University of Houston

Programming Assignment 1

COSC 3320
Algorithms and Data Structures

Gopal Pandurangan

NAME

Due: Sunday, February 14, 2021
11:59 PM

Read the University of Houston Academic Honesty policy.

Academic Honesty Policy

All submitted work should be your own. Copying or using other people's work (including from the Web) will result in $-$`MAX` points, where `MAX` is the maximum possible number of points for that assignment. Repeat offenses will result in a failing grade for the course and will be reported to the Chair. If you have any questions, please reach out to the professor and the TAs. The best way to ask is on Piazza.

By submitting this assignment, you affirm that you have followed the Academic Honesty Policy.

The writeup portion of your submission **must be typed**. We prefer you use LaTeX to type your solutions — LaTeX is the standard way to type works in mathematical sciences, like computer science, and is highly recommended; for more information on using LaTeX, please see this post on Piazza — but any method of typing your solutions (e.g., MS Word, Google Docs, Markdown) is acceptable. **Your writeup must be in pdf format.** The assignment can be submitted **up to two days late for a penalty of 10% per day.** A submission more than **two days late** will receive a **zero**.

Before you begin the assignment, create an account on LeetCode if you do not already have one.

Problem 1 ▶ Permutations

Given an array `nums` of distinct integers, return *all the possible permutations*. You must return the permutations in an order such that any permutation can be obtained from the previous permutation by *swapping a single pair of elements*.

Example 1

Input: `nums` $= [1, 2, 3]$
Output: $[1, 2, 3], [2, 1, 3], [2, 3, 1], [1, 3, 2], [3, 1, 2], [3, 2, 1]$
Note that each permutation in the above ordering can be obtained by swapping exactly one pair of elements from the previous permutation.

Example 2

Input: `nums` $= [0, 1]$
Output: $[0, 1], [1, 0]$

Example 3

Input: `nums` $= [1]$
Output: Output: $[1]$

It is important that you solve this problem using *recursion*. That is, you have to reduce the original problem into one or more subproblem, recursively solve the subproblems, and then combine the solutions to obtain the solution to the original problem. Your solution should take $\mathcal{O}(n!)$ time. A solution that does not use recursion **will receive a zero**.

Note that the LeetCode webpage allows the permutations to be output in any order. By contrast, **we require the permutations to be output such that any two consecutive permutations differ by swapping a single pair of elements**. Additionally, some solutions on LeetCode do not use recursion. These are **not** acceptable solutions. Some solutions posted may also be wrong. In any case, a solution that is largely copied from another source (e.g., verbatim or made to look different by simply changing variable names) will be **in violation of the Academic Honesty Policy**.

The following must be submitted.

(a) Writeup (50 Points)

- Pseudocode for your solution, with an explanation in words why your solution works. (25 points)
- Analysis, showing the correctness of your algorithm and its complexity (i.e., its runtime). (25 points).

(b) Source Code (50 Points)

- Write your solution in Python, C, C++, Java, or JavaScript.
- Your code should be well written and well commented.
- A comment with a link to your LeetCode profile (e.g., `https://leetcode.com/jane-doe/`) and a statement of whether or not your code was accepted by LeetCode. We will verify whether your code is accepted.
- We must be able to *directly copy and paste your code into LeetCode*. If your code does not compile **on LeetCode**, it will **will receive zero points**. Under no circumstances will we attempt to modify any submission, so be sure the code you submit works.

Please submit these files individually. **Do not submit as an archived file (zip file, tarball, etc.)**.

1 Pseudocode and Explanation

Algorithm 1 Permutations – Permutations of an Array

```
1: def PERMUTATIONS(A):
       Input  ▷ An array A of distinct elements.
       Output ▷ All n! permutations of A such that consecutive permutations differ by a single swap.
2:     n ← |A|
3:     if n = 1:                                                    ▷ Base Case
4:         Base Case Stuff
5:     else:                                                        ▷ Recursive Step
6:         Recursive Step Stuff
```

2 Analysis