Name:

Place number:

Permanent code:

Directives:

- Write you name, place number and permanent code.
- Put your student card in view.
- Read all questions and **answer directly on the questionnaire**.
- You can only use a pen or pencil, **no documentation, calculator, phone, computer or object**.
- This exam contains 25 questions of 4 points each, for a total of 100 points.
- This exam contains 17 pages, including 4 detachable pages at the end for drafting.
- For developing questions, **write clearly and detail your answers**.
- You have 165 minutes to complete this exam.

GOOD LUCK AND HAVE A GOOD SUMMER!

1	/ 4	11	/ 4	21	/ 4
2	/ 4	12	/ 4	22	/ 4
3	/ 4	13	/ 4	23	/ 4
4	/ 4	14	/ 4	24	/ 4
5	/ 4	15	/ 4	25	/ 4
6	/ 4	16	/ 4		
7	/ 4	17	/ 4		
8	/ 4	18	/ 4		
9	/ 4	19	/ 4		
10	/ 4	20	/ 4		
Total	/ 40	Total	/ 40	Total	/ 20
				Total	/100

Question 1 What will be displayed when executing the following `Python` code?:

```
x = [2*i for i in range(-32, 32, 2)]
print(x[-1])
```

Question 2 What will be displayed when executing the following `Python` code?:

```
def scale1(data, factor):
    for j in range(len(data)):
        data[j] *= factor
data = [i for i in range (4)]
scale1(data, 3)
print(data)
```

Question 3 What will be displayed when executing the following `Python` code?:

```
def scale2(data, factor):
    for val in data:
        val *= factor
data = [i for i in range (4)]
scale2(data, 3)
print(data)
```

Question 4 What will be displayed when executing the following `Python` code?:

```
def B(data):
    x = float('-inf')
    y = float('-inf')
    for i in data:
        if i > y:
            x, y = y, i
        elif i > x:
            x = i
    return (x, y)
data = [4, -8, 0, 3000, -1234, 45, 3, -6, -100]
print(str(B(data)))
```

Some reminders in Python:
1) in a sum of booleans, these are converted to integers (False = 0 and True = 1)
2) bool(i) = False if $i = 0$ and True otherwise
3) the evaluation of a clause like (A and B) or (A or B) is lazy : if the 1st boolean is sufficient, the 2nd one is not evaluated

When asked for a number of comparaisons, it has to be in function of n.

Question 5 Given the following recursive function:

```
def present_1(data, i, element):
    if i == len(data):
        return False
    else:
        return ( (data[i] == element) or present_2(data, i+1, element) )
```

When initially called with a 1st argument of length $n > 0$ and a 2nd argument equals to 0:

What does this function do?

How many comparaisons (==) does it perform in the **worst** case?

How many comparaisons (==) does it perform in the **best** case?

Question 6 Given the following recursive function:

```
def present_2(data, i, element):
    if i == len(data):
        return False
    else:
        return bool( (data[i] == element) + present_1(data, i+1, element) )
```

When initially called with a 1st argument of length $n > 0$ and a 2nd argument equals to 0:

Does the function `present_2` return the same result as the function `present_1`?

How many comparaisons (==) does it perform in the **worst** case?

How many comparaisons (==) does it perform in the **best** case?

Important
(1) Unless specified otherwise, all questions refer to data structures containing $\boldsymbol{n}$ **elements**.
(2) When asked for a complexity, we imply **in the worst case**.
(3) When asked for a complexity, you must give the set. For example, if the answer is $\mathcal{O}(n)$, **you will get no point for $\boldsymbol{n}$.**
(4) When asked for a $\mathcal{O}$ complexity, you must give the **lowest upper bound**. Similarly, when you are asked an Ω complexity, you must give the **greatest lower bound**. **No points will be given for a larger set.** For example, if the answer is $\mathcal{O}(n)$, you will get no point for $\mathcal{O}(n!)$.
(5) When asked for a complexity, you must give the most reduced expression describing it. For example, if the answer is $\mathcal{O}(n^2)$, **you will get no point for $\boldsymbol{\mathcal{O}(6n^2+7n+42)}$.**

Question 7 Let L be a **Dynamic** array-based **List**.

What is its $\mathcal{O}$ space complexity?

What is its Ω space complexity?

What is the $\mathcal{O}$ time complexity of accessing an element through an index?

What is the $\mathcal{O}$ time complexity of inserting an element at some index?

