CS 2413 001: Data Structures, Summer 2000
Programming Project #3: Hungry Students
Due in class Monday 24 July 2000

http://www.cs.ou.edu/~cs2413/

This project will give you experience using stacks, queues and trees. You will use the same development process as in Programming Projects #0, #1 and #2.

The University of Podunk is holding a Henryball tournament and has invited competitors from several schools:

- the University of Eastville, whose colors are blue and yellow;
- the University of Westburg, whose colors are red and green;
- the University of Northton, whose colors are orange and purple;
- the University of Southport, whose colors are orange and blue;
- the University of Jahunga, whose colors are red and blue;
- the University of Podunk, whose colors are yellow and green.

The tournament lasts all day, with a lunch break in the middle, served at the campus cafeteria. Every meal, the cafeteria serves three different entrees, two different desserts, and salad. On the day of the tournament, the entrees are: chicken ($4.75), fishsticks ($4.00) or meatless lasagna ($3.75). The desserts are: cheesecake ($2.50) or pudding ($1.25). Salad costs $1.50 per ounce.

The cafeteria works like this: there are two queues, one on the left and one on the right. Each queue has: two stacks of trays, a salad bar, an entree area and a dessert area. As students enter the cafeteria, they see which queue is shorter and enter it; if the queues are the same length, they choose one at random. They select a tray from the taller stack, or if the stacks are the same height, they select one at random.

Trays come in six colors: red, green, blue, yellow, orange and purple. The cafeteria has 20 trays of each color, randomly distributed throughout the cafeteria.

Henryball players have a lot of school spirit: they'll only eat off of trays that are one of their school colors. For example, an Eastville player will only eat from a blue tray or a yellow tray. So, an Eastville player will keep popping trays off their chosen stack to get to a blue or yellow tray, pushing the rejects onto the other stack; if they find no tray of an appropriate color, they'll reverse the process to go through the other stack, and if they still find no tray of the appropriate color, then they'll take the top tray of the remaining stack.

When a student enters the cafeteria, they already know what they plan to eat: how many ounces of salad, which entree and which dessert. Every student also has a name and a school. For example, Kim from Eastville might want 3 ounces of salad, fishsticks and pudding, while Lee from Southport might want 6 ounces of salad, lasagna and cheesecake.

Students are not charged individually for their food. Instead, at the end of the meal, the cafeteria adds up the cost for all students of each school, and presents each school with an itemized bill, alphabetized by student name.

For this project, write a program that simulates the cafeteria. Because you now have a lot of experience with object-oriented program design in C++, you will have considerable autonomy in how to design the program, under these conditions:

- the hungry student queues must be implemented using queues;

- the tray stacks must be implemented using stacks;
- the students from each school must be alphabetized using a binary search tree;
- the main program should be used only for inputting, outputting and controlling the overall system, with most of the algorithm implemented in classes representing such concepts as student, school, stack of trays, queue of hungry students, etc.

You may use the stack, queue and tree classes in *OODS*, or design your own. Bear in mind that every class should have at least one constructor as well as a copy constructor, a destructor, an overloaded assignment operator, an overloaded equality operator, and a friend `ostream` function.

For all of these classes, you should also declare appropriate exception classes, and you should handle exceptions as gracefully as possible.

Input files will be provided by Wednesday July 11 and will be found on `ecnalpha` in the directory `~neeman/CS2413pp3`. Input files will begin with a single integer representing the seed for the random number generator:

```
123456789
```

The seed will be followed by student entries, which will have the following form:

```
StudentName
SchoolName
#OuncesSalad
EntreeName
DessertName
```

For example:

```
Kim
Eastville
3
Fishsticks
Pudding
```

The last entry in the input file will have one word:

```
Done
```

Output should have the schools listed alphabetically, and should have the following form:

```
University of Eastville
  Kim:  $9.75
  Jan:  $12.25
  Total: $22.00
University of Jahunga ...
```

Programming Style

The style for this programming project is the same as described for Programming Project #1.

What To Turn In

You should turn in the same items, in the same format, as for Programming Projects #1 and #2.

You may turn your project in early if you choose; otherwise, turn it in during class on Monday 24 July. If you turn it in after the close of class (3:20pm), it will be considered late, at which point you will lose 25% of the maximum value but can turn it in any time through the close of class on Wednesday 26 July. Submissions after 3:20pm Wednesday 26 July will receive no credit.

References

S. Radhakrishnan, L. Wise & C. N. Sekharan, *Object-Oriented Data Structures Featuring C++*, 1999.