CSE211 DATA STRUCTURES

LAB 2 FALL 2024

STACK OPERATIONS

Prerequisites

Open the terminal and execute the following commands after downloading the `tarball` file:

```
cd /mnt/c/Users/user/Downloads && tar -xvf lab2_2.tar.gz --one-top-level=lab2_2
cd /mnt/c/Users/user/Downloads/lab2_2 && make all
code .
```

Introduction

In this lab, you will implement advanced operations on a Stack data structure using C++. The Stack is implemented as a template class that can store elements of any type T. Your task is to implement the following challenging operations:

1. `reverseInGroups`: Reverse elements in groups of K
2. `removeAlternate`: Remove alternate elements from the stack
3. `interleave`: Interleave first half with second half of the stack
4. `getNextGreater`: Find next greater element for each element in the stack

Project Structure

```
.
├── bin/
│   └── stack
├── include/
│   ├── Stack.hpp
│   └── Color.hpp
├── obj/
│   ├── Stack.o
│   ├── Color.o
│   └── main.o
├── src/
│   ├── Stack.cpp
│   ├── Color.cpp
│   └── main.cpp
├── instructions.md
└── Makefile
```

Implementation Details

1. reverseInGroups

- **Purpose**: Reverse elements in groups of K. If final group has less than K elements, reverse them too
- **Parameters**: k (group size)
- **Example**:

```
Input:  [8, 7, 6, 5, 4, 3, 2, 1] (top), k=3
Output: [7, 8, 4, 5, 6, 1, 2, 3] (top)
```

2. removeAlternate

- **Purpose**: Remove alternate elements from the stack (removes every second element from the stack)
- **Example**:

```
Input:  [6, 5, 4, 3, 2, 1] (top)
Output: [6, 4, 2] (top)
```

3. interleave

- **Purpose**: Interleave first half elements with second half
- **Example**:

```
Input:  [6, 5, 4, 3, 2, 1] (top)
Output: [6, 3, 5, 2, 4, 1] (top)
```

4. getNextGreater

- **Purpose**: Find next greater element for each element
- **Return**: New stack with next greater elements
- **Example**:

```
Input:  [10, 25, 2, 5, 4] (top)
Output: [25, 0, 5, 0, 0] (top)  // 0 when no greater element exists
```

Testing

1. Build and run:

```
make clean  # Clean previous builds
make all    # Compile all files
make run    # Execute the program
```

Restrictions

❌ Do not modify:

- Stack.hpp interface
- main.cpp test cases
- Project structure
- Build system

❌ Do not use:

- External libraries
- Global variables
- Additional data structures (except where specified)

Academic Integrity

- Individual work only
- No code sharing
- No plagiarism
- Violations result in zero grade

Submission

1. Test thoroughly
2. Clean build files: `make clean`
3. Send only the `Stack.cpp` file to the course portal

Good luck with your implementation!