What is the $\mathcal{O}$ time complexity (amortized) of adding an element at end of the list (*append*)?

Question 8 Let S be a **Dynamic** array-based **Stack**.

What is its $\mathcal{O}$ space complexity?

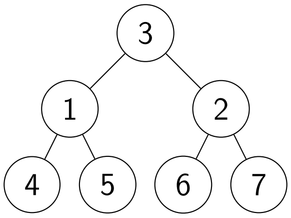
What is the $\mathcal{O}$ time complexity (amortized) of pushing an element on a stack?

What is the $\mathcal{O}$ time complexity (amortized) of pushing n elements on an initially empty stack?

Question 9 Let L be a **Doubly Linked List**.

What is the $\mathcal{O}$ time complexity of accessing the element at the tail?

What is the $\mathcal{O}$ time complexity of deleting an element on which a pointer is given?

Question 10 In what order would be visited the nodes of the tree  during:

a pre-order traversal?

an in-order traversal?

a post-order traversal?

a breadth-first traversal?

Question 11 Let T be a binary tree of which the following traversals would visit the keys in these orders:
in-order: H T U R A D R N E T
post-order: H U T R D N T E R A
In what order would be visited the nodes during:

a breadth-first traversal?

a pre-order traversal?

Question 12 We want to sort the numbers contained within the following array with the in-place heapsort algorithm. Fill in the following table, which must represent the state of the array after phase I of the sort algorithm, that is, after the heap construction. You will get points only if the table is exact.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	0

Question 13 Let H be a heap encoded in an array starting with index 0.

a) At which index(ices) of the heap's array could be found the 10th smallest key?

b) If the key k can be found at any index i such that $1 \leq i \leq 62$, what would be the rank of k if the keys were sorted $(1^{\text{st}}, 2^{\text{nd}}, 3^{\text{rd}}, ...)$?

Question 14 Draw the 11-entry hash table that results from using the hash function, $h(i) = (3i - 5) \bmod 11$, to hash the keys, in this order, 5, 11, 12, 13, 16, 20, 23, 39, 44, 88 and 94, assuming collisions are handled by linear probing.

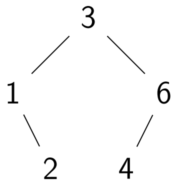
0	1	2	3	4	5	6	7	8	9	10

Question 15 Draw the 11-entry hash table that results from using the hash function, $h(i) = (3i - 5) \bmod 11$, to hash the keys, in this order, 94, 88, 44, 39, 23, 20, 16, 13, 12, 11 and 5, assuming collisions are handled by double hashing using the secondary hash function $h'(k) = 7 - (k \bmod 7)$.

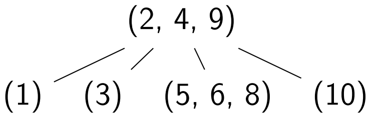
0	1	2	3	4	5	6	7	8	9	10

Question 16

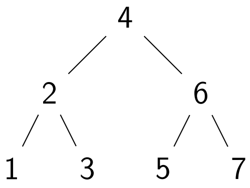
A function in the set $\mathcal{O}\left(n^2\right)$ is necessarily in the set $\mathcal{O}(n)$.	TRUE	FALSE
Which one(s) of these two data structures is(are) usually used to accumulate recursive calls?	a queue	a stack
What is the lowest superior bound for the complexity of swapping two nodes of a doubly-linked list containing n nodes, on which two pointers are given, in the worst case?	$\mathcal{O}(1)$	$\mathcal{O}(n)$
Which one(s) of these two types of traversal guarantee(s) to traverse the nodes of a binary search tree in a sorted order?	in-order	breadth-first
A hash table allows the traversal of its n keys in a sorted order in $\mathcal{O}(n)$.	TRUE	FALSE
The search of a key in an AVL tree, a Spay tree, a (2, 4) tree or a red-black tree containing n keys is guaranteed in $\mathcal{O}(n \lg n)$.	TRUE	FALSE
What is the lowest superior bound for the complexity of heap-sorting n elements, in the worst case?	$\mathcal{O}(n \lg n)$	$\mathcal{O}\left(n^2\right)$
A function in the set $\mathcal{O}\left(n\right)$ is necessarily in the set $\mathcal{O}\left(n^2\right)$.	TRUE	FALSE



