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_4.tar.gz --one-top-level=lab2_4
cd /mnt/c/Users/user/Downloads/lab2_4 && 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. `decodeString`: Decode an encoded string with numbers and brackets
2. `asteroidCollision`: Simulate asteroid collisions based on size and direction
3. `carFleet`: Calculate number of car fleets formed based on position and speed
4. `exclusiveTime`: Calculate exclusive execution time of functions from logs

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. decodeString

- **Purpose**: Decode a string encoded with numbers and brackets
- **Parameters**: encoded string (e.g., "3[a]2[bc]")
- **Return**: decoded string
- **Example**:

```
Input:  "3[a2[c]]"
Output: "accaccacc"

Input:  "2[abc]3[cd]ef"
Output: "abcabccdcdcdef"
```

2. asteroidCollision

- **Purpose**: Simulate asteroids colliding based on size and direction
- **Parameters**: vector of integers (positive = right, negative = left)
- **Return**: vector of surviving asteroids
- **Example**:

```
Input:  [5, 10, -5]
Output: [5, 10]  // -5 collides with 10 and is destroyed

Input:  [8, -8]
Output: []  // Both asteroids destroy each other
```

3. carFleet

- **Purpose**: Calculate number of car fleets that will form
- **Parameters**: target distance, position array, speed array
- **Return**: number of fleets formed
- **Example**:

```
Input:  target = 12, position = [10,8,0,5,3], speed = [2,4,1,1,3]
Output: 2  // Two fleets will form
// Fleet 1: cars 1,2,5 (arrive at same time)
// Fleet 2: cars 3,4 (slower group)
```

4. exclusiveTime

- **Purpose**: Calculate exclusive execution time of functions from logs
- **Parameters**: number of functions, vector of log strings
- **Return**: vector of exclusive times for each function
- **Example**:

```
Input:  n = 2, logs = ["0:start:0","1:start:2","1:end:5","0:end:6"]
Output: [3, 4]  // Function 0 runs for 3 units, Function 1 runs for 4 units
```

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!