Exercises — Generic Void List

version **#7be580532266ed398481e31366afcc24b1950c2a**

Assistants C/Unix 2023 <assistants@tickets.assistants.epita.fr>

Copyright

Contents

*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.