UNIVERSITY OF WEST LONDON

Engineering Software

Assignment 2

Manuel Alberto Suena Galindez

03/10/2017

Table of Contents

Introduction

This assignment consists of using C programming to solve a physics problem. The aim of the assignment is to show the coding abilities of the student by using multiple data types, different type of statements and the use of a user defined function.

This assignment is divided in to 6 sections, the first of which is the introduction. The other section are My Project, The Program, Main Function, Proof of Execution and Proof of Compilation. The section My Project consists of a description of the physics problem, the equation in use and a general explanation on how is going to be solved using C programming. In The Program section a discussion is present about how the program works and what it does to solve the problem. Meanwhile, in the section Main Function, there is a more detailed description of every line of code within the program, a brief explanation on how they work and an illustration of the program. In the last two section there are illustrations of the compilation and the execution of the program with a brief description of the figures.

My Project

In this project, I'm going to calculate the wavelength, frequency and propagation speed of a wave by using C programming. Through coding it is expected to obtain the calculation result by first asking the user to choose which variable he wants to calculate. If the user decides that he wants to calculate the wavelength, he will have to enter the other two variables so as to calculate the result by using the wavelength equation.

$$w = \frac{c}{f}$$

If instead the user chooses to calculate one of the other two variables of the wave, the equation is rearranged and the inputs change to account for the variable to solve:

$$f = \frac{c}{w} \qquad c = w \times f$$

In the case that the speed of the wave is constant the user will also have the option to get a table of all the values of the wavelength for a range of frequency that he chooses.

The Program

In this program the user will first observe a line of text explaining what the program does and then it will be ask to enter one of four numbers, from 1 to 4, to choose which operation to execute. The first operation will calculate wavelength. The user will be ask to enter the other two variables needed to execute this operation and afterwards he will be shown the result. By choosing the second option, the computer will ask the user to enter the values of speed and wavelength to calculate frequency. The third option instead will ask the user to enter wavelength and frequency to calculate speed of the wave. The last option, by using a loop, allows the user calculate all the values of the wavelength in a range of frequency that the user decides. Later, a table of values of *f* and *w* is shown to the user.

Main Function

The main function of this program starts by declaring the structure and setting the pointer *var* equal to the *make_var* function. Then, the header function executes a series of *printf* that explain the user how the program works and then the *set_number* function ask him to enter a number from 1 to 4, and a *scanf* saves the number that the user has entered. The saved number is used by a switch statement to determine which case to execute depending on the value of the number entered.

In case 1, the *calc_wavelength* function is executed. In this function the variables for frequency and speed of the wave are set and used to calculate the wavelength, then the getter function retrieves the result and returns its value so a *printf* can show the result to the user. Case 1 ends with a break that prevents the computer from executing all the cases.

In case 2 and 3, the same procedure is follow but the function executed are *calc_frequency* and *calc_speed* respectively.

In case 4, the table function is executes a series of setter function that ask and save the value of the variables *c*, *upper, lower* and *step*. Then, two *printf* are used to print the header of the table and after that *f* is declared to have the same value of *lower*. In this function a *while* loop with the values entered by the user is executed, printing a table of the values of *w* and *f*. The case ends with a *break*.

The last line of code before closing the *switch* is a *default*, which is required by the *switch* syntax, and it is used in case the user enters a value of number that wasn't defined in the *switch* case. In this function if the *default* is executed, a *printf* tells the user that the value entered is too high. The *default* ends with a *break*.

After the *switch* closes an *if* statement is executed that check if the *write_data* function failed. In case it failed it will print a fail message in *stderr* file otherwise it executes the function.

