**CS1101: Lab 4 – Using Lists of Structs to Build Computer Networks**

Computer networking is a very important concept in the study of Electrical and Computer Engineering. Networks allow for easy exchange of data, software, and devices among all users connected to the network. When analyzing networks, it is important to investigate the performance of the computers on the network. Performance is judged by how quickly a particular computer can execute a program or set of programs. In this lab, we will be using structs and lists of structs in Scheme to create computers and computer networks for analysis.

**Lab Motivation and Goals**

* Program functions to process lists of structs

**Exercises**

An electrical engineer has decided to build his own computer. He figures that since he is building one, he might as well build several in order to form a computer network. A computer has a name (string), processor (string), speed in GHz (number), power rating in Watts (number), clock cycles per instruction rating or CPI (number), and age (number). The data definition for a computer is provided. While constructing the computers, the electrical engineer decides that he should have custom programs to run on the computers. He asks his friends who are CS majors to create programs that he can run on his new network. A program has a name (string) and a number of instructions (number). A data definition for a program is also provided. Copy the data definitions into your Lab 4 file before proceeding with the exercises.

;; a computer is a

;; (make-computer string string number number number number)

(define-struct computer (name processor speed power CPI age))

;; a program is a

;; (make-program string number)

(define-struct program (name instructions))

;; a network is either

;; empty, or

;; (cons computer network)

1. Write the template for functions over networks and computers.
2. Create a sample network of computers.
3. Write a function **count-processors** that consumes a network of computers and produces a number representing the total number of computers on the network with a speed greater than 2 GHz.
4. Write a function **average-age** that consumes a network of computers and produces a number representing the average age of all computers on the network. (Hint: It is often beneficial to create helper functions when the list must be processed to obtain more than one quantity.)
5. Write a function **upgrade** that consumes a network and produces a network where all computers of age greater than 3 have been updated so that their clock speed is 15% faster and their CPI is 5% greater to compensate.

**Everybody should be able to finish up to this point.**

1. Write a function **remove-bad** that consumes a network and produces a network that is the same as the original network; however, all computers with a speed less than 3 GHz and a power rating less than 500 Watts are removed.
2. This problem will be completed in three-steps. When working with computer networks, it is often important to know which computer is fastest on the network. The overall speed of a computer is determined by how fast it can run a given program.
3. Write a function **execute-time** that consumes a computer and a program and produces a number representing the execution time of that program on the computer. The equation for execution time is as follows: $Execution Time= \frac{CPI×instruction count}{clock speed}$.
4. Write a function **find-faster** that consumes two computers and a program and produces the computer that executes the given program faster. \*Note: This program will either receive two computers or a computer and a symbol (if a network is empty). It should never receive 2 symbols. If one of the computers is actually a symbol return the other computer.
5. Write a function **find-fastest** that consumes a network and a program and produces the computer that executes this program fastest or ‘none if there are no computers in the network.
6. Write a function **make-portable** that consumes a network and a string representing the name of a computer. It returns the same network; however, the computer with the inputted name will have its power-rating reduced by 50%, since now it is a portable computer.
7. AMD decides to sponsor the electrical engineer’s computer network. In addition to funding, they decide to replace all computers on the network with AMD equivalents.

Write a function **replace-all** that consumes a network of computers and produces the same network with all computer processors converted to AMD. AMD cannot produce replacement processors for computers greater than 4 years old; therefore, these computers should be removed from the network.

 **Well Done! The last problem requires a bit more thought, but if you got this far you can do it!**

1. In this problem, we will be rating the performance of our network using a process called benchmarking. In benchmarking, a computer is rated on how well it performs a series of programs by comparing the execution times of the programs on the computer being ranked to the execution time on a reference computer. This rating is called a benchmark. In this problem, we would like to calculate the benchmark of the entire network, which is the average of the benchmarks for each individual computer on the network.

As a mathematical recap of the above paragraph, the benchmark for a single computer would be calculated as follows:

$Computer Benchmark=\frac{1}{n}×\sum\_{i=1}^{n}\frac{Reference Execution Time}{Execution Time of Program\left(i\right) on Computer}$.

The benchmark for the entire network would then be as follows:

$$Benchmark=\frac{1}{n}\sum\_{i=1}^{n}Computer Benchmark(i)$$

Write a function **benchmark** that consumes a network of computers, a list of programs, and a number representing the execution time of the programs on a reference computer. It should produce a number representing the benchmark of the computer network. (Hint: Look back to other problems in this lab and see if you can reuse code. This is a good habit to get into and will save you a lot of time in more complicated projects you may come across.)