SIM UOW CSCI251

ASSIGNMENT 1

Last updated: 14 January 2024

Timothy Mah

10258663

Lecture Group L01

Contents

[Introduction 2](#_Toc156479967)

[Usage 2](#_Toc156479968)

[Program flow 2](#_Toc156479969)

[Extra features 3](#_Toc156479970)

[Dynamic border character and grid spacing 3](#_Toc156479971)

[Input validation 4](#_Toc156479972)

[Main menu 4](#_Toc156479973)

[File not found 4](#_Toc156479974)

[Config file must be processed 4](#_Toc156479975)

[Config file ignores white spaces outside of the 5 lines 4](#_Toc156479976)

[Limitations 4](#_Toc156479977)

[Learnings 5](#_Toc156479978)

[How to update elements in vector with range type for loop 5](#_Toc156479979)

[Some functions want C style string, not regular string 5](#_Toc156479980)

[Map is just like a python dictionary with key:value 6](#_Toc156479981)

[Unordered\_map is map but unordered and slower 6](#_Toc156479982)

[Shortcut class constructor syntax 6](#_Toc156479983)

[Switch-case vs if-else 6](#_Toc156479984)

[Regex worked on Windows but not assignment Ubuntu 7](#_Toc156479985)

[Find() 7](#_Toc156479986)

[Thoughts and reflections 7](#_Toc156479987)

[Am I alone? 7](#_Toc156479988)

[Impressive teacher 7](#_Toc156479989)

[Assignment difficulty 7](#_Toc156479990)

[C++ is like C but with dynamic memory allocation 7](#_Toc156479991)

# Introduction

Hello this is my documentation for SIM UOW CSCI251 ASSIGNMENT 1.

# Usage

The final program file can only be run in Ubuntu because of the makefile. However, the main file can be compiled to support Winidows by compiling as .exe.

