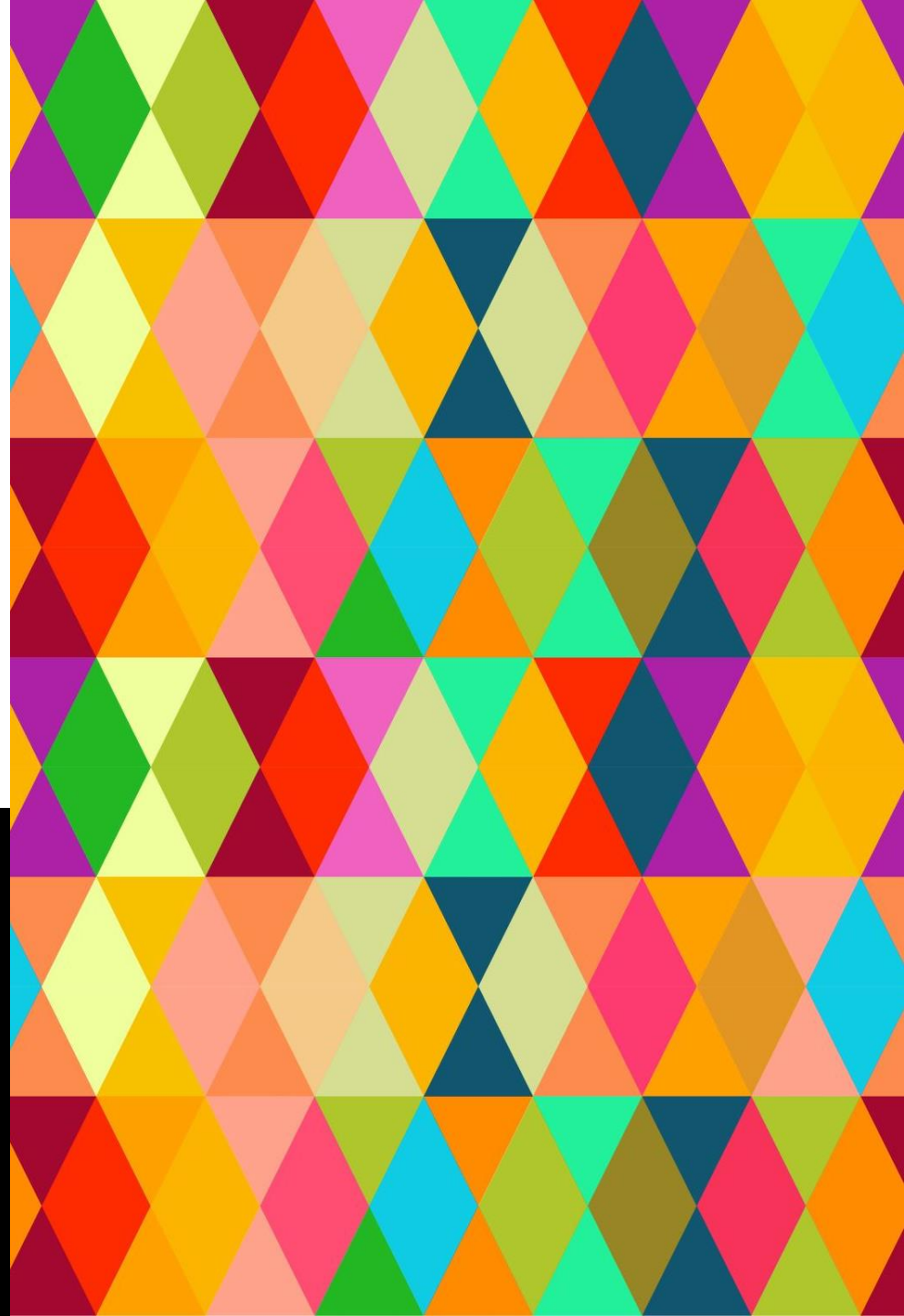
REVIEW BSQ

Arthur Decaen



CONSIGNE

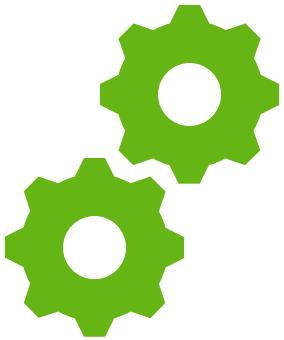
- Trouver le carré le plus grand dans une carte composée de "." et de "o"

- La zone trouvée doit être remplacée par des "x"

```
~/B-CPE-110> cat -e example_file
9$
.........................$
....o....................$
............o............$
.........................$
....o....................$
............o............$
.........................$
......o.............o.....$
..o.......o..............$
```

```
~/B-CPE-110> ./bsq example_file
.....xxxxxxx.............
....oxxxxxxx.............
.....xxxxxxxo............
.....xxxxxxx.............
....oxxxxxxx.............
.....xxxxxxx...o.........
.....xxxxxxx.............
......o.............o.....
..o.......o..............
```

BUT DU PROJET



Création d'un algorithme optimisé pour arriver au résultat souhaité.

