CSE 3150 — C++ Essentials — Yufeng Wu — Fall 2024
Programming Assignment 2: Basic programming in C++. Due: 11:59 pm @ 09/13/2024.

---

There are two objectives in this assignments.

1. Continue working on basic programming logic
2. C++ I/O.

# 1 Next permutation

This problem is related a problem in LeetCode. I actually implemented the almost exactly the same function that was needed for my work.

A permutation of an array of integers is an arrangement of its members into a sequence or linear order. Here, we assume the permutation is $n$ integers, starting from 1. For example, let $n = 3$, and we consider three integers $1, 2, 3$. The following are all the permutations of these three integers: [1,2,3], [1,3,2], [2, 1, 3], [2, 3, 1], [3,1,2], [3,2,1].

The next permutation of a permutation for the first $n$ integer is the next *lexicographically* greater permutation of its integer. More formally, if all the permutations of the array are sorted in one container according to their lexicographical order, then the next permutation of that array is the permutation that follows it in the sorted container. If such arrangement is not possible, the array must be rearranged as the lowest possible order (i.e., sorted in ascending order). For example, the next permutation of [1,2,3] is [1,3,2]. For [2,3,1], the next permutation is [3,1,2]. Also, if the current permutation is [3,2,1], the next permutaiton would be [1,2,3].

Now you are asked to implement a function called ECNextPermutation that will take a permutation (an array of integers, starting from 0) and its size. After the function returns, the array should be updated with the next permutation.

Note: you shouldn't allocate additional memory. That is, you should update the array in place. You can assume the input list is correctly formatted as a permutation of the first $n$ numbers. Your code must run in time that is linear to the size of array. We will have running time check in test cases.

# 2 Consecutive integers

You are given a string *str*. You need to implement a function to test whether *str* is formed by consecutive integers (without separators). The integers are in the standard decimal format. You may assume integers are non-negative. For example, *str* = 123467 is not formed by consecutive integers, while *str* = 1213141516 is (i.e., 12, 13, 14, 15, 16). You may assume integers are at most 999999.

# Getting familiar with C++ standard library

Try avoiding reinventing wheels by using library functions provided by C++. For example, you can use C++ string::substr to extract a substring from a string; there is a function called isalnum which tests whether a given character is alphanumeric or not and so on. If you don't know how these functions. use Google...

# 3 Instructions for submission

Please read the following carefully.

1. Submit your programs in GradeScope. A link is provided from HuskyCT. Note that you can submit multiple files by drag and drop into the submission UI.

2. You must use the filenames as specified. If you change the filename, the Autograder won't work. Don't change function names.

3. For automated grading, we mainly focus on functionalities. However, often we will also test on efficiency/run time. , (ii) code complexity, and (iii) code quality. We will use third-party tools for some of these tasks. While these tools have been around for sometime, they can still sometimes produce not so accurate report about your code. If this is the case, contact the TAs about the issues.