Elementary programming

BSQ

Teacher in charge
younes2.serraj@epitech.eu
Last update
04/12/2015_11h16

Contents

Instructions

- Name of the turn-in repository: CPE_year_BSQ
  `Example for the school year 2015-2016 : CPE_2015_BSQ`
- Binary name : bsq
- Your Makefile must be located at the root of your repository.
- The executable file created must be located at the root of your repository.

`Be careful: the norm will be checked for every single file you turn in`

Subject

- The goal of the project is to find the biggest possible square on a board, while avoiding obstacles.
- The board will be given to you in a file, passed as an argument to your program.
- The board is composed of lines of `'.'` and `'o'`.
- The first line of the file is a number indicating the number of lines in the board.
- All lines have the same length.
- The board will always be a rectangle.
- There will always be at least one line of at least one cell.
- At the end of every line, there is a `'\n'`.
- Example :

```
bash> cat example_file
9
.........................
....o....................
...........o.............
.........................
....o....................
..............o..........
.........................
......o.............o.....
..o.......o..............
bash>
```

- The goal of the program is to replace the `'.'` by `'x'` to represent the biggest square possible.
- When several solutions are possible, we will choose to represent the topest square. In case of equality, choose the leftmost one.
- Example :

```
bash> ./bsq example_file
.....xxxxxxx.............
....oxxxxxxx.............
.....xxxxxxxo............
.....xxxxxxx.............
....oxxxxxxx.............
.....xxxxxxx...o.........
.....xxxxxxx.............
......o.............o.....
..o.......o..............
bash>
```

Even if it does not visually look like a square, it is a square ...

Perl board generator

- The following perl script allows you to create boards:

```
1       #!/usr/bin/perl -w
2
3       if ((scalar @ARGV) != 3)
4       {
5               print "program x y density\n";
6               exit;
7       }
8
9       my $x = $ARGV[0];
10      my $y = $ARGV[1];
11      my $density = $ARGV[2];
12      my $i = 0;
13      my $j = 0;
14
15      print $y . "\n";
16
17      while ($i < $y)
18      {
19              $j = 0;
20              while ($j < $x)
21              {
22                      if (int(rand($y)*2) < $density)
23                      {
24                              print "o";
25                      }
26                      else
27                      {
28                              print ".";
29                      }
30                      $j++;
31              }
32              print "\n";
33              $i++;
34      }
35
```

Allowed functions

- open
- read
- write
- close
- exit
- malloc
- free
- stat