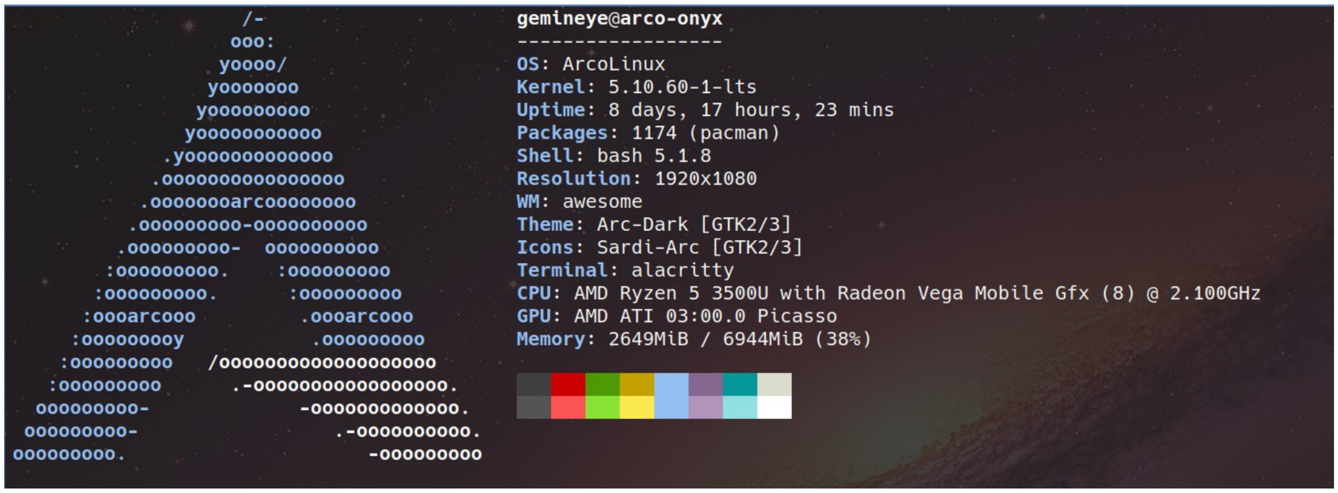
Assignment 1

Dylan Kapnias

COS 284

03/09/2021

Introduction



```
                  /-                    gemineye@arco-onyx
                 ooo:                   ------------------
                yoooo/                  OS: ArcoLinux
               yooooooo                 Kernel: 5.10.60-1-lts
              yooooooooo                Uptime: 8 days, 17 hours, 23 mins
             yooooooooooo               Packages: 1174 (pacman)
           .yooooooooooooo              Shell: bash 5.1.8
          .oooooooooooooooo             Resolution: 1920x1080
         .oooooooarcoooooooo            WM: awesome
        .ooooooooo-oooooooooo           Theme: Arc-Dark [GTK2/3]
       .ooooooooo-  oooooooooo          Icons: Sardi-Arc [GTK2/3]
      :ooooooooo.    :ooooooooo         Terminal: alacritty
     :ooooooooo.      :ooooooooo        CPU: AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx (8) @ 2.100GHz
    :oooarcooo         .oooarcooo       GPU: AMD ATI 03:00.0 Picasso
   :ooooooooy           .ooooooooo      Memory: 2649MiB / 6944MiB (38%)
  :ooooooooo   /ooooooooooooooooooo
 :ooooooooo      .-ooooooooooooooooo.
ooooooooo-             -ooooooooooooo.
ooooooooo-                .-oooooooooo.
ooooooooo.                    -ooooooooo
```

- This is the summary of the system I used to test all of the programs.

Explanation

To run and test the programs for the four languages provided I created a bash script that would run 50 lambda. Each lambda consisted of a bash command to get the start time (with nanoseconds), a loop to run the specified program 500 times, another bash command to get the end time (with nanoseconds), a one line python command (using the c flag) to subtract the end time from the start time to get the total run time and finally a command to echo the total time to the specific text file. All of the programs were pre compiled and thus only needed to be executed in the bash scripts. The commands to do this are:

- Assembly -- ./bin/hello
- C++ -- ./hello
- Java -- java hello.java
- Python -- python hello.py

Findings: (I used the CLI program datamash to filter the results)

Programming Language	Min. time (seconds)	Avg. time (seconds)
Assembly	0.097113370895386	0.10618929862976
C++	0.46184706687927	0.61982322692871
Java	261.43575620651	274.05472441673
Python	6.3145985603333	8.9558685302734

Ordered (based off of avg. time – fastest to slowest)

1. Assembly
2. C++
3. Python
4. Java

Conclusion

Based off the results I have gathered, we can assume that the more directly and closely that we interact with the physical architecture of the computer, the faster the execution time will be. The lowest level language Assembly preformed the best by and extremly large margin when compared to a high level language such as Java. I believe this is due to Java having to use the Java Virtual Machine (JVM) to execute the program, thus taking up more resources and hence the slower performance.