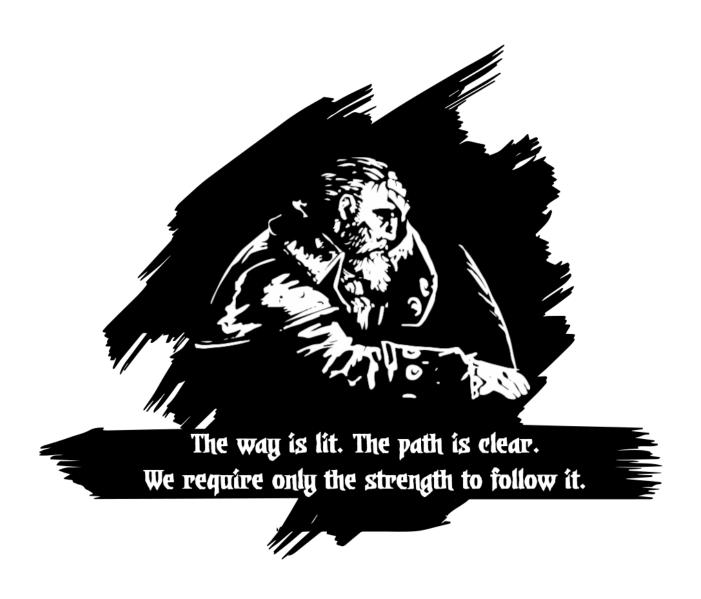


EXERCISES — King of the Hill, array edition

version #7be580532266ed398481e31366afcc24b1950c2a



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 |
|---|-----------|---|
| 2 | Prototype | 4 |
| 3 | Examples | 4 |

^{*}https://intra.assistants.epita.fr

File Tree

```
hill_array/
hill_array.c (to submit)
hill_array.h
```

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

printf(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

We consider that an array is a *hill* if it is composed of:

- A sequence of positive increasing integers (<n+1> being superior or equal to <n>)
- The Top of the hill: one or several equal integers.
- A sequence of positive decreasing integers (< n+1> being inferior or equal to < n>).

For example, this is a valid hill:

```
int arr[] =
{
    1, 2, 3, 4, 5, 6, 7, 7, 2, 1, 0, 0
};
```

And these are **not** hills:

```
int arr1[] =
{
     0, 2, 3, 2, 4, 5, 4, 3, 7, 1
};
int arr2[] =
{
     -1, 2, 3, 2, 0
};
```

Write a function that takes as input an array of int and its length, and returns the index of the top of the hill. You must also check that the hill is correct.

If the hill is invalid, the function returns -1. An empty array is considered invalid.

With an array like this:

```
int arr[] =
{
    0, 2, 3, 4, 6, 7, 9, 9, 7, 6, 5, 4, 2, 1
};
```

The top of the hill is the first 9, thus you must return the index 6.

2 Prototype

```
int top_of_the_hill(int tab[], size_t len);
```

3 Examples

```
int main(void)
{
   int tab1[] = { 1, 2, 3, 4, 6, 6, 4, 2, 0, 0 }; // Valid hill.

   printf("%d\n", top_of_the_hill(tab1, 10));

   int tab2[] = { 1, 2, 3, 4, 5, 6, 6, 6, 6, 6 }; // Valid hill.

   printf("%d\n", top_of_the_hill(tab2, 10));

   int tab3[] = { 1, 2, 3, 4, 6, 6, 4, 5, 0, 0 }; // Invalid hill.

   printf("%d\n", top_of_the_hill(tab3, 10));

   return 0;
}
```

```
42sh$ ./hill_array | cat -e
4$
5$
-1$
```

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