Exercises — Quick Sort

version **#7be580532266ed398481e31366afcc24b1950c2a**

Assistants C/Unix 2023 <assistants@tickets.assistants.epita.fr>

Copyright

This document is for internal use at EPITA (website) only.

Contents

*https://intra.assistants.epita.fr

File Tree

```
quick_sort/
├── quick_sort.c   (to submit)
└── quick_sort_example.c
```

Authorized headers : You are only allowed to use the functions defined in the following headers

- err.h
- errno.h
- assert.h
- stddef.h

Compilation : Your code must compile with the following flags

- -std=c99 -pedantic -Werror -Wall -Wextra -Wvla

Main function : None

1 Goal

You must implement the following function:

```c
void quicksort(int *tab, int len);
```

It takes a table and its size and sorts it using an in-place *quicksort* algorithm.

If the argument `tab` is `NULL`, your function must not do anything.

The algorithm is pretty simple:

1. Pick a pivot in the table (for example the first element of the table).
2. Change the array to have all the elements less than (or equal to) the pivot first, then the pivot and in the end the remaining elements.
3. Recursively use this algorithm on those two tables.

Obviously, stop the recursion on a *single element* or empty table.

Be careful!

We will watch the execution time on *really* large tables in order to check that you are not trying to trick us with a bubble sort.

2 Example

```
42sh$ gcc -Wall -Werror -Wextra -std=c99 -pedantic quick_sort.c quick_sort_example.c -o quick_
↪sort
42sh$ ./quick_sort
1 2 2 3 5 6 7 8 10 11 13 14 17 26 30
```

The way is lit. The path is clear. We require only the strength to follow it.