```
//Elaborate uninformed search algorithm for any suitable real time application.
//AI Missionaries and Cannibals Problem solved using DFS
//Submitted by: Sanskar Sharma
//PRN: 0120180381
//Roll numbetr: 090


#include<iostream>
#include<string.h>

using namespace std;

class state//state class
{
    public:
        //int index;
        int left_Missionaries;
        int left_Cannibals;
        int Boat; //if boat is left side of river 1, else 0.

};

//Data structure to add node names into queue or stack using LL
class node
{
    public:
        state data;
        node*down;//pointer of a node
};

//Stack implementation for DFS algorithm.

class stack//pushing and poping states of left side
{
    public:
        node *top;
        stack()
        {
            top=NULL;
        }
        void push(state val)
        {

            if(top==NULL)
            {

                top=new node();
                if(top==NULL)
                {
                    cout<<"Error in memory allocation";exit(-1);
                }
                top->data=val;
                top->down=NULL;
            }
            else
            {
                node*ptr=NULL;
                ptr=new node();
                if(ptr==NULL)
                {
                    cout<<"Error in Memory allocation";
                    exit(-1);
                }
                ptr->data=val;
                ptr->down=top;
                top=ptr;
            }

        }
        state* pop()
        {

            if(stack_empty()==1)
            {
                return NULL;

            }
            else
            {
                state val;
                val=top->data;
                node*ptr=top;
                top=top->down;
                delete(ptr);
                return &val;
            }
        }
        int stack_empty()
        {
            if(top==NULL)
```

```
            {
                return 1;
            }
            else
            {
                return 0;
            }
        }
        void Operation();
};

//State validation satisfying M and C problem
int state_validation(state current_state)
{
    int invalid=0;
    //Missionaries on left side should not be less than Cannibals on left
    iint Boat; //if boat is left side of river 1, else 0.
    {
        if(current_state.left_Missionaries!=0)
        {
            invalid=1;
        }
    }
    //Also Missionaries on right sie cannot be less than Cannibals on right
    int x=3-current_state.left_Missionaries;//M on right side
    int y=3-current_state.left_Cannibals;//C on right side
    if(x<y)
    {
        if(x!=0)
        {
            invalid=1;
        }
    }
    //[3,3,0] is invalid state as boat is on right
    if(current_state.left_Missionaries==3&&current_state.left_Cannibals==3&&current_state.Boat==0)
    {
        invalid=1;
    }
    //[0,0,1] if no M and C on left side, no boat can be there
    if(current_state.left_Missionaries==0&&current_state.left_Cannibals==0&&current_state.Boat==1)
    {
        invalid=1;
    }


    return invalid;
}

//M and C class
class Missionaries_Cannibals
{
    public:
        void operation(state initial,int ind);//To select algo
        void DFS(state initial,int ind);

};

//[0,0,0] checkin goal state
int check_goalstate(state *curr_state)
{
    int goal_state=0;
    if(curr_state->Boat==0&&curr_state->left_Cannibals==0)
    {
        if(curr_state->left_Missionaries==0)
        {
            goal_state=1;
        }
    }
    return goal_state;
}

//simple function to swap boat positions
int swap(int c)
{
    if(c==0)
        return 1;
    else
        return 0;

}


//Function to keep record of explored states, to avoid repeatation
