Lab03
Deadline: 11:59PM Jan 27

Requirements
Write a function that draws a line following coordinates. For this lab, there is no skeleton file provided, so you have to write your main function that interacts with files (descriptions to follow).

You are given an input file which contains a set of (x, y) coordinates. These coordinates are written, so that a line segment is connected from the coordinate in the preceding line to the coordinate. For example, let's assume that input.txt contains the following coordinates:

```
>> cat input.txt
0,0
0,1
1,1
1,0
0,0
```

The first coordinate is at coordinate 0,0 and the next coordinate is at 0,1. There is a line segment of a length 1 from 0,0 to 0,1 in the xy plane. The next line segment moves from 0,1 to 1,1 with a length of 1. This continues until line #5 is processed. In your output file (descriptions to follow), the line segments are represented by a sequence of *'s. From the previous input file, the output file will contain the following:

```
>>cat output.txt
**
**
```

Let's take a look at another example where we are drawing a rectangle this time.

```
>> cat input.txt
0,0
0,4
2,4
2,0
0,0
```

```
>> cat output.txt
***
*  *
*  *
*  *
***
```

Finally, your input and output can be the following as well.

```
>> cat input.txt
2,2
4,0
```

0,0
2,2

```
>> cat output.txt
   *
  * *
*****
```

Our program will read an input from a file and write an output to a file. There will not be any output to the console. In order to read/write from/to a file, we need to use file I/O libraries. I am not restricting on what functions you are using as long as they are a part of a standard library mentioned in learning hub. Here (https://www.tutorialspoint.com/cprogramming/c_file_io.htm) is a reference that might be useful to enable this.

Detailed specifications are below.

1. The command line argument is in this order <executable> input_file output_file
2. You are guaranteed that your line segment will be strictly vertical, horizontal, or at 45 degree angle. In our example, the triangle was at 45 degrees.
3. You will write a main function as well as other helper functions.
4. Your input is in X,Y format and order. No spaces allowed between , and Y coordinate.
5. You can assume that the coordinates can be any integer values. However, the test case will not request large memory where malloc fails due to insufficient free memory in the system.
6. There will not be line segment crossings.
7. The line segment will always come back to the original starting position.

Restrictions

- As you might have noticed, the size of your shape differs depending on your input. You are not allowed to use [] anywhere in your code except when you are passing command line arguments. This means argv[x] is the only place where you can use []. You can use argv[1] and argv[2] only once.
- You must use malloc or a variant of malloc to allocate necessary memory. You are not allowed to allocate arbitrarily large array. In our triangle example, the malloc is supposed to only allocate 15B. Your malloc must be the smallest to be able to draw the shape. Your draw does not need to start from 0,0. Any malloc that uses extra space will get automatic 50% penalty in the overall grade.
- Only use standard libraries listed in Learning Hub

Grading

Any grading failure due to not following specifications will result in 0. For full marks this week, you must:

- (1 point) Correctly use git/GitHub and the repository following the lab handout including the correct A number format
- (6 point) Generate a correct solution to the problem(s) in this lab

Submission Files

- You must deliver only one .c file named: **lab3.c**

- The file that you submit should be a .c file (not .txt, not .cpp or any other type)
- lab3.c (do not capitalize)
- AXXXX.txt (empty file, but with your A number as file name. Make sure to include 0's and match this A number with your A number in learning hub)
- In Github, there must be only lab3.c and AXXX.txt file. Nothing else, so make sure to remove all other files there.
- Github: TBD