Robert Saunders

ID: 0891696

rsaund05@uoguelph.ca

CIS3190 A3 Writeup

How to run:

For ease of use, typing “make” in the project root directory will compile all the completed programs. Note: sieve.py already has executable privileges (chmod u+x), so it only needs to be invoked via “./sieve.py”.

Executable names:

- sieveC
- sieveFortran
- sieveAda
- sieve.py

To run any of these, type “./(name)”, where the name is the name of the version of the algorithm to run. When a program is invoked, the desired upper bound is inputted, the program generates the list of primes and outputs it to a text file that corresponds to the language you used, Ex: running ./sieveAda and inputting a number will produce a text file called “outputAda.txt”.

Language: C

This language was by far the easiest to program in, simply because its what I personally have the most experience and time spent programming in.

One benefit of programming in C was the more logical layout of for loops, allowing for calculations inside a parameter, similar to a lambda in Python. Another benefit of programming in C is the more intuitive way that you may iterate through loops, compared to other languages.

One limitation to programming in C is manual memory allocation, as it can be potentially fatal to a program, and can sometimes just be a pain to manage.

Language: Python

This language was more difficult that I had anticipated, mostly because it has been some time since I’ve last programmed in it.

One benefit of programming in Python was the automatic memory management, which is a nice bonus over C.

One limitation to programming in python is the somewhat un-flexible way that Python handles for loops, as it didn’t immediately seem intuitive for the algorithm I was employing.

(Continued on next page)

Language: Fortran 95

Of the legacy languages, Fortran was the easier of the three to program in. Once I had the loops programmed in Python, transferring them to Fortran was a relatively painless task.

One benefit of programming in Fortran was the simple file I/O functions, as they seemed more intuitive and simple than the other languages.

One disadvantage of programming in Fortran was the way the language handles loops, as it was a bit too simple to be of real value.

Language: Ada

Programming the algorithm in this language proved to not be terribly difficult once I had completed the Fortran program, as the two languages have many similarities.

One benefit of programming in Ada is its similarities to C, which makes programming something in Ada a bit easier.

A significant disadvantage of Ada is the lack of natively supported dynamic array lengths, as doing this requires a strange workaround of declaring a new data structure.

Language: Cobol

I was unable to program the algorithm in Cobol in a timely manner.

A significant disadvantage of Cobol is the unintuitive sections that need to be managed before any actual programming may begin. Another disadvantage is the unintuitive method of variable declaration.

Which language had the best usability?

By far, of the languages used, C was superior in usability. It was the easiest and most intuitive language to program in, and while unintuitive for novice C programmers, pointers (specifically heap arrays) are relatively simple to maintain.

The worst language for usability was Cobol, as it was far too finicky to be of any real value in terms of a programming language.

Was there any difference in efficiency?

There was a somewhat significant difference in efficiency between the programming languages, with Python (Specifically python3) being the longer of the languages. While I was not able to program the algorithm in Cobol in time, it has been reported by classmates to be similarly slow as Python.