Pune Vidyarthi Griha's College of Engineering and Technology & G.K. Pate (Wani) Institute of Management, Pune- 411009.

(Affiliated to Savitribai Phule Pune University)

A Mini Project Report On

"TOWER OF HANOI"

Submitted in Partial Fulfillment for the Term-work of Fourth year in Computer Engineering of *Savitribai Phule Pune University*.

By

1. Samruddhi Dhon	0077
2. Akanksha Lokhande	0071
3. Jagruti Patil	2089

Under The Guidance of

Prof. U.M. Kalshetti

Department of Computer Engineering
Academic Year: - 2024-2025

Pune Vidyarthi Griha's College of Engineering and Technology & G.K. Pate (Wani) Institute of Management, Pune- 411009.

(Affiliated to Savitribai Phule Pune University)

CERTIFICATE

This is to certify that the Project report entitled "**Tower Of Hanoi**" submitted by,

Students Name	**Roll No**
1. Samruddhi Dhon	0077
2. Akanksha Lokhande	0071
3.Jagruti Patil	2089

is a record of bonafide work carried out by him/her, in the partial fulfilment of the Term-work of fourth year in Computer Engineering of Savitribai Phule Pune University at Pune Vidyarthi Griha's College of Engineering and Technology & G.K. Pate (Wani) Institute of Management, Pune under Savitribai Phule Pune University, Pune. This work is done during the academic year 2022-23.

Date: - 17/10/24 **Place: -** Pune

Prof. U.M.Kalshetti
Subject Teacher

Prof. U.M. kalshetti
H.O.D. (Computer Engg.)

ABSTRACT

In the medical system, the verification, preservation and synchronization of electronic medical records has always been a difficult problem, and the random dissemination of patient records will bring various risks to patient privacy. Therefore, how to achieve secure data sharing on the basis of ensuring users' personal privacy becomes the key. In recent years, blockchain has been proposed to be a promising solution to achieve data sharing with security and privacy preservation due to its advantages of immutability. So, a distributed electronic medical records searchable scheme was proposed by leveraging blockchain and smart contract technology. Firstly, we perform a hash calculation on the electronic medical data and store the corresponding value on the blockchain to ensure its integrity and authenticity. These operations not only can solve centralized data store of servers of several medical institutions, but also be good at lowering stress from data store and high-frequency access to blockchain.

Block chain-based implementation of EMR is a secured transaction and maintaining of medical records in various hospitals. Now technology has developed, but the technology in the medical record transaction has not developed. Still now, each hospital is maintaining a separate database to maintain their patient details. When the patient moved to another hospital, they need to carry document each and every time. If they missed the document, they need to take all the report from starting. It takes more time and cost. To avoid this, we need to maintain the globalized database to store the data in secure manner using block chain technology. Here the donor database also connected to it, when there is any emergency in the organ transplantation and any blood requirement, the hospital can approach the donor who are connected to this system and get the immediate transaction. It also reduces the time to get the donor at the necessary time.

TABLE OF CONTENT

Overview

The Tower of Hanoi also called the Tower of Brahma or Lucas is a mathematical game or puzzle. It consists of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

- Only one disk can be moved at a time.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
- No disk may be placed on top of a smaller disk.
- With three disks, the puzzle can be solved in seven moves. The minimum number of moves required to solve a Tower of Hanoi puzzle is 2n - 1, where n is the number of disks.

The puzzle can be played with any number of disks, although many toy versions have around seven to nine of them. The minimum number of moves required to solve a Tower of Hanoi puzzle is 2n - 1, where n is the number of disks.

