Level: 2nd year "Computer Science"	**Module:** Algorithmic and Data Structures 3	**January 2026**

Typical Correction

Exercise n° 1 (5 pts): Given the following array: 2 7 4 9 1 6 3

1. Provide the successive states of the array at the end of each step of the internal loop 'for' when i = 6. **(1.5 pts)**

   k=0: 2 7 4 9 1 6 3 k=1: 2 4 7 9 1 6 3 k=2: 2 4 7 9 1 6 3
   k=3: 2 4 7 1 9 6 3 k=4: 2 4 7 1 6 9 3 k=5: 2 4 7 1 6 3 **9**

2. Provide the successive states of the array at the end of each step of the external loop. **(1.5 pts)**

   i=6: 2 4 7 1 6 3 **9** i=5: 2 4 1 6 3 **7 9** i=4: 2 1 4 3 **6 7 9**
   i=3: 1 2 3 **4 6 7 9** i=2: 1 2 **3 4 6 7 9** i=1: 1 **2 3 4 6 7 9**

3. This bubble sort algorithm continues its iterations even if the array is sorted in the early iterations. Propose an improvement to this algorithm to allow it to stop as soon as the array is sorted and explain the time complexity in the best and worst cases. **(2 pts)**

```
void ImprouvedBubbleSort (int* t, int n) {
  int i, k;   int swapped ;
  for (i = n - 1; i > 0; i--) {
      swapped = 0;    // Set the swapped flag to false
      for (k = 0; k < i; k++)
          if (t[k] > t[k + 1]){
              swap(&t[k], &t[k + 1]);
              swapped = 1;   // Set the flag to true if a swap occurred
          }
       if (swapped == 0)  break;
  }
}
```

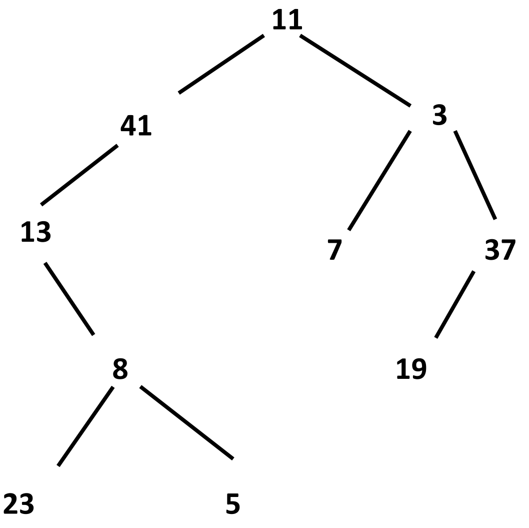
Worst-Case Time Complexity: O(n²)
Best-Case Time Complexity: O(n)

Exercise n° 2 (5 pts): Given the following table that represents a binary tree T in triplets (info, left, right):

5	3	8	23	7	37	13	11	41	19
-1	4	3	-1	-1	9	-1	8	6	-1
-1	5	0	-1	-1	-1	2	1	-1	-1

- Node 0 has info = 5, left = -1 (no left child), and right = -1 (no right child).
- Node 1 has info = 3, left = 4 (index of left child), and right = 5 (index of right child).
- Node 2 has info = 8, left = 3 (index of left child), and right = 0 (index of right child)….

1. Draw the binary tree T. **(1.5 pts)**



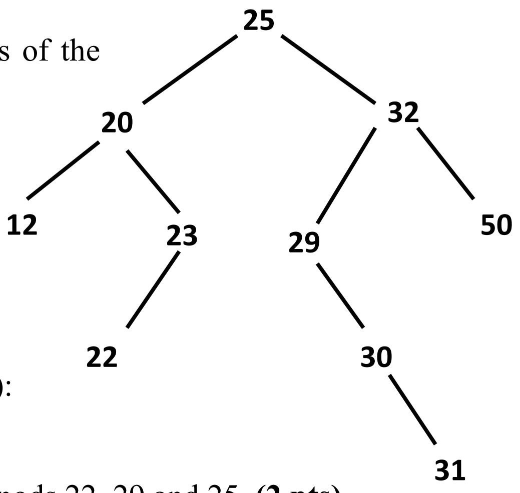
2. Write the C code for the data structure to represent tne tree T in this way. **(1.5 pts)**

```
typedef struct Node {
   int info;
   int left;
   int right;
} Node;
Node T[10] = {{5, -1, -1}, {3, 4, 5}, {8, 3, 0}, {23, -1, -1}, {7, -1, -1},
{37, 9, -1},{13, -1, 2}, {11, 8, 1}, {41, 6, -1}, {19, -1, -1}};
```

