Data Structures – final test

Name: _____ Student ID: _____ Date: _____

Duration: 60 minutes · **Maximum score:** 100 points · **Aids:** none allowed.
In the pseudocode, arrays are indexed **from 1**. The fields of list elements are `Elem[P].Value` and `Elem[P].Next`; the fields of tree nodes are `Elem[P].Key`, `Elem[P].Left` and `Elem[P].Right`. The empty pointer is denoted by **Nil**.

Task 1 – True or false?

14 × 1 = 14 points

Mark the correct answer with an X!

#	Statement	True	False
1.	The elements of an array are stored contiguously in memory, one after the other.		
2.	The *i*-th element of a singly linked list can be accessed in O(1) steps.		
3.	Binary search always gives a correct result on an unsorted array too.		
4.	Using dynamic data structures makes resource sharing possible.		
5.	An empty sentinel list needs no storage, since it stores no elements.		
6.	The stack is a LIFO (*last in, first out*) data structure.		
7.	In a circular queue, dequeue takes O(n) steps because the elements must be shifted forward.		
8.	Insertion sort finishes in O(n) steps on an already sorted array.		
9.	Quicksort takes $O(n^2)$ steps in the worst case.		
10.	Merge sort is a stable sorting algorithm.		
11.	A binary search tree is always a balanced binary tree.		
12.	Searching in a binary search tree can be as slow as searching in a linked list.		
13.	Incorrect use of dynamic data structures can cause memory leaks.		
14.	In Java the `==` operator must be used to compare the contents of two `String`s.		

Task 2 – Multiple choice

10 × 1 = 10 points

Circle the only correct answer!

1. Which property **cannot** a singly linked list have?

a) scattered storage | b) direct access | c) homogeneous | d) dynamic

2. The binary tree, as a data structure, is:

a) heterogeneous | b) directly accessible | c) hierarchical | d) none of these

3. The elements of a sentinel list can be accessed:

a) in inorder | b) directly | c) sequentially | d) none of these

4. Which data structure has **direct** access?

a) doubly linked list | b) one-dimensional array | c) linked stack | d) circular list

5. The keys of a binary search tree are listed in increasing order by the:

a) preorder traversal | b) inorder traversal | c) postorder traversal | d) level-order traversal

6. How many steps does binary search take in a sorted array of n elements?

a) O(1) | b) O(log n) | c) O(n) | d) O(n log n)

7. An `int A[6][10]` matrix is stored in row-major order from address 1000, with 4-byte elements, indexed **from 0**. What is the address of element `A[2][3]`?

a) 1092 | b) 1132 | c) 1128 | d) 1052

8. Which sorting algorithm performs at most $n - 1$ swaps?

a) bubble sort | b) selection sort | c) insertion sort | d) quicksort

9. What is the value of the postfix expression `5 1 2 + 4 * + 3 -`?

a) 14 | b) 12 | c) 18 | d) 20

10. In a circular queue of 8 elements (indexes 0–7), `front = 6` and `count = 3`. At which index is the next enqueued element placed?

a) 0 | b) 1 | c) 9 | d) 7

Task 3 – Number of steps

10 × 1 = 10 points

Give the number of steps of the operations in O notation (n is the number of elements)!

	Operation	Steps
a)	Accessing the i-th element of an array	
b)	Linear search in an array of n elements (worst case)	
c)	Inserting a new element into a sorted array of n elements so that it stays sorted	
d)	Insertion at the beginning of a singly linked list	
e)	Deletion from the end of a singly linked list	
f)	Push and pop in a stack implemented with an array	
g)	Removing the maximum from a heap of n elements	
h)	Search in a balanced binary search tree	
i)	Merge sort	
j)	Towers of Hanoi with n disks (the number of moves)	

Task 4 – Code tracing: stack

7 points

What does the following code print? In general, what does it do with an arbitrary text? (`Stack` is a stack of characters.)

```
String t = "DATA STRUCTURE";
Stack s = new Stack();
for (char c : t.toCharArray()) {
    if (c != ' ') s.push(c);
    else {
        while (!s.isEmpty()) System.out.print(s.pop());
        System.out.print(' ');
    }
}
while (!s.isEmpty()) System.out.print(s.pop());
```