Origin

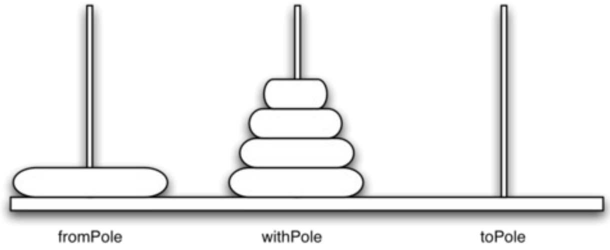
The puzzle was invented by the French mathematician Édouard Lucas in 1883. There is a story about an Indian temple in Kashi Vishwanath which contains a large room with three time-worn posts in it surrounded by 64 golden disks. Brahmin priests, acting out the command of an ancient prophecy, have been moving these disks, in accordance with the immutable rules of the Brahma, since that time. The puzzle is therefore also known as the Tower of Brahma puzzle. According to the legend, when the last move of the puzzle will be completed, the world will end. It is not clear whether Lucas invented this legend or was inspired by it.

If the legend were true, and if the priests were able to move disks at a rate of one per second, using the smallest number of moves, it would take them 264-1 seconds or roughly 585 billion years or 18,446,744,073,709,551,615 turns to finish, or about 127 times the current age of the sun.

Applications :

1. The Tower of Hanoi is frequently used in psychological research on problem solving.
2. It is also used as a Backup rotation scheme when performing computer data Backups where multiple tapes/media are involved.
3. Tower of Hanoi is popular for teaching recursive algorithms to beginning programming students.
4. The Tower of Hanoi is also used as a test by neuropsychologists trying to evaluate frontal lobe deficits.

An Example Arrangement of Disks for the Tower of Hanoi



Here is a high-level outline of how to move a tower from the starting pole, to the goal pole, using an intermediate pole:

- Move a tower of height-1 to an intermediate pole, using the final pole.
- Move the remaining disk to the final pole.
- Move the tower of height-1 from the intermediate pole to the final pole using the original pole.

As long as we always obey the rule that the larger disks remain on the bottom of the stack, we can use the three steps above recursively, treating any larger disks as though they were not even there. The only thing missing from the outline above is the identification of a base case. The simplest Tower of Hanoi problem is a tower of one disk. In this case, we need move only a single disk to its final destination. A tower of one disk will be our base case.

Here is the algorithm to move the disc from source to destination.

```
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")
```

output :-

```
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
```

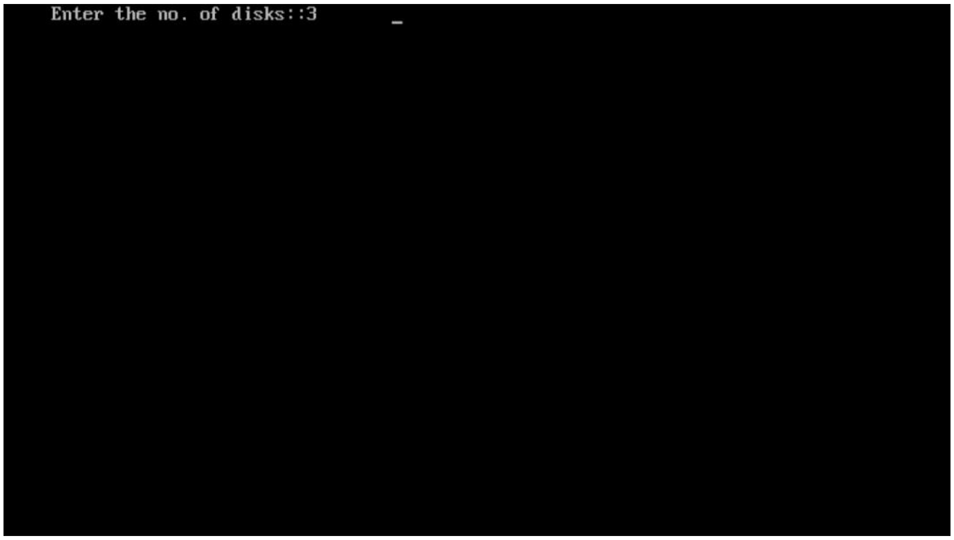
Source code of Tower of Hanoi

```
// Program To Implement Tower Of Hanoi Algorithm Using Recursion.
This program shows the movements of disk from one tower to another when a key is pressed.

#include<iostream.h>
#include<stdio.h>
#include<conio.h>

class tower
{
        int *t1,*t2,*t3;
        int x,y,z;
        public:
        void disp_tower();
        void move_disk(int tx,int ty);
        void toh(int n,int a,int b,int c);
        tower(int no);
        ~tower();
};

tower :: tower(int no)
{
        t1 = new int[no+1];
        t2 = new int[no+1];
        t3 = new int[no+1];
        x = no;
        y  = z = 0;

        for(int i=0,j=no ; i<no ; i++,j--)
        {
                t1[i] = j;
                t2[i] = t2[i] = 0;
        }
        t1[no] = t2[no] = t3[no] = 0;
}

tower :: ~tower()
{
        delete []t1;
        delete []t2;
        delete []t3;
}
void tower :: disp_tower()
{
        clrscr();
```

