**Lab: Lists as Stacks and Queues**

This document defines the exercises for the ["Python Advanced" course at @Software University.](https://softuni.bg/trainings/4370/python-advanced-january-2024)

Please submit your solutions (source code) to all the below-described problems in [Judge](https://judge.softuni.org/Contests/1830/Lists-as-Stacks-and-Queues-Lab).

## Reverse Strings

Write a program that:

* **Reads** an **input string**
* **Reverses** itusing a **stack**
* **Prints** the result back on the console

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| I Love Python | nohtyP evoL I |
| Stacks and Queues | seueuQ dna skcatS |

## Matching Parentheses

You are given an algebraic expression with **parentheses**. Scan through the string and extract **each set of parentheses**.

Print the result back **on the console**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 + (2 - (2 + 3) \* 4 / (3 + 1)) \* 5 | (2 + 3)  (3 + 1)  (2 - (2 + 3) \* 4 / (3 + 1)) |
| (2 + 3) - (2 + 3) | (2 + 3)  (2 + 3) |

### Hints

Scan through the expression searching for parentheses:

* If you find an **opening** **parenthesis**, **push** the index into the stack.
* If you find a **closing** **parenthesis, pop** the topmost element from the stack. This is the index of the last opening parenthesis.
* Use the current index and the popped one to **extract** a set of **parentheses.**

## Supermarket

Tom is working at the supermarket, and he needs your help to keep track of his clients. Write a program that **reads lines of input** consisting of a customer's **name** and **adds** it to the end of a **queue** until "**End**" is received. If, in the meantime, you receive the command "Paid", you should **print each customer** in the order they are served (from the first to the last one) and **empty** the queue.

When you receive **"**End**",** you should print the **count of the remaining** **people** in the queue in the format: "{count} people remaining.**"**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| George  Peter  William  Paid  Michael  Oscar  Olivia  Linda  End | George  Peter  William  4 people remaining. |
| Anna  Emma  Alexander  End | 3 people remaining. |

## Water Dispenser

*Write a program that keeps track of people getting water from a dispenser and the amount of water left at the end.*

On the first line, you will receive the starting **quantity** of water (integer) in a dispenser. Then, on the following lines, you will be given the **names** of some people who want to **get water** (each on a separate line) until you receive the command **"**Start**"**. Add those people to a **queue**. Finally, you will receive some commands until the command **"**End**"**:

* **"{liters}"** - litters (integer) that the current person in the **queue** wants to get. Check if there is **enough** water in the dispenser for that person.
  + If there is enough water, print **"{person\_name} got water"** and remove him/her from the queue.
  + Otherwise, print **"{person\_name} must wait"** and **remove the person** from the queue **without reducing** the water in the dispenser.
* **"refill {liters}"** - **add** the given litters in the dispenser.

In the end, print how many liters of water have left in the format: **"{left\_liters} liters left"**.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 2  Peter  Amy  Start  2  refill 1  1  End | Peter got water  Amy got water  0 liters left | We create a queue with Peter and Amy. After the start command, we see that Peter wants 2 liters of water (and he gets them). The water dispenser is left with 0 liters. After refilling, there is 1 liter in the dispenser. So when Amy wants 1 liter, she gets it, and there are 0 liters left in the dispenser. |
| 10  Peter  George  Amy  Alice  Start  2  3  3  3  End | Peter got water  George got water  Amy got water  Alice must wait  2 liters left |  |

## Hot Potato

Hot Potato is a game in which children form a circle and toss a hot potato. The counting starts with the first kid. **Every nth toss, the child holding the potato leaves the game**. When a kid leaves the game, it **passes** the potato to the next kid. It continues **until there is only one kid left**.

Create a program that simulates the game of Hot Potato. On the **first line,** you will receive kids' names, separated by a single space. On the **second line,** you will receive the **nth** toss (integer) in which a child leaves the game.

**Print** everykidwho is **removed** fromthe **circle** in the format **"Removed {kid}"**. In the end, **print** the **only kid left** in the format **"Last is {kid}"**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Tracy Emily Daniel  2 | Removed Emily  Removed Tracy  Last is Daniel |
| George Peter Michael William Thomas  10 | Removed Thomas  Removed Peter  Removed Michael  Removed George  Last is William |
| George Peter Michael William Thomas  1 | Removed George  Removed Peter  Removed Michael  Removed William  Last is Thomas |