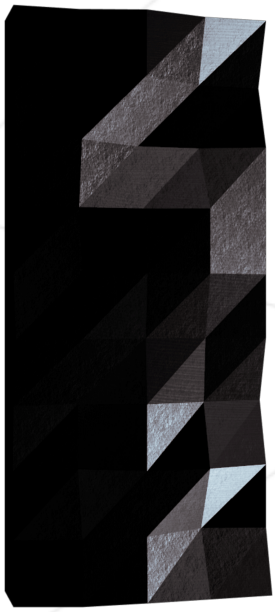


C Piscine's final Project

BSQ

Summary: Will you find the biggest square ?

Version: 8

Contents

Chapter I

Instructions

- This subject could change up to an hour before submission.
- Any question concerning the subject would complicate the subject.
- You have to follow the submission procedures for this project.
- 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 will have to handle errors coherently. Feel free to either print an error message, or simply return control to the user.
- You project must be done by the time you get to defense. The purpose of defense is for you to present and explain 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 the defense, you'll be asked questions and the final grade will be based on the worst explanations.
- It goes without saying, but 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 fair, that's just the way it is.

This project will be checked and graded by a program called Moulinette. Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated and there is no way to negotiate with it. So if you want to avoid bad surprises, be as thorough as possible.

Chapter II

Foreword

Extraits de Life, The Universe, and Everything :

« Important facts from Galactic history, number one:
(Reproduced from the Siderial Daily Mentioner's Book of popular Galactic History.)
The night sky over the planet Krikkit is the least interesting sight in the entire Universe. »

« The Krikkit Wars belonged to the ancient past of the Galaxy, and Zaphod had spent most of his early history lessons plotting how he was going to have sex with the girl in the cybercubicle next to him, and since his teaching computer had been an integral part of this plot it had eventually had all its history circuits wiped and replaced with an entirely different set of ideas which had then resulted in it being scrapped and sent to a home for Degenerate Cybermats, whither it was followed by the girl who had inadvertently fallen deeply in love with the unfortunate machine, with the result (a) that Zaphod never got near her and (b) that he missed out on a period of ancient history that would have been of inestimable value to him at this moment. »

« The game you know as cricket, [Slartibartfast] said, and his voice still seemed to be wandering lost in subterranean passages, is just one of those curious freaks of racial memory which can keep images alive in the mind aeons after their true significance has been lost in the mists of time. Of all the races on the Galaxy, only the English could possibly revive the memory of the most horrific wars ever to sunder the Universe and transform it into what I'm afraid is generally regarded as an incomprehensibly dull and pointless game. »

« Although it has been said that on Earth alone in our Galaxy is Krikkit (or cricket) treated as a fit subject for a game, and that for this reason the Earth has been shunned, this does only apply to our Galaxy, and more specifically to our dimension. In some of the higher dimensions, they feel they can more or less please themselves, and have been playing a peculiar game called Brockian Ultra-Cricket for whatever their transdimensional equivalent of billions of years is. »