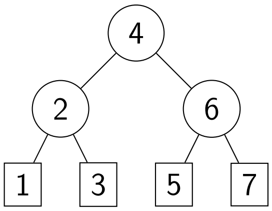
Question 17 Re-draw the AVL tree
after the **insertion** of **key 5**:



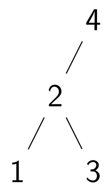
Question 18 Re-draw the (2, 4)-tree
after the **insertion** of **key 7**:



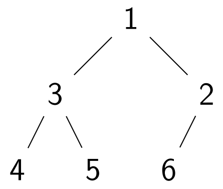
Question 19 Re-draw the Splay Tree
after the **deletion** of **key 7**:



Question 20 Re-draw the red-black tree
after the **insertion** of **key 8** (◯ = black, □ = red):

Question 21 Circle the correct answer. Can the tree  be:

A binary tree?	YES	NO
A heap?	YES	NO
A binary research tree?	YES	NO
An AVL tree?	YES	NO
A (2, 4)-tree?	YES	NO
A Splay Tree?	YES	NO

Question 22 Circle the correct answer. Can the tree  be:

A binary tree?	YES	NO
A heap?	YES	NO
A binary research tree?	YES	NO
An AVL tree?	YES	NO
A (2, 4)-tree?	YES	NO
A Splay Tree?	YES	NO

Question 23 Following is a dynamic programming table to find a longest common subsequence between two strings:

		m	y	m	m	e	c	a	c	e	c	o	i	t	
		0	1	2	3	4	5	6	7	8	9	10	11	12	13
t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
l	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
y	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1
m	3	0	0	1	1	1	1	1	1	1	1	1	1	1	1
m	4	0	1	1	2	2	2	2	2	2	2	2	2	2	2
e	5	0	1	1	2	3	3	3	3	3	3	3	3	3	3
p	6	0	1	1	2	3	4	4	4	4	4	4	4	4	4
f	7	0	1	1	2	3	4	4	4	4	4	4	4	4	4
c	8	0	1	1	2	3	4	4	4	4	4	4	4	4	4
u	9	0	1	1	2	3	4	5	5	5	5	5	5	5	5
p	10	0	1	1	2	3	4	5	5	5	5	5	5	5	5
i	11	0	1	1	2	3	4	5	5	5	5	5	5	5	5
t	12	0	1	1	2	3	4	5	5	5	5	5	5	6	6
	13	0	1	1	2	3	4	5	5	5	5	5	5	6	7

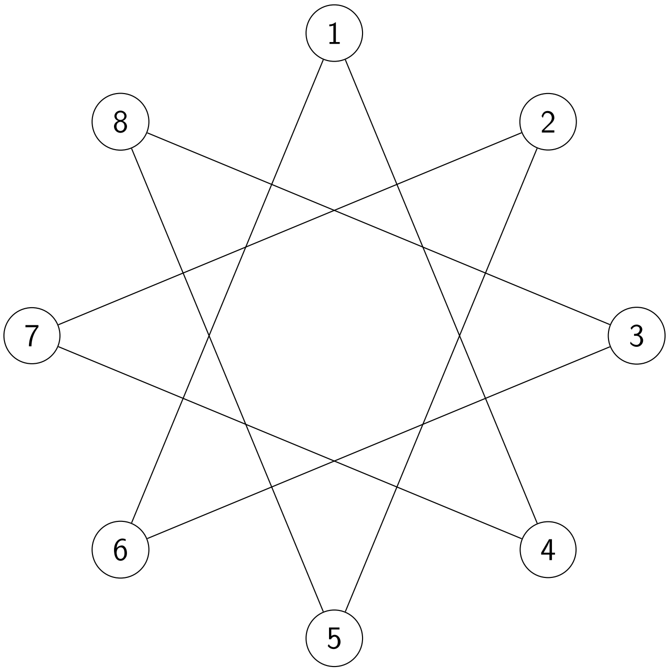
a) What is the longest common subsequence?

b) Blacken in the table the path to obtain this longest common subsequence.

Question 24 Draw the standard trie containing the following strings (conserve the alphabetic order of the childs):

```
{ arbre, trie, arc, tree, arete, cycle, clique }
```

Question 25 Let G be the following graph:



While respecting the increasing order of adjacent nodes, in what order will be visited the nodes, if starting at node 7:

a) during a depth-first search? (only one possible answer)

b) during a breadth-first search? (only one possible answer)

Draft:

Draft:

Draft:

Draft: