Lab 6

Thursday, October 26, 2023

You will get 1 point for attending the lab, making an effort to work on the problems, and **discussing your work with a lab instructor during the lab**.

If you finish during the lab, demonstrate your working code to a lab instructor and they will give you the full 2 points. If you do not finish during the lab, discuss what you've done with a lab instructor and they will give you an opportunity to finish outside of the lab and attend office hours to demonstrate your solution.

Practice Problems. Write a function `clock_game` that takes as input a price between 1 and 999 and prints the best sequence of guesses that a contestant can make in the clock game for the given price (as discussed in lecture). For example, calling `clock_game(625)` should print `500 750 625` to the screen.

The function will have the following prototype:

```
// clock_game(n) prints the optimal sequence of guesses that a player can
//   make when n is the price they are trying to guess
// requires: 1 <= n <= 999
void clock_game(const int n);
```

What is the big O running time of `clock_game` in terms of `n`?

Assessment Problem. Write a function `first_positive` that takes as input an array of positive and negative integers and returns the first positive integer. In the array given to `first_positive` all negative integers will appear before all positive integers (and 0 will not appear). For example, the array $[-5, -2, -9, 8, 1]$ satisfies this condition and 8 is the first positive integer.

Your function MUST run in $O(\log n)$ operations where n is the length of the array. You will need to explain to the lab instructor why your code has this running time.

The function will have the following prototype:

```
// first_positive(A, n) returns the first positive integer in the array A
//   or 0 if no positive integer is in A
// requires: A is of length n >= 1 and A does not contain 0
//           all negative entries in A appear before all positive entries
int first_positive(const int A[], const int n);
```

In `main` use `assert` to give at least five tests of `first_positive`, all of which meet the requirement on the input as specified in the function's purpose statement. Include tests for any edge cases.