CS206 – Monsoon 2018 — Homework Assignment 1

Sai Vineet Reddy Thatiparthi, Harsh Karamchandani

Collaborators: Harsh Karamchandani

Harsh Karamchandani : Go Sai Vineet Reddy - C Together : Rust and Python

Was it hard?

General We first created an algorithm and designed a pseudocode around it. The aim of this was to allow easy transltion of the algorithm into the four different languages.

User Input In Python, the user input could be directly saved in the list data type which has functionalities for manipulation that made the implementation much simpler for this problem.

Comparatively, in Go user input as space separated values required three different libraries to implement. It had to be read as a string using bufio library, split using strings library and converted to integers using strconv library. This had an impact on complexity of implementation and time taken to execute.

C had a very straight forward policy when it comes to user input. Accessing the command line inputs also was quite easy and it really helped us initialize arrays without having to specify the size beforehand. It was, however, a little more complex than Python.

Rust was by far the most difficult for us to figure out how to take input into arrays/vectors! It took us almost two hours just to figure out how to take input without the compiler throwing an error at us.

ariable Assignment In Python, variables were easily implementable due to lack of need to specify data types and implicit type conversion.

In Go, variables had to be initialized with data types and had lesser functionalities as compared to Python. In particular, Go used a data type slice which is the equivalent of Pythons list but with lesser functionality which made it more difficult to work with.

In C, we had to initialize every variable and also allocate the memory to it in case of an array. However, doing operations on the array were quite simple.

In Rust, there were so many intricacies involving variable initialization and assignment. First problem was figuring out what the whole mutable and non mutable variables were as it threw a lot of errors at us the first few times we had to compile the code. Secondly, there was quite a bit of learning involved for us to figure out how the vectors (which were similar to slices in Go) worked. This took up much of our time and we still have not completely figured it out.

Online Resources Python had extensive documentation, online resources, tutorials and Stack Overflow queries to assist in execution.

Go being a lesser known and used language had scarce resources which made it much more difficult to adapt to from scratch and implement the solution. There was a major lack of resources on user input.

C was very similar to Python in this regard. There was a huge database of online resources available in the form of Stack Overflow questions or other websites that catered to every issue we faced while coding in C.

Rust was, again, quite difficult to deal with. We only had access to the official documentation due to it being a lesser known language. Specific queries that we had were left unanswered as there were not many resources to look up from.

Adaptability Python was the easiest language to work with because of flexibility in terms of user input, list manipulation, libraries and shorthand math functions. It was also easiest to adapt to due to simple syntax which is closest to the English language, more intuitive and easier to get used to.

Go, on the other hand, was more difficult to get used to due to lack of resources and descriptive documentation and different data structures. Most structures were similarly implemented in Go such as loops, decision making and variables. One good thing about Go was that it identified variables, libraries and other things that were initialized/imported but were not used in the program at compile time which reduced redundancy of the code.

C was very easy for me (Vineet) as I had worked in it extensively before. However, as compared to Python lets say, it was not very adaptable. It could be variable initialization or a segmentation fault caused due to accessing a location in the array which was not within the specified size, we had to be very specific about everything we are doing.

Rust by far was the least adaptable. The rules with which the syntax was written probably make some sense to the developers but we were not able to figure it out and struggled every time we had to code and debut in Rust.

Speed Python : CPU Time: 0.03 sec(s), Memory: 8708 kilobyte(s), executed in 0.579 sec(s)
Go: CPU Time: 0.30 sec(s), Memory: 36288 kilobyte(s), executed in 1.149 sec(s)
C: CPU Time: 0.0001 sec(s), Memory: 8.7 kilobyte(s), execute in 0.000141 sec(s)

Work Distribution Harsh Karamchandani - Go
Sai Vineet Reddy - C
Together - Pytho