Ebad Babar

0954164

ebabar@uoguelph.ca

cis 3190 assignment 2

Ada word jumble

Synopsis:

When designing my word jumble program, I broke up the program into 3 different parts. 1 was the user, 2 was the file, and 3 and the search. For the user I decided to allow only one word at a time and after find the match, let them search again, until they press q to quit. I had a check to ensure that the size of the word was 6 or smaller, otherwise an invalid error was given. I then find all the possible anagram for the word and store it into a user array. The swap function was found on the internet at https://www.geeksforgeeks.org/write-a-c-program-to-print-all-permutations-of-a-given-string/ which after converting from java to Ada, helped get all the anagrams. Part 2 was reading in the file. I hardcoded the address and store into an array. Part 3, I ran loops comparing the 2 lists and storing the matched into a 3 list. I then checked to see if multiple instances of a word had made it to the 3rd list. I tried to remove them with limited success. And finally printed the matched and freed the user list so they can enter a new word.

Most of the assignment was mode using unbounded array data structures as it allowed the input and program to be more malleable. Looking back, it would have been better to use linked lists or queue for the user data or permutations to allow easier use, and hash tables or trees for the dictionary, as keys would make this code much faster. The declare struct was really useful as it allowed later variable initializations. This make making the code seem and feel more modularized.

While Ada isn't the worst language to make this project in, I would have preferred to have done it in Java or C. This is because C and Java are more dynamic languages, and I have more experience using them. The functionality the provide and the user friendliness is much better then in Ada.

The unbounded structures, especially for arrays and strings, made Ada a good choice. While similar to malloc, unbounded data was much easier to deal with.

Benefits of Ada include, being similar to many other languages. I can see how it influenced newer languages such as C, C++, or Java. It is very easy to pick up for someone who has even a little knowledge the big programming language. Ada also forces good programming habits as we have to declare everything at the top, while I personally prefer to declare data in groups around the functions I use them in, I can understand why Ada was developed this way. Ada doesn't have brackets and therefore forces tabs to make the code more readable. The range type for loops was handy and fun to use. I really liked the feedback from the compiler, letting us know what we did wrong, like putting a colon with out equals, or 2 equals sign.

Limitations of Ada consist of, being a strongly typed language. While not necessarily I bad thing and I did like how the complier gave feedback, its not my type. I prefer languages that let u go wild. Ada

forces many limitations on you as you code, most things have to be done a certain way even if there is a more efficient way to do.

How to run the program

Compile

gnatmake jumble.adb

to run

./jumble

Limitations of the program

Uses the small dictionary as my virtual machines did not have the longer version.

The path is hard coded. If changed the program will break as most list is hardcoded to be able to handle a certain amount of data.

It assumes the user will only enter one word at a time, or it will give invalid input error.

The program will break on windows or mac machine if the path differs or the files doesn't exist.