Assignment 2

Submission date: Sunday, May 19

NOTE: For benchmarking, you need not use BenchmarkTools.jl . Instead, you can use @time. (Which is easier to use)
Eg. @time sin(1) # prints time taken for computing sin(1)

1. BenchmarkTools package in Julia makes performance tracking of Julia code easy by supplying a framework for writing and running groups of benchmarks as well as comparing benchmark results. In this exercise you will be writing your own sort function and benchmark it against the sort function provided by Julia.

   First create a random array with 2^20 elements. Then, write your sort function to sort the contents of the array. Benchmark your custom sort against the default sort function in Julia.

2. Julia makes unit testing easy by Test.jl package. With this package, programmers can easily check the correctness of their program. In this exercise, you will be writing a custom function to find square root of a number using binary search. For checking the correctness of your implementation, you will be comparing your custom square root function with Julia's built-in sqrt by using Test.jl package.

   PS: Refer this link for the algorithm for finding sqrt using binary search.

3. Define a mutating function called array_sqrt, which takes the sqrt of all of it's elements. You should use the custom functions you defined in previous exercise. Also, You should use the assignment format of function declaration, i.e, f(x) = #function declaration.

   Example for assignment format:
   square(x) = x^2 # function to find square in assignment format.

   Benchmark array_sqrt on a random array with 2^20 elements against the builtin sqrt function.

4. A = [[2 1]; [4 3]; [6 5]] defines a 3X2 2-dimensional array. Perform sqrt.(A) and sort.(A). Explain the output. Now let A = [[2, 1], [4, 3], [6, 5]]. Perform sqrt.(A) and sort.(A). Explain the output.