# Exercises: Combinatorial Algorithms

Problems for exercises and homework for the "Data Structures and Algorithms Advanced" course from the official "Applied Programmer" curriculum.

You can check your solutions here: <https://judge.softuni.org/Contests/3641/Combinatorial-Algorithms-Exercises>

Use the provided skeleton:

A picture containing table

Description automatically generated

## Permutations without Repetitions

Given a **set** of elements, find all **permutations** **without** **repetitions**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B C | A B C  A C B  B A C  B C A  C B A  C A B |

### Solution

To write the algorithm we first need to write the **bottom** of our **recursion**. That will be our **index** becomes **higher** or **equal** to our permutation's **length:**

Graphical user interface, text, application

Description automatically generated

Now from **0** until the permutation's **length**,if we haven't used the current permutation, we should:

* **Mark** the **current** **permutation** as **used** to make sure we **don't** **repeat** the **same** letter **twice**.
* Set the **current** permutation as the one from the **input** to update our sequence.
* Call GeneratePermutations() for **index + 1** to go to the next possible letter.
* **Mark** the **current** **permutation** as **not used** so it is possible to use it **again** for our **next** sequence.

Graphical user interface, text, application

Description automatically generated

## Permutations with Repetitions

Given a **multi-set** of elements, find all **permutations**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B B | A B B  B A B  B B A |

## Variations without Repetitions

Given a **set** of elements, find all **variations** **of** **k elements** **without** **repetitions**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B C  2 | A B  A C  B A  B C  C A  C B |

### Hint

How **different** is this algorithm from the previous? Where does our **K** even come in play? **Before** or **during** the algorithm?

## Variations with Repetition

Given a **set** of elements, find all **variations** of **k elements** **with** **repetitions**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B C  2 | A A  A B  A C  B A  B B  B C  C A  C B  C C |

## Cinema

Write a program that prints all of the possible **distributions** of a **group of friends** in a **cinema hall**.

### Input

* On the **first line,** you will be given all of the friends' **names** separated by **comma and space**.
* Until you receive the command **"generate"** you will be given some of those friends' **names** and a **number of the place** that they want to have in format: **"{name} - {place}"**.
  + Those friends **wan**t to **sit** only in the place that they **have chosen**. They **cannot sit in other places**. For more clarification see the examples below.

### Output

Print all the **possible ways to distribute the friends** having in mind that some of them want a particular place and they will sit there only. The **order** of the output does **not matter**.

### Constrains

* The friends' **names** and the **number** of the place will always be valid.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| Peter, Amy, George, Rick  Amy - 1  Rick - 4  generate | Amy Peter George Rick  Amy George Peter Rick | Amy only wants to sit on the first seat and Rick wants to sit on the fourth, so we only switch the other friends |
| Garry, Liam, Teddy, Anna, Buddy, Simon  Buddy - 3  Liam - 5  Simon - 1  generate | Simon Garry Buddy Teddy Liam Anna  Simon Garry Buddy Anna Liam Teddy  Simon Teddy Buddy Garry Liam Anna  Simon Teddy Buddy Anna Liam Garry  Simon Anna Buddy Garry Liam Teddy  Simon Anna Buddy Teddy Liam Garry |  |

### Hints

In your Generate() **method** you can first check if the **index** is **higher or equal** to the number of **names** and if it is just **print** the **result** and **return**.

Then permute for **index + 1.**

For **backtracking** for **index + 1** until number of names:

* Swap the **index** with i.
* Permute for **index + 1.**
* Swap the **index** with i.

## Combinations without Repetition

Given a **set** of elements, generate all **combinations** of **k elements without** repetition.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B C  2 | A B  A C  B C |

### Solution

This algorithm differs from permutations in the following way:

In the **for** **loop** **start** from our **start** **parameter** and for **each** iteration set the **current** **combination** to equal the one from the **input**, and call **generate** with **index + 1** and currentIndex **+ 1**:

Graphical user interface, text, application

Description automatically generated

## Combinations with Repetition

Given a **set** of elements, generate all **combinations** of **k elements with** repetition.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A B C  2 | A A  A B  A C  B B  B C  C C |

## N Choose K Count

Given a **n** and **k**, calculate the **number** of possible **n choose k** **combinations** (**without** repetition).

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  2 | 3 |
| 5  3 | 10 |

## School Teams

Write a program that receives the **names** of **girls** and **boys** in a class and generates **all possible ways** to create **teams** with **3 girls** and **2 boys**. Print each team on a **separate line** separated by comma and space **", "** (**first** the **girls then** the **boys**). For more clarification see the examples below.

### Input

* On the **first line,** you will receive the **girls'** names separated by comma and space **", "**.
* On the **second line,** you will receive the **boys'** names separated by comma and space **", "**.

### Output

* On **separate lines** print all the possible **teams** with exactly **3 girls** and **2 boys** separated by comma and space and starting with the girls.

### Constrains

* There will always be **at least 3 girls** and **2 boys** in the input.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Linda, Amy, Katty  John, Bill | Linda, Amy, Katty, John, Bill |
| Lisa, Yoana, Marta, Rachel  George, Garry, Bob | Lisa, Yoana, Marta, George, Garry  Lisa, Yoana, Marta, George, Bob  Lisa, Yoana, Marta, Garry, Bob  Lisa, Yoana, Rachel, George, Garry  Lisa, Yoana, Rachel, George, Bob  Lisa, Yoana, Rachel, Garry, Bob  Lisa, Marta, Rachel, George, Garry  Lisa, Marta, Rachel, George, Bob  Lisa, Marta, Rachel, Garry, Bob  Yoana, Marta, Rachel, George, Garry  Yoana, Marta, Rachel, George, Bob  Yoana, Marta, Rachel, Garry, Bob |

## Fruit Drinks

Write a program that receives the **names** of **fruits** and **toppings** and generates **all possible ways** to create **drinks** with **2 fruits** and **1 topping**. Print each drink on a **separate line** separated by space.

### Input

* On the **first line,** you will receive the **fruit** names separated by comma and space.
* On the **second line,** you will receive the **topping** names separated by comma and space.

### Output

* On **separate lines** print all the possible **fruit drinks** with exactly **2 fruits** and **1 topping** separated by **space and comma**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Apple Peach Pear  Mint OrangeSlice | Apple, Apple, Mint  Apple, Apple, OrangeSlice  Apple, Peach, Mint  Apple, Peach, OrangeSlice  Apple, Pear, Mint  Apple, Pear, OrangeSlice  Peach, Peach, Mint  Peach, Peach, OrangeSlice  Peach, Pear, Mint  Peach, Pear, OrangeSlice  Pear, Pear, Mint  Pear, Pear, OrangeSlice |
| Apple Peach Cherries Orange  Mint Straw OrangeSlice NoTopping | Apple, Apple, Mint  Apple, Apple, Straw  Apple, Apple, OrangeSlice  Apple, Apple, NoTopping  Apple, Peach, Mint  Apple, Peach, Straw  Apple, Peach, OrangeSlice  Apple, Peach, NoTopping  Apple, Cherries, Mint  Apple, Cherries, Straw  Apple, Cherries, OrangeSlice  Apple, Cherries, NoTopping  Apple, Orange, Mint  Apple, Orange, Straw  Apple, Orange, OrangeSlice  Apple, Orange, NoTopping  Peach, Peach, Mint  Peach, Peach, Straw  Peach, Peach, OrangeSlice  Peach, Peach, NoTopping  Peach, Cherries, Mint  Peach, Cherries, Straw  Peach, Cherries, OrangeSlice  Peach, Cherries, NoTopping  Peach, Orange, Mint  Peach, Orange, Straw  Peach, Orange, OrangeSlice  Peach, Orange, NoTopping  Cherries, Cherries, Mint  Cherries, Cherries, Straw  Cherries, Cherries, OrangeSlice  Cherries, Cherries, NoTopping  Cherries, Orange, Mint  Cherries, Orange, Straw  Cherries, Orange, OrangeSlice  Cherries, Orange, NoTopping  Orange, Orange, Mint  Orange, Orange, Straw  Orange, Orange, OrangeSlice  Orange, Orange, NoTopping |