Data Structures and Algorithms BCSE202L

Dr. Ragala Ramesh

September 1, 2022

Algorithm Examples

- Algorithm for summation of N numbers

Algorithm Sum(a,n)

```
{
sum ← 0;
for i ← 1 to n do
    sum ← sum + a[i];
end for
}
```

- Algorithm for Finding Largest Number in a given set

Algorithm Max(a,n)

```
{
larger ← a[0];
for i ← 1 to n do
    if (a[i] > larger) then
        larger ← a[i];
    end if
end for
return larger
}
```

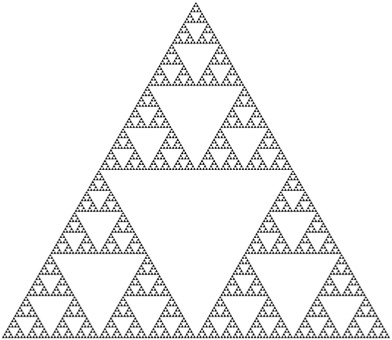
Algorithm Examples

- Algorithm for Matrix Addition

Algorithm MatrixAdd(a,b,n,m)

```
{
for i ← 1 to n do
    for j ← 1 to m do
        c[i][j] ← a[i][j] + b[i][j];
    end for
end for
}
```

- Analyse the process of drawing of a following picture



Recursive Algorithm

- An algorithm is said to be recursive if the same algorithm is invoked in the body of the algorithm
- It solves the problem by possibly using the result of applying itself to a simpler problem
- Properties of Recursive Calls
  - It solves the large problem by using its solution to a simpler sub-problem.
  - Example: Divide and Conquer Approach
  - Eventually the sub-problem is simple enough that it can be solved without applying the algorithm to it recursively. This is called Base Case

Recursive Algorithm

- **Base Case**
  - There should be at least one base case
  - it is used to avoid the infinite looping
  - Every possible chain of recursive calls must eventually reach a base case.
- **Recursive Call**
  - Calls to the current method
  - Each recursive call should be defined so that it makes progress towards a base case.
- **Two types of Recursive Algorithm**
  - Direct Recursive Algorithm:
    - An algorithm that, directly calls it self is called direct recursive
  - Indirect Recursive Algorithm:
    - An algorithm - A said to be indirect recursive, if it calls another algorithm which in turn calls Algorithm - A

Recursive Algorithm Examples

- Recursive Algorithm for Finding Largest Number in a given set

Algorithm RecurMax(a,n)

```
{
if (n = 1) then
    return a[0];
else
    max{ RecurMax(a,n-1),a(n-1) };
end if
}
```

Towers of Hanoi

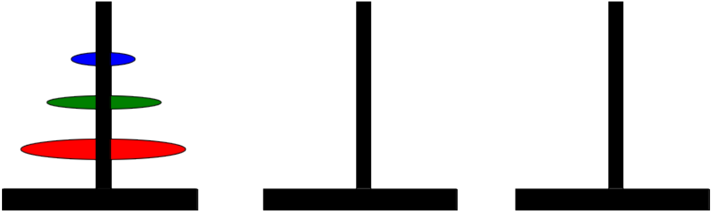
- Tower of Hanoi is a mathematical puzzle invented by a French Mathematician Edouard Lucas in 1883.
- The game starts by having few discs stacked in increasing order of size. The number of discs can vary, but there are only three Towers.
- The goal of Towers of Hanoi is to move all the disks from the leftmost Tower to the rightmost Tower, adhering to the following rules:
  1. Move only one disk at a time.

- Tower of Hanoi is a mathematical puzzle invented by a French Mathematician Edouard Lucas in 1883.
- The game starts by having few discs stacked in increasing order of size. The number of discs can vary, but there are only three Towers.
- The goal of Towers of Hanoi is to move all the disks from the leftmost Tower to the rightmost Tower, adhering to the following rules:
    1. Move only one disk at a time.
    2. A larger disk may not be placed on top of a smaller disk.

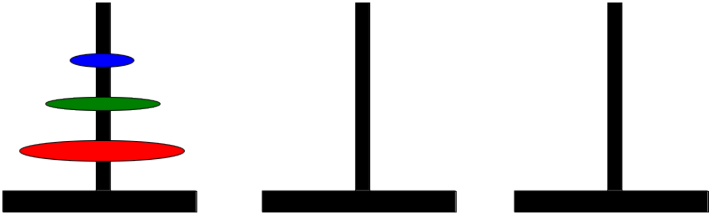
Towers of Hanoi

- Tower of Hanoi is a mathematical puzzle invented by a French Mathematician Edouard Lucas in 1883.
- The game starts by having few discs stacked in increasing order of size. The number of discs can vary, but there are only three Towers.
- The goal of Towers of Hanoi is to move all the disks from the leftmost Tower to the rightmost Tower, adhering to the following rules:
    1. Move only one disk at a time.
    2. A larger disk may not be placed on top of a smaller disk.
    3. All disks, except the one being moved, must be on a Tower.