3. Write a function that returns the number of leaf nodes. **(2 pts)**

```
int nberLeafnodes (Node T[ ], int size) {

   int s=0;

   for (int i = 0; i < size; i++) {

             if (T[i].left = = -1   &&  T[i].right  = = -1)

                    s = s+1;

  return s;

}
```

Exercise n° 3 (6 pts):



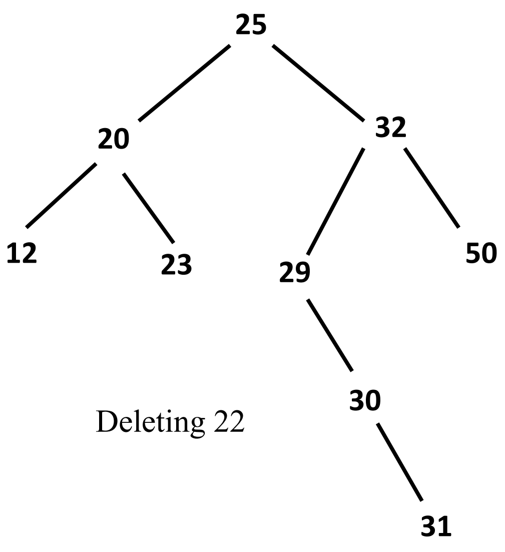
1. Give the result of the three depth-first traversals of the following binary search tree. **(1.5 pts)**

   In-Order Traversal (Left, Root, Right):
   12, 20, 22, 23, 25, 29, 30, 31, 32, 50.

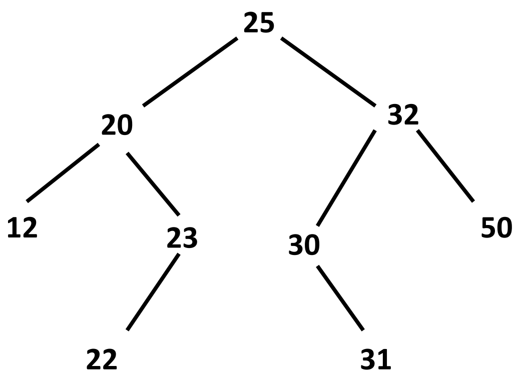
   Pre-Order Traversal (Root, Left, Right):
   25, 20, 12, 23, 22, 32 ,29, 30, 31, 50.

   Post-Order Traversal (Left, Right, Root):
   12, 22, 23, 20, 31, 30, 29, 50, 32, 25.

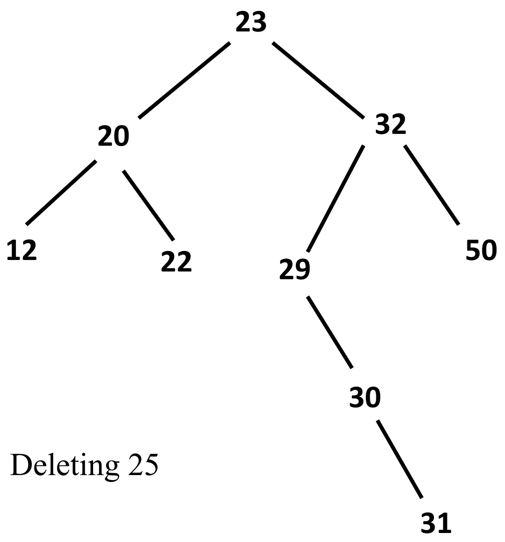
2. Show the new trees obtained after deleting these nods 22, 29 and 25. **(2 pts)**



