

## CMPT125, Fall 2022

Lab exam - 12:30pm-1:20pm

Thursday, November 10, 2022

You need to implement the functions in ***labexam.c***.

Submit only the **.c** file to Coursys

Coursys Assignment - **Lab Exam D105-D106 12:30-13:20.**

You have 50 minutes to solve all 3 problems.

The maximal score is 20 points.

The exam will be graded both **automatically** and by **reading your code**.

You can run your code using

```
>> make
```

```
>> ./run_test
```

**Correctness:** Make sure that your code compiles without warnings/errors, and works as expected.

**Readability:** Your code should be readable. Add comments wherever necessary. If needed, write helper functions to break the code into small, readable chunks.

**Compilation:** Your code **MUST** compile in CSIL with the Makefile provided. If the code does not compile in CSIL, the grade on the assignment is 0 (zero). Even if you can't solve a problem, make sure it compiles.

**Helper functions:** If necessary, you may add helper functions to the .c file.

**main() function:** do not add main(). Adding main() will cause compilation errors, as the main() function is already in the test file.

**Using printf()/scanf():** Your function should not have any unnecessary printf() statements. They may interfere with the automatic graders.

**Warnings:** Warnings during compilation will reduce points. More importantly, they indicate that something is probably wrong with the code.

**Testing:** An example of a test file is included.

Your code will be tested using the provided tests as well as additional tests.

You are *strongly encouraged to write more tests* to check your solution is correct, but you don't need to submit them.

### Question 1 [6 points]

Write a function that gets a number  $n$ , and returns an array with  $n$  strings, where the  $i$ 'th string consists of exactly  $i$  asterisks. Make such that the array and all strings are allocated on the heap.

```
// the function gets an int  $n > 0$ 
// returns the array ["", "*", "**", "***", ... "*****....*"]
char** get_array_of_asterisks(int  $n$ );
```

### Question 2 [7 points]

Write a function that gets an int  $n > 0$ , and returns the sum  $\text{fib}(0) + \text{fib}(1) + \text{fib}(2) + \dots + \text{fib}(n)$ . Recall, the fibonacci sequence is defined as:  $\text{fib}(0) = 0$ ,  $\text{fib}(1) = 1$ ,  $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$ . That is, the sequence is: **0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55...** For example:

- On input  $n=3$  the function returns  $1+1+2=4$
- On input  $n=6$  the function returns  $1+1+2+3+5+8=20$
- On input  $n=10$  the function returns  $1+1+2+3+5+8+13+21+34+55=143$

The function needs to run under one second for all  $n < 40$

```
// the function gets an int  $n > 0$ 
// and returns the sum of  $\text{fib}(1) + \text{fib}(2) + \dots + \text{fib}(n)$ 
long sum_fib_n(int  $n$ );
```

### Question 3 [7 points]

Write a function that gets a queue and a predicate, and removes all elements from the queue on which  $\text{pred}$  returns false. For example:

- Suppose that the queue was  $1 \leftarrow 2 \leftarrow 8 \leftarrow 5 \leftarrow 8 \leftarrow 7$ , and  $\text{pre}$  is  $\text{is\_even}()$
- Then, after applying the function, the queue becomes  $2 \leftarrow 8 \leftarrow 8$

See the file `lib/queue.h` for the functions you can use,

```
// the function gets a queue and a predicate
// and removes all elements from the queue
// on which  $\text{pred}$  returns false
// all other elements are kept in the same relative order
void queue_filter(queue_t*  $q$ , bool(* $\text{pred}$ )(int));
```