The main function ends with the free function and the returns 0.

```
int main()
{
	Variables *var;

	var = make_var(27122017);
```

```
	header(var);   // this executes the header

	int res = set_number(var);  //Here the set_number function is executed


	switch (res) //The computer uses the number entered by the user to determine which
operation to execute

	{
	case 1://In case 1 the user is asked for c and f

		calc_wavelength(var,res); // Here the wavelength is calculated
		get_result(var, res);      // This function gets the result from the
calc_wavelength function
		printf("The value of the wavelength is %.2f m\n", var->w);

		break;

	case 2: // In this case the user is asked for w and c to calculate f

		calc_frequency(var,res); //Here the frequency is calculated
		get_result(var, res);    // This function gets the result from the
calc_ferquency function
		printf("The value of frequency is %.2f Hz\n", var->f);

		break;

	case 3: //In case 3 the user is asked for w and f

		calc_speed(var,res); //Here the speed of the wave is calculated
		get_result(var, res);
		printf("The value of speed of wave is %.2f m/s\n", var->c);

		break;
	case 4: //In case 4 the user is asked for c ,lower ,upper and step

		table(var,res); //Here the function table is executed

		break;

	default: //This is executed in case a number higher then 4 is entered
		printf("This program only accepts options from 1 to 4. Inserting any other
number will result in ERROR\n");

		break;
	}

	if (write_data("data.txt", var) == 1)
	{
		fprintf(stderr, "Failed to open file!\n");
	}
	free_memory(var); // Here the memory is freed by deleting data stored

	return 0;
}
```

C Structure

This structure allows to reference all the variables used in this program by a single pointer

```
typedef struct {
	float w;
	float f;
	float c;
	int number;
	float lower;
	float upper;
	float step;
	int date;
}Variables;
```

Make function

In this function memory is allocated for the structure by a if statement that first checks if there is enough memory available and in the case there is not enough space the massage failed to allocated memory is printed by fprintf in the standard error file.

```
Variables *make_var(int date)
{
	Variables*var;
	if ((var = (Variables *)malloc(sizeof(Variables))) == NULL)
	{
		fprintf(stderr, "failed to allocated memory \n");
		exit(0);
	}
	var->w = 0;
	var->f = 0;
	var->c = 0;
	var->number = 0;
	var->lower = 0;
	var->upper = 0;
	var->step = 0;
	var->date =NULL;

	return var;
};
```

Setter functions

In this series of functions is where all the variables needed for the calculation are assigned a value set by the user through a series of printf and scanf.

```
void set_speed(Variables*var)
{
	float c;

    printf("Enter the value of speed of wave in m/s\n");
    scanf("%f", &c);
	printf("You entered the value %.2f\n", c);
	var->c = c;
```

```
}

void set_wavelenght(Variables*var)
{
	float w;

	printf("Enter the value of wavelength in m\n");
	scanf("%f", &w);
	printf("You entered the value %.2f\n", w);
	var->w = w;
}

void set_frequency(Variables*var)
{
	float f;

	printf("Enter the value of frequency in Hz\n");
	scanf("%f", &f);
	printf("You entered the value %.2f\n", f);
	var->f = f;

}

int set_number(Variables*var)

{
	int number;

	printf("Enter number\n");
	scanf_s("%d", &number);
	printf("You entered the number %d\n", number);
	var->number = number;

	return number;
}

void set_tabel_values(Variables*var)
{
	float lower, upper, step;

	printf("Enter the lowest value of frequency in Hz\n");
	scanf("%f", &lower);
	printf("You entered the value %.2f\n", lower);
	var->lower = lower;

	printf("Entre the highest value of frequency in Hz\n");
	scanf("%f", &upper);
	printf("You entered the value %.2f\n", upper);
	var->upper = upper;

	printf("Enter the increment of frequency\n");
	scanf("%f", &step);
	printf("You entered the value %.2f\n", step);
	var->step = step;

}
```

Getter function and Freeing

The getter function use on this program uses a switch to return the value of the case that was executed, and it return only that value.

The free function deletes all the data stored in the memory use for this program.

```
float get_result(Variables *var,int res)
{
	switch (res)
	{

	case 1:

    return var->w;

    break;

	case 2:

	return var->c;

	break;

	case 3:

	return var->f;

	break;

	default: printf("Error");
		break;
	}
}

void free_memory(Variables *var)
{
	free(var);
}
```

File Output

In this function all the data of the value of all the variables is saved in the fp file.

```
int write_data(char file_name[], Variables *var)
{
	FILE *fp;
	fp = fopen(file_name, "w+");
	if (fp == NULL) {
		fprintf(stderr, "ERROR: Can't open input file\n");
		return 1;
	}
	fprintf(fp, "Data:\n");
	fprintf(fp, "Speed of Sound %0.2f\nFrequency %0.2f\nWavelength %0.2f\n",var->c,
var->f, var->w);
	fprintf(fp, "date is %d\n",var->date);
```

```
        fclose(fp);
        return 0;
}
```

