Ryan Gouldsmith Assignment write-up

1 Introduction

I first started off the assignment by researching the different types of sentences from the website given in the specification. This took a bit longer than expected as, even though the website was comprehensive, I still felt sections were unclear and confusing. This was amended by Fred, when he gave a detailed explanation of what we needed to do. I then went and broke down the task into smaller sub sections. I then tried to think of an overall algorithm for each sub section that wasn't language specific. This worked quite well and I had a general algorithm for reading in the files, parsing and exporting to the .gpx file.

1.1 Challenges I faced

Throughout the assignment there were many challenges which I came across. First off, I thought that working out an algorithm to align the files was a little bit of a challenge - to which I eventually over come, but this took up a lot of the time. Another challenge I had was, especially with the implementations where you manage your own memory, was keeping track of all the pointers. I thought this was generally more of a concern in the C application whereas in the C++ implementation I could set the instance variable to the current object. A lot of the functions which I could have found useful, such as the for each loop, or regular expressions, were for C++11 and we were not allowed to use that, so I had to use other means of getting around these, some of which were longer than others.

1.2 Parts I found easier

There was a few parts of the assignment that I found easier than others. For example, checking to see if the fix itself was relatively simple to code, and this wasn't much of a challenge especially when I worked out a simple algorithm, this was simple to translate across the languages. Another aspect of the assignment in which I found not too difficult was checking the times in the files to check that they match, this wasn't too strenuous when I either stored the values into the structs or an Object, and checked the values from both the Object.

1.3 Assumptions and Simplifications

To make my application work to the level it does now there was some assumptions and simplifications in order for the application to work. First off, I don't check the time stamp in the first file, compared to the second time stamp. I make the assumption that they will both be around the same time, and if they're not then the files will sync up eventually. Another assumption I made was that I stored some results in the model before outputting them, this is used especially when reading in the number of GSV sentences. I read ahead a couple of sentences/ wait for the GSV sentences to come in, and read all the SNR values from them and process them. That it it's easier to see if a time stamp is valid or not. Other than this I treated all the other data as a stream.

2 Implementation

For the assignment I managed to successfully implement the application in the Java and C++ language; I couldn't get it to properly work in C. However for all languages I tackled the assignment in each of languages in the same way. First of I worked out how to read the file in line by line, so rather storing it all in memory I just read in one line at a time and parsed the data and did all the calculations accordingly. The process for this was: Open up the file reader, read in one line, check the sentence to see if it was a GSV or GGA and then process these sentences in separate methods/functions. This however, made a couple of assumptions. I only ever considered the GSV or GGA sentences, so I discarded all the other information in the other sentences. Additionally, the simplification I made was that when the next GSV sentence came in it corresponded to that last time-stamp read. There is information relating to the satellites in the GGA sentence, but I discard this information as it makes it easier to just assume that GSV is for the prior time-stamp.

Once I have read the first file up until the time stamp I then find the next time stamp in the second file. This makes the assumption that the first file time is the same or before the second files timestamp. Since the assignment said that they should start from the same, or discard the prior times - so if there was no match then the values are not really creditable, so I made the decision of not needing to check which time was first to determine which file to search first. A a result this simplified the resultant designs.

After I read in the file I parsed the information accordingly to each language: so for Java and C++ I used a class object, whereas for C I used a struct. I then parsed the information into the appropriate attributes. This additionally made some simplifications, where I only parsed the information I needed from the sentence (latitude and longitude values) as a result in the structures I only ever store the information actually needed for the exporting procedure rather than all the other information in the sentence.

Additionally when I find a GSV sentence in my implementation I store the number of sentences and read forward checking if there's more GSV sentences. This way I can check the SNR for the current time stamp and check to see whether that fix is good enough to accept, if not then I read in the second file and check that values. This simplified my assignment as I didn't have to keep a track of all the information prior to that sentence, I only ever considered the next block of sentences. In terms of the satellites I don't cross reference them with the ID in the other sentence, I make the assumption that the GSV sentence is for the prior GGA being read in.

Finally, I make the assumption based on the assignment specification that all I need to output is the waypoint item with the latitude and longitude values - this way I only output what is needed. Therefore I do not consider the time or any of the other values from the sentence; this isn't needed for the assignment and you can see the way point path from these values.

3 Libraries Used

For the assignment to work I had to include a few external libraries in order for the assignment to work properly. These will be discussed in the second report, but I am making it aware that there are mainly external libraries for the exporting of the GPSX file, for Java and C++. For C I used a exported it using the `fprintf` function in the C library.

4 Conclusion

Overall, I thought the assignment was challenging. It highlighted where the languages had strengths and they had weaknesses. Having been the first time writing C++ I found it a very nice language to write in, and I thought out of the three languages then this was my favourite one to use. I found once I implemented the version in Java then porting the code to C++ with the different syntax wasn't too strenuous, but when I tried to convert to C I had too many issues for the time frame. I found that being able to debug C++ was easier than that of C. Overall, I am slightly disappointed that I couldn't fully implement the C program but I am happy in knowing I could write the same application in C++, as well as along the way picking up on the similarities and differences in the languages.