CSCI 1120 (Low-Level Computing), Spring 2011

Homework 2

Credit: 20 points.

1 Reading

Be sure you have read the assigned readings for classes through 2/21.

2 Programming Problems

Do the following programming problems. You will end up with at least one code file per problem. Submit your program source (and any other needed files) by sending mail to `bmassing@cs.trinity.edu`, with each file as an attachment. Please use a subject line that mentions the course number and the assignment (e.g., "csci 1120 homework 2"). You can develop your programs on any system that provides the needed functionality, but I will test them on one of the department's Linux machines, so you should probably make sure they work in that environment before turning them in.

1. (10 points) Newton's method for computing the the square root of a non-negative number x starts with an initial guess r_0 and then repeatedly refines it using the formula

$$r_n = (r_{n-1} + (x/r_{n-1}))/2;$$

Repetition continues until the absolute value of $(r_n)^2$ - x is less than some specified threshold value. An easy if not necessarily optimal initial guess is just x.

Write a C program that implements this algorithm and compares its results to those obtained with the library function `sqrt`. Since we haven't talked yet about how to read values from a human user, put the code to compute and print results in a function and call it with different inputs, as we did in the program to compute the roots of a quadratic equation. You might also find it interesting to print the number of iterations. Also have the function print an error message if it is called with invalid (negative) input.

Sample output (for quickly-chosen inputs — you may want different ones):

```
square root of 0:
with newton's method (threshold 0.1):  0 (0 iterations)
using library function:  0
difference:  0

square root of -4:
unable to compute square root of negative number

square root of 4:
with newton's method (threshold 0.1):  2.00061 (3 iterations)
using library function:  2
```

```
difference:  0.000609756

square root of 2:
with newton's method (threshold 0.1):  1.41667 (2 iterations)
using library function:  1.41421
difference:  0.0024531

square root of 2:
with newton's method (threshold 0.01):  1.41667 (2 iterations)
using library function:  1.41421
difference:  0.0024531

square root of 2:
with newton's method (threshold 1e-06):  1.41421 (4 iterations)
using library function:  1.41421
difference:  1.59482e-12
```

You may find the library function `fabs` (which computes the absolute value of a `double`) useful.

2. (10 points) Complete the sample sort program we wrote in class by filling in the `sort` function. (You can find it linked from the course "sample programs" page here[1].)

   It's completely up to you which sorting algorithm to implement, though I'm inclined to recommend that you just do one of the simple-but-slow ones (e.g., bubble sort or selection sort). We may do at least one of the faster recursive ones (quicksort and mergesort) later in the semester. Please say in comments at the start of the program which sorting algorithm you're implementing. Feel free to make other alterations to the program (e.g., adding more functions, though depending on your choice of algorithm you might want to just do everything in `sort`).

[1] `http://www.cs.trinity.edu/~bmassing/Classes/CS1120_2011spring/SamplePrograms/Programs/sort.c`