**THE KNIGHT’S TRAVEL**

* **CODE:**

#include<conio.h>

#include<bits/stdc++.h>

using namespace std;

int dx[]={1,2,2,1,-1,-2,-2,-1};

int dy[]={2,1,-1,-2,-2,-1,1,2};

bool vis[105][105];

int dis[105][105];

bool isvalid(int x,int y,int n,int m){

    if(x<1 or y<1 or x>n or y>m)

        return false;

    if(vis[x][y])

        return false;

    return true;

}

bool isvalid1(int x,int y,int n,int m){

    if(x<1 or y<1 or x>n or y>m)

        return false;

    return true;

}

int n,m;

int startx,starty,endx,endy;

void solve(int arg1,int arg2)

{

    memset(vis,0,sizeof(vis));

    memset(dis,0,sizeof(dis));

    n=arg1;

    m=arg2;

    srand(time(NULL));

    startx=rand()%n+1;

    starty=rand()%m+1;

    endx=rand()%n+1;

    endy=rand()%m+1;

    queue<pair<int,int>>q;

    q.push(make\_pair(startx,starty));

    vis[startx][starty]=1;

    dis[startx][starty]=0;

    while(!q.empty())

    {

        int x=q.front().first;

        int y=q.front().second;

        q.pop();

        for(int i=0;i<8;i++)

        {

            int newx=x+dx[i];

            int newy=y+dy[i];

            if(isvalid(newx,newy,n,m))

            {

                dis[newx][newy]=dis[x][y]+1;

                vis[newx][newy]=1;

                q.push(make\_pair(newx,newy));

            }

        }

    }

    int  step=0;

    int maxstep=dis[endx][endy]+4;

    memset(vis,0,sizeof(vis));

    int xx,yy;

    int in=0;

    int sx=startx,sy=starty;

    int t[n+1][m+1],flag=0,ans=1;

    cout<<"\t\t\t\t\t\t\t\t\t\t\tMAXIMUM MOVES :"<<dis[endx][endy]+4<<endl;

    cout<<"\n\t\t\t\t\t\t\t\t\t\t\tS: START\n\t\t\t\t\t\t\t\t\t\t\tF:FINISH"<<endl;

    for(int p=1;p<=n;p++)

        {

            cout<<"\t\t\t\t\t\t";

            for(int q=1;q<=m;q++)

            {

                if(startx==p and starty==q)

                    cout<<"[S]"<<" ";

                else if (endx==p and endy==q)

                {

                        cout<<"[F]"<<" ";

                }

                 else if(dis[p][q]==1)

                         cout<<"[1]"<<" ";

                 else

                 {

                     cout<<"[x]"<<" ";

                 }

            }

            cout<<endl;

            cout<<"\t\t\t\t\t\t";

            for(int x=1;x<=n;x++)

            cout<<"----";

            cout<<endl;

        }

    for(step=1;step<=maxstep;step++)

    {

        memset(t,-1,sizeof(t));

        pair<int,int>xy;

        cout<<"\t\t\tEnter X-Coordinates :";

        cin>>xy.first;

        cout<<"\t\t\tEnter Y-Coordinates :";

        cin>>xy.second;

        for(int i=0;i<8;i++)

        {

            int nx=xy.first+dx[i];

            int ny=xy.second+dy[i];

            if(isvalid1(nx,ny,n,m))

            {

                t[nx][ny]=step+1;

            }

        }

        for(int p=1;p<=n;p++)

        {

            cout<<"\t\t\t\t\t\t";

            for(int q=1;q<=m;q++)

            {

                if(startx==p and starty==q)

                    cout<<"[S]"<<" ";

                else if (endx==p and endy==q)

                {

                        cout<<"[F]"<<" ";

                }

                else if(t[p][q]!=-1)

                    cout<<"["<<t[p][q]<<"] ";

                else if(p==xy.first and q==xy.second)

                    cout<<"["<<step<<"] ";

                else

                cout<<"[x]"<<" ";

            }

            cout<<endl;

            cout<<"\t\t\t\t\t\t";

            for(int x=1;x<=n;x++)

            cout<<"----";

            cout<<endl;

        }

        cout<<endl;

        int flag1=0;

        xx=xy.first;

        yy=xy.second;

        for(int i=0;i<8;i++)

        {

            if(xx==sx+dx[i] and yy==sy+dy[i])

            {

                flag1=1;

                break;

            }

        }

        if(!flag1)

        {

            step--;

            cout<<"\t\t\t\tInvalid move"<<endl;

            in=1;

            break;

        }

        else

        {

            if(xx==endx and yy==endy)

            {

                cout<<"\t\t\t\tFound in steps : "<<step<<endl;

                ans=0;

                break;

            }

            sx=xx,sy=yy;

        }

    }

    if(ans==0&&in==0)

    {

        if(step==dis[endx][endy])

        {

            cout<<"\t\t\t\tCONGRATULATIONS !!! YOU HAVE WON \n \t\t\t\tCompleted game in "<<step<<"steps"<<endl;

            cout<<"\t\t\t\tYour score: 20"<<endl;

