Starfleet

Interview

Staff 42 pedago@42.fr

Summary: This document is an interview question for the Starfleet Piscine.

Contents

Chapter I

General rules

- The interview should last between `45 minutes`.
- Both the interviewer and the interviewed student must be present.
- The interviewed student should write his code using a `whiteboard`, with the language of her/his choice.
- At the end of the interview, the interviewer evaluates the student based on the provided criteria.

Read carefully the interview question and solutions, and make sure you `understand` them before the interview. You can't share this document with other students, as they might be interviewed on the same question. Giving them the answer would prevent them from having to solve an unknown question during an interview.

I.1 During the interview

During the interview, we ask you to :

- Make sure the interviewed student `understands` the question.
- Give her/him any `clarification` on the subject that she/he might need.
- Let her/him come up with a solution before you guide her/him to the best solution given the constraints (time and space).
- Ask the student what is the `complexity` of her/his algorithm ? Can it be improved and how ?
- `Guide` her/him to the best solution without giving the answer. You may refer to the `hints` for that.
- You want to evaluate how the interviewed student thinks, so ask her/him to `explain everything` that she/he thinks or writes (there should be no silences).
- If you see a mistake in the code, wait untill the end and give her/him a chance to correct it by her/himself.
- Ask the student to show how the algorithm works on an `example`.
- Ask the student to explain how `limit cases` are handled.
- Bring out to the student any mistake she/he might have done.
- Give `feedback` on her/his performances after the interview.
- Be `fair` in your evaluation.

As always, stay mannerly, polite, respectful and constructive during the interview. If the interview is carried out smoothly, you will both benefit from it !

Chapter II

Array rotation

II.1 Interview question

`Given an integer array, perform a circular right shift by k.`

EXAMPLE :

```
Input : {0, 1, 2, 4, 5, 6, 7}, k = 4

Output : {4, 5, 6, 7, 0, 1, 2}
```

NOTE : k can be smaller of greater than the array length.

II.2 Acceptable answers (no constraint)

II.2.1 K rotations

- Perform k rotations to the right (worst case : O(n^2) time).

```
O(nk) time , O(1) space, where n is the number of elements in the array
```

Code:

```
void rot1(int *arr, int n) {
        int tmp;

        if (n <= 0)
                return ;
        tmp = arr[n - 1];
        for (int i = n - 1; i > 0; i--) {
                arr[i] = arr[i - 1];
        }
        arr[0] = tmp;
}

void rotate(int *arr, int n, int k) {
        k %= n;
        while (k > 0) {
                rot1(arr, n);
                k--;
        }
}
```

II.2.2 With a buffer

- Copy the array in a buffer.
- Fill the array from the buffer with a rotation by k.

```
O(n) time , O(n) space, where n is the number of elements in the array
```

Code:

```
void rotate(int *arr, int n, int k) {
        int buffer[n];

        k %= n;
        memcpy(buffer, arr, sizeof(int) * n);
        for (int i = 0; i < n; i++) {
                arr[i] = buffer[(i + n - k) % n];
        }
}
```

II.3 Follow up question

Provide a solution in-place with the best possible runtime.

Your algorithm must takes `O(1) space` apart from the input array and `O(n) time`.

II.3.1 Hints

- What if you reverse the array?
- What do you notice when you compare the reversed array to the final rotated array?

II.4 Best solution

II.4.1 Reverse the array

This can be done in 3 steps:

- Reverse the array from 0 to n-1 (whole array)
- Reverse the array from 0 to k-1
- Reverse the array from k to n-1

EXAMPLE:

```
Input : {0, 1, 2, 4, 5, 6, 7}, k = 4

Step 1 : {7, 6, 5, 4, 2, 1, 0}

Step 2 : {4, 5, 6, 7, 2, 1, 0}

Step 3 : {4, 5, 6, 7, 0, 1, 2}
```

```
O(n) time , O(1) space, where n is the number of elements in the array
```

code:

```
void reverse(int *arr, int start, int end) {
        while (start < end) {
                swap(arr, start, end);
                start++;
                end--;
        }
}

void rotate(int *arr, int n, int k) {
        k %= n;
        reverse(arr, 0, n - 1);
        reverse(arr, 0, k - 1);
        reverse(arr, k, n - 1);
}
```