To run,

./ass.app

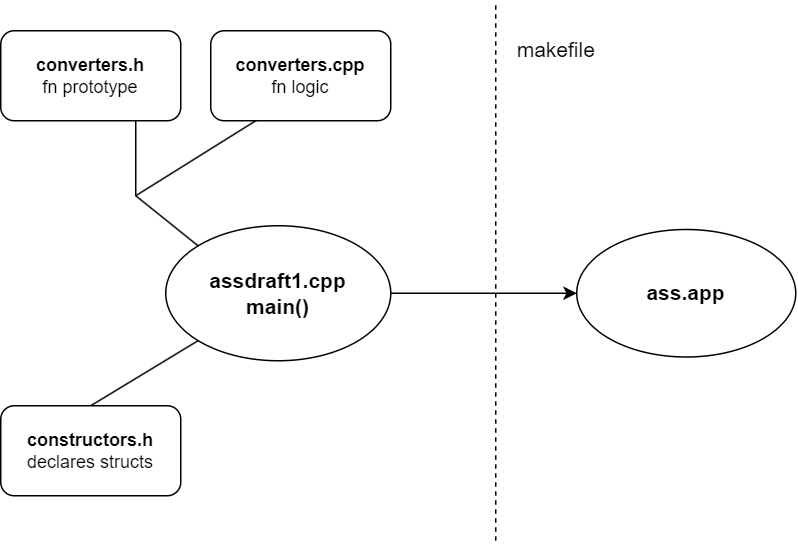
To compile,

make

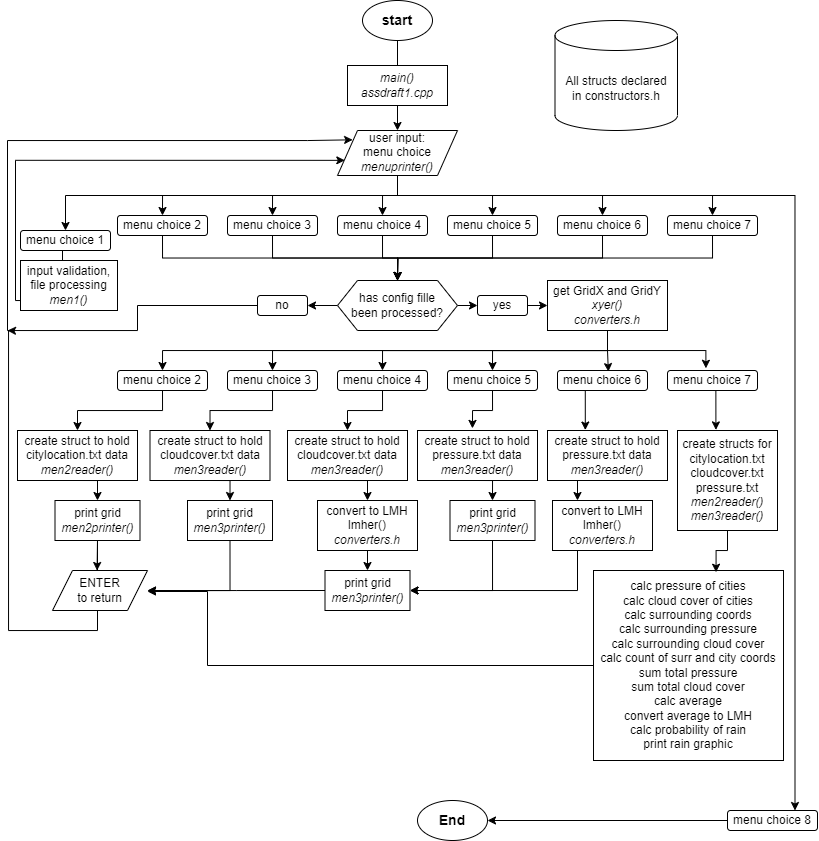
In same directory as makefile. Makefile should be in same directory as main file and the other helper files.

# Program flow

This is my program structure. assdraft.cpp is the main program file while converters.h and converters.cpp declare functions that transform values. All structs are declared in constructors.h.



This is my program flowchart



There are primarily 5 functions in main(), and the rest are either my converter functions that are all declared in converters.h, or other miscellaneous functions such as menuprinter(). As mentioned earlier, all structs are declared in constructors.h. The 5 primary functions are, men1(), men2reader(), men2printer(), men3reader(), and men3printer(). Initially I had wanted to create an individual function for each menu option however I realised that many of the same logic and code is actually reused for options 2 to 6. I could have reused men2reader() and men2printer() for options 3 to 6 as well however men2reader() returns a different struct than what’s needed for options 3 to 6 so I had to create similar functions but with different return type.

