



# EXERCISES — Generic Void List

---

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>](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	Goal	3
2	<code>list_prepend</code>	3
3	<code>list_length</code>	4
4	<code>list_destroy</code>	4

---

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

## File Tree

```
generic_void_list/  
├── list.c  (to submit)  
└── list.h
```

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

- free(3)
- malloc(3)
- calloc(3)
- memcpy(3)

**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

In this exercise, you will have to implement a generic linked list, along with its manipulation operations.

### Be careful!

An empty list is represented by a NULL pointer.

## 2 list\_prepend

- **Authorized functions:** malloc(3), memcpy(3)

```
struct list *list_prepend(struct list *list, const void *value,  
                          size_t data_size);
```

This function must insert a node containing `value` at the beginning of the list. It must return the new list, or NULL if an error occurred. You can use `memcpy(3)` to copy `value` into the data field of the list structure.

### 3 list\_length

- **Authorized functions:** none.

```
size_t list_length(struct list *list);
```

This function returns the length of the list.

### 4 list\_destroy

- **Authorized functions:** free(3)

```
void list_destroy(struct list *list);
```

This function releases all the memory used by list.

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