

**Last name(s)**

**Name**

**ID**

**Midterm EDA**

**Length: 2.5 hours**

**31/03/2016**

- 
- The exam has 4 sheets, 8 sides and 4 problems.
  - Write your full name and ID on every sheet.
  - Write your answers to all problems in the exam sheets within the reserved space.
  - Unless otherwise indicated, all your answers must be justified.
- 

**Problem 1**

**(1 point)**

Answer the following questions. In this exercise, you do **not** need to justify the answers.

1. (0.2 points) The cost of Strassen's algorithm for multiplying two matrices of size  $n \times n$  is  $\Theta(\text{ } \text{ } )$

2. (0.6 points) The master theorem for subtractive recurrences states that given a recurrence  $T(n) = aT(n - c) + \Theta(n^k)$  with  $a, c > 0$  and  $k \geq 0$  then:

$$T(n) = \begin{cases} \text{ } & \text{if } a < 1, \\ \text{ } & \text{if } a = 1, \\ \text{ } & \text{if } a > 1. \end{cases}$$

3. (0.2 points) The mergesort sorting algorithm uses  $\Theta(\text{ } \text{ } )$  auxiliary space when sorting a vector of size  $n$ .

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

**Last name(s)**

**Name**

**ID**

**Problem 2**

**(3 points)**

In this problem costs are only considered *in time*. Consider the following program:

```
void f(int m);
void g(int m);

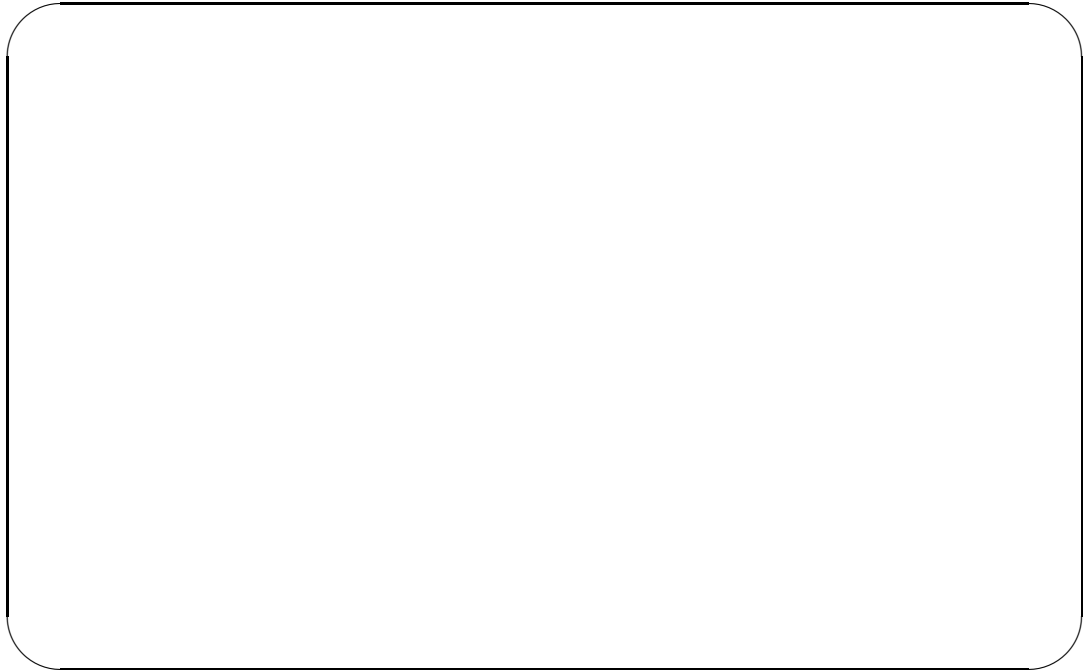
void h(int n) {
    int p = 1;
    for (int i = 1; i ≤ n; i++) {
        f(i);
        if (i == p) {
            g(n);
            p *= 2;
        }
    }
}
```

- (a) (1.5 points) Answer: if the cost of  $f(m)$  as well as the cost of  $g(m)$  is  $\Theta(m)$ , then the cost of  $h(n)$  as a function of  $n$  is  $\Theta(\text{ } \text{ } )$

Justification:

(b) (1.5 points) Answer: if the cost of  $f(m)$  is  $\Theta(1)$  and the cost of  $g(m)$  is  $\Theta(m^2)$ , then the cost of  $h(n)$  as a function of  $n$  is  $\Theta( \text{ } )$

Justification:



Last name(s)

