**Last name(s)** **Name** **ID**

# Midterm EDA    Length: 2h30m    23/3/2015

- *The exam has 4 sheets, 8 sides and 4 problems.*
- *Write your full name and ID on every sheet.*
- *Write your answers within the reserved space.*
- *All your answers must be justified.*

## Problem 1: Cost analysis (2 points)

The sieve of Eratostenes (276–194 BC) is a method to generate all prime numbers smaller or equal than a given $n$. The method goes as follows: traversing the sequence of numbers $2, 3, 4, \ldots, n$, look for the next number $x$ that is not yet marked, mark all its multiples $2x, 3x, 4x, \ldots$ up to $n$, and repeat with the next $x$. When it finishes, those $x \geq 2$ that are not marked are the prime numbers. In C++:

```
vector<bool> M(n + 1, false);
for (int x = 2; x ≤ n; ++x) {
    if (not M[x]) {
        for (int y = 2*x; y ≤ n; y += x) M[y] = true;
    }
}
```

For the first three questions below we ask for an exact (non-asymptotic) expression *as a function of* $n$. If needed, use the notation $\lfloor z \rfloor$ that rounds $z$ to the maximum integer smaller or equal than $z$; for example, $\lfloor \pi \rfloor = \lfloor 3.14\ldots \rfloor = 3$.

(a) (0.33 points) How many times is $M[y]$ = **true** executed when $x$ is 2?

Answer: Exactly ____ many times.

(b) (0.34 points) How many times is $M[y]$ = **true** executed when $x$ is 15?

Answer: Exactly ____ many times.

(c) (0.33 points) How many times is $M[y]$ = **true** executed when $x$ is 17?

Answer: Exactly ____ many times.

**Note**: More questions on the back.

(d) (0.5 points) It is known that

$$\sum_{\substack{p=2 \\ p \text{ prime}}}^{n} \frac{1}{p} = \Theta(\log \log n).$$

Use this, in conjunction with the answers to the previous questions, to determine the cost of the algorithm as a function of $n$, in asymptotic notation. Answer: $\Theta($ $\boxed{\phantom{xxxxxxxxxx}}$ $)$.

Justification:

(e) (0.5 points) An improvement consists in replacing the condition $x \leq n$ in the external loop by $x{*}x \leq n$. Would this improve the asymptotic cost?

Answer and justification:

**Last name(s)** **Name** **ID**

**Problem 2: Strassen & company** **(2 points)**

The school algorithm to multiply two $n \times n$ matrices makes $\Theta(n^3)$ arithmetic operations. In 1969 Strassen found an algorithm that makes $\Theta(n^{2.81})$ arithmetic operations. Twenty one years later, Coppersmith and Winograd discovered a method that makes $\Theta(n^{2.38})$ operations.

Assuming (for simplicity) that the implicit constants in the $\Theta$ notation are 1, 10 and 100, respectively, and that they apply to every $n \geq 1$ (that is, the costs are $n^3$, $10n^{2.81}$ and $100n^{2.38}$, respectively, for every $n \geq 1$), compute the least $n$ for which one of these algorithms makes less operations than another.

(a) (1 point) For $n \geq$ ________, Strassen improves the school method.

(b) (1 punt) For $n \geq$ ________, Coppersmith-Winograd improves Strassen.

Justifications:

*This side would be left blank intentionally if it were not for this note.*

**Last name(s)** **Name** **ID**

**Problem 3: Algorithm design** **(3 points)**

Given a sequence of $n$ non-empty intervals $[a_1, b_1], \ldots, [a_n, b_n]$, we want to compute their union in time $O(n \log n)$. The output is represented by a sequence of disjoint intervals sorted by their left boundary. For example, if the intervals in the input were $[17, 19]$ $[-3, 7]$ $[4, 9]$ $[18, 21]$ $[-4, 15]$, then the intervals in the output would be $[-4, 15]$ $[17, 21]$.

(a) (1 point) Describe an algorithm that solves this problem. Explain the algorithm in words, without writing code, but clearly enough so that the algorithm can be implemented. Assume that the input is given by the vectors $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_n)$, with $a_i \leq b_i$ for every $i = 1, \ldots, n$.

(b) (1 point) Now, in addition to the sequence of $n$ intervals, we are given a sequence $m$ different reals $p_1, \ldots, p_m$, and we want to determine how many fall in some interval of the union (only the number is needed; not which ones). Using the algorithm from the previous section, describe an algorithm that solves this problem in time $O(n \log n)$ when $m = n$. Assume that the input is given by the vectors $a$ and $b$ from the previous section, and the vector $(p_1, \ldots, p_m)$ with $p_i \neq p_j$ if $i \neq j$.

(c) (1 punt) If you happened to know that $m$ is bounded by a small constant independent of $n$, say $m \leq 5$, would you still use the same algorithm? If not, which one would you use instead? If you choose a different algorithm, make its cost explicit.

**Last name(s)** **Name** **ID**

**Problem 4: Quickies** **(3 points)**

- (0,5 points) Determine if they are equal (=) or different ($\neq$), and prove it:

$$\Theta(3^{\log_2(n)}) \ \boxed{\phantom{x}}\ \Theta(3^{\log_4(n)}).$$

- (0,5 points) Compute $2^1 \cdot 2^2 \cdot 2^3 \cdot \dots \cdot 2^{99} \cdot 2^{100} \mod 9$. This problem is not meant to be solved with the help of a calculator.

- (1 point) Sort the functions that follow increasingly according to their asymptotic growth rates: $n^4 - 3n^3 + 1$, $(\ln(n))^2$, $\sqrt{n}$, $n^{1/3}$. Exceptionally, you are not required to justify your answer.

- (1 point) Consider the following three alternatives for solving a problem:
  - A: divide an instance of size $n$ into five instances of size $n/2$, solve each instance recursively, and combine the solutions in time $\Theta(n)$.
  - B: given an intance of size $n$, solve two instances of size $n-1$ recursively, and combine the solutions in constant time.
  - C: divide an instance of size $n$ into nine instances of size $n/3$, solve each instance recursively, and combine the solutions in time $\Theta(n^2)$.

  Write down and solve the corresponding recurrences. Which alternative is the most efficient?