Seng 265 Term Portfolio Project

Adam Zheng

V01005151

NetlinkID: adamzheng

Q1: Describe your continued learning experience in SENG 265.

Lecture week 1:

```
- Learned of Jupyter Notebooks and how they can be great additions to resumes
- Basics of C with Bill Bird
```

Lecture week 2:

```
- Development cycle
- How important Git is when it comes to sharing and updating software worked on by teams of people
- How software development and technological augmentation can improve the lives of many.
- Lab 1:
    -accessing UVic servers remotely and some basic bash commands
```

Lecture week 3:

```
- More bash commands and aliases
    - how to move around with cd
    - attributes for ls
    - piping
    - grep to filter for things
- How chatGTP can help with learning bash
- Lab 2:
    - Vim and its functionalities
```

Lecture week 4:

```
- midterm prep questions
- file permissions and how to change with chmod
- begin assignment 1
- basic c functions like strtok(), scanf(), strcpy()
- Lab 3:
    - Git work flow
    - git add, git commit -m "", git push
    - how to view and access branches
```

Lecture week 5:

```
- memory addresses and pointers for c
- declare pointer with * and set to memory location of variable with &
- segmantation fault error occurs when trying to access memory you don't have access to
```

```
    - file IO for assignment 1
    - Lab 4:
        - Basic c programing and git
```

Lecture week 6:

```
    - c arrays and strings
    - typedef to give more info about the array EX: typedef int a[10]
    - sending values or pointers into function calls
        - if send value, need to return value if want to access changes
        - if send pointer any changed made in the function will hold for
    rest of program
    -Lab 5:
        - Differences between python and c
        - Basic python programing and pandas library
```

Q2: What is the core functionality of Jupyter Notebooks?

Jupyter Notebooks are used for literate programming, a mix of natural language and snippets of code. This allows for Jupyter Notebooks to become vital in recording and sharing information in a meaningful(code wise) yet understandable way(language and logic wise).

Q3: Summarize simple Jupyter Notebook markdown including how to insert links and images?

To insert links you can display the link directly or have a refrence to a link using other text using [text](link).

https://www.youtube.com/

YouTube

To insert images you can can click the **Edit** button in the jupyter notebook tool bar and selected insert image. This will allow you to add any image you put in there.You can also use the HTML img block to add them in but you will need to know the path to the image.

Q4: How can you typeset mathematical formulas including Greek letters using LaTeX Markdown in Jupyter Notebooks?

LaTeX Markdown has commands inside dollar signs. These commands are often mathematical equations, symbols, or greek letters. LaTeX formatting often relies on \begin and \end blocks to organise blocks of text. \ is used to indicate that the following text is related to some command like \sum to get the summation symbol.

Ex: \sum_{n=1}^{\infty} x_i bewteen double dollar signs gives:

$$\sum_{n=1}^{\infty} x_i$$

Q5: What is the history of Unix or Linux?

Linux is a computer operating system created in the early 1990s by Linus Torvalds (a Finnish software engineer) and the FSF. In 1994 the Linux kernel that acted as the core of the operating system and around this time the FSF and an American software developer were trying to create an open source operating system similar to Unix (made in the 1960s). Unlike Linus who was working on the core components the FSF were working on utilities first and when these utilities were added to the Linux kernel that made a complete operating system.

How prevalent is Unix or Linux in software development today?

While not as popular as Windows or MacOS it is very reliable and accounts for most of the servers we have today.

Q6: What is the core functionality of the Bash command and scripting language?

Bash commands are often typed in a command line interface where these commands are capable of changing files, making new directories, programing throught the use of something like vim, compiling and executing code, and much more. This makes Bash very powerful and useful tool in software development as it allows programmers to interact and change files without the need for a graphical user interface as bash can operate out of the terminal.

Q7: What is your background in the scripting language Bash?

For me personally I have never dabbled in the usage of Bash until this course. I have used the cmd terminal for a few things but nothing of the level that bash is capable of.

How prevalent is Bash in software development today?

Bash is very popular in software development today. This is because Bash with its capabilities became the go to interactive shell for most Linux users, and since most servers today run on Linux a lot of developers rely on Bash to maintain and operate their servers.

Q18: Describe the fundamental differences between C and Python.

	c	python
Type	Compiled	Interpreted
Line end	Requires ;	Nothing
Scope	Uses {}	Uses tabbed lines
Data structures	Arrays, Struct	Lists, Maps, etc
Memory	Can manually set	Automatic

Q19: How challenging were learning C and completing Assignment 1 for you?

The syntax for c is very similar to that of Java, which is the coding language I am most familiar with, so a lot of ideas were simple to remember. The hardest things to grasp were how in c there were no strings and you had to use char[] and pointers. The lack of strings made it hard to manipulate the icalander formatted time and the usage of pointers often tripped me up as I wasn't completly sure what was saved in each variable and how I could access and change the values.

Q20: What are your personal insights, aha moments, and epiphanies you experienced in the first part of the SENG 265 course?

The moment that truly resonated with me from the first part of SENG 265 was during the git introducation lecture and the git process lab. This is when I began to understand the importance of git and why it was that alot of game mod projects and such were released on github. My dad works as a tester so i've seen github being used in the context of his work but it wasn't until the lab where we delved into the git process where I began to understand why git is so redaly used in the industry. This introduction to git and the git workflow has helped me better grasp why the industry operates the way it does and I hope to learn more regarding the git workflow through using it in group projects, and learning about it from future lectures.

Q21: How did you experience chatGPT as a learning tool?

When I use chatGPT when studying I tend to get confirmation that my understanding is correct, that is I tend to use chatGPT to make myself feel more sure that I'm understanding the content correctly. When I use chatGPT for programing I use it to write basic code or programs that I can use to understand a concept or manipulate it to get my desired program output. As a learning tool chatGPT has helped me alot as it speeds up my coding proccess and boosts my confidence that my solution is correct.

Part 2

Q1: How useful is SENG 265 for building your resume for future co-op applications and future job applications?

SENG 265 has provided me with a pretty good understanding of two very powerful and desired programing languages, C and Python. This experience will surely come in handy and the examples in SENG 265 has helped with my logical thought process to break down problems too. These tools and Jupyter Notebooks will help display my understanding and present my skills in the best light possible. The section on interview stratages, striking a conversation, and asking questions to clarify tacit knowledge will definatly help during any interview phase of trying to get into a co-op or job. Overall the course has provided me with a strong foundation to learn and apply new skills that will be integral to my future in the industry.

Q3: Describe the notion of and motivation for typing and typing hints in Python?

As Python is a dynamic language where everything is an object and anything can change types and hold different data at any moment during the program. Type hints allow for a more strucutred process where each variable serves a distinct purpose and the programer can be confident that the data stored in that variable is what the type hint says. This allows for more orginized programing and makes working on group projects easier as you won't need to guess whether or not the variable that should be holding a string got accedently turned into an integer.

Q5: What is the difference between pytonic and non-pytonic code?

Pytonic code is code that is written in a the quote on quote proper python programing way while non-pytonic code may still achieve the same outcome but could and should be rewritten to match pytonic standards. This structure allows for easier times understanding code and finding/fixing errors.

Q11: How challenging was Assignment 2 for you?

As a whole I felt that the assignment wasn't too hard. It acted as a good introduction to Python and the Pandas library. As I took CSC 110 during my first semester at UVic I was already pretty familier with Python however I did all the programing in vim which did give me some trouble. One major thing I wish I took the time to learn was copy and pasting in vim as whenever I wanted to repeate a line of code or dupliacate and modify a line I had to write it all out. Vim also doesn't underline mispelled words or give hints that there are errors like in eclipse or wing which ment I had to be more careful when writing the code. Panadas wasn't too bad as the lab was a was a great help in showing what tools pandas provides and which ones we needed for the assignment.

Q12: How challenging was Assignment 3 for you?

Without Pandas and also having to code in C made assignment 3 much harder than assignment 2. It didn't take too long to figure out the steps I would need to take to write the program but it was dealing with memory allocation, pointers, and structs in C that took most of my time. I knew that I wanted to save the inforamtion for each song to add to the linked list into a song struct but didn't realize how much time I would need to spend to implemnt the loops to find the important song information, but to also make the nodes and add them inorder. After making the linked list though it was relaticly smooth sailing as creating the output file wasn't too bad. I do recall spending the first day trying to figure out why the tester wasn't working not realizing that the validator was in Python and required me to setSENG265 so that the type hints would not cause any syntax errors.

Q16: Summarize PEP 8.

PEP 8 is a style guide for Python code written by Guido van Rossum, Barry Warsaw, and Nick Coghlan in 2001. PEP 8 exists to regulate Python code to improve readability. PEP 8 includes standards on naming functions (use all lowercase with underscores to seperate words), variables (use all lowercase single letters, words, or multiple words seperated by underscores), constants (use all uppercase letters, and use underscore to seperate words). PEP 8 also recomends the usage of discriptive variable names rather than using a very genaric one to improve code understanding. PEP 8 also provides guidelines on proper spacing in and out of classes and function, and proper commenting. Overall PEP 8 is integral to making Python sharable and viable in a group coding enviorment.

Q17: Discribe at least five of your favourite libraries in the Python ecosystem

Pandas:

```
- very useful in data management and analysis
- helps in storing, filtering, and presenting data in a more human
readable way
- saves time during data preping processes
```

Random:

```
- used to make random selections from a collection
- used to shuffle a collection
- very good to use when you want to mass test inputs as Random can be
used to stream line the process by using it to generate tons of test
cases.
```

NumPy:

```
- used in alot of math based programing as it provides plenty of
useful math operations and systems
- great for working with large amounts of numerical data
```

os:

```
- easy access to and modification tools for the operating system
- used to create, change, and move files and directories
- used to interact with the operating system and change permission
(can make for some intresting game mechanics that allow user to play
around with file made by the program)
```

datetime:

```
- always useful when handling anything that has to do with time
- integral to callanders, and planners
- lots of built in functions like seeing which date comes first which
simplifies the program as you don't need to write your own comparing
function.
```

Q18: Document your continued learning experience in SENG 265.