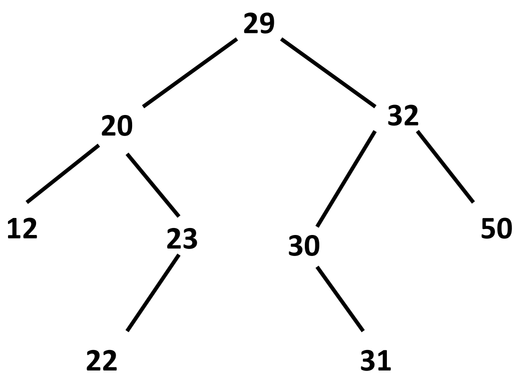
Deleting 22



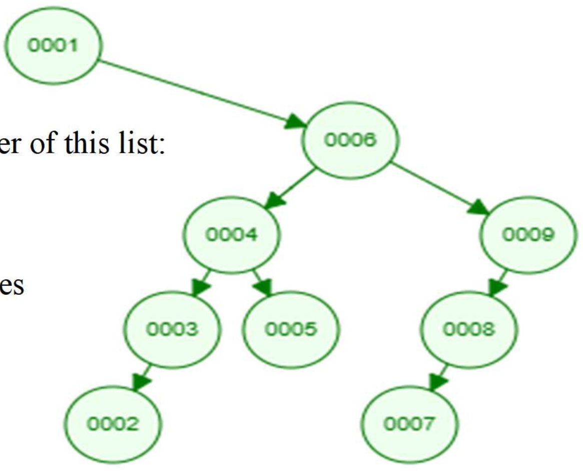
Deleting 29



Deleting 25



3. Construct the associated **BST** in the order of this list:
   1, 6, 9, 8, 7, 4, 3, 2, 5 **(1 pt)**



4. Write a recursive function that compares two binary search trees. The function returns 1 if the two trees are identical and 0 otherwise. **(1.5 pts)**

```
int IdenticalTrees(Node *R1, Node *R2) {

   if (R1 == NULL && R2 == NULL)      return 1;

   if (R1 == NULL || R2 == NULL)      return 0;

   return ((R1->data == R2->data) && IdentivalTrees(R1->left, R2->left)

                                  && IdenticalTrees(R1->right, R2->right));    }
```

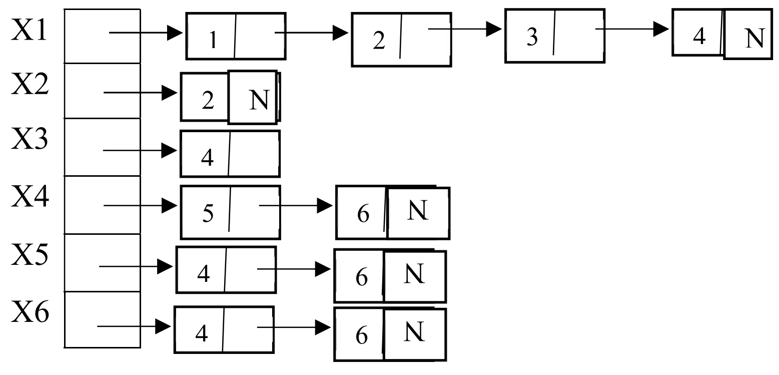
Exercise n° 4 (4 pts):

- Give all possible representations of this graph. **(2 pts)**

Adjacency Matrix Representation:

	X1	X2	X3	X4	X5	X6
X1	1	1	1	1	0	0
X2	0	1	0	0	0	0
X3	0	0	0	1	0	0
X4	0	0	0	0	1	1
X5	0	0	0	1	0	1
X6	0	0	0	1	0	1

Adjacency List Representation:



- Give the traversal results of this graph. **(2 pts)**
  BFS: 1,2,3,4,5,6
  DFS: 1,2,3,4,5,6