INFO1111: Computing 1A Professionalism

2023 Semester 1

Self-Learning Report

Submission number: ??

Github link: https://github.com/yousiefhassan156/INFO1111-Self-Learning-Assignment-Self-Submission-1.git

Student name	Yousief Amr Elsayed Hassan
Student ID	530193973
Topic	Topic 5: C++
Levels already achieved	N/A
Levels in this report	A and B

Contents

Instructions

Important: This section should be removed prior to submission.

You should use this LaTeX template to generate your self-learning report. Keep in mind the following key points:

- **Submissions**: There will be three opportunities during the semester to submit this report. For each submission you can attempt 1 or 2 levels. Each submission should use the same report, but amended to include new information.
- **Assessment**: In order to achieve level B, you must first have achieved level A, and so on for each level up to level D. This means that we will not assess a higher level until a lower level has been achieved (though we will review one level higher and give you feedback to help you in refining your work).
- **Minimum requirement**: Remember that in order to pass the unit, you must achieve at least level A in the self-learning (unless you achieve level B in both the skills and knowledge categories).
- **Using this template**: When completing each section you should remove the explanation text and replace it with your material.
- **Referencing**: You should also ensure that any resources you use are suitably referenced, and references are included into the reference list at the end of this document. You should use the IEEE reference style [**?**] (the reference included here shows you how this can be easily achieved).

1. Level A: Initial Understanding

1.1. Level A Demonstration

- Install C++ and a compiler that is compatible with VS Code and create a local file to have data stored in.

- Complete basic unary, arithmetic, and rational operations with the use of a variety of data types while understanding how the different standard libraries perform.

- Create a simple user input program which creates a conversation with the written program and uses it's written data.

1.2. Learning Approach

After picking Topic 5 - C++, I went on to research the language and understand its basics of it. As a person with little programming knowledge, this self-learning task will enhance my computer science path. I started by learning how to download C++ on my device and download a compiler to suit the language on VsCode. After doing that, I started to learn the basics of C++ which included the basics of data types, how standard libraries work, the different mathematical notations and simple lines of code such as "Hello World!". I started off by hopping from one website to another to understand the basics, as one website didn't cover everything I needed to understand.

With background knowledge of python, understanding how C++ functions were easy and I was able to implement the basics into writing code. I went with a different approach for Level A, rather than just showing what I have learnt online. I chose to explain my learnings through examples to refer back to the C++ file in the future. The user input took time to understand and I am still learning how to write perfect user input statements as errors arise when running the file.

1.3. Challenges and Difficulties

As I started learning how to code in C++, I started facing issues from the beginning. For example, learning the different standard libraries and how to write a function for it was a hard process. The wide variety of standard libraries left me in doubt when picking one for my code block. A part which I found more challenging than the rest was the user input function. After reading online and trying to implement my new knowledge, the difference between "cout" and "cin" and using "«" and "»" caused my file to bug a couple of times. I would either place them in the wrong order or mess up my direction arrows. In addition to that, using curly brackets began to become an issue when the user input conversation increased in length. I was messing up the placement of the curly bracket therefore causing errors. To deal with my difficulties, I would either return to the internet and search my errors or use a trial and error system until I understand the faulty code.

1.4. Learning Sources

Learning Source - What source did you use? (Note: Include source details such as links to websites, videos etc.). Contribution to Learning - How did the source contribute to your learning (i.e. what did you use the source for)?

Learning Source - What source did you use? (Note: Include source details such as links to websites, videos etc.).	Contribution to Learning - How did the source contribute to your learning (i.e. what did you use the source for)?
Downloading both C++ and a Compiler.	How to download C++ on VSCode and which compiler to download, which in this case, I downloaded Code Runner.
Using a Hyperlink on LaTex.	Issue with the line arose as it couldn't run due to the link going to the next box, had to learn how to put in a hyperlink to make the issue go away.
Understanding the different data types.	Learn the different data types and how to use them
Rational Operators.	Understanding the rational operators as I had issues with them.

1.5. Application artifacts

In my code file, I started by naming my files so that the processor knows which files to choose from. After that, I defined my standard library and established the code by naming the upcoming set of codes as "int main". This means that integers will be used in the code. As I learnt, by using "slashslash" (or double slash) it shows that it is a note and therefore it will not show when running the program. I used the note feature to name each code part to make it easier when trying to understand the explanations.