Proof of execution

As we can see in all images, the program was executed successfully. In figure 1, the computer executed case 1 so it asked for *c* and *f* and then calculated the values entered by the user to get *w.* In this case *w* was *52,03*m. In figure 2, case 2 is executed and the values entered are used to calculate *f*. In this case *f* is *25,50* Hz. In figure 3, case 3 is executed, calculating *c* which in this case was *2011,29* m/s. In figure 4, case 4 is executed and prints a table of the values of *w* and *f*. Figure 5 and 6, shows what happens when the user enters a value not expected by this program so it prints "ERROR".

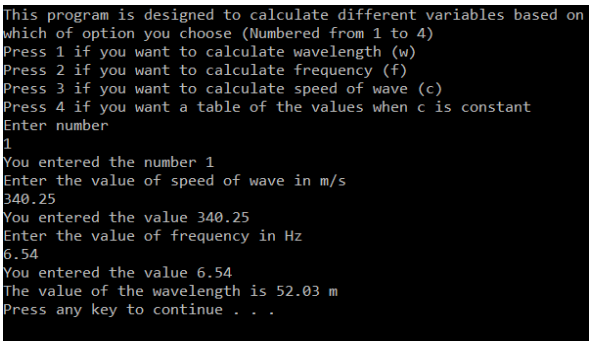

Figure 1: illustrates case 1.

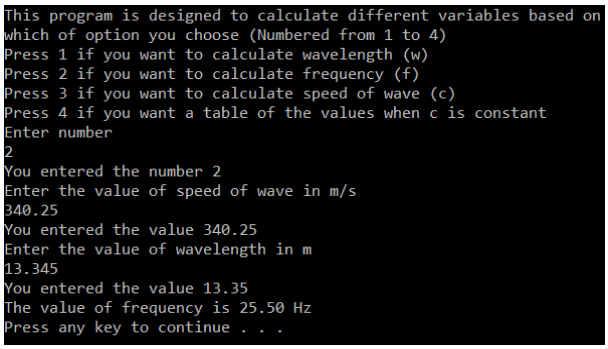

Figure 2: illustrates case 2.

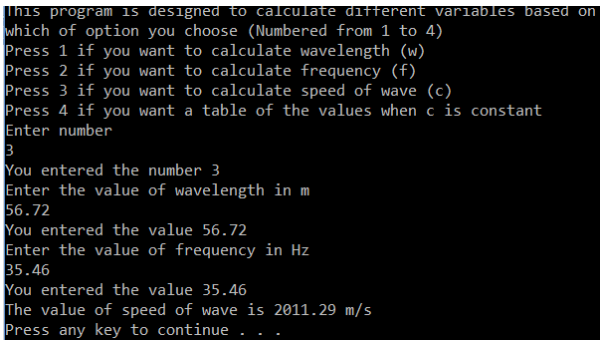

Figure 3: illustrates case 3.

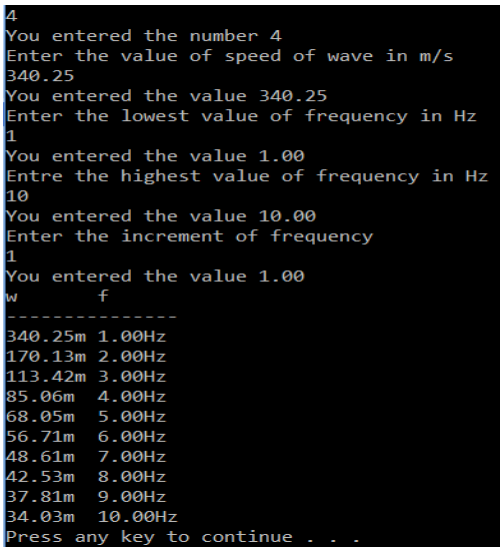

Figure 4: illustrates case 4.

