TDDB56 DALGOPT – Algoritmer och Optimering
Kontrollskrivning DALG 2005-10-21, 14–18

Examinator: Jan Maluszynski
Jour: Jan Maluszynski, 013-281483 eller 0707-524408
Maxpoäng: 14p (som kan tillgodoräknas på de tre närmast följande ordinarie tentamina i kursen om så önskas - detaljerade anvisningar om detta se kurshemsidan.)

Tillåtna hjälpmedel:
- Engelsk-Svensk ordbok
- Goodrich, Tamassia: Data Structures and Algorithms in Java
- Lewis, Denenberg: Data Structures and Their Algorithms
- Cormen, Leiserson, Rivest: Introduction to Algorithms

Generella instruktioner:
• Läs igenom alla uppgifter innan du börjar.
• Redovisa maximalt en uppgift per inlämnat ark. Består uppgiften av flera deluppgifter kan dessa redovisas på samma ark. Skriv namn och personnummer överst på varje ark.
• Skriv tydligt. Oläsbara lösningsförslag beaktas icke.
• Motivera tydligt alla steg i svaret/lösningen. Avsaknad av motivering kan medföra poängavdrag.
• The solutions can be written in English or in Swedish, as you prefer.

Warning: This is an additional kontollskrivning organized exceptionally due to some exam scheduling conflicts. You are not allowed to take it if you attended kontollskrivningen on October 18.

Lycka till!
1. Complexity (5 p)

(a) Which of the following are true and which are false? Answers without justification give no points.

- \( \sqrt{n^5} \in O(n^2) \)
- \( \log_2 n^3 \in O(n \log n) \)
- \( \log_2 n^3 \in \Theta(n \log n) \)
- \( \min(700, n^2) \in \Theta(1) \)

(b) The greatest common divisor \( \gcd(m, n) \) of two positive integers \( m \) and \( n \) is the largest positive integer that divides both, e.g. \( \gcd(28, 42) = 14 \). Analyze the worst case time complexity of the following algorithms computing the greatest common divisor \( \gcd(m, n) \):

- Starting from the smaller of \( m \) and \( n \) search for the gcd by counting down and checking the consecutive numbers for being a divisor of \( m \) and \( n \).
- function \( GCD(\text{integer} \ m, n) \): integer

\[
\text{while } n > 0 \text{ do}
\]
\[
r \leftarrow m \mod n
\]
\[
m \leftarrow n
\]
\[
n \leftarrow r
\]
\[
\text{return } m
\]

Hint: prove that if \( m > n \) then \((m \mod n) < m/2\).

Statement of complexity without justification gives no points.

2. Queues and Stacks. (3 p)

(a) Queues.
A queue \( Q \) is implemented as a circular array (called also ring buffer).

- What is the minimal size of the ring buffer necessary for correct execution of the following operations:

\(\text{enqueue}(5, Q), \text{enqueue}(3, Q), \text{dequeue}(Q), \text{enqueue}(2, Q), \text{enqueue}(8, Q), \text{dequeue}(Q), \text{enqueue}(9, Q), \text{enqueue}(1, Q), \text{dequeue}(Q), \text{dequeue}(Q)\)

Justify your answer.

- Show the sequence of output values obtained by execution of the above operations and the final state of the ring buffer after execution of these operations.

(b) Stacks
We consider postfix arithmetic expressions over natural numbers. Such an expression is either a natural numeral or a pair of postfix expressions followed by a binary arithmetic expression, e.g. \([5], [4 \ 3 \ 5 \ + \ +], [12 \ 2 \ + \ 3 \ 4 \ \times \ +]\). The value of an expression which is a numeral is the number represented by this numeral. The value of an expression of the form \(e_1 \ e_2 \ \text{op}\) is obtained by applying the operation \(\text{op}\) to the values of expressions \(e_1\) and \(e_2\). Thus the values of the example expressions are: 5, 12 and 26.

- Describe the idea of an iterative algorithm for computing the value of a given expression in time linear on the size of the expression. The algorithm should use a stack. Justify that the algorithm has the required complexity.
• Show the state of the stack in consecutive steps of the computation of the value of the expression:
  \[2 \ 1 \ 2 + 3 \ 2 \times \times +\]

3. Sorting and Selection
   (3 p)
   (a) Illustrate the operation of Insertion Sort on the following Sequence of Integers:
       \([9, 4, 15, 8, 5]\)
   (b) Illustrate the operation of Radix Sort on the following array of keys, where the characters of the keys form the subkeys for sorting: \([\text{abd, cad, bce, aba, ccc, dab}]\). The ordering of keys is defined as usual alphabetic ordering. You need only to show the form of the array after each phase of subkey sorting.
   (c) Illustrate the operation of Quick Select while finding the median of the set given as the following sequence \([5, 8, 3, 2, 7, 9, 12]\).

4. Search Trees and Heaps
   (3 p)
   (a) Consider the binary search tree whose preorder traversal gives the following sequence of keys: \(41, 16, 39, 67, 49, 47, 61, 74\).
      • Prove that it is an AVL tree. (0.5)
      • Show step by step how the AVL-tree operations \(\text{Insert}(40), \text{Insert}(53), \text{Delete}(67)\) are performed on this tree (each of the operations is to be applied to this tree). (1.5)
   (b) Show the heaps that result from inserting \(6, 5, 2, 4, 7, 1\), and \(3\) into an initially empty heap. (Draw the partially ordered trees, and their array representations). (1)