



# EXERCISES — Lookup table

---

version #



# Copyright

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

Copyright © 2021-2022 Assistants <[assistants@tickets.assistants.epita.fr](mailto:assistants@tickets.assistants.epita.fr)>

**The use of this document must abide by the following rules:**

- ▷ You downloaded it from the assistants' intranet.\*
- ▷ This document is strictly personal and must **not** be passed onto someone else.
- ▷ Non-compliance with these rules can lead to severe sanctions.

## Contents

1	Lookup table	3
1.1	Goal . . . . .	3
1.2	Example . . . . .	3

---

\*<https://intra.assistants.epita.fr>

# 1 Lookup table

**Files to submit:**

- lookup\_table/lookup\_table.c

**Provided files:**

- lookup\_table/lookup\_table.h

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

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

## 1.1 Goal

A lookup table is an array which stores the result of a computation for all its possible inputs. It is usually used to replace a costly computation with an array indexing, which is often faster. It means that for a function `func`, we can write  $lut[i] = func(i)$ . The matrix is filled with indices of the lut value that you must lookup. You would not have any error case to handle for this exercise.

```
void apply_lut(unsigned char mat[4][4], const unsigned char lut[256]);
```

Implement the `apply_lut` function which changes the matrix `mat` in-place according to the lookup table:

## 1.2 Example

Using this matrix:

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 121 & 122 & 123 & 124 \\ 125 & 126 & 127 & 128 \\ 252 & 253 & 254 & 255 \end{bmatrix}$$

With the following lookup table:  $[255, 254, 253, \dots, 2, 1, 0]$ , it will give:

$$\begin{bmatrix} 255 & 254 & 253 & 252 \\ 134 & 133 & 132 & 131 \\ 130 & 129 & 128 & 127 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

*It is my job to make sure you do yours.*