CST238 Fall 2019

Lab 08 - Classes with Dynamic Memory

Objectives:

- Use multiple files to develop a program
- Revise a standard data structure to use dynamic memory, rather than static memory
- Implement the Big 3 member functions for dynamic memory management

Part 1. (5 points) Fork the Stack with Static Array code from earlier in the course, and separate into three files:

- Stack.h
- Stack.cpp
- main.cpp

Note that case DOES count.

Part 2. (20 points)

1. Modify the stack to use a dynamic array, rather than a static array
2. Add a myCapacity member variable to store the possible number of elements in the stack
3. Add a member function called `getCapacity()` to return the capacity
4. Modify the default constructor to set that capacity to 5
5. Add a constructor with a parameter to set the capacity to an arbitrary size
6. Correctly implement the Big 3 (destructor, copy constructor, and assignment operator)

Once you are done, test each function, as with the main() function below:

```
int main(){
   cout << "Create a stack with 5 capacity: ";
   Stack s(5);

   cout << "Pushing 100, 200, 300... ";
   for (int i = 1; i <= 3; i++)
      s.push(100*i);

   cout << "Contents of stack s:\n";
   s.display();
   cout << endl;

   // Create t with the copy constructor and push 77 on it.
   Stack t(s);
   t.push(77);

   cout << "Contents of stack s:\n";
   cout << "s:\n";
   s.display();

   cout << "Contents of stack t:\n";
   cout << "t:\n";
   t.display();
   cout << endl;
   // Test assignment operator.
   Stack u(2);
   u.push(88);
   u.push(99);
   u = s;
   cout << "Contents of stack u after u = s:\n";
   cout << "u:\n";
   u.display();
   cout << endl;

   return 0;
}
```

If everything is implemented correctly, you should not have any memory leaks, and should print:

```
Create a stack with 5 capacity: Pushing 100, 200, 300... Contents of stack s:
300
200
100

Contents of stack s:
s:
300
200
100
Contents of stack t:
t:
77
300
200
100

Contents of stack u after u = s:
u:
300
200
100
```