Full Name:

Roll No:

Data Structures and Algorithms

(ESO211)

Exam's duration: 1 hour

13 September 2010

Directions

- This is a **closed book exam**.
- Write your answers directly on the exam paper, in the frames. The size of the frame is a hint of the maximal space you should need to answer a question.
- Do not give to much details or justifications unless you are asked for them.
- Do not forget to write your name atop the two sheets of paper.
- Solutions will be in `http://www.lrde.epita.fr/~adl/ens/iitj/eso211/` by the end of the day.

1 The Θ Notation

1. Prove rigorously that for any positive constants a, b, c we have $a(bn + c)^2 = \Theta(n^2)$

Answer:

2. Which of the constants a, b and c could be zero without invalidating the above equality?

Answer:

2 Special Heap

Usually, a heap of n elements is stored as an array. The father of the element at index i is located at index $Parent(i) = \lfloor i/2 \rfloor$ and can be accessed in constant time.

Here is an algorithm that inserts a value v in a heap represented by the first n values of an array A:

```
HEAPINSERT(A, n, v)
1  i ← n + 1
2  A[i] ← v
3  while i > 1 and A[Parent(i)] < A[i] do
4      A[Parent(i)] ↔ A[i]
5      i ← Parent(i)
```

1. What is the complexity of this algorithm when the heap has n elements?

Answer:

2. Imagine that we replace the array A by a doubly linked list. The access to the father of the element at position i is now done in $\Theta(i/2)$ operations, because we have to unwind the list by $i/2$ positions.

   What becomes the complexity of HEAPINSERT if A is now a doubly linked list? **Justify your answer.**

Answer:

3. Consider heap represented by the following array:

13	9	12	3	6	8	5	2

   Give the state of the heap after the successive removals of its three largest values.

Full Name:

Roll No:

3 Merge Sort

1. Recall the complexity of running Merge Sort on an array of n values.

Answer:

2. Consider a variant of the Merge Sort algorithm, where the array is split in three parts (instead of two) that are then sorted recursively before being merged back together.

   What is the complexity of merging three sorted arrays of size $n/3$ in one sorted array of size n?

Answer:

3. Give a recursive definition of the complexity of this "3-part" Merge Sort algorithm.

Answer:

4. What is the solution of this recursive equation?

Answer:

4 Dynamic Arrays

Consider a dynamic array on which the following operations are defined:

- ACCESS(A,i) returns the value at position i in the array A in $\Theta(1)$ operations.
- INSERTBACK(A,v) inserts the value v at the end of the array A, enlarging the array if necessary.

Assume INSERTBACK is implemented using the following algorithm:

```
If the array is not full:
    insert the element at the first empty place
Else (the array is full)
    allocate a new array twice as large
    copy all elements from the old array to the new one
    free the memory of the old array
```

insert the value at the first empty place

The size of the array, as well as the position of the first empty place are of course stored in two variables, so they are known in constant time.

We have seen in class why doubling the size of the dynamic array during reallocation is important to obtain *amortized* constant time.

1. What would be the amortized complexity of INSERTBACK if instead of doubling the size of the array, we just multiply its size by $5/4$. **Justify your answer.**

Answer:

2. What would be the amortized complexity INSERTBACK if instead of doubling the size of the array, we augment it by $\sqrt{n}$ elements. (For instance if an array of 16 elements is full, we reallocate it in an array of $16 + \sqrt{16} = 20$ elements.) **Justify you answer** without worrying about rounding the result of the square root.

   For what it is worth: if $x = y + \sqrt{y}$ then $y = x - \frac{1}{2}\sqrt{4x+1} + \frac{1}{2}$.

Answer:

5 Recursion

Explain what is a *tail recursion*.

Answer:

End of Exam.