int is_explored(int explored[50][3],state current_state)
{
    int flag=0;

    for(int i=0;i<50;++i)
    {
        if(explored[i][0]==current_state.left_Missionaries)
```

```
        {
            if(explored[i][1]==current_state.left_Cannibals)
            {
                if(explored[i][2]==current_state.Boat)
                {
                    flag=1;
                }
            }
        }
    }
    return flag;
}

//Depth-First search algorithm
void Missionaries_Cannibals::DFS(state initial,int ind)
{
    int check;
    //Explored 2D array of 50x3
    int explored[50][3];
    for(int i=0;i<50;++i)
    {
        for(int j=0;j<3;++j)
        {
            explored[i][j]=-1;//initiailising as -1 to positions in 2D array
        }
    }
    explored[0][0]=initial.left_Missionaries;
    explored[0][1]=initial.left_Cannibals;
    explored[0][2]=initial.Boat;
    int counter=1;
    check=check_goalstate(&initial);
    if(check==1)
    {
        cout<<"\n\t\tSUCCESSFULLY! Reached goal state!";
        exit(0);
    }
    stack s;//object of stack
    s.push(initial);
    state current_state;
    int flag=0,c,f;
    //current_state=*s.pop();
    state next;
    cout<<"\n\n";
    while(s.stack_empty()!=1)
    {

        current_state=*s.pop();
        cout<<"\tCurrent State: ";
        cout<<"["<<current_state.left_Missionaries<<","<<current_state.left_Cannibals<<","<<current_state.Boat<<"]\n\n";
        //push all possible next states

        //s=successor_states(s,current_state);

        //ind=current_state->index+1;
        if(current_state.Boat==1)//when boat is on right side
        {
            cout<<"\t\t=> Possible ways to move right side of river bank\n";
            cout<<"\t\tStates pushed in the stack: ";;
            for(int i=0;i<=current_state.left_Missionaries;++i)
            {
                for(int j=0;j<=current_state.left_Cannibals;++j)
                {
c=i+j;

next.left_Missionaries=current_state.left_Missionaries-i;
next.left_Cannibals=current_state.left_Cannibals-j;
next.Boat=swap(current_state.Boat);
//cout<<"\n"<<i<<"   "<<j;
//cout<<"["<<next.left_Missionaries<<","<<next.left_Cannibals<<","<<next.Boat<<"]";
if(c==1||c==2)//only 1 or 2 people can travel in boat at a time
{
    if(state_validation(next)!=1)//not invalid
    {
        if(check_goalstate(&next))
        {
            cout<<"["<<next.left_Missionaries<<","<<next.left_Cannibals<<","<<next.Boat<<"]";
            cout<<"\n\t\tSUCCESSFULLY! Reached goal state!";
            exit(0);
        }
        else
        {
            //next.index=ind++;
            //cout<<"["<<current_state.left_Missionaries<<","<<current_state.left_Cannibals<<","<<current_state.Boat<<"]
            f=is_explored(explored,next);
            //cout<<f;
            if(f==0)
            {
                //cout<<"f";
                cout<<"["<<next.left_Missionaries<<","<<next.left_Cannibals<<","<<next.Boat<<"]";
                explored[counter][0]=next.left_Missionaries;
```

```
                            explored[counter][1]=next.left_Cannibals;
                            explored[counter][2]=next.Boat;
                            counter+=1;
                            cout<<"  ";
                            s.push(next);

                        }



                        //ind+=1;
                        //delete(&next)
                    }

                }
            }
        }
    }
}
else//when boat is on left side
{
    cout<<"\t\t<= Possible ways to move left side of river bank\n";
    cout<<"\t\tStates pushed in the stack: ";
    for(int i=0;i<=3-current_state.left_Missionaries;++i)
    {
        for(int j=0;j<=3-current_state.left_Cannibals;++j)
        {
c=i+j;