Name

ID

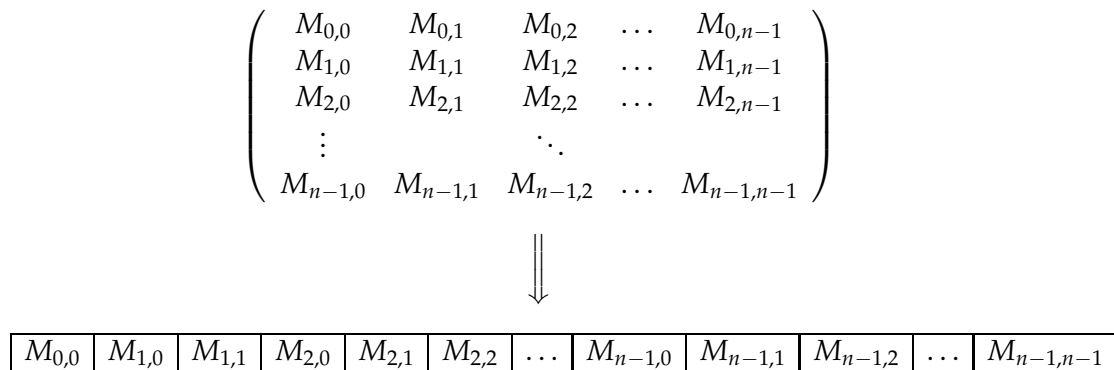



### Problem 3

(3 points)

We say that a square matrix  $M$  of integer numbers of size  $n \times n$  is *symmetric* if  $M_{i,j} = M_{j,i}$  for all  $i, j$  such that  $0 \leq i, j < n$ .

Let us consider the following implementation of symmetric matrices with unidimensional vectors, which only stores the “lower triangle” to minimise the consumed space:



For example, the symmetric matrix  $3 \times 3$

$$\begin{pmatrix} 1 & 2 & 0 \\ 2 & -2 & 3 \\ 0 & 3 & -1 \end{pmatrix}$$

would be implemented with the unidimensional vector

1	2	-2	0	3	-1
---	---	----	---	---	----

- (a) (0.5 points) Answer: the cost in space of this representation for a symmetric matrix  $n \times n$  as a function of  $n$  is  $\Theta(\text{  })$ .

Justification:

- (b) (1 point) Given  $n > 0$  and a value  $k$ , the function

```
pair<int,int> row_column(int n, int k);
```

returns the pair  $(i, j)$  where  $i$  is the row and  $j$  is the column of the coefficient pointed by  $k$  in a unidimensional vector that implements a symmetric matrix  $n \times n$ . If  $k$  is not a valid index, it returns  $(-1, -1)$ . For example, `row_column(3,2)` returns  $(1, 1)$ , `row_column(3,3)` returns  $(2, 0)$ , and `row_column(3,6)` returns  $(-1, -1)$ .

Complete the following implementation of `row_column`:

```
int p(int x) { return  ;}

int mystery(int k, int l, int r) {
    if (l+1 == r) return l;
    int m = (l+r)/2;
    if (p(m) ≤ k) return mystery(k, , r);
    else return mystery(k, l,  );
}

pair<int,int> row_column(int n, int k) {
    if (k < 0 or k ≥ p(n)) return  ;
    int i = mystery(k, 0, n);
    return {i,  };
}
```

- (c) (1 point) Analyse the cost in time in the worst case of the implementation of `row_column(int n, int k)` of the previous exercise as a function of  $n$ .

- (d) (0.5 points) Using the functions `double sqrt(double x)` and `int floor(double x)` which respectively compute  $\sqrt{x}$  and  $\lfloor x \rfloor$  in time  $\Theta(1)$ , give an alternative implementation of `row_column` with cost  $\Theta(1)$ .

Last name(s)

Name

ID

**Problem 4**

**(3 points)**

Assume that  $n$  is a power of 2, that is, of the form  $2^k$  for a certain  $k \geq 0$ .

