# MCGILL UNIVERSITY – COMP 251
# TEST 1

October 6, 2009

| | |
|---|---|
| **Student Number:** | |
| **Email Address:** | |
| **Family Name(s):** | |
| **Given Name(s):** | |

- There are 10 pages in total (including this page).
- You have 60 minutes for this test.
- This test is worth 10% of your final mark.
- Answer each question directly on the test paper, in the space provided. Use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and indicate clearly the part of your work that should be marked.

## The Master Theorem

Suppose that for some $a \geq 1$, $b > 1$, $d > 0$:

$$T(n) = aT(n/b) + \Theta(n^d)$$

then

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log(n)) & \text{if } a = b^d \\ \Theta(n^{\log_b(a)}) & \text{if } a > b^d \end{cases}$$

| | |
|---|---|
| Question 1 (out of 8) | |
| Question 2 (out of 7) | |
| Question 3 (out of 12) | |
| Question 4 (out of 8) | |
| **Total (out of 35)** | |

**Question 1 [8]** (a)[4] Answer Yes or No to the following questions:

1. Is Quicksort stable?
2. Is Counting sort stable?
3. Is Merge sort stable?
4. Is Insertion sort stable?

(b)[4] Consider the ZZZ-sort algorithm given in Assignment 1:

ZZZ-sort$(A, i, j)$:

```
1. if $A[i] > A[j]$ do
2.     swap $A[i]$ and $A[j]$
3. end if
4. if $i + 1 \geq j$
5.     return
6. end if
7. $k \leftarrow \lfloor (j - i + 1)/3 \rfloor$    % round down one third of the length of $A[i \dots j]$
8. ZZZ-sort$(A, i + k, j)$    % call recursively on the last two-thirds
9. ZZZ-sort$(A, i, j - k)$    % call recursively on the first two-thirds
10. ZZZ-sort$(A, i + k, j)$    % call recursively on the last two-thirds again
```

Answer Yes or No to the following questions:

1. Is ZZZ-sort in-place?
2. Is ZZZ-sort stable?

**Question 2 [7]** Suppose hypothetically that the product $C$ of two $n \times n$ matrices $A$ and $B$ (where $n$ is a power of 3):

$$C = A \times B$$

can be computed as follows: Divide each $A$, $B$, $C$ into 9 $\frac{n}{3} \times \frac{n}{3}$ submatrices $A^{i,j}$, $B^{i,j}$ and $C^{i,j}$ ($1 \le i \le 3$, $1 \le j \le 3$):

$$A = \begin{pmatrix} A^{1,1} & A^{1,2} & A^{1,3} \\ A^{2,1} & A^{2,2} & A^{2,3} \\ A^{3,1} & A^{3,2} & A^{3,3} \end{pmatrix} \qquad B = \begin{pmatrix} B^{1,1} & B^{1,2} & B^{1,3} \\ B^{2,1} & B^{2,2} & B^{2,3} \\ B^{3,1} & B^{3,2} & B^{3,3} \end{pmatrix} \qquad C = \begin{pmatrix} C^{1,1} & C^{1,2} & C^{1,3} \\ C^{2,1} & C^{2,2} & C^{2,3} \\ C^{3,1} & C^{3,2} & C^{3,3} \end{pmatrix}$$

Then the $\frac{n}{3} \times \frac{n}{3}$ submatrices of $C$ can be computed by performing in total 30 additions and substractions, and 20 multiplications between the $\frac{n}{3} \times \frac{n}{3}$ submatrices of $A$ and $B$.

(a)[3] Describe briefly a divide-and-conquer algorithm that computes the product of $n \times n$ matrices (for $n$ a power of 3) in the style of Strassen's algorithm using the above assumption.

(b)[2] Let $T(n)$ be the total number of arithmetic operations required for this algorithm. Write down the recurrence relation for $T(n)$ that allows you to apply the Master Theorem to derive $T(n)$ asymptotically.

(c)[2] What is $T(n)$? Give your answer using the $\Theta$ notation.

**Question 3 [12]** Consider the following problem, known as the "Counting Inversion Problem". The input is an array of $n$ distinct integers $A[1 \ \ldots \ n]$. An inversion in this array is a pair of indices $(i, j)$ such that $i < j$ and $A[i] > A[j]$. For example, suppose that $A = (3, 9, 5, 2)$, then there are 4 inversions in $A$: $(1, 4)$, $(2, 3)$, $(2, 4)$, and $(3, 4)$.

In the following parts you are asked to give a divide-and-conquer algorithm for this problem, explain how it works, and verify its running time. Your algorithm must run in time $\mathcal{O}(n \ln(n))$. (Hint: You may want to write a procedure Count-and-sort$(A, \ell, m, r)$ which, assuming that the subarrays $A[\ell \ \ldots \ m]$ and $A[(m+1) \ \ldots \ r]$ are already sorted, outputs the number of inversions $(i, j)$ where $\ell \leq i \leq m$ and $(m+1) \leq j \leq r$.)

(a)[3] Briefly explain how your algorithm works.

(b)[7] Give the pseudo-code for your algorithm. (You may use "swap $A[i] \leftrightarrow A[j]$" for a piece of code that exchanges the elements in $A$ at positions $i$ and $j$.)

**Question 3(b) continues here**

**Question 3 continues here**

(c)[2] Verify that your algorithm does run in time $\mathcal{O}(n \ln(n))$.

**Question 4 [8]** Throughout this question, $A$ is the array of length 9 whose element $A[i]$ is the $i$-th digit in your student number, for $1 \leq i \leq 9$. For example, if your student number is 123456789 then $A = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. This question is about running the Heapsort algorithm on this array.

(a)[2] State the Max-heap property.

(b)[1] Write down $A$, and draw the binary tree that $A$ represents, such that $A[1]$ is at the root, and the left child of $A[i]$ is $A[2i]$, the right child of $A[i]$ is $A[2i + 1]$.

(c)[2] Now make $A$ a heap by calling the procedure Build-max-heap(A). Draw the tree anew to show how it looks like every time it is changed during the execution of Build-max-heap(A).

**Question 4(c) continues here**

**Question 4 continues here**

(d)[1] Write down A as it is after Build-max-heap(A).

(e)[2] The next phase in Heapsort is to put elements of $A$ into their sorted position. Draw the tree and write down $A$ anew to show how they look like every time they are changed during this phase. Stop when three largest elements of $A$ are in their right place.

**Question 4(e) continues here**