepness values greater than the average wave steepness
		for(i=0; i < waves; i++)
		{
			if (wave_steepness[i] > average_wave_steepness)
				b++; //number of waves greater than the average
		} //end for loop

	//percentage of wave steepness values greater than the average
	perc = (1.0 * b / waves) * 100; //turn int values into a double

	//print output
	cout << "\nThere are " << waves << " total values. The sum of those values is equal to "
<< sum << endl;
	cout << perc << "% of the values exceed the average wave steepness of " <<
average_wave_steepness << endl;
	cout << "In total, " << b << " of the values exceeds the average wave steepness" << endl
<< endl; //number of values exceeding the average

	system("pause");
	return 0;
} //end of main
```

Screenshot of Outputs



C:\Northeastern University\Dev C++\GE 1502\HW 2 - Let's hang ten.exe

```
Please enter file name to read the table from --> waves.txt

Year      Month      Day       Hour      Minute      Wave Height     Wave Length       Calculated Wave Steepness

2016        1         2         10         43           1.23            6.50                0.19
2016        1         4         9         31           1.33            7.20                0.18
2016        1         7         14         6           1.10            6.80                0.16
2016        1         11         16         19           1.00            5.40                0.19
2016        1         23         22         53           1.61            6.20                0.26
2016        1         29         8         22           1.54            6.78                0.23
2016        2         2         21         45           2.00            8.00                0.25
2016        2         5         7         15           2.35            10.83                0.22
2016        2         8         18         1           1.93            21.93                0.09
2016        2         14         14         23           2.29            5.41                0.42
2016        2         17         0         50           1.40            14.81                0.09
2016        2         20         24         22           2.08            13.21                0.16
2016        2         26         18         29           1.03            16.49                0.06
2016        3         5         9         15           1.27            5.80                0.22
2016        3         8         21         24           0.97            5.74                0.17
2016        3         10         7         6           0.71            6.79                0.10
2016        3         17         22         5           2.11            20.31                0.10
2016        3         19         20         1           2.05            16.82                0.12
2016        3         27         11         30           2.65            20.58                0.13
2016        3         28         14         6           1.54            16.54                0.09
2016        3         31         14         13           1.03            13.11                0.08
2016        4         2         11         25           1.51            11.20                0.13
2016        4         3         12         4           2.28            11.62                0.20
2016        4         6         23         6           0.99            18.88                0.05
2016        4         9         23         39           1.88            16.24                0.12
2016        4         18         13         41           1.23            6.15                0.20
2016        4         26         1         10           2.52            17.91                0.14
2016        4         30         11         58           1.34            19.14                0.07
2016        5         1         6         12           1.89            12.40                0.15
2016        5         8         9         6           1.30            3.11                0.42
2016        5         13         4         29           1.18            18.93                0.06
2016        5         15         8         38           1.68            1.05                1.60
2016        5         21         18         55           2.75            8.51                0.32
2016        5         24         21         20           1.34            4.40                0.30
2016        5         28         14         22           2.32            2.43                0.95
2016        6         3         12         22           0.94            21.62                0.04
2016        6         6         6         34           0.79            19.80                0.04
2016        6         12         0         31           2.60            6.43                0.40
2016        6         17         13         49           1.53            5.52                0.28
2016        6         20         6         20           2.00            21.48                0.09
2016        6         22         9         11           2.36            11.33                0.21
2016        6         29         13         51           2.02            12.70                0.16
2016        7         2         15         17           1.30            12.19                0.11
2016        7         5         23         2           0.73            11.40                0.06
2016        7         17         8         18           1.12            18.73                0.06
2016        7         20         17         36           1.00            1.97                0.51
```



C:\Northeastern University\Dev C++\GE 1502\HW 2 - Let's hang ten.exe

```
2016        7         2         15         17           1.30            12.19                0.11
2016        7         5         23         2           0.73            11.40                0.06
2016        7         17         8         18           1.12            18.73                0.06
2016        7         20         17         36           1.00            1.97                0.51
2016        7         23         1         28           1.72            11.15                0.15
2016        7         26         23         50           2.25            9.33                0.24
2016        7         29         14         39           1.72            12.80                0.13
2016        8         6         3         41           2.61            19.63                0.13
2016        8         8         1         28           1.92            13.75                0.14
2016        8         12         23         30           1.07            4.46                0.24
2016        8         19         8         45           1.79            9.47                0.19
2016        8         21         8         26           2.13            21.01                0.10
2016        8         26         0         50           1.75            6.32                0.28
2016        8         30         15         46           2.14            3.10                0.69
2016        9         1         16         11           2.54            17.45                0.15
2016        9         4         5         57           2.64            7.04                0.38
2016        9         11         3         40           2.43            12.90                0.19
2016        9         19         13         25           0.70            12.02                0.06
2016        9         27         17         22           1.26            20.35                0.06
2016        9         29         12         17           1.78            4.05                0.44
2016        10         4         12         31           1.31            7.40                0.18
2016        10         7         16         15           2.00            1.24                1.61
2016        10         8         1         40           0.96            3.80                0.25
2016        10         14         14         33           1.91            2.62                0.73
2016        10         22         5         53           2.44            18.83                0.13
2016        10         24         24         21           1.51            17.62                0.09
2016        10         28         17         43           1.45            11.98                0.12
2016        10         31         5         21           1.90            13.01                0.15
2016        11         7         20         15           1.85            9.83                0.19
2016        11         9         17         57           2.26            8.33                0.27
2016        11         10         2         37           1.88            19.10                0.10
2016        11         14         6         3           0.58            21.06                0.03
2016        11         18         17         50           1.34            13.22                0.10
2016        11         23         11         26           1.43            17.61                0.08
2016        11         27         1         23           1.99            12.29                0.16
2016        11         29         21         55           0.55            5.52                0.10
2016        12         1         17         45           1.49            8.52                0.17
2016        12         7         13         2           1.36            15.16                0.09
2016        12         9         20         46           2.36            5.92                0.40
2016        12         18         9         9           0.91            18.13                0.05
2016        12         22         0         34           2.30            17.01                0.14
2016        12         24         16         7           0.77            4.79                0.16
2016        12         26         5         41           0.80            17.86                0.04
2016        12         30         0         39           1.42            15.53                0.09

There are 86 total values. The sum of those values is equal to 19.21
26.74% of the values exceed the average wave steepness of 0.22
In total, 23 of the values exceeds the average wave steepness

Press any key to continue . . .
```