A small tree: How to think like a computer

An example of storing a general tree; which for simplicity here is assumed to be rooted and directed.

The first drawing depicts a tree with a root in which some key value F is stored. The root has two children with key values A and F.

The second drawing represents the tree using records and pointers. Each vertex is represented by a record with three fields. The first field stores the key, the second field stores a pointer to the oldest child, and the third field stores a pointer to the next sibling.

Pointers are just memory addresses. The third drawing shows how this tree could be stored in the memory. We have a memory of length 24 with each location identified by its number (address) from 0 to 23. There is a variable Tree in our program, which stores the location of the record corresponding to the root. This variable is stored in location 13. That location stores the value of 16. The root is stored in three consecutive locations, starting with location 16. The value stored are, the key F, the location of the oldest child 8, the location of the next sibling, here Nil, denoted here by -1. Continuing, the oldest child is stored in three consecutive locations, starting with location 8. Etc. The empty locations in the drawing are not empty, there is always something stored in every location of the memory. We just do not care in our example what is stored there.

Of course, the memory layout is not completely fleshed out. The location storing a pointer could be 8 bytes in a 64-bit system and location storing a key requires the length able to store the key. To make everything simple for the drawing, we assume that every item of interest fits in location of size of one byte.