        }

        else

        {

            cout<<"\t\t\t\tHARD LUCK BUT THERE IS A MORE EFFICIENT WAY TO REACH HERE WITH STEPS :"<<dis[endx][endy]<<endl;

            cout<<"\t\t\t\tYour score:"<<(20-(step-dis[endx][endy])\*(2))<<endl;

        }

    }

    else if(in==0)

    {

        cout<<"\t\t\t\tYOU LOSE !!! YOU EXCEEDED THE MAXIMUM MOVES"<<endl;

        cout<<"\t\t\t\tYour score: 0"<<endl;

    }

    return ;

}

int main()

{

    int choice;

    cout<<"\n\n\n\t\t\t\t\t\t WELCOME TO KNIGHT TRAVEL"<<endl;

    cout<<endl<<endl<<endl;

    cout<<"\t\t\t\t For correct answer in efficient steps you will be awarded 20 points\n\n\n\t\t\t\t For each extra steps (-2) points and for exceeding max limit 0 points\n\n"<<endl;

    cout<<"\t\t\t\t\t\t Choose your level of game :"<<endl<<endl<<endl;

    cout<<"\n\t\t\t\t\t\t 1: EASY ( 4 X 4 GRID)"<<endl<<endl;

    cout<<"\n\t\t\t\t\t\t 2: MEDIUM ( 8 X 8 GRID)"<<endl<<endl;

    cout<<"\n\t\t\t\t\t\t 3: HARD ( 12 X 12 GRID)"<<endl<<endl;

    cout<<"\n\t\t\t\t\t\t 4: QUIT"<<endl<<endl;

    cout<<"\n--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------"<<endl;

    cin>>choice;

    switch(choice)

    {

        case 1:

        {

            system("cls");

            cout<<"\n\n\n";

            solve(4,4);

            break;

        }

        case 2:

        {

            system("cls");

            cout<<"\n\n\n";

            solve(8,8);

            break;

        }

        case 3:

        {

            system("cls");

            cout<<"\n\n\n";

            solve(12,12);

            break;

        }

        case 4:

        {

            system("cls");

            cout<<"\n\n\n";

            cout<<"\t\t\t\t\t\tThank you for playing"<<endl;

            break;

        }

        default:

        {

            system("cls");

            cout<<"\n\n\n";

            cout<<"\n \t\t\t\t\t\tWrong choice entered"<<endl;

            break;

        }

    }

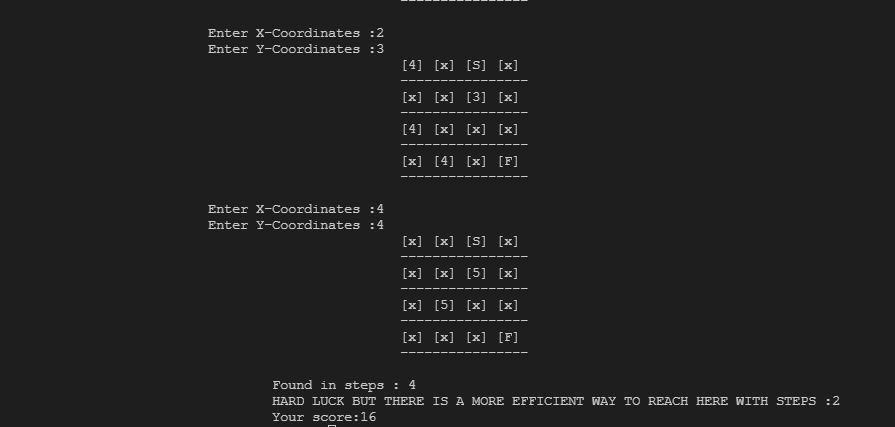
    return 0;