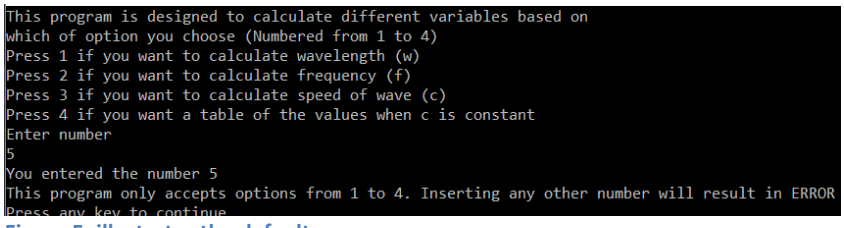

Figure 5: illustrates the default.

```
This program is designed to calculate different variables based on
which of option you choose (Numbered from 1 to 4)
Press 1 if you want to calculate wavelength (w)
Press 2 if you want to calculate frequency (f)
Press 3 if you want to calculate speed of wave (c)
Press 4 if you want a table of the values when c is constant
Enter number
4
You entered the number 4
Enter the value of speed of wave in m/s
340.25
You entered the value 340.25
Enter the lowest value of frequency in Hz
45
You entered the value 45.00
Entre the highest value of frequency in Hz
4
You entered the value 4.00
Enter the increment of frequency
67
You entered the value 67.00
Error the increment/lowest value can't exceed the highest value
Press any key to continue . . .
```

Figure 6

Proof of compilation

Figure 6, shows that the compilation was successful and that it executed one build.

