

CMPT125 D103-104, Fall 2025
Lab exam - Thursday, 10:30am - 11:20am
November 20, 2025
You need to implement the functions in ***labexam.c***.
Submit only the **.c** file to Coursys
Coursys Assignment - **Lab Exam D103-D104**.

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 string and returns a new copy of the given string.

```
// the function gets a string
// returns a new copy of the given string
char* str_make_copy(char* str);
```

Question 2 [7 points]

Write a function that gets an int a , a function f , and another int n . It returns an array (allocated on the heap) of length n , where

- $\text{array}[0] = a$
- $\text{array}[1] = f(a)$
- $\text{array}[2] = f(f(a))$... and so on.
- In general, $\text{array}[k] = f(f(f(\dots(a))))$, where f is composed k times for all $k=0\dots n-1$.

For example,

- On input $a = 0$, $f = \text{plus1}$, and $n=6$, the function returns $[0, 1, 2, 3, 4, 5]$.

```
// The function gets an int a, a function f, and another int n.
// It returns an array (allocated on the heap) of length n
// such that for all  $k=0\dots n-1$  we have  $\text{array}[k] = f(f(f(\dots(a))))$ ,
// where  $f$  is composed with itself  $k$  times
int* f_compositions(int a, int n, int(*f)(int));
```

Question 3 [7 points]

Write a function that gets a linked list of ints, removes the last element from the list and returns it. You may assume the list is not empty. For example,:

- On input $3 \rightarrow 2 \rightarrow 8 \rightarrow 10 \rightarrow 5$
The function returns 5, and the resulting list will be $3 \rightarrow 2 \rightarrow 8 \rightarrow 10$.

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

```
// the function gets a Linked List of ints
// removes the last element from the list and returns it
// Assumption: the list is not empty
int remove_last(LL_t* list);
```