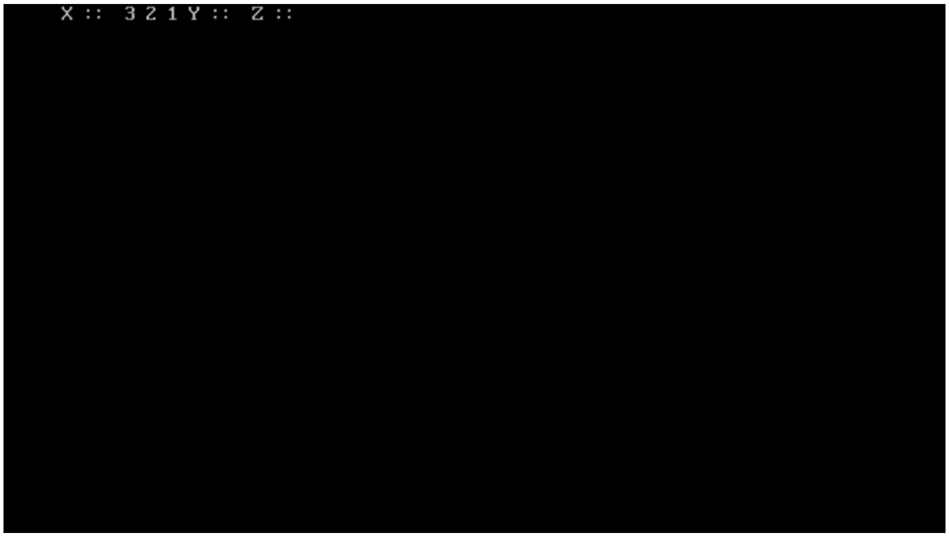
```
	cout<<"

 X :: ";
	for(int i=0;i<x;i++)
	{
		cout<<" "<<t1[i];
	}
	cout<<"

 Y :: ";
	for(i=0;i<y;i++)
	{
		cout<<" "<<t2[i];
	}
	cout<<"

 Z :: ";
	for(i=0;i<z;i++)
	{
		cout<<" "<<t3[i];
	}
	getch();
}

void tower :: toh(int n,int tx,int ty,int tz)       //x to y using z
{

	if(n>=1)
	{
		toh(n-1,tx,tz,ty);
		move_disk(tx,ty);                    //x to y
		disp_tower();
		toh(n-1,tz,ty,tx);
	}
}

void tower :: move_disk(int tx,int ty)
{
	switch(tx)
	{
		case 1:
			{
				if(ty==2)
					t2[y++] = t1[--x];
				else
					t3[z++] = t1[--x];
```

```
            }
    break;
        case 2:
            {
                if(ty==1)
                    t1[x++] = t2[--y];
                else
                    t3[z++] = t2[--y];
            }
break;
        case 3:
            {
                if(ty==1)
                    t1[x++] = t3[--z];
                else
                    t2[y++] = t3[--z];
            }
            break;
    }//end of switch
}
// beginning of main function

int main(void)
{
    clrscr();
    cout<<"Enter the no. of disks::";
    int no;
    cin>>no;
    tower obj(no);
    obj.disp_tower();
    obj.toh(no,1,2,3);
    getch();
    return 0;
}
```

