Ex.No.6
Date:1/4/2021

Implementation of Minimax algorithm for an application.

AIM: to Implementation of Minimax algorithm

Algorithm:
The whole process can be categorized as follows:
1. Create a function called minimax.
2. This Function takes the arguments like int depth, int nodeIndex, bool isMax , int scores[], int h.
3. This Function is used to return the the optimal value a maximizer can obtain.
4. The arguments are used for:
 depth is current depth in game tree.
 nodeIndex is index of current node in scores[].
 isMax is true if current move is
 of maximizer, else false
 scores[] stores leaves of Game tree.
 h is maximum height of Game tree
5. Next check whether it is MAX or MIN with ismax function
6. If current move is maximizer find the maximum attainable value
7. If not , find the Minimizer value.
8. Based on this traverse the tree accordingly with the given MAX and MIN and find the Optimal value.
9. Print the Value and Path.

Code:
```
#include<bits/stdc++.h>
using namespace std;

int minimax(int depth, int nodeIndex, bool isMax,
 int scores[], int h)
{
 if (depth == h)
 return scores[nodeIndex];

 if (isMax)
 return max(minimax(depth+1, nodeIndex*2, false, scores, h),
 minimax(depth+1, nodeIndex*2 + 1, false, scores, h));

 else
 return min(minimax(depth+1, nodeIndex*2, true, scores, h),
```

```
 minimax(depth+1, nodeIndex*2 + 1, true, scores, h));
}

int max1;
int max2;
int max3;
int max4;
int min1;
int min2;
int Optimal;
int log2(int n)
{
 return (n==1)? 0 : 1 + log2(n/2);
}
int main()
{
 int scores[1234];
 int nn;
 cout<<"Enter The no. of Nodes: ";
 cin>>nn;
 cout<<"Enter The costs of the Nodes: ";
 for(int i =0; i<nn; i++)
 {cin>>scores[i];}
 int n = sizeof(scores)/sizeof(scores[0]);
 int h = log2(n);
 int res = minimax(0, 0, true, scores, h);
 if(scores[0]>scores[1])
 {
 cout<<"For D: cost= "<< scores[0]<<"\n";
 max1 = scores[0];
 }
 else{
 cout<<"For D: cost= "<< scores[1] <<"\n";
 max1 = scores[1];
 }

 if(scores[2]>scores[3])
 {
 cout<<"For E: cost= "<< scores[2]<<"\n";
 max2 = scores[2];
 }
 else{
 cout<<"For E: cost= "<< scores[3] <<"\n";
 max2 = scores[3];
```

```
}

if(scores[4]>scores[5])
{
cout<<"For F: cost= "<< scores[4]<<"\n";
max3 = scores[4];
}
else{
cout<<"For F: cost= "<< scores[5] <<"\n";
max3 = scores[5];
}

if(scores[6]>scores[7])
{
cout<<"For G: cost= "<< scores[6]<<"\n";
max4 = scores[6];
}
else{
cout<<"For G: cost= "<< scores[7] <<"\n";
max4 = scores[7];
}

if(max1>max2)
{
cout<<"For B: cost= "<<max2<<"\n";
min1=max2;
}
else{
cout<<"For B: cost= "<<max1<<"\n";
min1=max1;
}

if(max3>max4)
{
cout<<"For C: cost= "<<max4<<"\n";
min2=max4;
}
else{
cout<<"For C: cost= "<<max3<<"\n";
min2=max3;
}
```

```
cout<<"Cost values in First(MAX) level is:";
cout<<max1<<" ";
cout<<max2<<" ";
cout<<max3<<" ";
cout<<max4<<" "<<"\n";
cout<<"Cost values in First(MIN) level is:";
cout<<min1<<" ";
cout<<min2<<" "<<"\n";

if(min1>min2)
{
Optimal=min1;

}

else{
Optimal=min2;

}
cout<<"Path Is:"<<"I "<<"D "<<"B "<<"A "<<"\n";

cout << "The Optimal value is : " <<Optimal<< endl;

return 0;
}
```

Aim:-
To implement a minimax algorithm for a tic-tac-toe Problem.