next.left_Missionaries=current_state.left_Missionaries+i;
next.left_Cannibals=current_state.left_Cannibals+j;
next.Boat=swap(current_state.Boat);
//cout<<"\n"<<i<<"   "<<j;
if(c==1||c==2)
{
    if(state_validation(next)!=1)//not invalid
    {
        if(check_goalstate(&next))
        {
            cout<<"["<<next.left_Missionaries<<","<<next.left_Cannibals<<","<<next.Boat<<"]";
            cout<<"\n\t\tSUCCESSFULLY! Reached goal state!";
            exit(0);
        }
        else
        {



            //next.index=ind++;

            f=is_explored(explored,next);
            if(f==0)
            {
                cout<<"["<<next.left_Missionaries<<","<<next.left_Cannibals<<","<<next.Boat<<"]";
                explored[counter][0]=next.left_Missionaries;
                explored[counter][1]=next.left_Cannibals;
                explored[counter][2]=next.Boat;
                counter+=1;
                cout<<"  ";
                s.push(next);
            }


            //ind+=1;
            //delete(&next);


                    }

                }
            }


        }
    }
 }
 cout<<"\n\n\tPop()"<<endl;


/* while(s.stack_empty()!=1)
 {
     f=is_explored(explored,current_state);
     if(f==0)
     {

         explored[counter]=current_state;
         counter+=1;
         break;
```

```
            }
            else
            {
                current_state=*s.pop();
            }
        }

        if(check_goalstate(&current_state))
        {
            flag=1;
            break;
        }*/
    /*  while(s.stack_empty()!=1)
        {

            cout<<"\n\t["<<s.pop()->left_Missionaries<<","<<s.pop()->left_Cannibals<<","<<s.pop()->Boat<<"]\n\t\t";
        }
    */  //current_state=s.pop();
        //cout<<"\n\t["<<current_state->left_Missionaries<<","<<current_state->left_Cannibals<<","<<current_state->Boat<<"]\n\t\t";


    }

    if(flag==1)
    {
        cout<<"\n\t\tSUCCESSFULLY! Reached the goal state";
    }
    else
    {
        cout<<"\n\t\tFAILED! to reach goal state.";
    }



}

void Missionaries_Cannibals::operation(state initial,int ind)
{
    DFS(initial,ind);
}



//Main function
int main()
{
    //Input Initial state!
    cout<<"\n\t\t\tAI Missionaries and Cannibals Problem";
    cout<<"\n\t\t\t       By Sanskar Sharma";
    cout<<"\n\t\t\t           0120180381\n";
    cout<<"\n\t\tEnter the initial state of system: ";
    state initial;
    int ind=1;
    //initial.index=ind++;
    cout<<"\n\t\tEnter number of Missionaries on left side: ";
    cin>>initial.left_Missionaries;
    while(1)
    {
        if(initial.left_Missionaries>=0&&initial.left_Missionaries<=3)
        break;
        else
        {
            cout<<"\n\t\tNo. of missionaries should be between 0 and 3.";
            cout<<"\n\t\tEnter again: ";
            cin>>initial.left_Missionaries;
        }
    }
    cout<<"\n\t\tEnter number of Cannibals on left side: ";
    cin>>initial.left_Cannibals;
    while(1)
    {
        if(initial.left_Cannibals>=0&&initial.left_Cannibals<=3)
        break;
        else
        {
            cout<<"\n\t\tNo. of Cannibals should be between 0 and 3.";
            cout<<"\n\t\tEnter again: ";
            cin>>initial.left_Cannibals;
        }
    }
    cout<<"\n\t\tIs boat left side of the river? (1/0)";
    cin>>initial.Boat;
    if(initial.Boat!=1)
    {
        initial.Boat=0;
    }
    cout<<"\n\tINITIAL STATE: ";
    cout<<"["<<initial.left_Missionaries<<","<<initial.left_Cannibals<<","<<initial.Boat<<"]\n\n";
    int c=state_validation(initial);
    if(c==1)
```

```
    {
        cout<<"\n\t\tInitial state is invalid.";
        cout<<"\n\t\tFAILED! to reach goal state.";
        exit(-1);//invalid initial state.