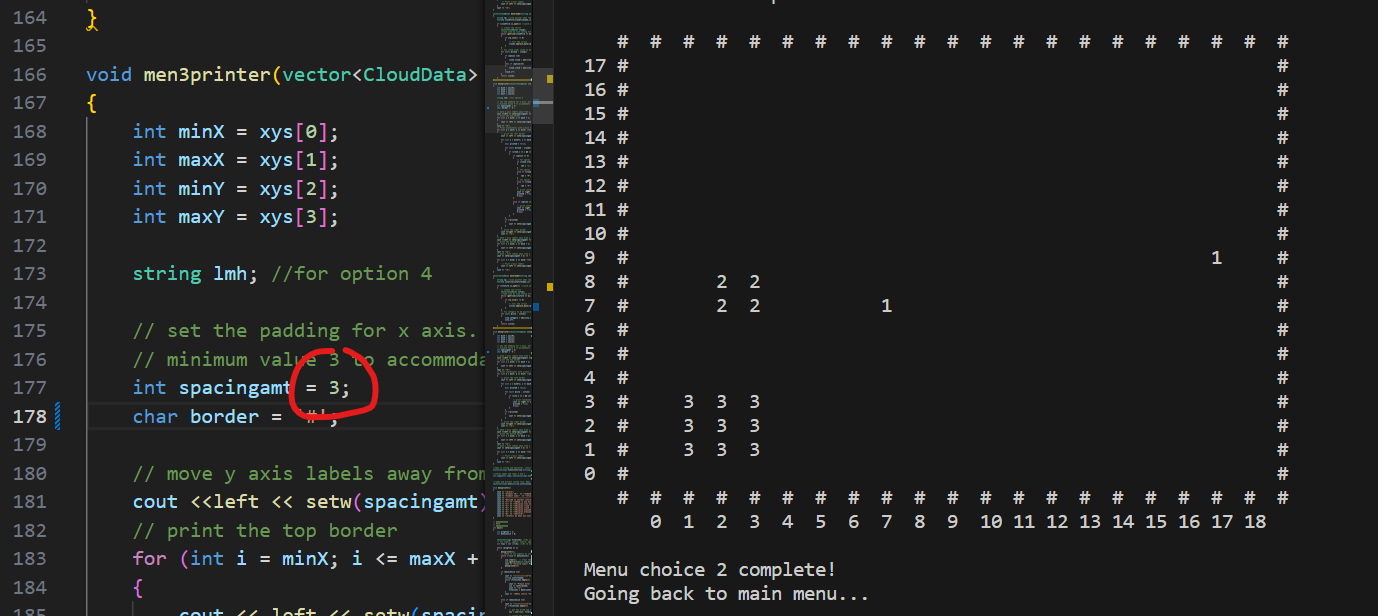
I suppose I could have use the same struct for CityData (stores city name, city ID, and X Y) and CloudData(stores pressure or cloud cover and X Y). But then the struct that holds pressure and cloud cover data would have to hold city name and city ID. This method would mean that I have less functions created but structs take up more memory. I could have tested both styles to see which one performed better but this assignment program is so small that any difference would have been miniscule and may not even be noticed significantly. Regardless, my current code works.

# Extra features

Hopefully these features give me extra marks ☺

## Dynamic border character and grid spacing

I have written my grid algorithm such that the # border can be set to any character. But it must be a single character long. The grid can also be dynamically adjusted to be horizontally wider or narrower. Wider grid is easier to read while a narrower grid can fit more things within the same terminal.



Above is a grid with spacing of 3, and a border of #



Above is a grid with spacing of 5, and a border of @

As illustrated, the change in grid size does not affect the coordinates in the grid. The x and y coordinates remain consistently accurate.

## Input validation

### Main menu

If the user inputs anything that’s not within the menu options, the program will ask for input again.

### File not found

In option 1, if the file is not found or if it cannot be read, the program will ask to input another file name.

### Config file must be processed

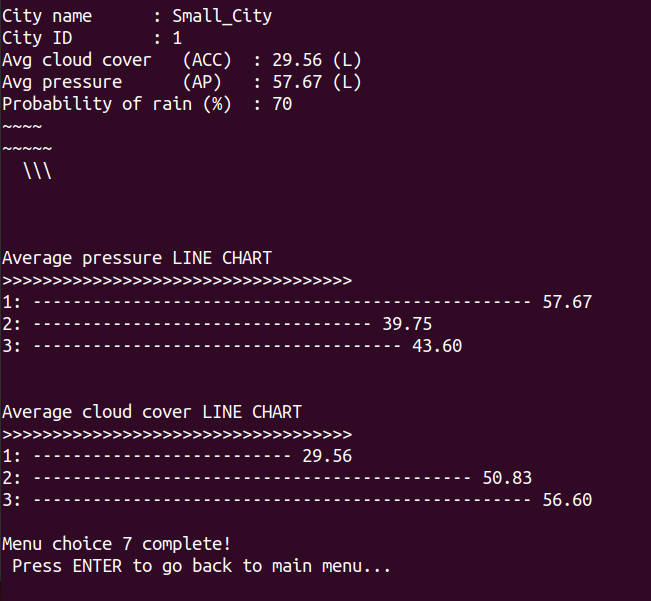
If the user selects any options from 2 to 7, without processing the config file in option 1, the selected option will not run.