I then went on to explain the different mathematical data types and use the "cout «" feature to show the output (same as the print function in python). After this step, I distinguished the differences between the Unary, Arithmetic, and Rational Operators by showing mathematical examples such as "cout « "a + b = " « a + b « endl;". Whenever I can, I would replace the "endl" function by "slashn" to make it easier for me when needing to create a new line. I would also use the "slashn" twice back to back to create a two-line space when needing to differentiate between things in my explanation. I always started each explanation by using "int" and then writing the variable numbers to allow the program to understand that an integer is placed and not a float.

When coming to the use of the standard library fully, I used the "double" function and then wrote my variables to identify them as floats with decimals. As always, I explained them in detail by using mathematical operators such as "sqrt" and "log".

I then wrote a user input interference using the learnt knowledge about C++. I attempted to use all of the data types which I haven't used before to show my clear understanding. And used the "cout" and "cin" functions well to allow the user to input their response. I used within the conversation "getline()" instead of "cin" to read multiple lines of string rather than one. The "setprecision()" function was used to allow only a certain decimal point to be shown, and within the bracket, I plugged in 3 to only limit the answer to three decimal points.

2. Level B: Basic Application

Whilst level A is about doing something simple with the topic to just show that you have started to be able to use the tool or technology, level B is about doing something practical that might actually be useful.

2.1. Level B Demonstration

I attempted to create a User-Input Adventure Game, where I'll demonstrate the basics of the C++ language such as data types, classes, and user input to program the adventure game. It includes a series of decisions to be made while solving a variety of mathematical questions as user input. To solve the game, the users need to complete the arguments given by the written codes. The different math data types which are used are +, -, *, /, <, >, −=, ++=.

2.2. Application artifacts

In my code file, like in Level A, I started by naming my files so that the processor knows which files to choose from. After that, I defined my standard library and established the code by naming the upcoming set of codes as "int main". That is because the only main data type I used was "int".

I went on to create a user interference game using the learnt knowledge from Level A.

I was not able to complete Level B to the fullest.

3. Level C: Deeper Understanding

Level C focuses on showing that you have actually understood the tool or technology at a relatively advanced level. You will need to compare it to alternatives, identifying key strengths and weaknesses, and the areas where this tool is most effective.

3.1. Strengths

What are the key strengths of the item you have learnt? (50-100 words)

3.2. Weaknesses

What are the key weaknesses of the item you have learnt? (50-100 words)

3.3. Usefulness

Describe one scenario under which you believe the topic you have learnt could be useful. (50-100 words)

3.4. Key Question 1

Note: This question is in the table in the 'Self Learning: List of Topics' page on Canvas. (50-100 words)

3.5. Key Question 2

Note: This question is in the table in the 'Self Learning: List of Topics' page on Canvas. (50-100 words)

4. Level D: Evolution of skills

4.1. Level D Demonstration

This is a short description of the application that you have developed. (50-100 words). **IMPORTANT:** *You might wish to submit this as part of an earlier submission in order to obtain feedback as to whether this is likely to be acceptable for level D.*

4.2. Application artifacts

Include here a description of what you actually created (what does it do? How does it work? How did you create it?). Include any code or other related artefacts that you created (these should also be included in your github repository).

If you do include screengrabs to show what you have done then these should be annotated to explain what it is showing and what the application does.

4.3. Alternative tools/technologies

Identify 2 alternative tools/technologies that can be used instead of the one you studied for your topic. (e.g. if your topic was Python, then you might identify Java and Golang)

4.4. Comparative Analysis

Describe situations in which both your topic and each of the identified alternatives would be preferred over the others (100-200 words).

5. Bibliography

- How to Write And Run C and C++ Code in Visual Studio Code (2023). Available at: https://www.freecodecamp.org/news/how-to-write-and-run-c-cpp-code-on-visual-studio-code/ (Accessed: 30 March 2023).

- How to Write And Run C and C++ Code in Visual Studio Code (2023). Available at: https://www.freecodecamp.org/news/how-to-write-and-run-c-cpp-code-on-visual-studio-code/ (Accessed: 30 March 2023).

- C++ Tutorial (2023). Available at: https://www.w3schools.com/cpp/default.asp (Accessed: 30 March 2023). - C++ Data Types (2023). Available at: https://www.programiz.com/cpp-programming/data-types (Accessed: 31 March 2023).

- How to Make Your Own C++ Game Engine (2022). Available at: https://www.gamedeveloper.com/blog to-make-your-own-c-game-engine (Accessed: 31 March 2023).