Midterm

The lectures notes, personnal notes and a calculator are allowed for the test.

Exercice 1 Proof of algorithm

One considers an unsorted array T of integers. One denotes by m and n two indices $m \leqslant n$ (The first index in the array is 1, i.e. $1 \leqslant m \leqslant n$) and let x be an integer. The algorithm returns True or stop without any message.

Algorithme 1

Search(T,m, n, x)
if $n = 1$ and $T[1] = x$ **then**
  Print('True')
end if
$k \leftarrow \lfloor \frac{m+n}{2} \rfloor$ % Closest integer smaller than $\frac{m+n}{2}$
Search($T, k+1, n, x$)
Search(T, m, k, x)

1. One example : One considers the array $T = [1,3,2,0,6,4,9,5]$, what does Search(T,1,8,6) returns ? Same question for Search(T,1,8,7). What does the algorithm do ?
2. Prove by induction that the algorithm is correct.
3. Explain why the algorithm converge ?
4. What is the complexity of this algorithm (count 1 for each call of Search) ?
5. Provide an iterative version of Search. What is the complexity of the iterative version ?
6. Explain why the complexity of all algorithms that solve the same problem can not be better than $\mathcal{O}(n)$

Exercice 2 Complexity

In this exercise one considers the Fibonacci sequence :

$$\begin{cases} u_0 &= & 0 \\ u_1 &= & 1 \\ u_n &= & u_{n-1} + u_{n-2} \quad \forall n \geqslant 2 \end{cases} \tag{1}$$

1. Write an iterative algorithm, named Fibbonacci1, which, given n, retruns u_n.
2. What is the complexity of your algorithm ?
3. One considers the following recursive version :

Algorithme 2

Fibonacci2(n)
if $n < 2$ **then**
  return(n)
else Fibonacci2($n-1$)+Fibonacci2($n-2$)
end if

a. Let $s(n)$ be the function that counts the number of additions done in the call Fibonacci2(n). Express $s(n)$ in terms of $s(n-1)$ and $s(n-2)$.

b. Deduce by induction that $s(n) \geqslant u_n$ for all $n \geqslant 2$

c. We recall the following formula :

$$u_n = \frac{1}{\sqrt{5}}(\phi^n - \phi'^n), \text{ with } \phi = \frac{1+\sqrt{5}}{2} \text{ and } \phi' = -\frac{1}{\phi} \tag{2}$$

i. Show that $(u_n) \in \Theta(\phi^n)$.

ii. What is the complexity of Fibonacci2 ?

4. For all n one denotes $F_n = \begin{pmatrix} u_n \\ u_{n-1} \end{pmatrix}$ and $F_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

a. Find a 2×2 matrix A such that $F_n = AF_{n-1}$.

b. Deduce that $F_n = A^n F_0$.

c. What would be the complexity of an algorithm Fibonacci3 that calculates the terms of the Fibonnacci sequence using a matrix-version of the fast powering algorithm ?

5. The following Table provides in secondes the running times of the algorithms Fibonacci1, Fibonacci2 and Fibonacci3. The names of the algorithms have been changed.

n	100	1000	10000	20000	50000	80000	10^6	10^7
algo_f	∞	∞	∞	∞	∞	∞	∞	∞
algo_g	0	0	0.187	0.656	3.750	9.563	14.531	∞
algo_h	0.016	0.015	0.031	0.063	0.359	0.75	1.031	103

(3)

Identify algorithms algo_f, algo_g and algo_h with the correct Fibonacci1, Fibonacci2 and Fibonacci3.

6. The Fibonacci sequence (u_n) satisfy the following relations

$$\begin{cases} u_{2k} &= u_k(2u_{k+1} - u_k) \\ u_{2k+1} &= u_{k+1}^2 + u_k^2 \end{cases} \tag{4}$$

Without too much details, explain why there is a strategy which allows to propose an algorithm that compute the terms of the Fibonacci sequence in $\mathcal{O}(\ln(n))$.

Exercice 3 RSA/FFT

Alice and Bob are exchanging information using a RSA protocol. Alice has set up the protocol (n_A, e_A, d_A) where (n_A, e_A) is the public key and d_A is the secret key.

1. Certification. Bob knows that (n_A, e_A) is Alice public's key but he wants to make sure the person he is sending his encrypted message to is indeed Alice.

a. Suppose x is a sequence of integers which encodes the message 'I am Alice'. If Alice sends $x^{d_A} \bmod n_A$ to Bob what can Bob do to recover the message x ?

b. Why is Bob sure that only to Alice could have sent this message ?

2. Attack. One gives $n_A = 187$ $e_A = 7$.

a. Encode the message $x = 121$.

b. Assume you are Eve the eavesdropper and you intercepted a message $y = 48$ from Bob to Alice. What can you try to do to decode the message ?

c. Decode it !