C Swimming pool

Rush 00

Summary: This document corresponds to the statement of Rush 00 of the C Piscine of 42.

Version: 5.2

General index

Chapter I

Instructions

- The group will be registered AUTOMATICALLY for the evaluation
- Do not cancel the evaluation, you will not have another one.
- Any request for additional clarification on the instructions could lead to a subsequent complication of the statement.
- You must respect the delivery procedure for all your exercises.
- This statement may change up to one hour before delivery.
- The program must be compiled with the -Wall -Wextra -Werror flags and use cc.
- If your program does not compile, you will have 0.
- You will have to manage errors consistently. Feel free to display an error message or return control to the user.
- The program must be written according to the Standard. If you have bonus files/functions, these are included in the Standard check and the grade will be 0 if they fail to comply.

- Rush exercises should be performed in groups of 2, 3 or 4.
- The rush number imposed on your group will follow the following rule: alphabetical range of the first letter of the team leader's login (from 1 to 26) module 5.
- Therefore, you must carry out the project together with your team members and you must all appear for the evaluation at the agreed time.
- The project must be finished when it is submitted for evaluation. The purpose of the evaluation is for you to present and explain your work in detail.
- Each member of the group will have to be perfectly aware of the work carried out. If you choose to divide the work, make sure you all understand what everyone else has done. During the evaluation, questions will be asked and the group's grade will be based on the worst explanation.
- Obviously, it is your responsibility to assemble the work team. You have all the means available to contact the rest of the team members: telephone,

e-mail, carrier pigeon, seance, etc. No excuse will be accepted regarding group problems. Life is unfair, but it is what it is.

- In any case, if after really trying everything you can't contact a member of your group: rush and turn it in. We will try to find a solution during the evaluation. Even if the missing member is the team lead-er, you all have access to the repository.

- Optionally, you can answer several statements to obtain possible extra points or use program arguments to test your function.

It is absolutely mandatory to have answered perfectly the mandatory statements to access the extra statements. If a extra statement works, but the mandatory one does not, you will have a 0.

Chapter II

Introduction

Here are the lyrics from the credits of Pinky and the Brain:

Pinky: Brain, what are we doing tonight?
Brain: The same thing we do every night, Pinky: try to conquer the world!

They are Pinky and the Brain
Pinky and the Brain
One is a genius
The other one is not sane
From the laboratory they are
With genes inserted They
are Pinky
They are Pinky and the Brain, bro, bro, bro, bro,
bro, bro, bro!

Before dawn
They will develop their plan
And when the sun rises
They will conquer the world

They are Pinky and the Brain
Pinky and the Brain
Your motivation
It's easy to explain To prove
their worth The world they
will conquer They are Pinky
They are
Pinky and the Brain, bro, bro, bro, bro, bro, bro,
bro!

Instead of conquering the world, it is better that you dedicate yourself to conquering this rush!

Chapter III

Statement

	Exercise: 00
rush0X	
Delivery directory: ex00/ Files to	
deliver: main.c, ft_putchar.c, rush0X.c Authorized functions: write	

- Files to deliver: main.c, ft_putchar.c and rush0X.c, where 0X will correspond to the rush number. For example, rush00.c.
- All three files will be compiled together.
- The ft_putchar.c file must contain the ft_putchar function.
- Main.c example:

```
int {            main()

            rush(5, 5); return
            (0);
}
```

- Therefore, you must write the rush function that receives as parameters two variables of integer type that are called respectively x and y.
- Your rush function will have to display a rectangle of x characters on the screen width and height characters.
- Your function should not crash or go into an infinite loop.
- Your main will be modified during the evaluation to be able to change the call parameters to the rush function. For example, these types of things will be tested:

```
int {            main()

          rush(123, 42);
          return (0);
}
```

Chapter IV

Rush 00

- rush(5,3) should show this:

```
$>./a.out
o---o
|   |
o---o
$>
```

- rush(5, 1) should show this:

```
$>./a.out
o---o
$>
```

- rush(1, 1) should show this:

```
$>./a.out
o
$>
```

- rush(1, 5) should show this:

```
$>./a.out
o
|
|
|
o
$>
```

- rush(4, 4) should show this:

```
$>./a.out
o--o
|  |
|  |
o--o
$>
```

Capítulo V

Rush 01

- rush(5,3) should show this:

```
$>./a.out /***\

* *
\***/ $>
```

- rush(5, 1) should show this:

```
$>./a.out /***\
$>
```

- rush(1, 1) should show this:

```
$>./a.out / $>
```

- rush(1, 5) should show this:

```
$>./a.out /

*
*
*
\
$>
```

- rush(4, 4) should show this:

```
$>./a.out /**\

* *
* *
\**/
$>
```

Chapter VI

Rush 02

- rush(5,3) should show this:

```
$>./a.out
ABBBA
B B
CBBBC
$>
```

- rush(5, 1) should show this:

```
$>./a.out
ABBBA
$>
```

- rush(1, 1) should show this:

```
$>./a.out A $>
```

- rush(1, 5) should show this:

```
$>./a.out A

B
B
B

C $>
```

- rush(4, 4) should show this:

```
$>./a.out ABBA

B B
B B
CBBC
$>
```

Chapter VII

Rush 03

• rush(5,3) should show this:

```
$>./a.out
ABBBC
B B
ABBBC
$>
```

• rush(5, 1) should show this:

```
$>./a.out
ABBBC
$>
```

• rush(1, 1) should show this:

```
$>./a.out A $>
```

• rush(1, 5) should show this:

```
$>./a.out A

B
B
B
A
$>
```

• rush(4, 4) should show this:

```
$>./a.out ABBC

B B
B B
ABBC
$>
```

Chapter VIII

Rush 04

• rush(5,3) should show this:

```
$>./a.out
ABBBC
B B
CBBBA
$>
```

• rush(5, 1) should show this:

```
$>./a.out
ABBBC
$>
```

• rush(1, 1) should show this:

```
$>./a.out A $>
```

• rush(1, 5) should show this:

```
$>./a.out A

B
B
B

C $>
```

• rush(4, 4) should show this:

```
$>./a.out ABBC

B B
B B
CBBA
$>
```

Chapter IX

Delivery and evaluation

Deliver your project to your Git repository as usual. Only the work delivered to the repository will be evaluated during the defense. Don't hesitate to check the file names several times to verify that they are correct.

Since this project is not verified by a program, you can organize your files as you see fit, as long as you submit the required files and they meet the requirements.

You only need to deliver the files required by the statement of this project.