# B1 - Phoenix Bootcamp

B-BOO-101

# Day 05

The Starry Day

# Day 05

| | |
|---|---|
| **binary name:** | star |
| **repository name:** | BOO_phoenix_d05_*$ACADEMICYEAR* |
| **repository rights:** | ramassage-tek |
| **language:** | C |
| **compilation:** | via Makefile, including re, clean and fclean rules |

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

**Allowed system function:** `write`.

For this last day of the bootcamp, you'll have only one **big** task (or a mini-project depending on how you look at it).
Create a program called `star` that displays a star, based on its given size.
If the size is 0, don't display anything (but it is not an error).

Don't forget that you need a coherent test policy to ensure your program outputs are correct. To do so:

- split your functions in **as many small functions as possible**, so that each function is responsible for one single thing (according to the Coding Style),
- Test each of your functions **individually** using automated unit tests.

Here's some output with different sizes, it is up to you to deduce the rules regarding the star formation.

These examples (and maybe more) can also be found in a simple text files given with this subject.

Terminal

```
~/B-BOO-101> ./star toto 1>/dev/null
Usage: ./star
~/B-BOO-101> ./star -1 1>/dev/null
Usage: ./star
~/B-BOO-101> ./star 0 1>/dev/null
~/B-BOO-101>
```


Terminal
~/B-BOO-101> ./star 1 | cat -e
*$
*** ***$
* *$
*** ***$
*$
~/B-BOO-101> ./star 2 | cat -e
*$
* *$
***** *****$
* *$
* *$
***** *****$
* *$
*$
~/B-BOO-101> ./star 5