CS 240

Data Structures and Algorithms

Spring 2014

1 Lab 08

1.1 Goal

The goal of this assignment is to becomes more familiar with the STL and all of the options that it provides you.

2 Assignment

You are to use the STL to solve one of the following problems. You may use any of the containers provided by the STL to solve this problem, however, the correctness of your solution will be graded based upon the correctness of the data structure that you select.

3 Problems

3.1 Reverse Polish Notation Calculator

A string is an arithmetic expression in Reverse Polish Notation if it satisfies the following properties:

- It is a number (positive, negative, or floating point)
- It is of the form `A,B,op`, where A and B are expressions in Reverse Polish Notation, and op is one of the basic operators (i.e. +,-,*,/)

The evaluation of a Reverse Polish Notation expression is determined recursively. The base case corresponds to an instance of a number, and the recursive case corresponds to the evaluation of the `A,B,op` in the expected way. (i.e. `2,3,*` will evaluate to 6).

Note that this syntax frees us from the need to have parenthesis as it is unambiguous.

Write a program that takes an arithmetical expression in Reverse Polish Notation (as described above) as a program argument and returns the number to which that expression evaluates.

3.2 Moving Tiered Cakes

You have been hired by a bakery and tasked with helping to arrange tiered cakes. The cakes have been arranged on a specially designed plate where cakes that are rested on the plate will not be damaged provided the smaller tiers are sitting atop the larger tiers. These plates are expensive and the bakery only has three such plates: one red, one green, one blue

Some customer had ordered a tiered cake, with T tiers on a green plate, but as the cake tiers were being made, a colorblind baker accidentally placed the entire tiered cake on the red plate instead of the green.

The rules for shifting the cake tiers are as follows:

- Only one tier of cake may be moved at a time.
- A tier of cake is slid off the top of one plate (possibly stacked with cake) and onto another plate (possibly also stacked with cake).
- A cake can only be placed on top of a larger piece of cake.

Since this happens frequently, you are to write a program that prints the steps to solve the problem following the rules for shifting cakes so that you can move the cake from the red plate to the green plate. The argument to your program will be T, the number of tiers that need to be moved. Your output will be of the form:

```
Move tier 1 from red to green.
```

(This will be the result when T = 1.)

Initially you should number the tiers with increasing integer values, starting from 1 and incrementing by 1 for each tier.

3.3 Musical Chairs Simulator

The game "musical chairs" is played with N players and N-1 chairs. In each round of the game, the players move from chair to chair in the same direction (in a circle) while a song is played for a random amount of time. When the song stops, players take the closest available chair and the player who does not have a chair is eliminated. In each successive round one chair is eliminated, and the game is played until only one player remains.

Since this game relies on a random amount of time you will have to use a random number generator. (Examples of this are shown in the Driver.cpp files from Lab 07.)

Write a program that simulates musical chairs. Note that the game relies on a random amount of time. (Computers cannot physically generate truly random numbers, they can only generate numbers programmatically. The randomness comes from changing the first input called the seed. As a result, if the provided seed is the same, each sequence of random numbers generated from the seed will be the same.) The first argument to your program will be the seed value. The second argument will be a song length (i.e. the modulus that will be used for the random numbers to keep them in range). The remainder of the arguments will be the names of the players (no spaces will occur in a name), in the order they begin the game. Using this information, generate a random number in the appropriate range, and the player with in that position from the front of the list is eliminated. The next round begins with the player who was after the eliminated player appearing at the front of the list.

3.4 Animal Shelter Inventory

An animal shelter takes in dogs and cats and operates on a strict "First In First Out" basis. When someone comes to adopt an animal they can request either a cat or a dog or both. At current, the system is a log book of when the animals came in and a list of the people who have made a request to adopt the animals.

You have been hired by the animal shelter to upgrade this system to the 21st century. Your program will have a text based menu for processing these requests.

The first menu will read:

```
What would you like to do?
```

```
1.  Add an animal.
2.  Remove an animal.
3.  Exit.
```

The "Add an animal" menu will read:

```
Would you like to add a dog or a cat?
1.  Dog
2.  Cat
```

Both of these sub-menus will read:

```
Provide the animal's name:
```

The "Remove an animal" menu will read:

```
Would you like to adopt a dog or a cat?
1.  Dog
2.  Cat
3.  Either One
```

All three of these sub-menus will read:

```
Provide the first name of the customer:
```

If an option is unavailable, (i.e. someone wants to adopt only a cat but there are currently no cats) print:

```
That option is unavailable.
```

After adding an animal print:

```
Added a XXX named YYY
```

where XXX is the type of animal (dog or cat) and YYY is the name of the animal.

After adopting out an animal print:

```
XXX adopted a YYY named ZZZ
```

where XXX is the new owner's name, YYY is the type of animal (dog or cat) and ZZZ is the name of the animal.

The menu's options will be chosen by their number, and all information will be added through cin. (DO NOT MAKE A GUI, A TEXT-BASED INTERFACE IS MORE THAN SUFFICIENT.) You should assume that names will not contain spaces.