Gareth Duffy - g00364693

Higher Diploma in Computing (Data Analytics)

Shop assignment: Multi Paradigm Programming - CSV/Live Shop in C & Java

Overview

Upon reflection, I found this assignment to be notably challenging. Getting to grips with two relatively complex languages that I wasn't very familiar with was oftentimes a mind-numbingly frustrating endeavour. That said, I also found it to be notably rewarding, in the sense that it afforded me an autodidactic opportunity to tackle a relatively complex programming project across two languages from two different paradigm approaches. I have to be honest, I found myself spending a substantial number of hours researching the basics of C and Java, reading up on basic control flow, variable assignment, functions and much more, doggedly searching for useful strategies on how I could approach the "problem" (or problems, to put it bluntly). While I am quite adept with Python, I am new to C and Java, and compared to Python, their static nature was tricky for me to adapt to at first. Nevertheless, I was very curious and eager to learn as I went, i.e. through research and trial-and-error programming I began to get some basic functionality working, and that grew into the catalyst that fuelled the rest of my efforts. Although I have persevered through some very difficult academic projects in the past, this was one of the most challenging I have encountered. Indeed, at times it seemed like I had an endless multitude of issues to work through, but I stuck with it due to my genuine desire to create a unique functional program built out of two languages that I had never used before.

Reflections and difficulties faced in C and Java

I found it quite difficult to get the price finder method to work the way I wanted it to across my entire program, especially for the customer processing section i.e. the checkout. At the start I couldn't get the finder to retrieve the prices at all, and at one point I even had to put the price variable inside my customer CSVs in order to progress with testing my functionality and keep making headway with the program. I spent a large amount of time doing this, testing out strategies to pass in the customer's shopping list item price. For example, I tried to use "global" variables, structs within structs, pointers, functions within functions and more. I eventually resolved the issue by declaring the itemprice variable outside of my main method, and by constructing for loops which iterated over a string compare method I made inside my printCustomer method. Relatively speaking, apart from this issue, the other issues I encountered were not as time consuming. I worked through each one in a piecemeal fashion, adopting a methodical trial-and-error approach and some good old-fashioned logic.

Understanding differences and similarities of C and Java: Notes on comparisons

While similar in many ways, the languages also differ in others. C suffers from a lack of built-in methods, a fact which added a little to my frustration. Another painstaking feature was the memory allocation differences. In C, the malloc function and pointers are essential, whereas basic keywords for allocation and assignment are all you need in Java. In C you must also declare variables in the beginning of a code block while in Java you can make declarations almost anywhere. C's error messages were also a little frustrating in that they are very overly general and vague, not really specifying what the problem in the code is. Java (especially

because I used Eclipse) on the other hand was really great for interpreting error/warning messages. I found I could resolve most of my Java issues a lot quicker than the raft of C issues. Moreover, there is no real exception handling for C, whereas Java let allowed me to incorporate "try and catch methods" when I needed to.

I found one of the most attractive features of Java is its primary property of inheritance, which affords reusability. C by contrast, doesn't support inheritance and as a result is a more laborious animal to work with. I also found it interesting how data hiding in C is achieved through "static" declaration whereas in Java we use "private". Another appealing aspect of Java is how its object-oriented style breaks all the code up into focused pieces (classes) to make each piece of code simpler, which we can combine them together to solve the shop problem. Classes support both state and functionality so in this way it is more methodical and "cleaner" than C which instead relies on structs which don't support functionality. Separate container objects (classes) support a divide-and-conquer approach, and when complexity continues to grow in a program, this approach becomes a very valuable one.

C is very function-oriented while Java is object-oriented, wherein functions vs objects underlie the very basic programming units of each language. Both languages are fast even though C uses a compiler while Java uses bytecode on a VM. I also learned quite quickly just how similar C and Java really are regarding syntax and static style. They share some very similar properties, for example, even though a for loop can be created in a number of ways with Java, both C and Java often adopt the traditional: "for (i = 0; i < N; i++)" method. Indeed this is the method I used myself mostly for Java. Similarly, the while loops are also identical, as are the if-else-else conditional statements. Their operators are also the same e.g. +.-. != etc, as are their primitive data types and keywords. They both operate programmatically from a MAIN function, both use the "null" value, both have the same commenting methods (/**/ or //), and similar data types e.g. "int", "string", "double", "char". An interesting difference is how arrays are declared, e.g. for an integer array in C its typically: "int *a = malloc(N * sizeof(*a));", while in Java its more like: "int[] a = new int[N];". I also found the print methods I both to be a bit laborious compared to Python. The process of data structures is also notably different as I have already mentioned, in that we have a struct in C but a class in Java. Furthermore, classes come combined with much more optional methods for manipulating data while structs don't at all. That said, I found it nice how they have an almost grandfather/grandchild relationship. Indeed, their similarities afford translation from one language to another, which is what I spent a large portion of my time doing in the last couple of weeks up to submission. That said, while almost perpetually frustrated, I was always determined and enjoyed the challenge. I also learned a notable amount about both languages within the whole project experience, and I honestly never expected to. I would like to think I have produced very similar functionality and aesthetic details across both languages for the shop program in both CSV and live modes. While difficult, I found the project to be a very valuable programming experience.

