NAME_____

MEDGAR EVERS COLLEGE
The City University of New York

CS246 - 001: Data Structures

Quiz 1 (Take Home)

- All tasks should be completed in your class repository on GitHub.
- Please place all files in the folder Quizzes/Quiz01.
- Only use the following libraries: iostream, string, sstream, cstdlib, ctime.
- Every header file should have a header guard.
- All content of each header file (excluding the header guard) should be in a namespace called dsq.

Task 1: (20 pts.)

In a header file named Util.h, define the following two functions:

- Swap() that takes two integer reference parameters. The function should assign the value of the first parameter to the second and the second parameter's value to the first.

- getRandom() that takes two integer parameters: max, min. The second parameter should have a default value of 0. If max is less than min, swap their values using the Swap() function. Declare a static boolean variable named seeded with an initial value of false. If seeded is false, call srand() with time(0) as its argument and set seeded to true. Return a random number within the range of the minimum and maximum numbers inclusively. Use the following formula: rand() % (max – min + 1) + min.

Task 2: (20 pts.)

In a header file named Object.h define a class named Object with the following methods:

- A public pure virtual string constant method named toString() that takes no parameters.

- A public friend ostream reference method named operator<<() that takes two parameters: ostream& out, const Object& obj. The function should insert (<<) the result of the second parameter's toString() method into the first parameter then return the first parameter.

Task 3: (40 pts.)

In a header file named UniqueList.h define a class named UniqueList that publicly inherits the Object class from Task 2. Include Object.h. The class should consist of the following:

- A private integer pointer named arr.
- A private integer variable named capacity.
- A private integer variable named size.
- A public overloaded constructor that takes an integer parameter. Assign the parameter to capacity, initialize size to 0. Assign a newly allocated array, with capacity as its size, to arr.
- A public default constructor that calls the overloaded constructor with 100 as its argument.
- A public copy constructor.
- A public assignment operator.
- A public destructor that deallocates arr.
- A private method named resize() that takes no parameters. The function should make a pointer to a new array with a capacity equal to twice that of the current capacity. Assign every element from the arr to the new array. Deallocate arr, then assign the new array to arr. Finally update the capacity to the new capacity.
- A public constant method named getSize() that takes no parameters and returns size.
- A public constant method named getCapacity() that takes no parameters and returns capacity.
- A public constant method named contains() that takes an integer parameter. If the parameter is in the arr array, return true, otherwise return false.
- A public method named insert() that takes an integer parameter. If the parameter isn't already in the arr array, check if the arr array is full. If it is, resize it, then assign the element at index size to the parameter and increase size by 1.
- A public method named remove() that takes an integer parameter. Search for the parameter in arr. If you find it, assign that element the last element in the arr array, decrease size by 1 and return true. Otherwise return false.
- A public overridden constant string method named toString() that takes no parameters. The method should return all elements of the arr array encased in square brackets with each element separated by a comma. (e.g. [1,2,3,4,5])
- A public boolean constant method named operator==() that takes a const UniqueList reference parameter. If the parameter contains every element from arr and they share the same size, return true otherwise return false.

Task 4: (20 pts.)

In a main.cpp file include Util.h, UniqueList.h and define the following functions:

- lottery() that takes no parameters. Declare a UniqueList object named numbers with an initial capacity of 6. Insert a random integer between 1 and 20 (inclusively) into the object. Keep doing so until the numbers object has a size of 6. Return the object.
- main() that assigns a UniqueList object named winning_numbers the call of lottery(). Declare another UniqueList object named ticket the call of lottery(). If the numbers of ticket equal the numbers of winning_numbers, congratulate the user. Otherwise, inform the user of their loss.

NAME_____

Task 5 (Extra Credit: 10 pts.)

In UniqueList.h, define a public method named sort() that takes no parameters and returns a new UniqueList containing the elements of the current list in ascending order.

- The method should not modify the original list.
- The method should work as follows:
  1. Create a copy of the current UniqueList.
  2. Create a new empty UniqueList named sorted.
  3. While the copy still has elements:
     - Find the smallest element in the copy.
     - Insert this element into sorted.
     - Remove this element from the copy.
  4. Return sorted.

Example:
If the list contains [7,3,9,1,5], calling sort() should return [1,3,5,7,9].