CSC 311-01 QUIZ 2 Spring 2015

5 min. 5 questions

This is a CLOSED textbook quiz.

USE SCANTRON FORM NO. 882-E LIKE THIS:

Select <u>one</u> answer to each question.

1. What is an Abstract Data Type? Select exactly one answer.

   (A) A class based on abstract concepts borrowed from discrete mathematics.

   (B) A type of data that does not have concrete implementation.

   (C) A collection of methods with well-defined functionality that work on some unspecified data structure.

   (D) A collection of abstract methods that work on some specific data structure.

   (E) None of the above.

2. Give the big-Θ characterization of the *average number* of copyings *per insertion* needed to "resize" an array while consecutively inserting n elements into a priority queue implemeted as an array if the <u>additive</u> capacity increment is used. Select exactly one answer.

   (A) $\Theta(1)$

   (B) $\Theta(n)$

   (C) $\Theta(n^2)$

   (D) $\Theta(2^n)$

   (E) None of the above.

3. Give the big-Θ characterization of the *total number* of copyings needed to "resize" an array while consecutively inserting n elements into a priority queue implemeted as an array if the <u>multiplicative</u> capacity increment is used. Select exactly one answer.

   (A) $\Theta(1)$

   (B) $\Theta(n)$

   (C) $\Theta(n \log n)$

   (D) $\Theta(n^2)$

   (E) None of the above.

4. Assumming that `PQ.insert(A[i])` takes $f(i)$ time and `A[i]=PQ.remove()` takes $g(i+1)$ time, where f and g are increasing functions, give the best big "Oh" characterization of the worst-case running time of the following code:

   ```
   for (int i = 0; i < n; i++) PQ.insert(A[i]);

   for (int i = n-1; i >= 0; i-) A[i]=PQ.remove();
   ```

   Select exactly one answer.

   (A) $O(n^2)$

   (B) $O(\max(f(n), g(n)))$

   (C) $O(n \max(f(n), g(n)))$

   (D) $O(\max(f(n) + g(n)))$

   (E) None of the above.

5. Assumming that `count` $= n$ at the time of call, give the best big "Oh" characterization of the worst-case running time of the following method. Select exactly one answer.

   ```
   public void insert(int newItem) {
       int insertPosition = count;
       while ((insertPosition > 0) &&
       (itemArray[insertPosition - 1] > newItem)) {
           insertPosition-;
           itemArray[insertPosition+1] = itemArray[insertPosition];
       }
       itemArray[insertPosition] = newItem;
       count++;
   }
   ```

   (A) $O(1)$

   (B) $O(\log n)$

   (C) $O(n)$

   (D) $O(n^2)$

   (E) None of the above.