Stack Allocation of Space :-

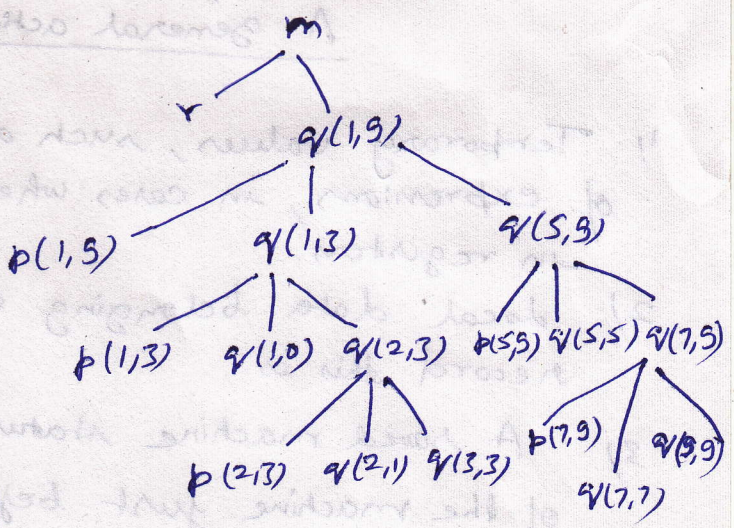
Almost all compilers for languages that use procedures, functions, or methods as units of user-defined actions manage at least part of their run-time memory as a stack. Each time a procedure is called, space for its local variables is pushed onto a stack.

Activation Trees :

Stack allocation would not be feasible if procedure calls, or activations of procedures, did not nest in time. Activation of procedures during the running of an entire program represented by a tree called an activation tree. Each node corresponds to one activation, and the root is the activation of the 'main' procedure that initiates execution of the program. At a node for an activation of procedure p, the children correspond to activations of the procedures called by this activation of p. We show these activations in the order that they are called, from left to right. One child must finish before the activation to its right can begin.

```
enter main()
    enter readArray()
    leave readArray()
    enter quicksort(1,9)
        enterpartition(1,9)
        leavepartition(1,9)
        enter quicksort(1,3)
        ...
        leave quicksort(1,3)
        enter quicksort(5,9)
        ...
        leave quicksort(5,9)
    leave quicksort(1,9)
```

Possible activations for the program



Activation tree representing calls during an execution of quicksort