



EXERCISES — The sieve of Eratosthenes

version #



IT IS MY JOB TO MAKE SURE YOU DO YOURS.

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	The sieve of Eratosthenes (Advanced)	3
1.1	Goal	3
1.2	Examples	4

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

1 The sieve of Eratosthenes (Advanced)

Files to submit:

- sieve_eratosthenes_advanced/Makefile
- sieve_eratosthenes_advanced/sieve.c

Makefile: Your makefile should define at least the following targets:

- all: Generate sieve.o

Authorized functions: You are only allowed to use the following functions:

- printf(3)
- malloc(3)
- free(3)
- calloc(3)

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

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

1.1 Goal

The goal here is to create a sieve of Eratosthenes. It is a simple iterative sieve for finding prime numbers.

It works with an array of numbers from 2 to n . The first number, 2, is identified as a prime number. Then, all multiples of 2 are marked. When encountering a new unmarked number, it is identified as a prime number, and its multiples are marked.

You must write a function taking an integer n and printing the number of prime numbers from 2 to n , excluded.

```
void sieve(int n);
```

If n is equal or lower than 2, the function will not print anything. Your function will have to be optimized as it will be tested with large values.

1.2 Examples

You can use a `main` to test your program, but it must not be included in your submission. Calling the `sieve` function with the first program argument as parameter, will produce the following output:

```
42sh$ ./sieve 11 | cat -e
4$
42sh$ ./sieve 0 | cat -e
42sh$ ./sieve 1000000 | cat -e
78498$
```

Be careful!

You must **not** include a `main` function. Your sieve function will be used on large numbers, you will have to optimize your function.

It is my job to make sure you do yours.