Output	
In general	

Task 5 – Code tracing: queue

7 points

What does the following code print? In general, which well-known problem does it solve if the numbers 1, 2, …, *n* are put into the queue?

```
Queue q = new Queue();
for (int i = 1; i <= 6; i++) q.enqueue(i);
while (q.size() > 1) {
    q.enqueue(q.dequeue());
    System.out.print(q.dequeue() + " ");
}
System.out.print(q.dequeue());
```

Output	
In general	

Task 6 – Executing algorithms on paper

4 + 4 = 8 points

a) Sort the array `[6, 2, 9, 1, 5]` with **insertion sort**! Write down the state of the array after every step of the outer loop!

b) Find `45` with **binary search** in the array below (indexed from 1)! Write down the values of `left`, `right` and `k` in every step! `A = [4, 9, 13, 21, 30, 38, 45, 52, 60, 71]`

Task 7 – Binary search tree

3 + 3 + 3 + 3 = 12 points

a) Insert the following keys in this order into an empty binary search tree, and draw the tree!
`45, 20, 70, 10, 30, 60, 80, 25, 35`

b) Give the preorder, inorder and postorder traversals of the tree!

	Traversal
Pre	
In	
Post	

c) Delete the key `20` from the tree obtained in a)! Draw the result, and state which case of deletion it is!

d) What is the height of the tree obtained in a)? How many key comparisons are needed to find `35`?

In Tasks 8–10 complete the pseudocode! Each missing part is an expression, a pointer or a **relational operator** (=, ≠, <, >, ≤, ≥). Write your answers into the table next to the code!

Task 8 – Search in a binary search tree

5 × 2 = 10 points

The function returns a pointer to the node with key `x`, or Nil if there is no such key. The **Root** pointer points to the root of the tree.

```
Function Search(Root, x)
  P ← __(1)__
  While P __(2)__ Nil and
        Elem[P].Key __(3)__ x
    If x __(4)__ Elem[P].Key then
      P ← Elem[P].Left
    else
      P ← __(5)__
    End if
  End while
  Return P
End function
```

	Missing part
(1)	
(2)	
(3)	
(4)	
(5)	

Task 9 – Sentinel search

5 × 2 = 10 points

The sentinel list is given by the **Head** and **End** pointers; End points to the sentinel element at the end of the list. The function returns a pointer to the first element containing `x`, or Nil if `x` is not in the list.

```
Function SentinelSearch(Head, End, x)
  Elem[End].Value ← __(1)__
  P ← __(2)__
  While Elem[P].Value __(3)__ x
    P ← __(4)__
  End while
  If P __(5)__ End then
    Return Nil
  else
    Return P
  End if
End function
```

	Missing part
(1)	
(2)	
(3)	
(4)	
(5)	

Task 10 – Insertion at the end of a singly linked list

6 × 2 = 12 points

The list is given by the **Head** pointer. The function `Reserve(P)` allocates a new element: on success P points to the allocated element and the function returns true.

```
Function InsertLast(Head, Data)
  If not Reserve(P) then Return false
  Elem[P].Value ← Data
  Elem[P].Next ← __(1)__
  If Head __(2)__ Nil then
    Head ← __(3)__
  else
    Q ← Head
    While Elem[Q].Next __(4)__ Nil
      Q ← __(5)__
    End while
    __(6)__ ← P
  End if
  Return true
End function
```

	Missing part
(1)	
(2)	
(3)	
(4)	
(5)	
(6)	

Scoring

Task	1	2	3	4	5	6	7	8	9	10	Total
Maximum	14	10	10	7	7	8	12	10	10	12	**100**
Achieved											

Score	0–49	50–61	62–74	75–86	87–100
Grade	fail (1)	pass (2)	satisfactory (3)	good (4)	excellent (5)

Marked by: ..... **Grade:** .....