CIS1500 Assignment 1: Student Finance Calculator

Total Marks: 10 marks
Weight: 10%
Due: Friday, 6th Oct 2023, at 11:59 pm

Welcome to your first CIS1500 assignment! Please ensure you read all the sections and **follow the instructions exactly** as we will grade your submission with an auto-grader. You must **test your assignment on the SoCS Linux server**, and it will be similarly graded on the SoCS Linux server for consistency.

Your program should only use concepts already taught in this course (ie. no arrays, custom functions, pointers, structs, etc.). **You will receive a 0 if your code includes: global variables** (variables declared outside of functions), **goto, malloc, calloc, regex, or if your code fails to compile without errors.**

This pdf goes in conjunction with videos available on courselink. The pdf document does not contain all required information for the assignment. There are multiple videos, each has different inputs, scenarios, with an invalid input, etc.

There is also a file containing "Skeleton Code"; you should use this as your starting point. It is a .c file with a few lines of code already written for you, and some comments to help guide you on how to get started. For example the program begins by printing the text: "Welcome to the CIS1500 Student Finance Planner. We have a few questions for you:" the skeleton code will already print that for you.

This document has 11 pages. Read them all and understand the requirements.

REMEMBER: You are to submit **one zip file containing one source code file** for your program.

- The name of the **zip file** follows the following format: **studentNumberA1.zip**
  - Example: John Snow's student number is 1770770 then the zip file name is 1770770A1.zip
- The single file within the zip you are to submit is a non-compiled code C file (ends with .c) containing your code
  - Do not include any other files or folders in your submission; they will be ignored.
  - Do not include this file inside another folder; it will be ignored by the auto-grader.
- The name of the **C file** follows the following format: **studentNumberA1.c**
  - Example: John Snow's student number is 1770770 then the C file name is 1770770A1.c
- **Incorrect submission will result in a grade of 0.**
- More details are outlined in Section 4: Program Submission and Administration Information.

Start Early. Get Help Early. Test Often.

1. Background

In this assignment, you will create a program to calculate student finances based on values received from the user. The program will ask users for information like: What does tuition cost per semester? How much money did you save over the summer? Do you have a part-time job? What does your rent cost? etc. After gathering information from the user, the program will calculate some values:

We will follow the 50/30/20 finance rule: 50% of your income is for essentials like food and rent, 30% for wants like games and going out, and 20% for savings and debt payment. We will assume students will not have debt in first-year, so the 20% can go to savings.

This program is intended to test your ability to:

- Read user inputs using scanf.
- Utilize conditionals (if and/or switch) to control program flow.
- Create and manipulate floating-point and integer variables.
- Perform mathematical operations.
- Format and display output using printf.

The course and assignment is not intended as actual finance advice. However you may find after entering your own details that you should make some financial adjustments.

2. Program Requirements

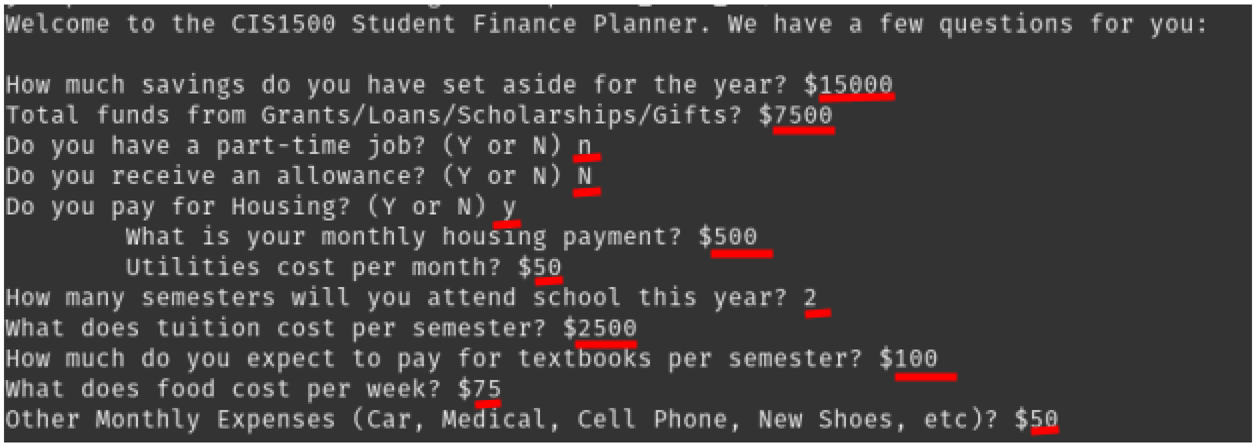
Your program will start with a welcome message, then prompt the user to answer some questions about savings and income, then ask questions about expenses; after getting the user information the program will display a table with calculated finance values, and finally display a few sentences summarizing the information.

2.1 Part 1: Getting User Input

Your program will ask 10 questions of the user. Some of these 10 questions will prompt additional questions. For example the fifth question is: “Do you pay for housing? (Y or N)”, if the user enters the letter N the program will move to the next question (about semesters); if the user enters the letter Y then the program will ask two follow up questions about the housing (monthly payments and utilities).

Below is an example of getting user input, the responses from the user are underlined in red to help differentiate. **Additional sub-questions are shown in the associated Sample Flow Videos on courselink.** In the example below a user entered N for “Do you have a part-time job? (Y or N)”, sample flow videos will show the sub-questions that are prompted when the user enters y.

Your program must prompt the users for the exact same questions as the samples do. If your questions are in a different order you will lose marks from sections 3.1 as the autograder expects the questions in the same order. If your program phrases the questions differently, uses different spelling or punctuation you will lose marks from section 3.2 as you did not correctly follow the formatting guidelines. Etc. Remember we are looking for precise, exact, solutions.



```
Welcome to the CIS1500 Student Finance Planner. We have a few questions for you:

How much savings do you have set aside for the year? $15000
Total funds from Grants/Loans/Scholarships/Gifts? $7500
Do you have a part-time job? (Y or N) n
Do you receive an allowance? (Y or N) N
Do you pay for Housing? (Y or N) y
        What is your monthly housing payment? $500
        Utilities cost per month? $50
How many semesters will you attend school this year? 2
What does tuition cost per semester? $2500
How much do you expect to pay for textbooks per semester? $100
What does food cost per week? $75
Other Monthly Expenses (Car, Medical, Cell Phone, New Shoes, etc)? $50
```

Your program will record all of the user input values into variables and store them to be used in Part 2. If at any point a user enters an invalid value (a negative value, or a character not listed as an option) you will display the message "Error: Invalid Input" and exit the program. Ex. `How often do you receive an allowance? (W for Weekly, M for Monthly, S for every Semester, A for annually): D`
Error: Invalid Input

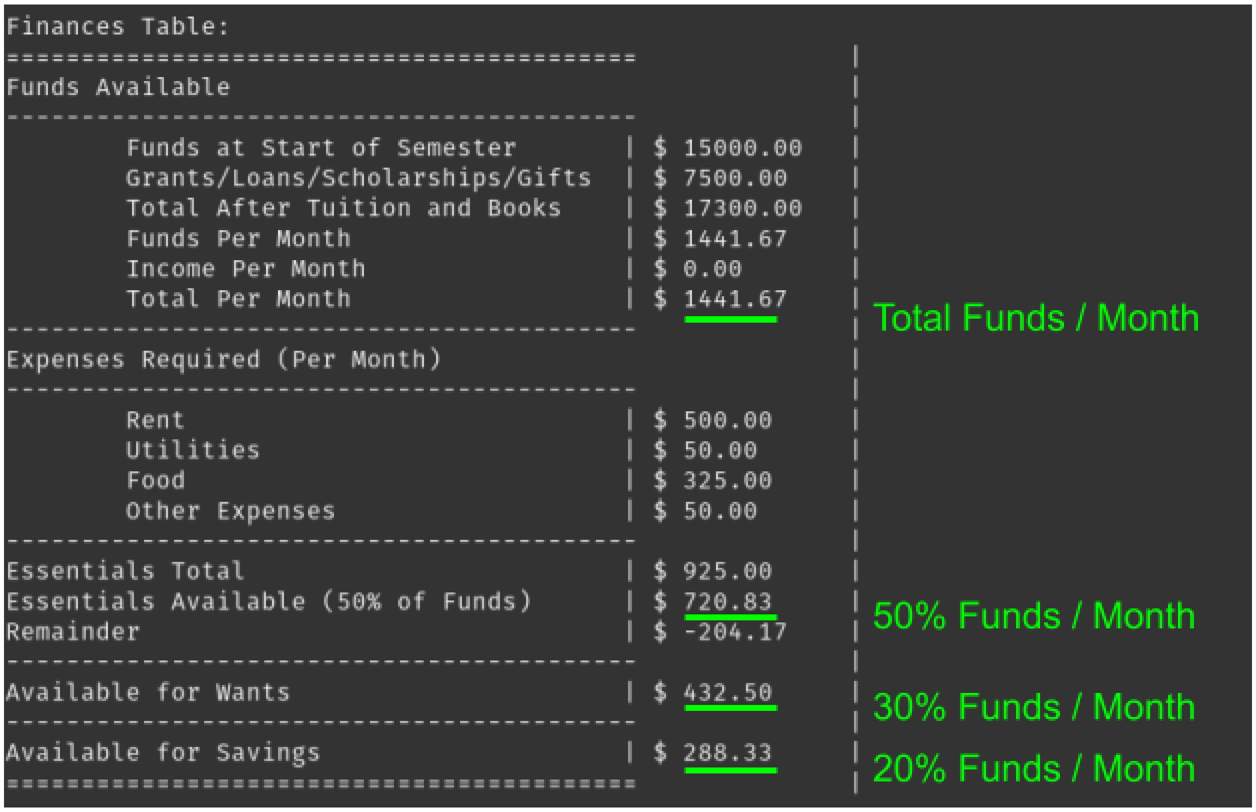
The input character should be case insensitive, meaning a value of 'y' or a value or 'Y' should both be accepted. We will not test for incorrect data types, ex asking for an integer and receiving a character.

Tips:
* Always start small. All these inputs sound daunting at first, pick one question and start with it; create the required variable(s), display the message, read the input to the variable, then print the variable as a test (remember to remove the print before submitting.
* Remember Edit-Compile-Run: regularly test your code by compiling and running it.
* Once you figured out how to do one question, think about how to copy and modify it for another. Repeat the steps from the previous tip. Repeat until you can see all the required data (temporarily) printed as tests.
* You can exit the program at any time by using the return statement in the main.

2.2 Part 2: Generating a Finances Table:

After verifying you have all the required data entered in part 1, you need to print a formatted table of the data. The table format is shown below with some sample data. All monetary values should be displayed with 2 decimals places for cents. The skeleton code we provided on courselink already prints some of the text below, this will help ensure you are able to format the text in the same way as we are expecting. Remember it needs to match exactly!

A few key lines are underlined in green and green text to help explain. Remember that we are following the 50/30/20 rule; so we calculate total funds per month, then assign 50% as available for Essential Expenses, 30% for Wants, and 20% for Savings.



```
Finances Table:
==========================================
Funds Available
------------------------------------------
        Funds at Start of Semester      | $ 15000.00
        Grants/Loans/Scholarships/Gifts | $ 7500.00
        Total After Tuition and Books   | $ 17300.00
        Funds Per Month                 | $ 1441.67
        Income Per Month                | $ 0.00
        Total Per Month                 | $ 1441.67
------------------------------------------
Expenses Required (Per Month)
------------------------------------------
        Rent                            | $ 500.00
        Utilities                       | $ 50.00
        Food                            | $ 325.00
        Other Expenses                  | $ 50.00
------------------------------------------
Essentials Total                        | $ 925.00
Essentials Available (50% of Funds)     | $ 720.83
Remainder                               | $ -204.17
------------------------------------------
Available for Wants                     | $ 432.50
------------------------------------------
Available for Savings                   | $ 288.33
==========================================
```

Note that if a payments/income is entered as weekly, we will convert it to yearly first (weekly x 52) then to monthly (yearly / 12). So if someone earns 250 / week from a part-time job, we calculate their monthly as ($250.00 x 52) /12 = $1083.33

If you enter extremely large values ($1000000.00) the table may lose some formatting, we will NOT be testing for this.

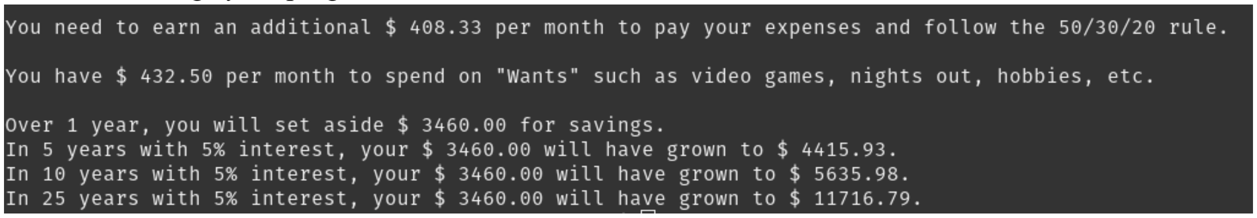
2.3 Part 3: Display Summary

Based on the values from the table, you will now print a summary of Expenses, Wants, and Savings. For Expenses you will either state that the user needs to earn additional income to meet their expenses:
"You need to earn an additional $XXX.XX per month to pay your expenses and follow the 50/30/20 rule."
 or you should recommend they set aside the remaining for a rainy day:
"You are following the 50/30/20 rule perfectly. The extra $XXX.XX per month can be set aside for a rainy day!"

The Wants and Savings sections will always display the same format (as shown below). **Note that wants and savings cannot be negative in the summary, if it is below 0 it should be set to 0** (Hint. the "fmin" function in math.h could help here). Ex. If a user's Funds, Grants, and income are less than tuition, the monthly funds available would be negative in the above table; you will need to ensure the value shows as a minimum of 0 in the summary (See Sample Flow Videos).

If we wanted to calculate the total growth of variable called yearlySavings with interest rate of 5% over 25 years, then we would **use the following interest equation**: *total = (yearlySavings * 1.05* ^{25}*)* . Hint: use the pow function in math.h.

After this message your program will end.



```
You need to earn an additional $ 408.33 per month to pay your expenses and follow the 50/30/20 rule.

You have $ 432.50 per month to spend on "Wants" such as video games, nights out, hobbies, etc.

Over 1 year, you will set aside $ 3460.00 for savings.
In 5 years with 5% interest, your $ 3460.00 will have grown to $ 4415.93.
In 10 years with 5% interest, your $ 3460.00 will have grown to $ 5635.98.
In 25 years with 5% interest, your $ 3460.00 will have grown to $ 11716.79.
```

Note: if your *essentials available* is $100 short of the *essentials required*, you will need to increase your *monthly funds available* by $200. This is because the essentials budget is ½ of the total budget.

3. Assessment

The assignment is graded out of 10 marks. The last page of this document shows an example of what a rubric/feedback file might look like. The 10 marks are split between these three categories:

3.1 The First 6 Marks

This is the bulk of the assignment, in which your program will receive input from a user, do a calculation, and output the result. We will run several tests using different input values then compare your output against the expected output. This is the majority of your grade, each test we run will be worth marks.

3.2 The Next 2 Marks

Your output text formatting will be graded out of 2 i.e. it will be worth 20% of this assignment. We will flag differences in spellings, spaces, tables, new lines, etc. and test against different input values using the auto-grader, so make your you follow the output exactly. For each unique error, 0.5 marks will be deducted. The formatting grade will not be negative, it will only range from 0 to 2.

Refer to the sample flows provided for how your program is expected to run (4 Sample Flows). Altering/customizing sentences will not be accepted, and you will lose grades in this category if you do so.

3.3 The Final 2 Marks

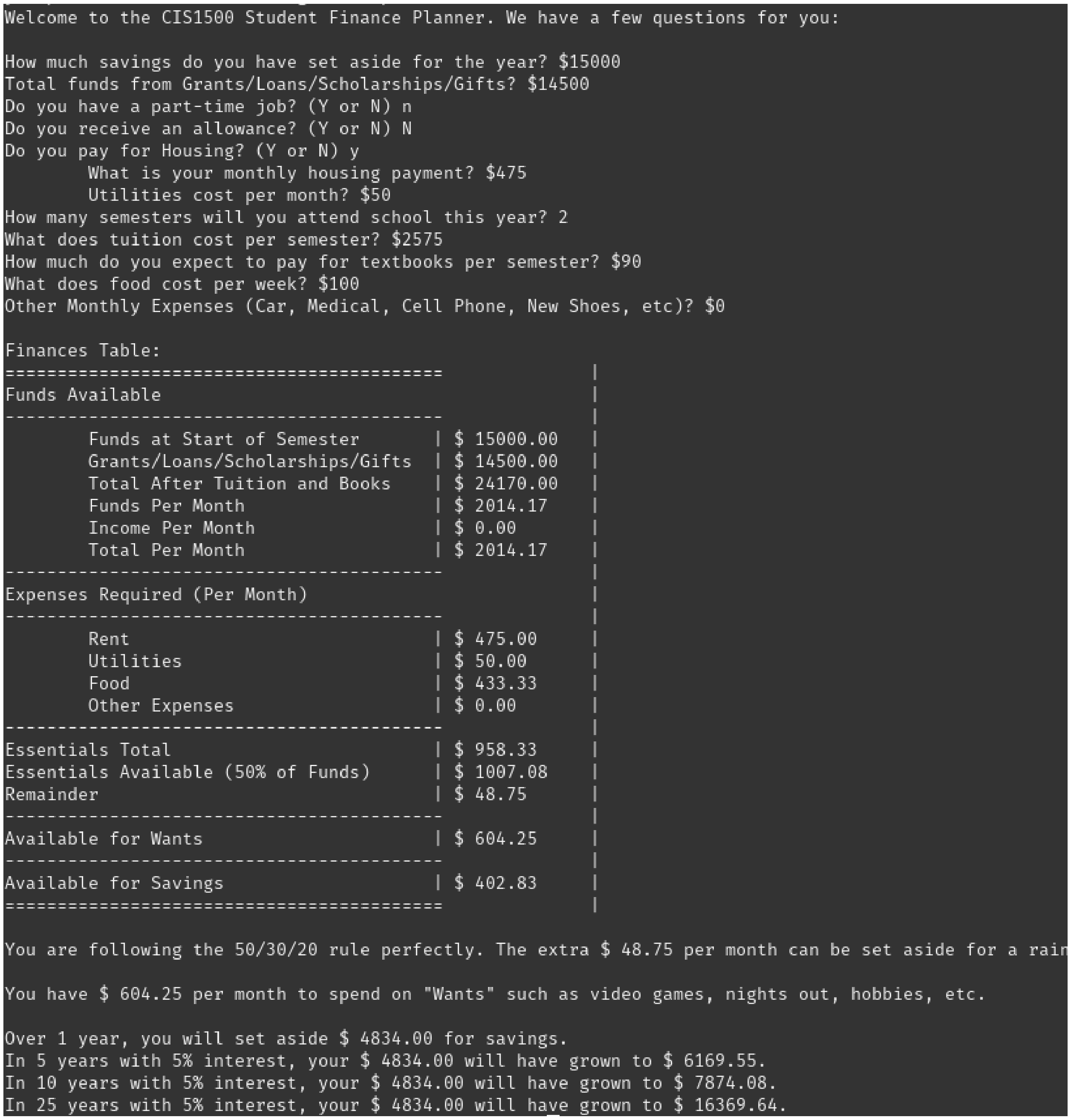
Style: Indentation, variable names, and comments.

Comments are most commonly used to explain confusing or not easily understood parts of code or for depicting that the following code carries out a certain piece of logic. These are also used to explain the program and add a header to it. A file header comment is a type of comment that appears at the top of your program before the #include line(s). **Comment grades include header comments (see 5.3 File Header Comment Format) and meaningful comments throughout your code.**

Each code block should be **indented for readability** (if/else statements are an example of a code block). We are looking for **meaningful variable names** that depict the values they are representing – this will make it easier for you to remember what each variable is for and for us to grade you!

4. Sample Flows

Here is another example of the program being run. This has none of the highlighting from the previous samples, so this should be exactly what your program looks like with these inputs. There are still questions not shown that you will need to review the Sample Flow Videos to see. However this should help to understand the overall flow of the program. Make sure you follow the formatting exactly. The skeleton code contains some of the table formatting already, so you can use that as a basis. Sub-questions are intended with a single tab (hint \t in printf).



```
Welcome to the CIS1500 Student Finance Planner. We have a few questions for you:

How much savings do you have set aside for the year? $15000
Total funds from Grants/Loans/Scholarships/Gifts? $14500
Do you have a part-time job? (Y or N) n
Do you receive an allowance? (Y or N) N
Do you pay for Housing? (Y or N) y
        What is your monthly housing payment? $475
        Utilities cost per month? $50
How many semesters will you attend school this year? 2
What does tuition cost per semester? $2575
How much do you expect to pay for textbooks per semester? $90
What does food cost per week? $100
Other Monthly Expenses (Car, Medical, Cell Phone, New Shoes, etc)? $0

Finances Table:
==========================================              |
Funds Available                                         |
------------------------------------------              |
        Funds at Start of Semester       | $ 15000.00   |
        Grants/Loans/Scholarships/Gifts  | $ 14500.00   |
        Total After Tuition and Books    | $ 24170.00   |
        Funds Per Month                  | $ 2014.17    |
        Income Per Month                 | $ 0.00       |
        Total Per Month                  | $ 2014.17    |
------------------------------------------              |
Expenses Required (Per Month)                           |
------------------------------------------              |
        Rent                             | $ 475.00     |
        Utilities                        | $ 50.00      |
        Food                             | $ 433.33     |
        Other Expenses                   | $ 0.00       |
------------------------------------------              |
Essentials Total                         | $ 958.33     |
Essentials Available (50% of Funds)      | $ 1007.08    |
Remainder                                | $ 48.75      |
------------------------------------------              |
Available for Wants                      | $ 604.25     |
------------------------------------------              |
Available for Savings                    | $ 402.83     |
==========================================              |

You are following the 50/30/20 rule perfectly. The extra $ 48.75 per month can be set aside for a rain

You have $ 604.25 per month to spend on "Wants" such as video games, nights out, hobbies, etc.

Over 1 year, you will set aside $ 4834.00 for savings.
In 5 years with 5% interest, your $ 4834.00 will have grown to $ 6169.55.
In 10 years with 5% interest, your $ 4834.00 will have grown to $ 7874.08.
In 25 years with 5% interest, your $ 4834.00 will have grown to $ 16369.64.
```

Note that **your submission will be either fully or partially graded by an auto-grader**. To get full grades you need to match our output requirements exactly.

5. Program Submission and Administration Information

The following section outlines what is expected when submitting the assignment and various other administration information with respect to the assignment. To submit your assignment, upload a single zip file to the Dropbox box for A1 on Courselink.