### Config file ignores white spaces outside of the 5 lines

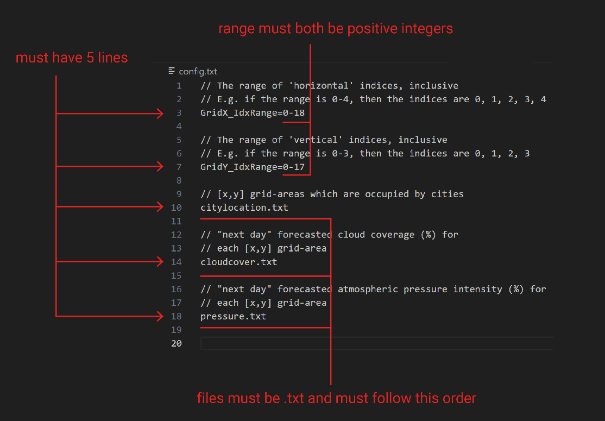
When the program reads in the config file, all comments and empty lines are ignored.

## Data visualisation

In the weather summary report, I have included a line chart to visually compare average pressure and average cloud cover among the cities.



# Limitations

1. Regex string may be system agnostic. However, in today’s context where many programs run over the web, server side processing would mitigate this issue.
2. The ncurser library is only available on Linux. But I stopped using it
3. The x and y axes of the grid can only be 2 digits big. Any bigger and the printing would have issue. But I think it’s fine because if the x and y axes are 3 digits big, the issue won’t just be the printing. The issue is that the user cannot even see the whole grid properly in terminal because it’s too big.
4. The application assumes that the x and y axes are integers. If it’s a decimal number, the decimal values will be truncated
5. The config file requires a specific format else the program would not run properly.
   1. 
6. User cannot re-enter config file. User would need to end program and start again
7. If there are duplicate coordinates in any of the data sets, the program will not print accurately. It will print the last duplicate entry in the data set.

# Learnings

Some cool stuff I learnt

## How to update elements in vector with range type for loop

Personally I prefer to code in python style. Hence, the python style for loop is more intuitive to me.

        vector<string> strings;

        for (string s: strings)

        {

            s = "some string";

        }

In the above code snippet, the strings within the vector would be updated with the value, “some string”. However, the value “some strings” will only stay persistent within this for loop. Outside of the for loop, the values in strings is still empty.

To make the value persistent globally, reference the memory location of elements within the vector.

        vector<string> strings;

        for (string &s: strings)

        {

            s = "some string";

        }

Just one simple & is needed.

Of course I could also do the traditional for loop style with counter++ and all but I like this python style for loop. So this is like a mix of C++ style with python. Lol

## Some functions want C style string, not regular string

I used this function called mvwprintw() from the ncurser library. It’s fourth parameter takes in a string. I had an int type variable that I wanted to use. I looked up ways to convert int to string, however I still received an error message. Upon closer inspection, the function wanted C style string const char\*. So I searched up how to convert string to C style string. In the end I mashed two functions together to become to\_string(gridy).c\_str()

But in the end I realised I cannot use external libraries so I scrapped it lol.

## Map is just like a python dictionary with key:value

I discovered this type called map that acts just like a python dictionary. Useful when I want to look for something based on key:value pairs.

## Unordered\_map is map but unordered and slower

Apparently maps, aka c++ dicts lol, have an unordered version however it takes up more memory. It’s use case is specific to the developer’s application. For my case, it seems both work fine so I used map instead.

## Shortcut class constructor syntax

I copied some code online to create my struct

// Struct to store coordinates and category

struct surrounddata

{

    int x;

    int y;

    int category;

    surrounddata(int x, int y, int category) : x(x), y(y), category(category) {}

};

I found the constructor function quite unusual especially with the : sign. Apparently this is just a shortcut syntax to construct struct with member elements. It is the same as the java style constructor.

