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

Sorted merge

II.1 Interview question

`You are given two sorted arrays, A and B, where A has a large enough buffer at the end to hold B. Write a method to merge B into A in sorted order.`

II.1.1 Hints

- Try moving from the end of the array to the beginning.

II.2 Best solution

II.2.1 Start from the end

EXAMPLE :

```
Input :  A = {1, 5, 6, 10, 12} ... ... ... ...
         B = {0, 3, 5, 7}

Output : A = {0, 1, 3, 5, 5, 6, 7, 10, 12}
```

Since we know that A has enough buffer at the end, we won't need to allocate additional space. The logic should involve simply comparing elements of A and B and inserting them in order, until all elements in A and in B are exhausted.

The only issue with this is that if we insert an element into the front of A, then we'll have to shift the existing elements backwards to make room for it. It's better to insert elements into the back of the array, where there's empty space.

The code below does just that. It works from the back of A and B, moving the largest elements to the back of A.

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

code:

```
void merge(int *a, int lenA, int *b, lenB) {
        int indexA = lenA - 1;
        int indexB = lenB - 1;
        int indexMerged = lenA + lenB - 1;

        while (indexB >= 0) {
                if (indexA >= 0 && a[indexA] > b[indexB]) {
                        a[indexMerged] = a[indexA];
                        indexA--;
                } else {
                        a[indexMerged] = b[indexB];
                        indexB--;
                }
                indexMerged--;
        }
}
```

Note that you don't need to copy the contents of A after running out of elements in B. They are already in place.