Fourth written examination of

Algoritmos e Estruturas de Dados

November 30, 2015 Duration: no more than 30 minutes

Name:

Student number:

11.0 **1:** **The following C code declares a data type that can be used to store a node of an ordered binary tree.**

```
typedef struct tree_node
{
  tree_node *left;  // smaller descendants
  tree_node *right; // larger descendants
  int value;        // the value (an integer)
}
tree_node;
```

The root of the tree is stored in the variable `root`**.**

3 **1a:** **Consider an initially empty ordered binary tree. Draw the tree after the integers 7, 2, 9, 3, 4, 12, 1, and 8 have been put in it.**
Answer:

4 **1b: Write a recursive function, named `tree_depth`, such that `tree_depth(root)` returns the height of the tree (i.e., the largest depth of a node of the tree). Assume that the root of the tree has depth 1, so that an empty tree as depth 0.**

2 **1b: As an alternative, write a recursive function, named `tree_size`, such that `tree_size(root)` returns the number of nodes of the entire tree.**

Answer:

4 **1c: Write a recursive function, named** `reverse_print`**, such that** `reverse_print(root,0)` **prints the contents of the tree in decreasing order, placing before each node value its printing order (0 for the first printed number, 1 for the second, etc.).**

2 **1c: As an alternative, write a recursive function, also named** `reverse_print`**, such that** `reverse_print(root)` **prints the contents of the tree in decreasing order.**

Answer:

4.0 **$\boxed{\textbf{2:}}$ Explain how mergesort works.**

Answer:

5.0 **$\boxed{\textbf{3:}}$ Explain the differences between a depth-first search and a breadth-first search. Which data structure is usually used to implement each one?**

Answer: