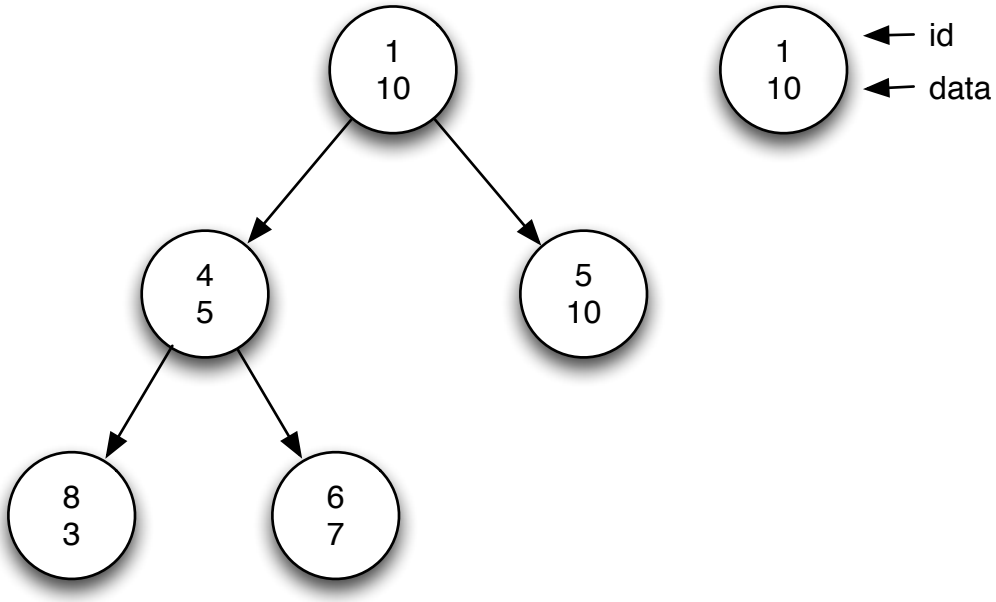
## Software Engineering II: Algorithms and Data Structures - Assignment 2

Aim:
The objective of the assignment is to test students' ability to operate on the binary tree structure and to develop functions / procedures to operate on that structure.

Details:
The assignment assumes an **ordered** tree to be in place (ascending order as it has been defined in the lectures). The figure below illustrates an instance of a such a binary tree data structure. Each node stores a unique identifier (i.e. *id*) and an integer (i.e. *data*) in the range [1, 200].




Binary Tree structure

The structure to store each node of the tree is defined as follows:

```
struct CP {
        int id;
        int data;
        CP *left;
        CP *right;
}
```

A template program is given to you where the constructStructure() function constructs an instance of the above tree. Extend the program by including functions / procedures that perform the following:

1) Write a recursive function / procedure that takes as argument the pointer to the root of the tree, and returns the number of nodes. Illustrate the functionality of your function by calling the function and display on the console the resulting

number.

2) Write a recursive function / procedure that takes as argument the pointer to the root of the tree and returns the number of nodes in the structure whose *data* field is an even number. Illustrate the functionality of your function by calling the function and display on the console the resulting number.

3) Write a recursive function / procedure that takes as argument the pointer to the root of the tree, and returns a *pointer* to the node with the maximum *data* value stored in the tree (in case of multiple nodes with the same value, the function can return any valid pointer Illustrate the functionality of your function by calling the function and displaying on the console the resulting node's *id* and *data* fields.

4) Write a recursive function / procedure that takes as argument the pointer to the root of the tree and returns the sum of all data stored in the binary tree structure. Illustrate the functionality of your function by calling the function and display on the console the resulting number.

5) Write a recursive function / procedure that takes as arguments a pointer to the root of the tree and an *id* number, and returns the depth of the node with the given *id*. Illustrate the functionality of your function by calling the function and displaying on the console the resulting number. If such a node does not exist, your function should indicate that, and your program should display on the console a relevant message.

Development issues:

1. Functions/procedures related to questions 1-5 shouldn't contain any I/O (shouldn't read or write to or from console or files) but only communicate with the main or other functions using the parameter list and/or the return value. The display of the results should be done by your program in the *main( )* function, after the control has returned from the specialised function/procedure that addresses questions 1-5.
2. Your functions/procedure should also handle an empty tree case.

Marking scheme:

This is a not assessed assignment and it is used to help you to learn how to program. Please make use of the computing lab if you have problems doing this exercise.

Deliverables:

There is no deadline for this submission, but please make use of the computer lab hours to get feedback on your work.