



EXERCISES — Run Length Encoding

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

1 Goal

3

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

File Tree

```
run_length_encoding/  
└─ rle.c  (to submit)
```

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

- err.h
- errno.h
- assert.h
- stddef.h
- stdlib.h
- string.h

Compilation : Your code must compile with the following flags

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

Main function : None

1 Goal

The RLE is an encoding format used to compress text, consisting in the replacement of successive occurrences of one character by an indication of the number of occurrences. For example, `rle_encode("AAAAAACBBBBB") = "7A1C6B"`.

You have to implement the following functions:

```
char *rle_encode(const char *s);  
char *rle_decode(const char *s);
```

These two functions respectively return the encoding and decoding of the strings passed as argument. The strings returned are allocated by the called functions. If any error occur you have to return NULL.

Be careful!

AAAAAAAAAAAAAAAA will result to 9A6A

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