Algorithm:

- Minimax is a recursive algorithm which is used to choose an optimal move for a player assuming that the other player is also playing optimally.

- It is used in games such as tic-tac-toe, go, chess, Isola, checkers, and many other two-player games.

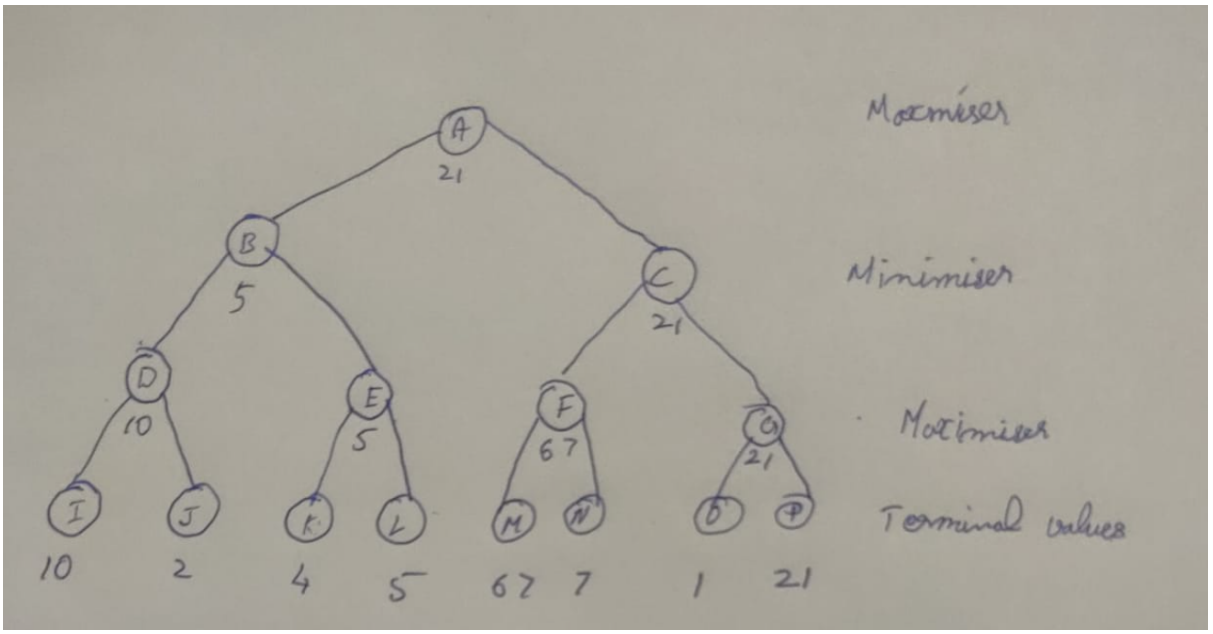
- The minmax function used is as follow

```
def minimax(board,player):
   x=analyzeboard(board);
   if(x!=0):
      return (x*player);
   pos=-1;
   value=-2;
   for i in range(0,9):
```

```
    if(board[i]==0):
        board[i]=player;
        score=-minimax(board,(player*-1));
        if(score>value):
            value=score;
            pos=i;
        board[i]=0;

if(pos==-1):
    return 0;
return value;
```

Sample graph:



Output:

```
Running /home/ubuntu/environment/RA1811003010303_minimax.cpp
Enter The no. of Nodes: 8
Enter The costs of the Nodes: 10
2
4
5
67
7
1
21
For D: cost= 10
For E: cost= 5
For F: cost= 67
For G: cost= 21
For B: cost= 5
For C: cost= 21
Cost values in First(MAX) level is:10 5 67 21
Cost values in First(MIN) level is:5 21
The Optimal value is : 21
```

Result:-

We Successfully Implemented Minimax algorithm .