CO322: Data structures and algorithms

Stacks

Stacks are Last-In-First-Out (LIFO) data structures which are useful in number of situations.

One particular application of a stack is when calling functions. Current function state (local variables, position etc.) would be pushed to a stack and then one can go and execute the called function. When the called function returns one can restore the stack and thereby return to calling function.

Recursion is a special case of a function call where the same function is calling on itself. Recursion is a natural fit to many different algorithms. One classic example of recursion is the *Tower of Hanoi* problem (see https://en.wikipedia.org/wiki/Tower_of_Hanoi).

You are provided with a C implementation of a solution to this problem. You may find the complete implementation from Moodel. The gist of the implementation is given below:

```
void move(char from, char to, char via, unsigned int disks)
{
  if(disks) {
    move(from, via, to, disks-1);
    printf("Move disks %d from %c to %c\n", disks, from, to);
    move(via, to, from, disks-1);
  }
}
```

Tasks 1: Start with a small number of disks (say 3 or more) and trace the recursive calls and what will be stored in the thread stack and how that would be resorted.

You do not have to submit this; but one of you would be selected randomly during the lecture to explain how this works for 3 disks.

Tasks 2: Do a Java based implementation of the Tower of Hanoi problem. Specifications are as follows:

1. You are **not allowed to use recursion**.
2. Number of disks and the towers (from, to, via) should be passed via the command-line.
3. Suitable **error checking** must be done.

Submit your answer via Moodle on or before 13th February 2018 2300hr.