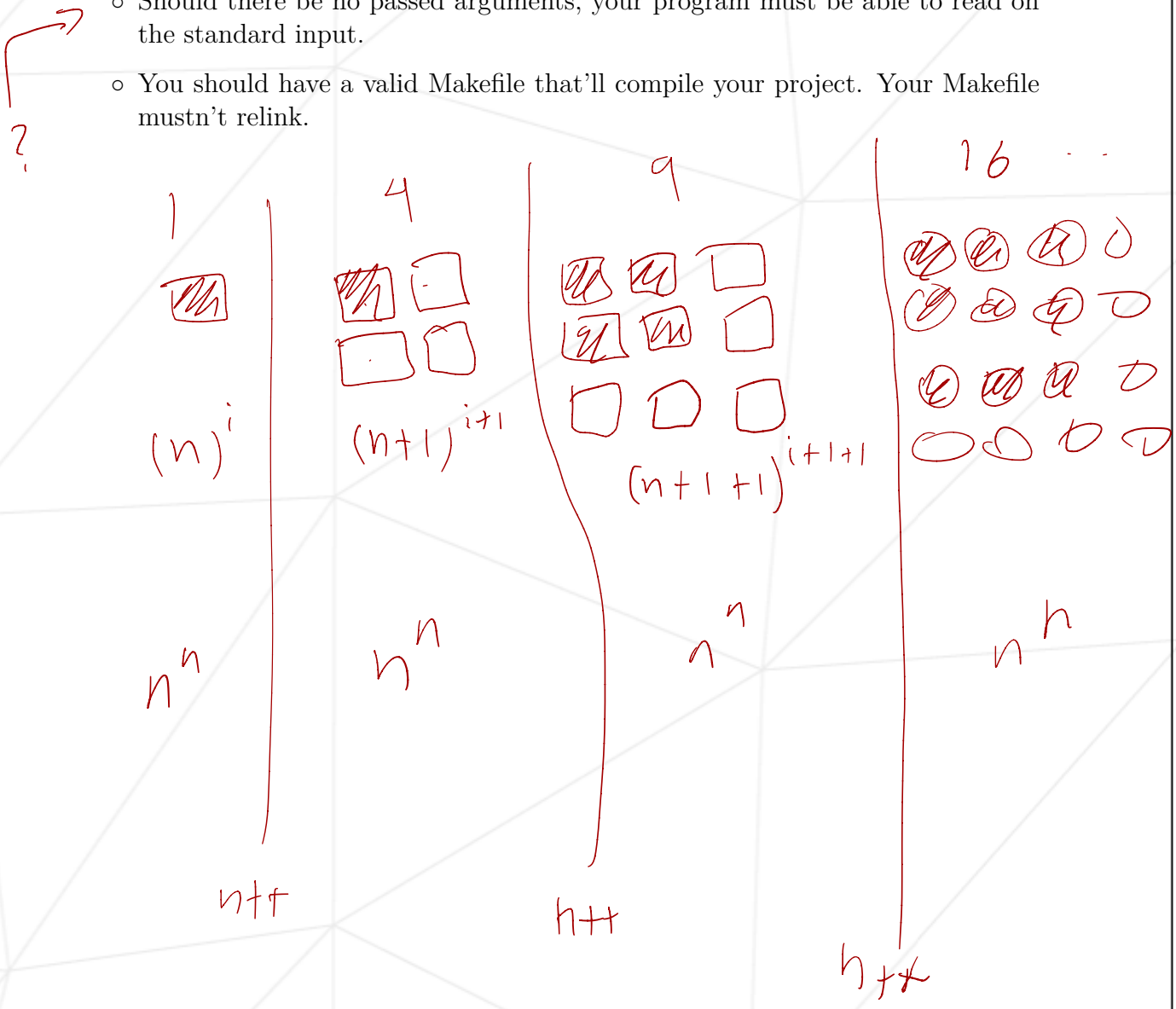
Chapter III

Subject

Program name	`bsq`
Turn in files	`Makefile and all the necessary files`
Makefile	`Yes`
Arguments	`File(s) in which to read the square`
External functs.	`open, close, read, write, malloc, free, exit`
Libft authorized	`Not applicable`
Description	`Write a program that draws and print the biggest possible square in the given area`

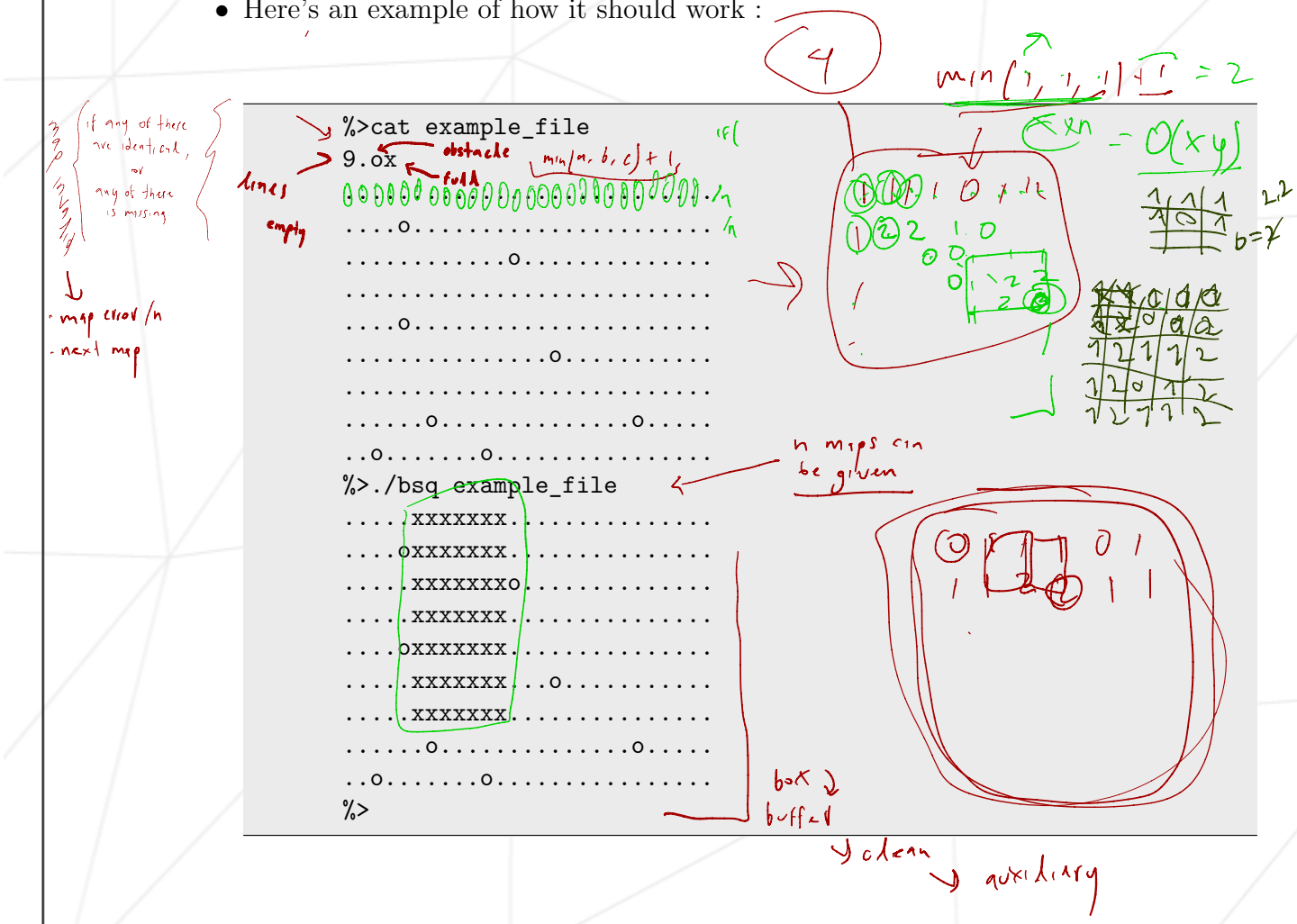
- The biggest square :
  - The aim of this project is to find the biggest square on a map, avoiding obstacles.
  - A file containing the map will be provided. It'll have to be passed as an argument for your program.
  - The first line of the map contains information on how to read the map :
    - The number of lines on the map;
    - The "empty" character;
    - The "obstacle" character;
    - The "full" character.
  - The map is made up of `'"empty" characters'`, lines and `'"obstacle" characters'`.
  - The aim of the program is to replace `'"empty" characters'` by `'"full" characters'` in order to represent the biggest square possible.
  - In the case that more than one solution exists, we'll choose to represent the square that's closest to the top of the map, then the one that's most to the left.
  - Your program must handle 1 to n files as parameters.

