

Lab: Stacks and Queues

This document defines the exercises for ["Java Advanced" course @ Software University](#). Please submit your solutions (source code) of all below described problems in [Judge](#).

I. Working with Stacks

1. Browser History

Write a program which takes 2 types of browser instructions:

- Normal navigation: a URL is set, given by a string
- The string **"back"** that sets the current URL to the last set URL

After each instruction the program should print the current URL. If the **back** instruction can't be executed, print **"no previous URLs"**. The input ends with **"Home"** command, then simply you have to stop the program.

Examples

Input	Output
https://softuni.bg/ back https://softuni.bg/trainings/courses back https://softuni.bg/trainings/2056 back https://softuni.bg/trainings/live https://softuni.bg/trainings/live/details Home	https://softuni.bg/ no previous URLs https://softuni.bg/trainings/courses https://softuni.bg/ https://softuni.bg/trainings/2056 https://softuni.bg/ https://softuni.bg/trainings/live https://softuni.bg/trainings/live/details

Hints

- Use **ArrayDeque<>**
- Use **String** to keep current page
- Use **push()**, when moving to next page
- Use **pop()**, when going back

2. Simple Calculator

Create a simple calculator that can evaluate simple expressions that will not hold any operator different from addition and subtraction. There will not be parentheses or operator precedence.

Solve the problem using a Stack.

Examples

Input	Output
2 + 5 + 10 - 2 - 1	14
2 - 2 + 5	5

Hints

- Use an **ArrayDeque<>**
- Consider using the **add()** method
- You can either
 - add the elements and then pop them out
 - or push them and reverse the stack

3. Decimal to Binary Converter

Create a simple program that **can convert a decimal number to its binary representation**. Implement an elegant solution **using a Stack**.

Print the **binary representation** back at the terminal.

Examples

Input	Output
10	1010
1024	10000000000

Hints

- If the given number is 0, just print 0
- Else, while the number is greater than zero, divide it by 2 and push the remainder into the stack

```
while (decimal != 0) {  
    stack.push(decimal % 2);  
    decimal /= 2;  
}
```

- When you are done dividing, pop all reminders from the stack, that is the binary representation

4. Matching Brackets

We are given an arithmetical expression with brackets. Scan through the string and extract each sub-expression.

Print the result back at the terminal.

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

- Use a stack, namely an **ArrayDeque()**
- Scan through the expression searching for brackets



- If you find an opening bracket, push the index into the stack
- If you find a closing bracket pop the topmost element from the stack. This is the index of the opening bracket.
- Use the current and the popped index to extract the sub-expression

```
if (ch == '(') {
    stack.push(index);
}
else if (ch == ')') {
    int startIndex = stack.pop();
    String contents = expression.substring(startIndex, index + 1);
    System.out.println(contents);
}
```

II. Working with Queues

5. Printer Queue

The printer queue is a simple way to control the order of files sent to a printer device. We know that a single printer can be shared between multiple devices. So to preserve the order of the files sent, we can use queue. Write down a program which takes filenames until **"print"** command is received. Then as output print the filenames in the order of printing. Some of the tasks may be **canceled** if you receive **"cancel"** you have to remove the first file to be printed. If there is no current file to be printed print **"Printer is on standby"**.

Examples

Input	Output
Lab.docx	Canceled Lab.docx
cancel	Printer is on standby
cancel	Canceled Presentation.pptx
Presentation.pptx	NoteNothing.txt
NoteNothing.txt	SomeCode.java
SomeCode.java	
cancel	
print	

Hints

- Use an **ArrayDeque<>**
- Use **offer()**, when adding file
- Use **pollFirst()**, when printing file

6. Hot Potato

Hot potato is a game in which **children form a circle and start passing a hot potato**. The counting starts with the first kid. **Every n^{th} toss the child left with the potato leaves the game**. When a kid leaves the game, it passes the potato forward. This continues repeating **until there is only one kid left**.

Create a program that simulates the game of Hot Potato. **Print every kid that is removed from the circle**. In the end, **print the kid that is left last**.

Examples

Input	Output
Mimi Pepi Toshko 2	Removed Pepi Removed Mimi Last is Toshko
Gosho Pesho Misho Stefan Krasi 10	Removed Krasi Removed Pesho Removed Misho Removed Gosho Last is Stefan
Gosho Pesho Misho Stefan Krasi 1	Removed Gosho Removed Pesho Removed Misho Removed Stefan Last is Krasi

7. Math Potato

Rework the previous problem so that a **child is removed only on a composite (not prime) cycle** (cycles start from 1)

If a **cycle is prime**, just **print the child's name**.

As before, print the name of the child that is left last.

Examples

Input	Output
Mimi Pepi Toshko 2	Removed Pepi Prime Mimi Prime Toshko Removed Mimi Last is Toshko
Gosho Pesho Misho Stefan Krasi 10	Removed Krasi Prime Pesho Prime Misho Removed Stefan Prime Gosho Removed Gosho Prime Misho Removed Pesho Last is Misho

8. Browser History Upgrade

Extend "Browser History" with a **"forward"** instruction which visits URLs that were navigated away from by **"back"**

Each forward instruction visits the next most-recent such URL. If a normal navigation happens, all potential forward URLs are removed until a new back instruction is given. If the forward instruction can't be executed, print **"no next URLs"**.

Examples

Input	Output
forward https://softuni.bg/ https://softuni.bg/trainings/courses back forward https://softuni.bg/trainings/2056 back forward forward https://softuni.bg/trainings/courses Home	no next URLs https://softuni.bg/ https://softuni.bg/trainings/courses https://softuni.bg/ https://softuni.bg/trainings/courses https://softuni.bg/trainings/2056 https://softuni.bg/trainings/courses https://softuni.bg/trainings/2056 no next URLs https://softuni.bg/trainings/courses

Hints

- Use the solution from Browser History
- Use **ArrayDeque<>** as queue to keep the forward pages
- Use **clear()** method to reset the forward pages
- Use **addFirst()** when adding page to the forward pages