Exercises — Lookup Table

version **#7be580532266ed398481e31366afcc24b1950c2a**

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

Copyright

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

Copyright © 2022-2023 Assistants `<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

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

File Tree

```
lookup_table/
├── lookup_table.c   (to submit)
│
└── lookup_table.h
```

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

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:

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: $[255, 254, 253, \ldots, 2, 1, 0]$, it will give:

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

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