Gestion de l'ouverture et la lecture de fichiers depuis un programme en C

ALGORITHME

RÉCUPÉRATION DES DIMENSIONS

```c
// Returns the height of the field
int height(char *file)
{
    int height = 0;

    for (int i = 0; file[i] != '\0'; i++)
        if (file[i] == '\n')
            height += 1;
    return height;
}

//Returns the width of the field
int width(char *file, int height)
{
    int width = 0;

    for (int j = len_nbr(height) + 1; file[j] != '\n'; j++)
        width += 1;
    return width;
}
```

```c
int len_nbr(int nb)
{
    int count = 0;

    if (nb != 0)
        for (count; nb > 0; count++)
            nb /= 10;
    else
        count = 1;
    return count;
}
```

.	.	.	o	.
.	.	o	.	.
o	o	.	.	o
.	.	.	.	.
.	.	.	o	.
.	o	.	.	.
.	.	.	.	o

1	1	1	0	1
1	.	o	.	.
0	o	.	.	o
1	.	.	.	.
1	.	.	o	.
1	o	.	.	.
1	.	.	.	o

1	1	1	0	1
1	-1	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	-1	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

X	X	.	O	.
X	X	O	.	.
O	O	.	.	O
.	.	.	.	.
.	.	.	O	.
.	O	.	.	.
.	.	.	.	O

ALGORITHME (COMPLET)

.	.	.	o	.
.	.	o	.	.
o	o	.	.	o
.	.	.	.	.
.	.	.	o	.
.	o	.	.	.
.	.	.	.	o

1	1	1	0	1
1	.	o	.	.
0	o	.	.	o
1	.	.	.	.
1	.	.	o	.
1	o	.	.	.
1	.	.	.	o

1	1	1	0	1
1	-1	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	-1	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	-1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	-1	-1
0	0	-1	1	0
1	1	-1	-1	-1
1	-1	-1	0	-1
1	0	-1	-1	-1
1	-1	-1	-1	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

1	1	1	0	1
1	2	0	1	1
0	0	1	1	0
1	1	1	2	1
1	2	2	0	1
1	0	1	1	1
1	1	1	2	0

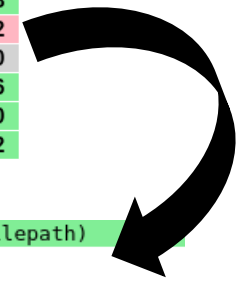
x	x	.	o	.
x	x	o	.	.
o	o	.	.	o
.	.	.	.	.
.	.	.	o	.
.	o	.	.	.
.	.	.	.	o

CRITERION ET GCOVR

GCC Code Coverage Report

Directory:	src/		Exec	Total	Coverage
Date:	2020-11-21 21:16:29	Lines:	87	88	98.9 %
Legend:	low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %	Branches:	47	48	97.9 %

File	Lines			Branches	
create_dup.c		100.0 %	15 / 15	100.0 %	14 / 14
len_array.c		100.0 %	11 / 11	100.0 %	6 / 6
load_2d.c		100.0 %	14 / 14	100.0 %	8 / 8
load_file.c		83.3 %	5 / 6	50.0 %	1 / 2
read_file.c		100.0 %	6 / 6	- %	0 / 0
set_value.c		100.0 %	10 / 10	100.0 %	6 / 6
set_x.c		100.0 %	14 / 14	100.0 %	10 / 10
start.c		100.0 %	12 / 12	100.0 %	2 / 2



```
8  char *load_file_in_mem(char const *filepath)
   {
       struct stat *buf;

8      buf = malloc(sizeof(struct stat));
8      stat(filepath, buf);
8      if (buf->st_size == 0) {
           exit(ERROR);
       }
8      return fs_cat_x_bytes(filepath, buf->st_size);
   }
```

MY EPITECH

Score

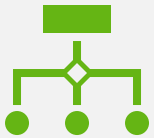
09 - Error handling

50% Passed

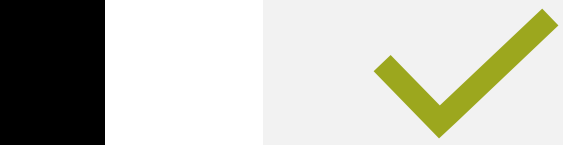
Total: 4
Passed: 2
Crashed: 0
Failed or skipped: 2

01 - different sizes > Test failure: The output must match the regular expression '^OK $', but it was 'KO: Invalid exit status. Got 0 but expected 84. '
02 - non-existant > PASSED
03 - empty map > PASSED
04 - less lines > Test failure: The output must match the regular expression '^OK $', but it was 'KO: Invalid exit status. Got 0 but expected 84. '

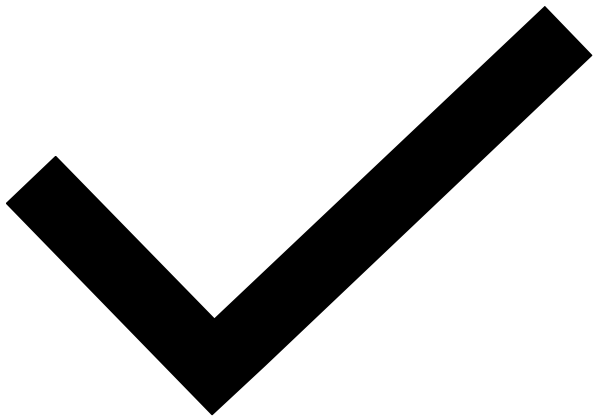
DIFFICULTÉS



- Pas de soucis pour les 2d arrays grâce au bootstrap.



Algorithme déjà **connu.**



ÇA DONNE QUOI GITHUB ?