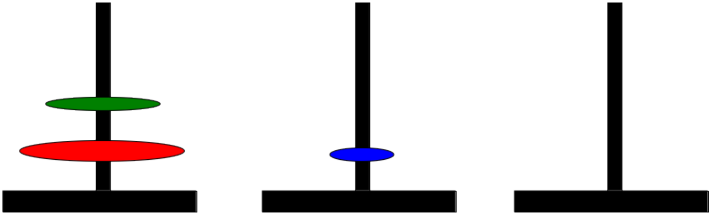
- Towers of Hanoi Problem : Initial Position



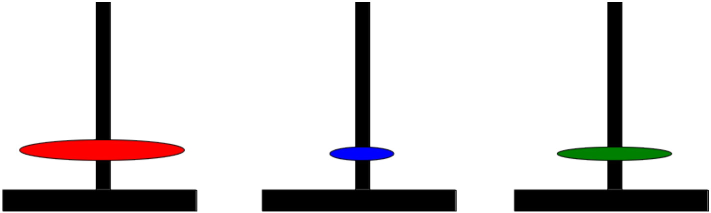
- Towers of Hanoi Problem : Step - 1



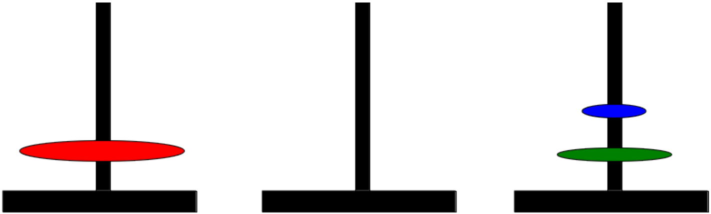
- Towers of Hanoi Problem : Step - 2



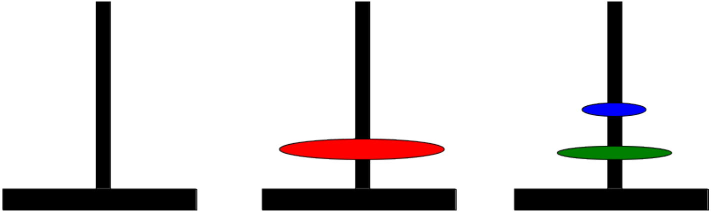
- Towers of Hanoi Problem : Step - 3



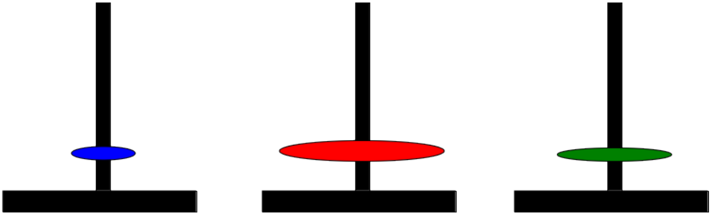
- Towers of Hanoi Problem : Step - 4



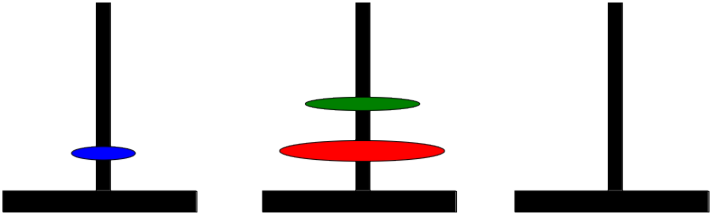
- Towers of Hanoi Problem : Step - 5



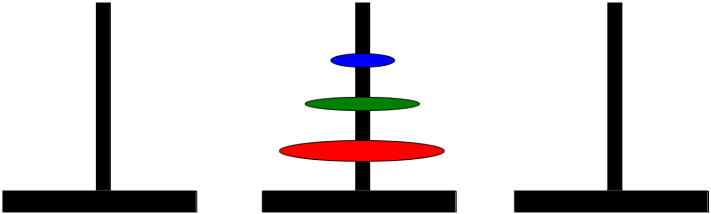
- Towers of Hanoi Problem : Step - 6



- Towers of Hanoi Problem : Step - 7



- Towers of Hanoi Problem : Step - 8



Recursive Algorithm Examples

- Recursive Algorithm for Towers of Hanoi

Algorithm ToH(n,x,y,z)

```
{
if (n ≥ 1) then
    ToH(n-1,x,z,y);
    Write("Move to disk from tower" ,x, "to top of tower", y);
    ToH(n-1,z,y,x);
end if
}
```

- The Tower of Hanoi puzzle can be solved in 7 moves for a given 3 disks.
- The minimal number of moves required to solve a Tower of Hanoi puzzle is 2^n - 1, where n is the number of disks.