- When your program receives more than one map in argument, each solution or `map error` must be followed by a line break.
- Should there be no passed arguments, your program must be able to read on the standard input.
- You should have a valid Makefile that'll compile your project. Your Makefile mustn't relink.



- Definition of a valid map :
  - All lines must have the same length.
  - There's at least one line of at least one box.
  - At each end of line, there's a line break.
  - The characters on the map can only be those introduced in the first line.
  - The map is invalid if a character is missing from the first line, or if two characters (of empty, full and obstacle) are identical.
  - The characters can be any printable character, even numbers.
  - In case of an invalid map, your program should display `map error` on the error output followed by a line break. Your program will then move on to the next map.
- Here's an example of how it should work :

```
%>cat example_file
9.ox
....o.....................
............o.............
.........................
....o.....................
..............o..........
.........................
......o..............o.....
..o.......o...............
%>./bsq example_file
....xxxxxxx...............
....oxxxxxxx...............
....xxxxxxxo...............
....xxxxxxx...............
....oxxxxxxx...............
....xxxxxxx...o...........
....xxxxxxx...............
......o..............o.....
..o.......o...............
%>
```



It is a square indeed. Even though it might not look like it visually.

Chapter IV

Annex

- Perl map generator

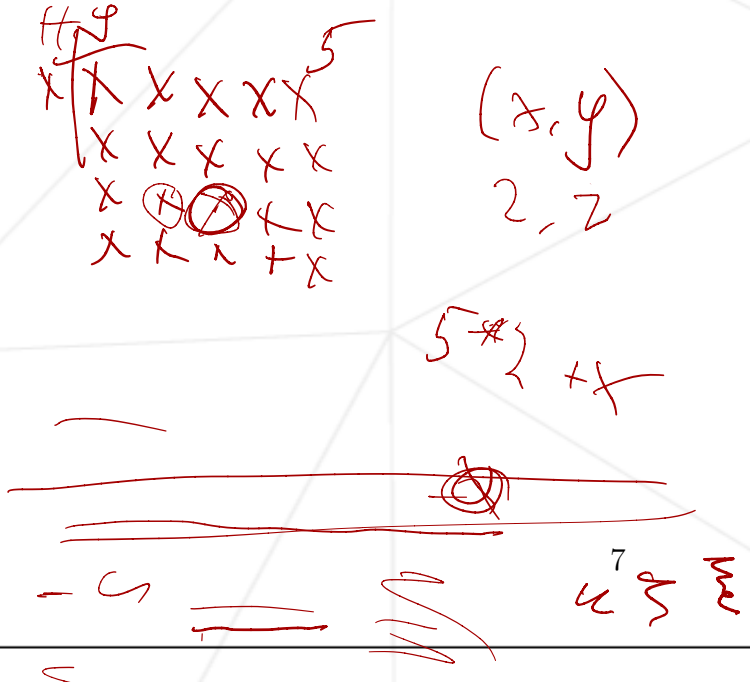
```
#!/usr/bin/perl

use warnings;
use strict;

die "program x y density" unless (scalar(@ARGV) == 3);

my ($x, $y, $density) = @ARGV;

print "$y.ox\n";
for (my $i = 0; $i < $y; $i++) {
    for (my $j = 0; $j < $x; $j++) {
        if (int(rand($y) * 2) < $density) {
            print "o";
        }
        else {
            print ".";
        }
    }
    print "\n";
}
```



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.

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

This project will be checked and graded by a program called Moulinette. Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated and there is no way to negotiate with it. So if you want to avoid bad surprises, be as thorough as possible.