5.1 The Submission File

The following is expected for the file that is to be submitted upon completing the assignment:

- You are to submit **one zip file containing one source code file** for your program.
- The name of the **zip file** follows the following format: **studentNumberA1.zip**
  - Example: John Snow's student number is 1770770 then the zip file name is 1770770A1.zip
- The single file within the zip you are to submit is a non-compiled code C file (ends with .c) containing your code
  - Do not include any other files or folders in your submission; they will be ignored.
  - Do not include this file inside another folder; it will be ignored by the auto-grader.
- The name of the **C file** follows the following format: **studentNumberA1.c**
  - Example: John Snow's student number is 1770770 then the C file name is 1770770A1.c
- **Incorrect zip file or C file name will result in a grade of 0.**
- To ensure you zip correctly and in the right format, we require you to zip your submission through the command line using the following command:

```
zip studentNumberA1.zip studentNumberA1.c
```

Make sure to replace studentNumber with your own student number.
You may wish to download your own submission from dropbox, transfer a copy back to the SoCS server and re-test after submission. This way you can be 100% certain you have submitted the correct document.

5.2 Program Expectations

Your program is expected to follow the outlined information exactly. Failure to do so will result in deductions to your assignment grade.

- Your program should be compiled and tested on the SoCS Linux server.
- You must use the following command to compile your code:

```
gcc -Wall -std=c99 studentNumberA1.c -lm
```

  - Example: For John Snow's submission, the command would be:
    `gcc -Wall -std=c99 1770770A1.c -lm`
- Programs that produce warnings upon compilation will receive a deduction of 1 mark for each type/category of warning
- The program file must contain instructions for the TA on how to compile and run your program in a file header comment (see 3.3 File Header Comment Format).

You will receive a 0 if:

- your program does not compile/creates errors when compiled with gcc -std=c99 -Wall on the SoCS Linux server.
- your program contains any global variables (variables declared outside of a function)
- you program uses:goto statements
- you program uses malloc
- you program uses calloc
- you program uses regex
- you program uses concepts not already taught in this course (ie. arrays, custom functions, pointers, structs, etc.).

5.3 File Header Comment Format

Note: This sample uses the name John Snow; the **file name, student name, student ID, file descriptions, etc. must be changed**.

```
/************************** 1770770A1.c **************************

Student Name: John Snow                                Student ID: 1770770

Due Date: Friday, 6th Oct 2023, at 11:59 pm

Course Name: CIS*1500

I have exclusive control over this submission via my password.

By including this statement in this header comment, I certify that:

1) I have read and understood the University policy on academic integrity; and

2) I have completed assigned video on academic integrity.

I assert that this work is my own. I have appropriately acknowledged any and all material (code, data,
images, ideas or words) that I have used, whether directly quoted or paraphrase. Furthermore, I certify
that this assignment was prepared by me specifically for this course.

***************************************************************

This file contains…

Describe the program, functions, important variables, etc.

***************************************************************

The program should be compiled using the following flags:

-std=c99

-Wall

-lm

Compiling:

gcc -Wall -std=c99 1770707A1.c -o A1 -lm

OR

Gcc -Wall -std=c99 1770707A1.c -lm

Running the program:

./A1

OR

./a.out

***************************************************************/
```

5.3 Grading Scheme:

This is an example of what feedback might look like. Each of the 6 Test Cases will look at a different specific situation. For example one test case might have the user enter W as a response to the question “Do you have a part-time job? (Y or N)”, this is an invalid input so we would expect the program to print “Error: Invalid Input” and immediately end. If your program did anything else (continued running, didn’t display the error message, etc.) it would not pass that test and you would not receive the grade for that test.

A1 Marking Scheme: 1770770

Style (out of 2.0)

* Header Comment (0.5/0.5)
* Indentation (0.5/0.5)
* Commenting/Readability (1.0/1.0)

Test Cases and Outputs (out of 6.0)

* Test Case 1 (1.0/1.0)
* Test Case 2 (1.0/1.0)
* Test Case 3 (0.5/1.0)
* Test Case 4 (1.0/1.0)
* Test Case 5 (1.0/1.0)
* Test Case 6 (0.0/1.0)

Grade for Output Formatting (1.5/2.0)

Final grade: 8.5 /10.0

TA Comments:

Should say "Welcome to the CIS1500 Student Finance Planner. We have a few questions for you:" not "Hello and Welcome". Program did not exit after invalid input from use.