EE231002 Introduction to Programming

Lab05. Permutations

Due: Oct. 27, 2018

Given a set of N distinct elements, there are $N!$ possible permutations to arrange these elements. For example, given the set {1, 2, 3} they can be arranged in the following 6 permutations:

1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1

Your assignment is to write a program to generate all possible permutations given the set of N integers from 1 to N. The output should follow the format below.

Assuming the array `A[N]` stores a permutation, the following algorithm by 14th century Narayana Pandita of India produces the next lexicographic permutation:

---

1. Find the largest index `j` such that `A[j] < A[j + 1]`.
   If no such index exists, the permutation is the last permutation.
2. Find the largest index `k` such that `A[j] < A[k]`.
3. Swap `A[j]` with `A[k]`.
4. Reverse the sequence from `A[j + 1]` up to and including the last element `A[N - 1]`.

If the array `A` is initialized to {1, 2, $\cdots$, `N` }, then repeatedly applying Pandita's algorithm it would generate all possible permutations. Please implement Pandita's algorithm such that all possible permutations are generated in the following format:

Example output (N=4):

```
$ a.out
permutation #1: 1 2 3 4
permutation #2: 1 2 4 3
permutation #3: 1 3 2 4
permutation #4: 1 3 4 2
permutation #5: 1 4 2 3
permutation #6: 1 4 3 2
permutation #7: 2 1 3 4
permutation #8: 2 1 4 3
permutation #9: 2 3 1 4
permutation #10: 2 3 4 1
permutation #11: 2 4 1 3
permutation #12: 2 4 3 1
permutation #13: 3 1 2 4
permutation #14: 3 1 4 2
permutation #15: 3 2 1 4
permutation #16: 3 2 4 1
permutation #17: 3 4 1 2
permutation #18: 3 4 2 1
permutation #19: 4 1 2 3
permutation #20: 4 1 3 2
permutation #21: 4 2 1 3
permutation #22: 4 2 3 1
permutation #23: 4 3 1 2
permutation #24: 4 3 2 1
  Total number of permutations is 24
```

Please note that your loop of finding all permutations should be terminated using the step 1 of the Pandita algorithm.

Notes.

1. Create a directory **lab05** and use it as the working directory.
2. Name your program source file as **lab05.c**.
3. The first few lines of your program should be comments as the following.

```
/* EE231002 Lab05. Permutations
   ID, Name
   Date:
*/
```

4. The number of elements, N, should be defined as a macro as following:

```
#define N 7
```

5. After finishing editing your source file, you can execute the following command to compile it,

```
$ gcc lab05.c
```

If no compilation errors, the executable file, **a.out**, should be generated, and you can execute it by typing

```
$ ./a.out
```

6. After you finish verifying your program, you can submit your source code by

```
$ ~ee2310/bin/submit lab05 lab05.c
```

If you see a "submitted successfully" message, then you are done. In case you want to check which file and at what time you submitted your labs, you can type in the following command:

```
$ ~ee2310/bin/subrec lab05
```

It will show the last few submission records.