Checklist

KEY: Functionality demonstrated:

- *The shop CSV should be modified to also hold the initial cash value for the shop:* ***YES***
- *Read in customer orders from a CSV:* ***YES***
- *That file should include all the products they wish to buy and in what quantity:* ***YES***
- *It should also include their name and their budget:* ***YES***
- *The shop must be able to process the orders of the customer:* ***YES***
- *Update the cash in the shop based on money receive:* ***YES***
- *It is important that the state of the shop be consistent:* ***YES***
- *You should create customer test files which cannot be completed by the shop:* ***YES***
- *Know whether or not the shop can fill an order:* ***YES***
- *Thrown an appropriate error:* ***YES***
- *Operate in a live mode, where the user can enter a product by name, specify a quantity, and pay for it:* ***YES***

File list

**Please note that I have kept the package name "ShopVideoVersion" the same for my project.*

C: (To run the shop program in C, run it from: "shopC.c")

shopC.c

stock.csv

customer.csv

customer2.csv

customer3.csv

Java: (To run the shop program in Java, run it from: "Shop.java")

Shop.java

Product.java

ProductStock.java

Customer.java

liveshop.java

shopScanner.java (Unused file)

stock.cv

customer.csv

customer2.csv

customer3.csv

References

C:

[0]: https://www.geeksforgeeks.org/data-structures/

[1]: https://rosettacode.org/wiki/Loops/Increment_loop_index_within_loop_body#C

[2]: https://beginnersbook.com/2014/01/c-for-loop/

[3]: https://stackoverflow.com/questions/122616/how-do-i-trim-leading-trailing-whitespace-in-a-standard-waySS

[4]: https://stackoverflow.com/questions/29147785/how-to-compare-length-of-4-strings-according-to-strlen

[5]: https://www.tutorialspoint.com/cprogramming/c_strings.htm

[6]: https://www.programiz.com/c-programming/c-if-else-statement

[7]: https://www.programiz.com/c-programming/c-for-loop

[8]: https://www.geeksforgeeks.org/data-types-in-c

[9]: https://stackoverflow.com/questions/23970329/how-to-store-tokensstrtok-in-a-pointer-on-an-arrays

[10]: https://courses.cs.washington.edu/courses/cse351/16wi/sections/1/Cheatsheet-c.pdf

[11]: http://www.cprograms4future.com/p/online-shopping.html

[12]: https://www.geeksforgeeks.org/switch-statement-cc/

[13]: https://www.tutorialspoint.com/cprogramming/switch_statement_in_c.html)

[14]: https://www.tutorialspoint.com/cprogramming/c_do_while_loop.htm

[15]: https://www.quora.com/What-is-the-importance-of-a-default-statement-in-C-programming

[16]: https://www.tutorialspoint.com/cprogramming/nested_if_statements_in_c.htm

[17]: http://www.cprograms4future.com/p/list-of-all-c-programs-in-my-blog.html

Java:

[0]. https://www.mkyong.com/java/java-display-double-in-2-decimal-points/

[1]. https://crunchify.com/how-to-iterate-through-java-list-4-way-to-iterate-through-loop/

[2]. https://www.geeksforgeeks.org/arraylist-listiterator-method-in-java-with-examples/

[3]. https://www.geeksforgeeks.org/classes-objects-java/

[4]. https://codereview.stackexchange.com/questions/71527/shopping-list-application/71575

[5]. https://docs.oracle.com/javase/tutorial/java/nutsandbolts/switch.html

[6]. https://stackoverflow.com/questions/39228386/user-input-if-statements-in-java-not-working

[7]. https://stackoverflow.com/questions/15857846/how-to-return-to-main-menu-in-switch-case-after-executing-a-method

[8]. https://docs.oracle.com/javase/tutorial/java/data/numberformat.html

[9]. https://stackoverflow.com/questions/15857846/how-to-return-to-main-menu-in-switch-case-after-executing-a-method

[10]. https://stackoverflow.com/questions/39228386/user-input-if-statements-in-java-not-working

[11]. https://stackoverflow.com/questions/24130399/whiletrue-loop-throws-unreachable-code-when-isnt-in-a-void

[12]. https://stackoverflow.com/questions/15218892/running-a-java-program-from-another-java-program