

lab06

```
1 // EE231002 Lab06. Permutations
2 // 108061112, 林靖
3 // Oct. 25, 2019
4
5 #include <stdio.h> // Standard input and output library
6 #define N 7       // Given a set of N distinct elements
7
8 int main(void)    // Called at program startup
9 {
10     int A[N];     // To store the permutation
11     int i;        // To initialize & print A[N], to reverse sequence in step 4
12     int j = N;    // To find such index j in step 1
13     int k;        // To find such index k in step 2
14     int temp;     // To swap in step 3, to reverse sequence in step 4
15     int count = 0; // Number of permutations generated
16
17     for (i = N; i > 0; i--) { // Initialize the set of integers from 1 to N
18         A[N - i] = i;        // In reversed order
19     }
20     while (j >= 0) { // If j < 0, no such index exists (the end)
21         for (i = N; ++j < --i; ) { // Step 4 to reverse A[j + 1] ~ A[N - 1]
22             i = N; ++j < --i;
23             temp = A[j]; //
24             Empty comments are not needed!
25             A[j] = A[i]; // Step 4 mirror the elements by reflection
26             A[i] = temp; //
27         }
28         printf("permutation # %d: ", ++count); // # of permutations generated
29         for (i = 0; i < N; i++) { // To print out all the elements in A[N]
30             printf("%d ", A[i]); // To print out the permutation
31         }
32         printf("\n"); // (new line)
33         for (j = N - 2; j >= 0 && A[j] < A[j + 1]; j--) ; // Step 1 find index j
34         for (k = N - 1; A[j] < A[k]; k--) ; // Step 2 find index k
35         temp = A[j]; //
36         A[j] = A[k]; // Step 3 swap A[j] with A[k]
37         A[k] = temp; //
38     }
39     printf(" Total number of permutations is %d\n", count); // (total number)
40     return 0; // Indicates normal program termination
41 }
```

[Format] can be improved.

[Program] should match the algorithm's 4 steps exactly.

[Program] logic can be simplified.

Score: 93