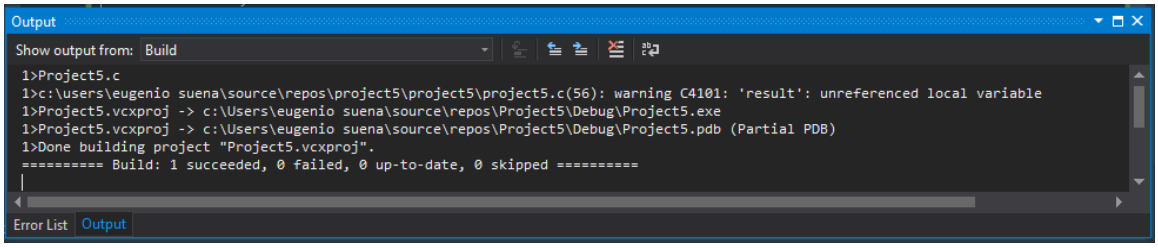

Figure 7: illustrates the compilation

Code listing

```c
#define _CRT_SECURE_NO_WARNINGS
#include "stdio.h"
#include "stdlib.h"


typedef struct {
	float w;
	float f;
	float c;
	int number;
	float lower;
	float upper;
	float step;
	int date;
}Variables;
```

```
Variables *make_var(int date)
{
	Variables*var;
	if ((var = (Variables *)malloc(sizeof(Variables))) == NULL)
	{
		fprintf(stderr, "failed to allocated memory \n");
		exit(0);
	}
	var->w = 0;
	var->f = 0;
	var->c = 0;
	var->number = 0;
	var->lower = 0;
	var->upper = 0;
	var->step = 0;
	var->date =NULL;

	return var;
};

void set_speed(Variables*var)
{
	float c;

    printf("Enter the value of speed of wave in m/s\n");
    scanf("%f", &c);
	printf("You entered the value %.2f\n", c);
	var->c = c;
}

void set_wavelenght(Variables*var)
{
	float w;

	printf("Enter the value of wavelength in m\n");
	scanf("%f", &w);
	printf("You entered the value %.2f\n", w);
	var->w = w;
}

void set_frequency(Variables*var)
{
	float f;

	printf("Enter the value of frequency in Hz\n");
	scanf("%f", &f);
	printf("You entered the value %.2f\n", f);
	var->f = f;

}

int set_number(Variables*var)

{
	int number;

	printf("Enter number\n");
```

```
	scanf_s("%d", &number);
	printf("You entered the number %d\n", number);
	var->number = number;

	return number;
}

void set_tabel_values(Variables*var)
{
	float lower, upper, step;

	printf("Enter the lowest value of frequency in Hz\n");
	scanf("%f", &lower);
	printf("You entered the value %.2f\n", lower);
	var->lower = lower;

	printf("Entre the highest value of frequency in Hz\n");
	scanf("%f", &upper);
	printf("You entered the value %.2f\n", upper);
	var->upper = upper;

	printf("Enter the increment of frequency\n");
	scanf("%f", &step);
	printf("You entered the value %.2f\n", step);
	var->step = step;

}
void header(Variables*var) // This function explains the program to the user
{
	printf("This program is designed to calculate different variables based on \n");
	printf("which of option you choose (Numbered from 1 to 4)\n");
	printf("Press 1 if you want to calculate wavelength (w)\n");
	printf("Press 2 if you want to calculate frequency (f)\n");
	printf("Press 3 if you want to calculate speed of wave (c)\n");
	printf("Press 4 if you want a table of the values when c is constant\n");
}

void calc_wavelength(Variables*var, int res) //Here the wavelength is calculated
{
	float w;

	set_speed(var);

	set_frequency(var);

	w = var->c / var->f;

	var->w = w;

}

void calc_frequency(Variables*var,int res) //Here the frequency is calculated
{
	float f;

	set_speed(var);

	set_wavelenght(var);
```

```
	f = var->c / var->w;

	var->f = f;

}

void calc_speed(Variables*var,int res) // Here speed of the wave is calculated
{
	float c;

	set_wavelenght(var);

	set_frequency(var);

	c = var->w * var->f;

	var->c = c;

}

void table(Variables*var,int res)
{
	float f,w;

	set_speed(var);

	set_tabel_values(var);

	if(var->upper>var->lower && var->upper>var->step) {

	printf("w\tf\n");

	printf("---------------\n");

	f = var->lower;

	while (f <= var->upper)
	{
		w = var->c / f;
		printf("%.2fm\t%.2fHz\n", w, f);
		f = f + var->step;
	}
	}
	else { printf("Error the increment/lowest value can't exceed the highest
value\n"); }
}

float get_result(Variables *var,int res)
{
	switch (res)
	{

	case 1:

    return var->w;

    break;
```

```
	case 2:

	return var->c;

	break;

	case 3:

	return var->f;

	break;

	default: printf("Error");
		break;
	}
}

void free_memory(Variables *var)
{
	free(var);
}

int write_data(char file_name[], Variables *var)
{
	FILE *fp;
	fp = fopen(file_name, "w+");
	if (fp == NULL) {
		fprintf(stderr, "ERROR: Can't open input file\n");
		return 1;
	}
	fprintf(fp, "Data:\n");
	fprintf(fp, "Speed of Sound %0.2f\nFrequency %0.2f\nWavelength %0.2f\n",var->c,
var->f, var->w);
	fprintf(fp, "date is %d\n",var->date);
	fclose(fp);
	return 0;
}


int main()
{
	Variables *var;

	var = make_var(27122017);

	header(var);   // this executes the header

	int res = set_number(var);  //Here the set_number function is executed


	switch (res) //The computer uses the number entered by the user to determine which
operation to execute

	{
	case 1://In case 1 the user is asked for c and f

		calc_wavelength(var,res); // Here the wavelength is calculated
```

```
                get_result(var, res);      // This function gets the result from the
calc_wavelength function
                printf("The value of the wavelength is %.2f m\n", var->w);

                break;

        case 2: // In this case the user is asked for w and c to calculet f

                calc_frequency(var,res); //Here the frequency is calculated
                get_result(var, res);    // This function gets the result from the
calc_ferquency function
                printf("The value of frequency is %.2f Hz\n", var->f);

                break;

        case 3: //In case 3 the user is asked for w and f

                calc_speed(var,res); //Here the speed of the wave is calculated
                get_result(var, res);
                printf("The value of speed of wave is %.2f m/s\n", var->c);

                break;
        case 4: //In case 4 the user is asked for c ,lower ,upper and step

                table(var,res); //Here the function table is executed

                break;

        default: //This is executed incase a number higher then 4 is entered
                printf("This program only accepts options from 1 to 4. Inserting any other
number will result in ERROR\n");

                break;
        }

        if (write_data("data.txt", var) == 1)
        {
        fprintf(stderr, "Failed to open file!\n");
        }
        free_memory(var); // Here the memory is freed by deleting data stored

        return 0;
}
```