// Struct to store coordinates and category

struct surrounddata

{

    int x;

    int y;

    int category;

    surrounddata(int X, int Y, int CATEGORY)

    {

        x=X;

        y=Y;

        category=CATEGORY;

    }

};

I guess I will use the java style because it is easier to understand despite being less clean.

## Switch-case vs if-else

There were several instances where I needed to evaluate switch-case or if-else. From what I read online, performance wise, both are similar, and it depends more on the situation and even the IDE plays a part. So I guess it boils down to code readability.

I decided to use switch-case when the case only has single simple condition such as checking if switch is 1 or 2 or 3. This was used in my menu choice.

Meanwhile, I used if-else when the if statement has multiple conditions. This was the case for my function that takes in a value and outputs it’s corresponding LMH value.

## Regex worked on Windows but not assignment Ubuntu

I developed my program on my windows laptop. It works fine. I tested it on my latest Ubuntu VM and the assignment Ubuntu VM. The regex did not work. I used a different method to get the 0-8 range and the file names. I guess regex is indeed system agnostic however as mentioned in my program limitations, server side processing would mitigate this issue.

## Find()

This find() function looks so simple and I assumed it just finds a specified value in a string. Yes, it essentially does that but it comes with some unique components.

With reference to <https://www.geeksforgeeks.org/string-find-in-cpp/> ,

find() introduced me to size\_t and string::npos . Apparently when using find(), you need these 2 other things for it to work as intended. size\_t in this context is basically an int that cannot be negative. This is useful because when comparing a size\_t variable with string::npos , this action is basically checking if the target string has been found. If it has not been found, string::npos would be -1. The logic is a bit hard to follow but I’m sort of treating it like a black box.

I prefer using regex, but unfortunately, my program behaves differently on Windows compared to Ubuntu.

# Thoughts and reflections

## Am I alone?

I entered this course with a 3-year diploma in cybersecurity. I was given the maximum amount of exemptions – 1.5 years. I feel I have more prior knowledge and experience than my peers. I found it very shocking to see classmates searching on their laptop, “what is Ubuntu”, “what is variable”, these are to name of but a few. I’m not trying to gate keep or look down on them. I think it’s great that they’re learning but I am concerned for myself because I was hoping to meet more people who are on the same path as me. I feel that many of my classmates are not as keen on a cybersecurity career as me. Perhaps it’s good that this first assignment does not require group work.

## Impressive teacher

I know it sounds like I am trying to angkat, but I will give credit where it is due. Mr Tian is rather condescending in class. I think it’s just because he has been jaded over the years by not so stellar students. Nonetheless he still answers questions well, and explains concepts clearly. I found it most impressive that he prepared generous resources for students who do not know how to use virtual machines. He provided instructions to install and configure both vmware and virtual box type virtual machines.

He is also able to write functions from memory without having to refer to online sources and documentation. Something I definitely won’t ever be able to do.

HOWEVER, although he has answered my questions in class, he has not replied to my email.

## Assignment difficulty

I found this assignment manageable in terms of difficulty, but very very time consuming. This is only because I have taken a module on C before, and my numerous programming projects I have done not just during poly, but in my personal time and internships. For students without this same experience as me, I think they would struggle GREATLY.

During my C module in poly, my teacher would have weekly practise exercises for us to do. He would explain his answer each week. These exercises were actually all cumulating to the assignment. If we had completed and understood each week’s exercise, we would have been able to complete the assignment because the concepts required for the assignment were covered progressively in each exercise. I found this incredibly helpful for my learning. However, this course it seems this course is not as conducive as my poly course. Quite disappointing honestly. But not a problem I directly faced since I did not find the assignment totally foreign. Perhaps this is not so much of a fault of the teacher but the course structure and schedule. Perhaps I am misunderstanding the situation for students that are doing the full 3 years as opposed to my 1.5 years.

## C++ is like C but with dynamic memory allocation

Vectors are my best friends.

File encapsulation is also a pretty cool concept but I suppose in a modern world with version tracking such as Github and Gitlab, obfuscating code may not be best practice.