Lecture week 7:

```
- more on c strings, and arrays
- c structs
- struct declaration and usages
- operation precedence
- midterm B solutions
- intro to python
- python history and basic syntax
- importance of pep 8
- Lab 6:
    - More on python data structures
    - sets, tuples, dictionaries, lists
    - some string slicing
    - list operations
```

Lecture week 8:

```
- python intro to f strings
- dunder variables
    - __doc__
    - __name__
- python built in functions
- useful functions
    - global() returns the dictionary implementing the current module namespace
    - locals() returns a dictionary representing the current local symbol table
    - id(obj) returns the identity of the obj
- python type hints
- python creating functions and variables
- how python handles memory allocation
- immutable and mutable objects in python
- function parameters and arguments
    - *args - positional arguments
    - **kwargs - keyword arguments
- python call by refrence vs call by assignment
- python return vs yield and their benifits
- python built in data types and collection properties
- namespaces, scope, and LEGB rule
- Lab 7:
    - malloc for c
    - linked lists in c
    - allocation and releasing heap-allocated memory
```

Lecture week 9:

- python module and function doc strings
- more information on sets and list operations
- deque and the operations associated with them
- timeit() and some information on lists and deques performences
- list, tuple, and dictionary slicing
- python slicing and how it allows for negative indexs
    - sequence[start:stop:step] where start is inclusive and all three could be positive or negative
- lazy slicing
- Software Development Life Cycle
- benifits of incremental software development
- tacit knowledge and interview stratagies
- regular expressions - re module
    - re module offers a set of functions that help with string searching and matching
    - most languages have a form of regular expressions
- Lab 8:
    - two methods to sort lists
    - sort while adding so that every time you add a node it is added into its sorted order
    - sort after linked list is done by turning into a list and using qsort()

Lecture week 10:

- how chatGPT can help with eliciting tacit knowledge
- python file I/O
- python generators using the yield keyword
- python calling functions that take in functions as parameters
    - allows for alot of versitility and flexability in function writing
- python decorators
    - allow for modifications/enchancements to a function without changing the functions code
    - can be used to make a simple logging system to see when calls are made
- module, package, library, framework
- Lab 9:
    - testing automation
    - python doctest to test functions
    - assertions to ensure that new bugs are not introduced when adding/fixing fetures
    - unit testing

Lecture week 11:

```
- midterm C solutions
- memory layout of pragrams
- heap layout
- back to c with malloc()
- dynamic storage allocation
- abstract lists like stacks, queues, and deque
- recursice data strucures
- power of APIs
- more on linked lists, doubly linked and circular lists
- defining some of these strucutres in c
- traversal and insertion into these lists
- binary trees and graphs
- Lab 10:
    - using regular expresions to search and filter
    - more on python decorators
```

Lecture week 12:

```
- introduction to assignment 4 and writing html to draw shapes
- introduction to the use of randomness in python to later draw
shapes
- more on python classes
```

Q19: How did you ecperience chatGPT?

I've notice that I've become less relient on chatGPT and how inaccurate it can be sometimes. Ive used it more so to check answers or expand my understanding, or give me a jumping point when im stuck rather then giving a full idea of what to do. This is most likly due to my better understanding of the course material and how to apply what I've learned better so my relience on chatGPT has dwindled. While I don't rely on it as much the tool is still very useful to have as I can bounce ideas off of it, and get clariffication to parts that I'm not entirely certian about.

Q20: What are your personal insights, aha moments, and epiphanies you experienced in the first and or second part of the SENG 265 course?

During the second half of SENG 265 I think the aha moment for me was when we learned Pandas in labs and linked lists in C. As for Pandas I was surprised that CSC 110 never touched on this library as it would have been quite useful in an assignment or two in that course and it also showed me that there is so much more Python to learn and that I need to seek out these reasources if I want to improve and expand my understanding. For linked lists (and trees) in C it was that moment where theses concepts that I've always seen in the contesxt of a Java program and only delt with in Java program have been presented in a different language. They provided me insight into how these concepts are no where as locked to Java as I thought and that the implementation of them were reflections of the coding languages.

Bibliography

"Literate Programming." Wikipedia, 7 June 2023, en.wikipedia.org/wiki/Literate_programming.

"Linux." Encyclopædia Britannica, 15 May 2023, www.britannica.com/technology/Linux.

"History of Unix." Wikipedia, 23 May 2023, en.wikipedia.org/wiki/History_of_Unix.

"Bash (Unix Shell)." Wikipedia, 7 June 2023, en.wikipedia.org/wiki/Bash_(Unix_shell).