Exercises — Array binary search

version #

Assistants C/Unix 2022 <assistants@tickets.assistants.epita.fr>

Copyright

This document is for internal use at EPITA (website) only.

Contents

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

1 Array binary search

Files to submit:

- binary_search/binary_search.c

Main function: None

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

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

1.1 Goal

```
int binary_search(const int vec[], size_t size, int elt)
```

When looking for an element in a sorted array, it is possible to get the result with a logarithmic complexity using *dichotomy*. Let us remind how dichotomy works:

- If the minimum and maximum indices of the sub-vector are equal or reversed (max < min), the search is negative (we did not find the element).
- Otherwise, the middle element of the array is chosen as the pivot:
  - If the searched item is equal to the pivot, we return its position.
  - If the searched item is greater than the pivot, we restart the search on the sub-vector starting at the `pivot + 1` position.
  - If the searched item is less than the pivot, we restart the search on the sub-vector ending at the `pivot - 1` position.

1.2 Example

```
Search of 42 inside: [ 0 1 4 5 9 10 18 22 42 51 69 ]

[ 0 1 4 5 9 10 18 22 42 51 69 ]
  <          ^              >

[ 0 1 4 5 9 10 18 22 42 51 69]
                <     ^     >
```

Write the function that returns the index of the searched element in a sorted vector of integers, or -1 if it is not present. The size of the array will always be correct.

It is my job to make sure you do yours.