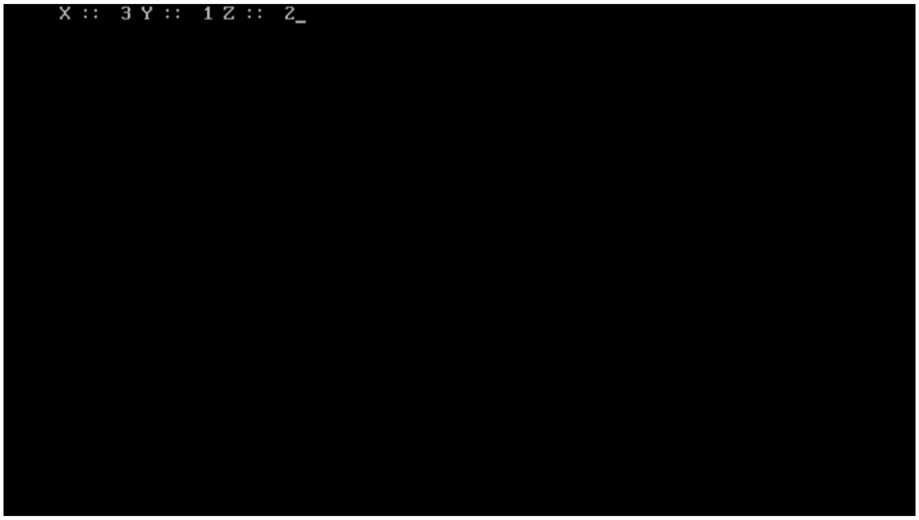
Output:



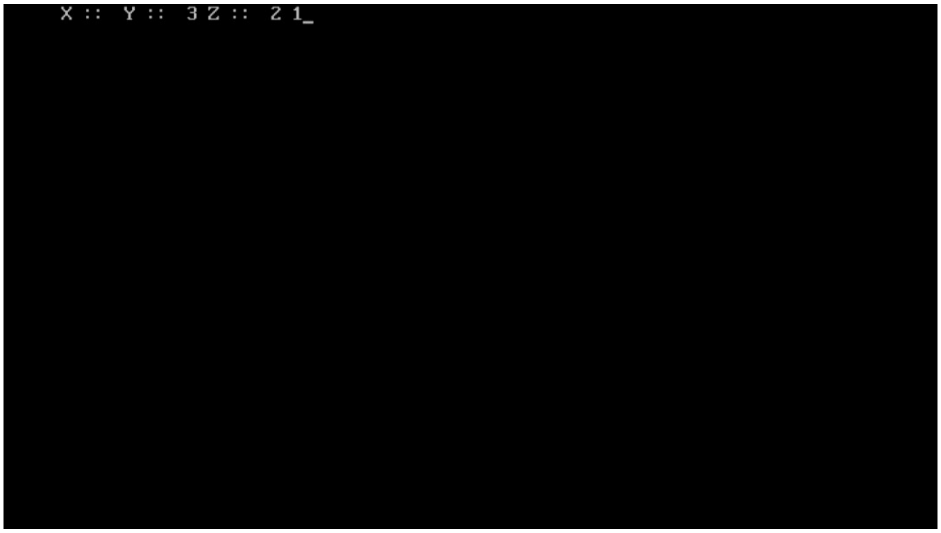
```
Enter the no. of disks::3
```



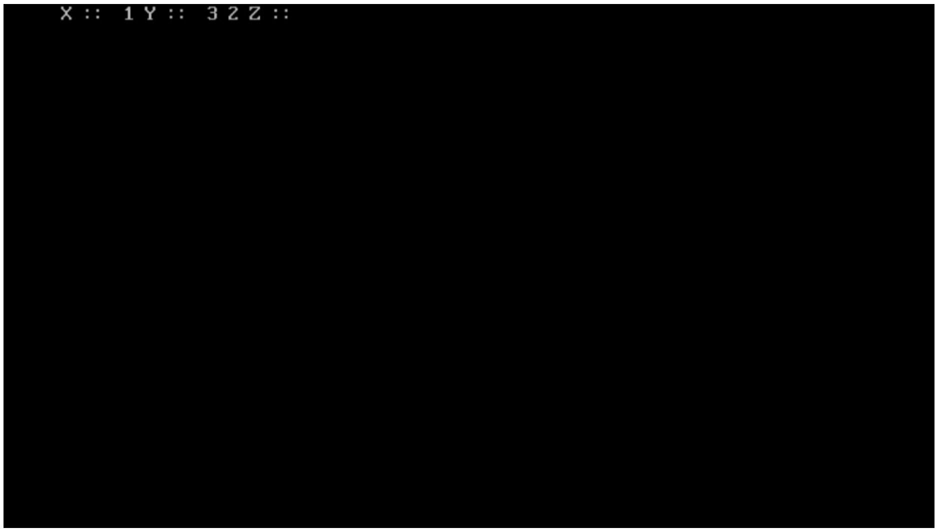
```
X ::  3 2 1 Y ::  Z ::
```



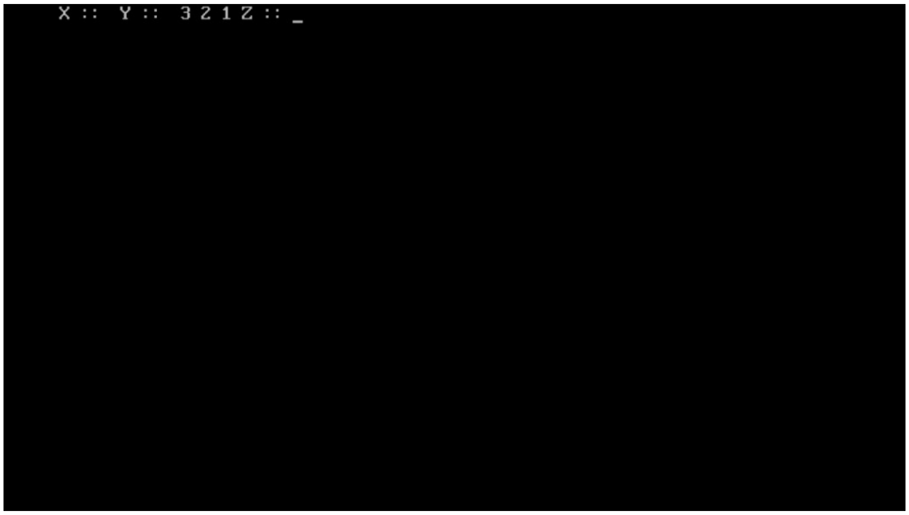
```
X ::  3 Y ::  1 Z ::  2
```



```
    X ::  Y ::  3 Z ::  2 1_
```



```
    X ::  1 Y ::  3 2 Z ::
```



```
    X ::  Y ::  3 2 1 Z :: _
```

Explanation of coding

This program uses recursion this program shows the movement of disk from one tower to another .In this program of Tower of Hanoi.
 we first create a class tower and in this class we create three pointer variable ie. T1,t2,t3 for three rods and correspondingly we can take three variables as c,y,z. here we take three methods:-

Void disp_tower();
Void move_disk();
Void toh();

Here we also use the constructor and destructor. In the constructor we initializes the values of each variable. In the destructor we use the delete method to freethe allocated memory.
In the disp_tower() method we display the values of the respective t1,t2 & t3 variables.
In the toh() method , it checks the condition whether the input size is greater than 1 or not . & then we call the toh(), move_disk(),disp_tower() and again the toh() method.
In the move_disk() method , we can use the switch case to move the disk between source , dest. And intermediate nodes.

At last in the main method , we enter the number of disks to move the disks to destination point , then the constructor is automatically called as we create the object of the tower class , then we call the disp_tower() method and toh() method .

Conclusion

From the proper analysis , this project tells that we can move the number of discs from source to destination rod with the help of one intermediate rod in such a way that smallest disc is placed at the top of the rod. And largest one is placed at the bottom, thus making a conical shape.

This is used in psychological research on problem solving.

This concept is also used in the development of the TURF framework for the representation of human computer Interaction.

Bibliography

Books :- c++ programming by E.balagurusamyData
structure by t.schaum series

Website :- happycoding.com
Codingfox.com