Programming Assignment 4

Part A

Create a program that generates all unique permutations of a given input string using backtracking. The input string is passed as a program argument. The output is written in a text file *output.txt* where each unique permutation are separated by a new line. You are not allowed to use built-in C++ algorithms to solve the problem. You are not allowed data structures with methods to remove repeating elements.

Sample Input/Output A

Command line

```
PA4@EEE121: ~$ ./PA4A DO6O
PA4@EEE121: ~$
```

output.txt

```
DOO6
DO6O
D6OO
ODO6
OD6O
OOD6
OO6D
O6DO
O6OD
6DOO
6ODO
6OOD
```

Part B

Suppose that you are working on the cash register and you are to provide a change amount of M to the customer. Create a program that determines the number of ways to provide the amount of change M assuming that you have an infinite amount of coins and bills using dynamic programming. The input is read through a text file *input.txt*. The first line of the input contains the number of denomination (coins and bills) N, followed by the amount of change M, separated by a space. The succeeding N lines contains the value of the denominations. The output is displayed through the standard output / command line. You may assume that the exact change is possible for the given input, that is the minimum output value is always 1.

Sample Input/Output B

input.txt

```
3 10
1
5
10
```

Command line

```
PA4@EEE121: ~$ ./PA4B
PA4@EEE121: ~$ 4
```

Grading Breakdown

30%	Part A - working (may have repeated string)
20%	Part A - optimal backtracking without repeated string
30%	Part B - working for a fixed number of denominations
20%	Part B - can handle any number of denominations

Notes

1. **DO NOT** use tabs to separate output.
2. If you have a question involving source code, please do post the code snippet and/or error messages. Always resolve compile-time errors in the order they appear in your code. If you think that the code snippet might be too revealing of your work, post it as a private question first and I'll decide if said code may be made public to your classmates.
3. Make sure that you are using a g++-8 compiler which is either from GCC 8.1 or GCC 8.2 so that you will not have issues during checking. If you want to use a new C++ specification kindly comment what compile command you are using as a comment at the beginning of your source code.
4. You may use any library in the C++ STL.
5. If you have a question just post it at Piazza for better traction.
6. Several test input/output will be provided three full days before the soft deadline; however, sample input/output for the bonus will not be provided.

Submission

1. Soft deadline is set to 9:00 PM of 14 May 2019.
2. Hard deadline is set to 9:00 PM of 22 May 2019.