Part B: Write the code

(N.B: **DO NOT** make any changes in the classes or in the main() except for those noted!)

Given the class definition below:

1. **Declare** a copy constructor in the arrayclass class defined below.

2. After the arrayclass definition, in the space provided, **define** the copy constructor (non-inline) to  
 create a **deep copy**. Watch out for memory leaks.

3. In the arrayStack definition, finish writing the **definition** for the **pop** method

--should return the popped item; should throw an exception if it can’t pop.

4. Finish writing main() such that:

a) The program uses C++ exception handling for errors in the classes and program. Be sure to  
 provide **specific** type catches to any and **all** throws anywhere in the class or program. If any

**bad\_alloc** exception is thrown, make sure that any memory that **was previously** allocated is   
 cleaned up. For all catches, the exitCode should be set to EXIT\_FAILURE.

b) The program dynamically allocates memory for **st** to be an array of two elements, the first of  
 which points to a dynamically allocated arrayStack with a capacity of 3, the second of which  
 points to a dynamically allocated arrayStack with a capacity of 2.

c) The program uses a **loop** to fill the arrayStack at st[0] with 3, 4, and 5, and the arrayStack at   
 st[1] with 6 and 7.

d) The program uses a *for* loop to go through both elements of the array, and

* Using an inner *for* loop, the program should use a const\_iterator to print each element of the *arrayStack* to *cout*.
* Using an inner while loop, pop each element of each *arrayStack*, add “one” to it, and push it on to *SingleStack*.
* Using an inner while loop, pop each element of *SingleStack* and push it back on to the original *arrayStack*.
* Use the *copy* function (*from <algorithm>*) to print each element of the a*rrayStack* to the console (*cout*) with ‘|’ as a delimiter.

e) The program cleans up all allocated memory.

The output of the program should look something like:

3 4 5

4 | 5 | 6 |

6 7

7 | 8 |