    }
    else
    {
        cout<<"\n\t\tInitial state is valid proceed ahead";
    }

    //state i;
    //stack s;
    //s.push(&initial);
    //cout<<"\n\t"<<(s.pop())->Boat;

    Missionaries_Cannibals MC;
    MC.operation(initial,ind);

    return 0;
}


/*
Output: left side= [M,C,B]

1. Valid, Start Initial state: [3,3,1]

                AI Missionaries and Cannibals Problem
                      By Sanskar Sharma
                          0120180381

        Enter the initial state of system:
        Enter number of Missionaries on left side: 3

        Enter number of Cannibals on left side: 3

        Is boat left side of the river? (1/0)1

INITIAL STATE: [3,3,1]


        Initial state is valid proceed ahead

Current State: [3,3,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [3,2,0]  [3,1,0]  [2,2,0]

Pop()
Current State: [2,2,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [3,2,1]

Pop()
Current State: [3,2,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [3,0,0]

Pop()
Current State: [3,0,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [3,1,1]

Pop()
Current State: [3,1,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [1,1,0]

Pop()
Current State: [1,1,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [2,2,1]

Pop()
Current State: [2,2,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [0,2,0]

Pop()
Current State: [0,2,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [0,3,1]
```

```
        Pop()
        Current State: [0,3,1]

                => Possible ways to move right side of river bank
                States pushed in the stack: [0,1,0]

        Pop()
        Current State: [0,1,0]

                <= Possible ways to move left side of river bank
                States pushed in the stack: [0,2,1]  [1,1,1]

        Pop()
        Current State: [1,1,1]

                => Possible ways to move right side of river bank
                States pushed in the stack: [0,0,0]
                SUCCESSFULLY! Reached goal state!
--------------------------------
Process exited after 3.091 seconds with return value 0
Press any key to continue . . .

2. Valid, Mid Initial State: [3,1,0] (any valid state)


                        AI Missionaries and Cannibals Problem
                               By Sanskar Sharma
                                0120180381

        Enter the initial state of system:
        Enter number of Missionaries on left side: 3

        Enter number of Cannibals on left side: 1

        Is boat left side of the river? (1/0)0

INITIAL STATE: [3,1,0]


        Initial state is valid proceed ahead

Current State: [3,1,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [3,2,1]  [3,3,1]

Pop()
Current State: [3,3,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [3,2,0]  [2,2,0]

Pop()
Current State: [2,2,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack:

Pop()
Current State: [3,2,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack:

Pop()
Current State: [3,2,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [3,0,0]

Pop()
Current State: [3,0,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [3,1,1]

Pop()
Current State: [3,1,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [1,1,0]

Pop()
Current State: [1,1,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [2,2,1]

Pop()
```

```
Current State: [2,2,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [0,2,0]

Pop()
Current State: [0,2,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [0,3,1]

Pop()
Current State: [0,3,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [0,1,0]

Pop()
Current State: [0,1,0]

        <= Possible ways to move left side of river bank
        States pushed in the stack: [0,2,1]  [1,1,1]

Pop()
Current State: [1,1,1]

        => Possible ways to move right side of river bank
        States pushed in the stack: [0,0,0]
                SUCCESSFULLY! Reached goal state!
--------------------------------
Process exited after 4.393 seconds with return value 0
Press any key to continue . . .

3. Invalid, Initial State: [1,3,0] (M<C)

                        AI Missionaries and Cannibals Problem
                                By Sanskar Sharma
                                    0120180381

                Enter the initial state of system:
                Enter number of Missionaries on left side: 1

                Enter number of Cannibals on left side: 3

                Is boat left side of the river? (1/0)0

        INITIAL STATE: [1,3,0]


                Initial state is invalid.
                FAILED! to reach goal state.
--------------------------------
Process exited after 4.54 seconds with return value 4294967295
Press any key to continue . . .

4. Valid, Goal state as Initial State: [0,0,0]


                        AI Missionaries and Cannibals Problem
                                By Sanskar Sharma
                                    0120180381

                Enter the initial state of system:
                Enter number of Missionaries on left side: 0

                Enter number of Cannibals on left side: 0

                Is boat left side of the river? (1/0)0

        INITIAL STATE: [0,0,0]


                Initial state is valid proceed ahead
                SUCCESSFULLY! Reached goal state!
--------------------------------
Process exited after 6.356 seconds with return value 0
Press any key to continue . . .

*/
```