



C Piscine

Rush 01

*Summary: This document is the subject for Rush01 of the C Piscine @ 42.*

*Version: 7.2*

# Contents

<b>I</b>	<b>Instructions</b>	<b>2</b>
<b>II</b>	<b>Foreword</b>	<b>3</b>
<b>III</b>	<b>subject</b>	<b>5</b>
<b>IV</b>	<b>Annexe</b>	<b>7</b>
<b>V</b>	<b>Submission and peer-evaluation</b>	<b>9</b>

# Chapter I

## Instructions

- The group WILL be registered to defense automatically.
- Do not cancel it, you won't get a second one.
- Any question concerning the subject would complicate the subject.
- You have to follow the submission procedures for all your exercises.
- This subject could change up to an hour before submission.
- The program must compile with the following flags: `-Wall -Wextra -Werror`; and uses `cc`.
- If your program doesn't compile, you'll get 0.
- Your program must be written in accordance with the Norm. If you have bonus files/functions, they are included in the norm check and you will receive a 0 if there is a norm error inside.
- You must do the project with the imposed team and show up at the defense slot you've selected, with all of your teammates.
- Your project must be done by the time you get to defense. The purpose of defense is for you to present and explain any and all details of your work.
- Each member of your group must be fully aware of the works of the project. Should you choose to split the workload, make sure you all understand what everybody's done. During defense, you'll be asked questions, and the final grade will be based on the worst explanations.
- Gathering the group is your responsibility. You've got all the means to get in contact with your teammates: phone, email, carrier pigeon, spiritism, etc. So don't bother blurping up excuses. Life isn't always fair, that's just the way it is.
- However, if you've really tried everything one of your teammates remains unreachable : do the project anyway, and we'll try and see what we can do about it during defense. Even if the group leader is missing, you still have access to the submission directory.
- Enjoy !

# Chapter II

## Foreword

Here are some cool quotes from some random movies :

1. "Find a truly original idea. It is the only way I will ever distinguish myself. It is the only way I will ever matter."

-A Beautiful Mind

2. "You don't have to be the bad guy. You are the most talented, most interesting, and most extraordinary person in the universe. And you are capable of amazing things. Because you are the Special. And so am I. And so is everyone. The prophecy is made up, but it's also true. It's about all of us. Right now, it's about you. And you... still... can change everything."

-The Lego Movie

3. "Sometimes it is the people who no one imagines anything of who do the things that no one can imagine." -The Imitation Game

4. "There should be no boundaries to human endeavor. We are all different. However bad life may seem, there is always something you can do, and succeed at. While there's life, there is hope."

-The Theory of Everything

5. "Just because someone stumbles and loses their path, doesn't mean they're lost forever."

-X-Men Days of Future Past

6. "Where we're going we don't need roads"

-Back to the futur

7. "I'm bad, and that's good. I will never be good, and that's not bad. There's no one I'd rather be than me."

-Wreck-it Ralph


8. "KA-ME-HA-ME-HAAAAAAAAAAAA"

-Various movies

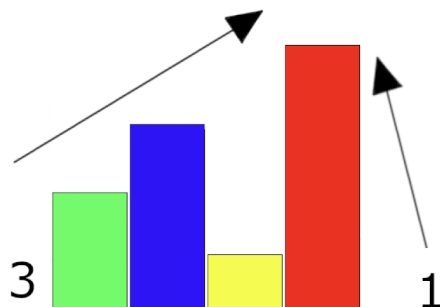
Movie culture won't help for this project even if it's important.

# Chapter III

## subject

	Exercise 00
Rush-01	
Turn-in directory : <i>ex00/</i>	
Files to turn in : <b>All necessary files</b>	
Allowed functions : <b>write, malloc, free</b>	

- Your source code will be compiled as follows: `cc -Wall -Wextra -Werror -o rush-01 *.c`
- Your submission directory must have all files required to compile your program.
- Create a program that solves the following problem:
- Given a map of 4x4, place boxes of height 1 to 4 on each available square in a way that every row and column sees the correct number of boxes from each the possible points of view (left/right for rows, up/down for columns).
- Ex: The box of height 3 will hide the box of height 1 from the left, so we have 3 visible boxes, and only one from the right, because the box of height 4 is hiding everything.



- Each of the views (2 per row and 2 per column) will have a given value. Your program must place the boxes correctly, while making sure each row and column only has one box of each size.
- Your output will contain the first solution you encounter
- Here's how we'll launch your program :

```
> ./rush-01 "col1up col2up col3up col4up col1down col2down col3down col4down row1left row2left  
row3left row4left row1right row2right row3right row4right"
```

- (cf. annex 1)
- "col1up" is the value for the left column upper point of view. Each of these represent a characters string of values ranged between '1' and '4'.
- This is the ONLY acceptable input for your program. Any other input must be considered an error.
- Here's an example of intended input/output for a valid set.

```
./rush-01 "4 3 2 1 1 2 2 2 4 3 2 1 1 2 2 2" | cat -e  
1 2 3 4$  
2 3 4 1$  
3 4 1 2$  
4 1 2 3$
```

- (cf. annex 2 and 3)
- In case of error or if you can't find any solutions, display "Error" followed by a line break.
- If you want bonus points, you may try to handle other map size (up to 9x9 !!!!).
- As usual if a bonus works, but the mandatory one fails the tests, you'll get 0.

# Chapter IV

## Annexe

What follows is an artistic view of your program. Obviously, you need to turn in a program as described in the previous chapter.

These representations' only goal is to help you understand the project.

- Annex 1:

	col1up	col2up	col3up	col4up	
row1left					row1right
row2left					row2right
row3left					row3right
row4left					row4right
	col1down	col2down	col3down	col4down	

- Representation of your program using col\_up, col\_down, row\_left and row\_right
- Annex 2:

	4	3	2	1	
4					1
3					2
2					2
1					2
	1	2	2	2	

- By replacing col\* et row\*, we get this.



- Annexe 3:

	4	3	2	1	
4	1	2	3	4	1
3	2	3	4	1	2
2	3	4	1	2	2
1	4	1	2	3	2
	1	2	2	2	

- Your program must fill in the blanks inside using the rules given in the first part.

# Chapter V

## Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your files to ensure they are correct.

As these assignments are not verified by a program, feel free to organize your files as you wish, as long as you turn in the mandatory files and comply with the requirements.



You need to return only the files requested by the subject of this project.