Artificial Intelligence (CS13217)

Lab Report

Name:	Sameea Naeem
Registration #:	CSU-XS16-139
Lab Report #:	01
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 1
Implementing Tower Of Hannoi

Objective
To understand and implement tower of hannoi.

Software Tool
1. Sub lime text
2. python
3. miktex

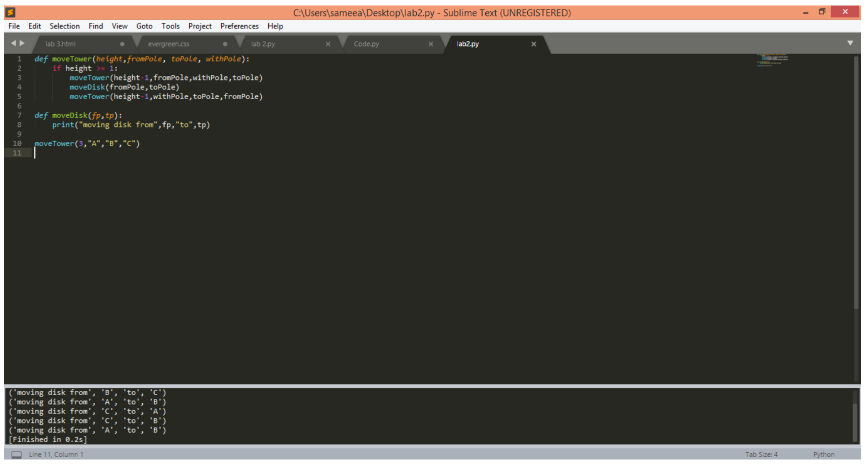
1 Theory

The rules of the game are very simple, but the solution is not so obvious. The game "Towers of Hanoi" uses three rods. A number of disks is stacked in decreasing order from the bottom to the top of one rod, i.e. the largest disk at the bottom and the smallest one on top. The disks build a conical tower.

The aim of the game is to move the tower of disks from one rod to another rod.

The following rules have to be obeyed: Only one disk may be moved at a time. Only the most upper disk from one of the rods can be moved in a move It can be put on another rod, if this rod is empty or if the most upper disk of this rod is larger than the one which is moved. Let's number the disks as D1 (smallest), D2 and D3 (largest) and name the pegs as S (SOURCE peg), A (AUX), T (TARGET). We can see that we move in three moves the tower of size 2 (the disks D1 and D2) to A. Now we can move D3 to T, where it is finally positioned. The last three moves move the tower consisting of D2D1 from peg A to T to place them on top of D3.

There is a general rule for moving a tower of size n (n ¿ 1) from the peg S to the peg T: move a tower of n - 1 discs Dn-1 ... D1 from S to A. Disk Dn is left alone on peg S Move disk Dn to T move the tower of n - 1 discs Dn-1 ... D1 on A to T, i.e. this tower will be put on top of Disk Dn The algorithm,



```
def moveTower(height,fromPole, toPole, withPole):
    if height >= 1:
        moveTower(height-1,fromPole,withPole,toPole)
        moveDisk(fromPole,toPole)
        moveTower(height-1,withPole,toPole,fromPole)

def moveDisk(fp,tp):
    print("moving disk from",fp,"to",tp)

moveTower(3,"A","B","C")
```

```
('moving disk from', 'B', 'to', 'C')
('moving disk from', 'A', 'to', 'B')
('moving disk from', 'C', 'to', 'A')
('moving disk from', 'C', 'to', 'B')
('moving disk from', 'A', 'to', 'B')
[Finished in 0.2s]
```

Figure 1: Impplementation of Tower Of Hannoi using python

which we have just defined, is a recursive algorithm to move a tower of size n. It actually is the one, which we will use in our Python implementation to solve the Towers of Hanoi. Step 2 is a simple move of a disk. But to accomplish the steps 1 and 3, we apply the same algorithm again on a tower of n-1. The calculation will finish with a finite number of steps, because very time the recursion will be started with a tower which is 1 smaller than the one in the calling function. So finally we will end up with a tower of size n = 1, i.e. a simple move.

2 Code

```
 def moveTower(height,fromPole, toPole, withPole):
     if height >= 1:
         moveTower(height-1,fromPole,withPole,toPole)
         moveDisk(fromPole,toPole)
         moveTower(height-1,withPole,toPole,fromPole)

def moveDisk(fp,tp):
    print("moving disk from",fp,"to",tp)

moveTower(3,"A","B","C")
```

3 Conclusion

```
('moving disk from', 'A', 'to', 'B')
('moving disk from', 'A', 'to', 'C')
('moving disk from', 'B', 'to', 'C')
('moving disk from', 'A', 'to', 'B')
('moving disk from', 'C', 'to', 'A')
('moving disk from', 'C', 'to', 'B')
('moving disk from', 'A', 'to', 'B')
```