Assignment 1 - 16+ Ways to Stack a Cat

Principles of Programming Languages-II
Deadline - February 12th, 2020

[Individual assignment]

The goal of this assignment is to implement a stack in different ways, as mentioned in the paper 16 ways to stack a cat. The language of choice is C++. The purpose of this assignment is to demonstrate the data-hiding techniques and feature richness of C++, and this is not a stack implementation focused assignment.

The paper simply assumes that a “cat” is being stacked. For this assignment, we will stack integers which will input from stdin, and the output will be directed to stdout.

Input format

The first line has `m` and `n`, where `m` is the number of stacks and `n` is the number of commands. The next `n` lines each have a command of one of the following two types:

- `<stack_id> pop` - pop operation
- `<stack_id> push <number>` - push operation

Stack IDs will always be a positive number. All the numbers will be within the range of `long long` in C++.

Sample input:

```
3 10
1 push 3
1 pop
1 push 23
1 push 24
2 push 25
1 pop
1 push 26
2 push 30
3 push 23
3 pop
```

Output format

Display the state of `m` stacks after those `n` operations. Display the stacks in increasing order of stack IDs, and for every stack the output format is:

- `<stack_id> <tail> <tail-1> … <head>`

Sample output:

```
1 26 23
```

```
2 30 25
3
```

File structure

Submission file structure:

```
assignment
|-Makefile
|-README.md
|-stack0
| |-...
| |-...
|-stack1
| |-...
| |-...
|-...
|-stack15
  |-...
  |-...
```

- The layout above shows the compulsory files. There can be other utility files to avoid redundant code.
- The number of ways to stack the cat need not be restricted to 16 (at least 16 have to be implemented), but the extra implementations should be adequately explained in README.md
- The ‘Makefile’ at the root level must generate 16 binaries with the names ‘bin/stack0’, ‘bin/stack1’, ‘bin/stack2’ etc. (in a folder named ‘bin’). These binaries will be used for evaluation.
- Stack specific implementation should be in its folder. Try to keep the directory structure minimalist and avoid any unnecessary files.
- **DO NOT INCLUDE BINARIES OR ANY OTHER GENERATED FILES IN YOUR SUBMISSION.**
- The README.md will have the steps to run your binaries and a brief 1-2 line description of all the ways to stack a cat.
- Potential things to note: **file-structure, make-file, executable, design, modularity, any-and-all-C++/OOP-features, commenting.**
- Zip this folder in a file named “<Roll number>.zip” (e.g. cs19btech11000.zip) such that the folder `assignment` is the only folder in this zip file. Make sure you adhere to this submission format because the evaluation will be script-based.

For any clarifications in this assignment, mail the TAs on or before 9th February 2020.