

CMPT125, Spring 2022
Lab exam - D105-D106

Thursday, March 17, 2022, 11:30am-12:20pm
You need to implement the functions in **labexam.c**.

Submit only the .c file to Coursys
Coursys Assignment - Lab Exam 11:30-12:20

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
```

Submit only labexam.c: Please make sure to submit the file to the *correct section* in Coursys.

Correctness: Your file must compile without warnings/errors, and work 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 *labexam.c* file.

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

Using printf()/scanf(): Your function should have no 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 the function that gets a string and returns the most frequent char in the string.
If there is more than one most frequent chars, the function returns the one that appears first in the array

If str is the empty string, the function returns '\0'

For example:

- `most_frequent_char("ABCDAA")` returns 'A' - 'A' appears three times
- `most_frequent_char("EAACDEEDE")` returns 'E' - 'E' appears four times
- `most_frequent_char("ABCD CDB")` returns 'B'. - several chars appear twice, and 'B' is the first among them.

// the function gets a string and returns the most frequent char in it.

// if there are multiple chars, returns the one that appears first

// if str is the empty string, the function returns '\0'

```
char most_frequent_char(const char* str);
```

Question 2 [7 points]

Write a function that gets an array of length n and a function foo, and counts the number of indices i such that `foo(i)==ar[i]`. For example:

- `arra_to_LL([0,3,-4,0,9,7], n=6, plus_two)` returns 2, because `plus_two(1)=3`, and `plus_two(5)=7`.

// the function gets an array of length n and a function foo

// it returns the number of indices i such that `foo(i)==ar[i]`

```
int count_foo(int* ar, int n, int(*foo)(int));
```

Question 3 [7 points]

Write a function that gets a linked list and a node.

If node is in the list, the function deletes the node from the linked list, releases the memory of the node and returns 1.

If node is not in the list, the function does not change the list and returns 0.

See `lib/LL.c` and `lib/LL.h` for details.

// gets a Linked List and a node

// it deletes the node from the Linked List, frees the node and returns 1

// otherwise, the function does not change the List and returns 0

```
int LL_delete_node(LL_t* List, node_t* node);
```