A square matrix  $A$  of real numbers of size  $n \times n$  is a *Monge matrix* if for all  $i, j, k$  and  $l$  such that  $0 \leq i < k < n$  and  $0 \leq j < l < n$ , it holds that

$$A_{i,j} + A_{k,l} \leq A_{i,l} + A_{k,j}$$

In other words, whenever we take two rows and two columns of a Monge matrix and consider the four elements at the intersections of the rows and the columns, the sum of the elements at the upper left and lower right corners is less than or equal to the sum of the elements at the upper right and lower left corners. For example, the following matrix is a Monge matrix:

$$\begin{pmatrix} 10 & 17 & 13 & 28 \\ 16 & 22 & 16 & 29 \\ 24 & 28 & 22 & 34 \\ 11 & 13 & 6 & 6 \end{pmatrix}$$

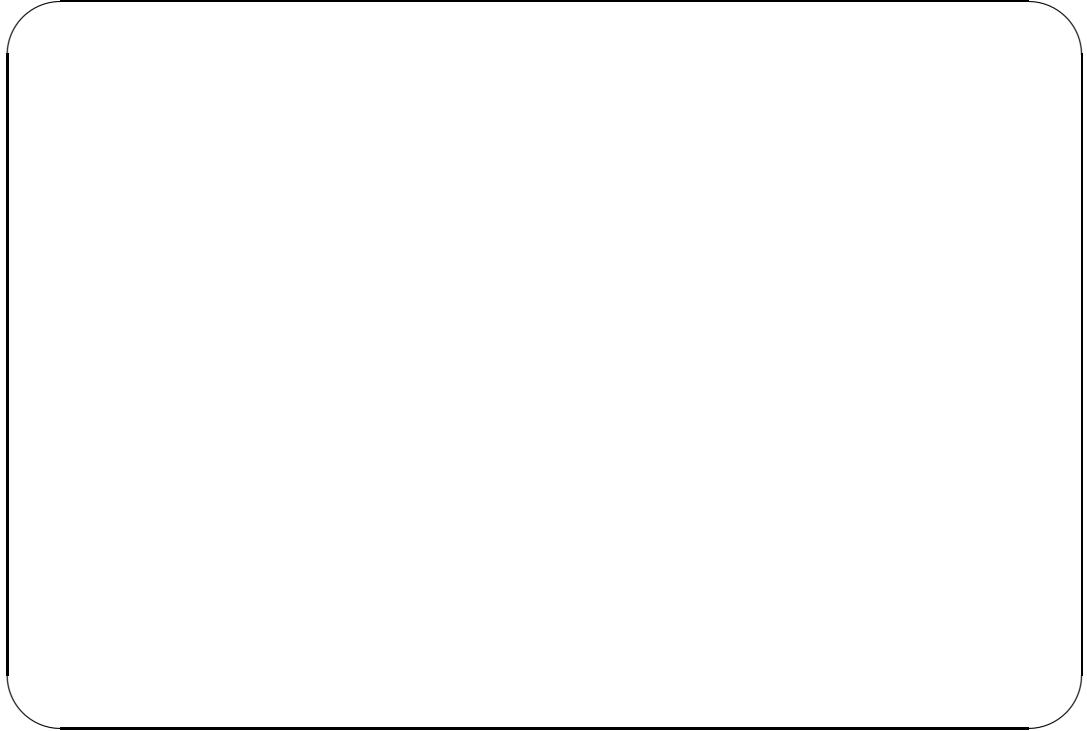
- (a) (1 point) Let  $f_A(i)$  be the index of the column where the the minimum element of the  $i$ -th row appears (breaking ties if needed by taking the leftmost one). For example, in the matrix above  $f_A(0) = 0$ ,  $f_A(1) = 0$ ,  $f_A(2) = 2$  i  $f_A(3) = 2$ .

Show that  $f_A(0) \leq f_A(1) \leq \dots \leq f_A(n-1)$  for any Monge matrix  $A$  of size  $n \times n$ .

(b) (1 point) In what follows a divide-and-conquer algorithm is described which computes the function  $f_A$  for all rows of a Monge matrix  $A$ :

- (1) Build two square submatrices  $B_1$  i  $B_2$  of the matrix  $A$  of sizes  $\frac{n}{2} \times \frac{n}{2}$  as follows:  $B_1$  is formed by the rows of  $A$  with even index and the columns between 0 and  $\frac{n}{2} - 1$ , and  $B_2$  is formed by the rows of  $A$  with even index and the columns between  $\frac{n}{2}$  and  $n - 1$ .
- (2) Recursively determine the column where the leftmost minimum appears for any row of  $B_1$  and for any row of  $B_2$ .
- (3) Find the column where the leftmost minimum appears for any row of  $A$ .

Explain how, by using the result of (2), one can accomplish (3) in time  $\Theta(n)$ .



(c) (1 point) Compute the cost as a function of  $n$  of the algorithm proposed in the previous exercise for computing the function  $f_A$  for all rows of a Monge matrix  $A$  of size  $n \times n$ . Assume that step (1) can be carried out in time  $\Theta(n)$ .