}

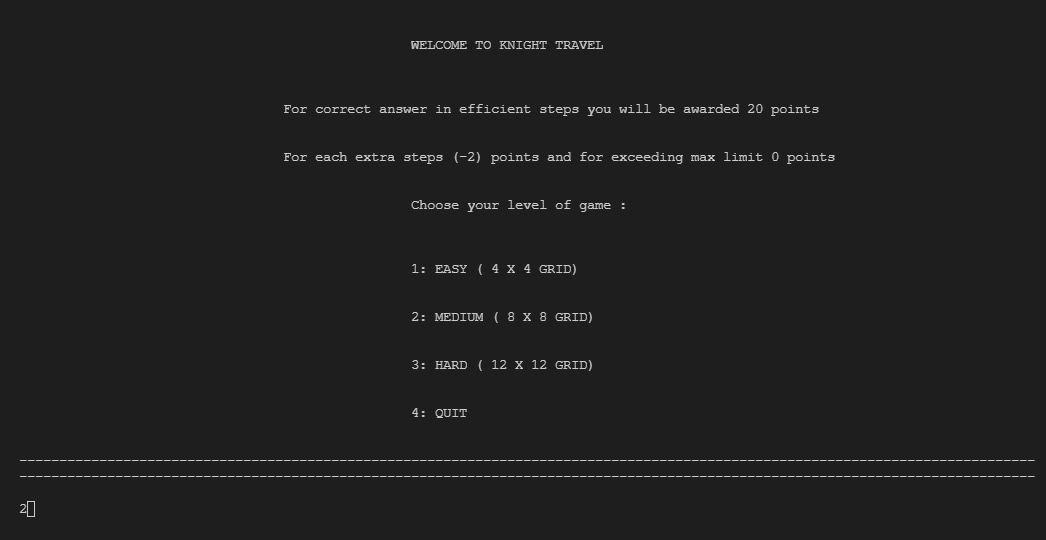
* **OUTPUT:**

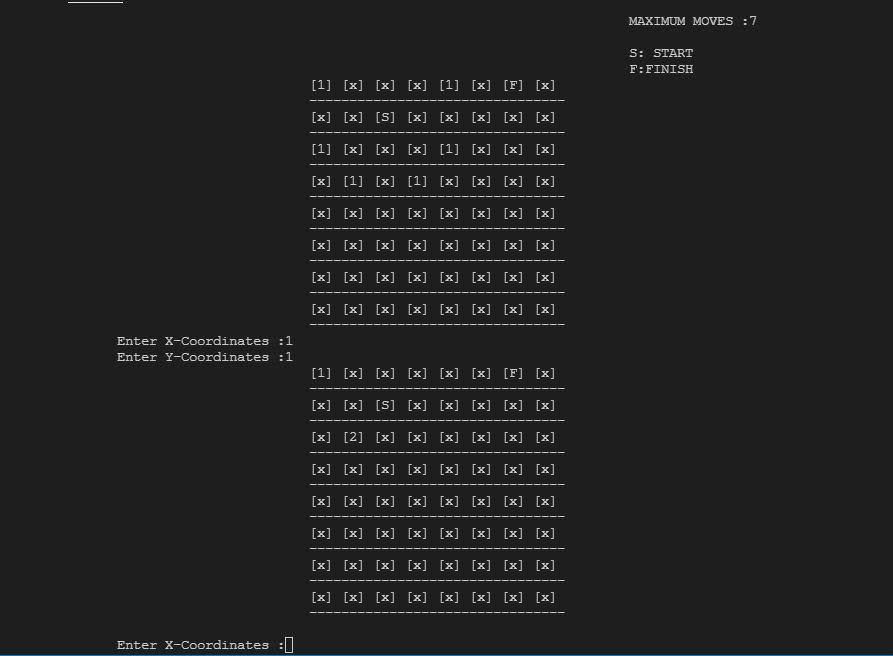
* **Case-1**: Easy level but not Efficient

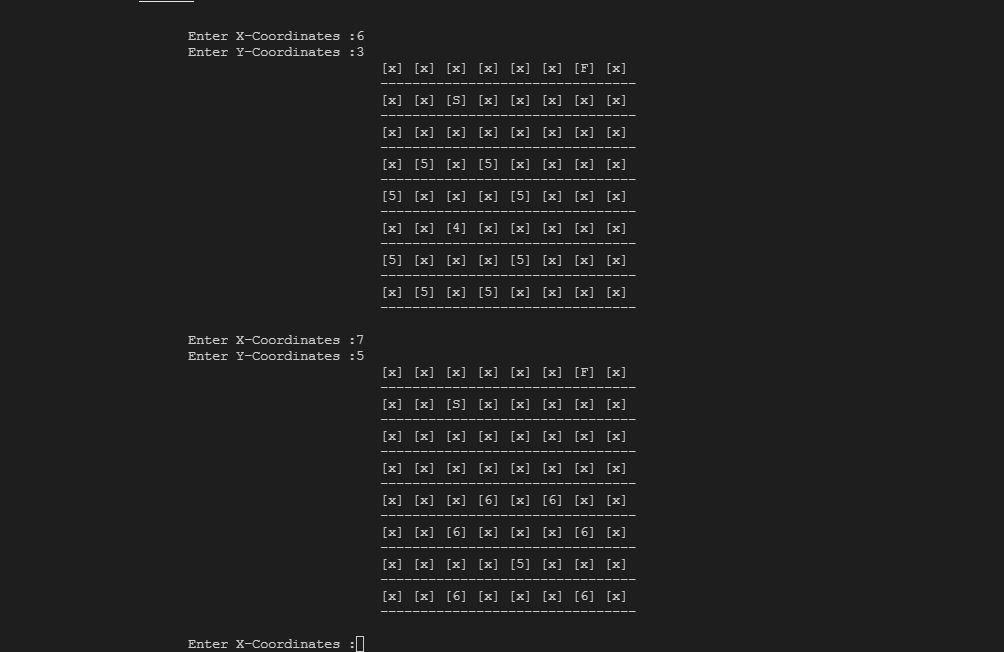


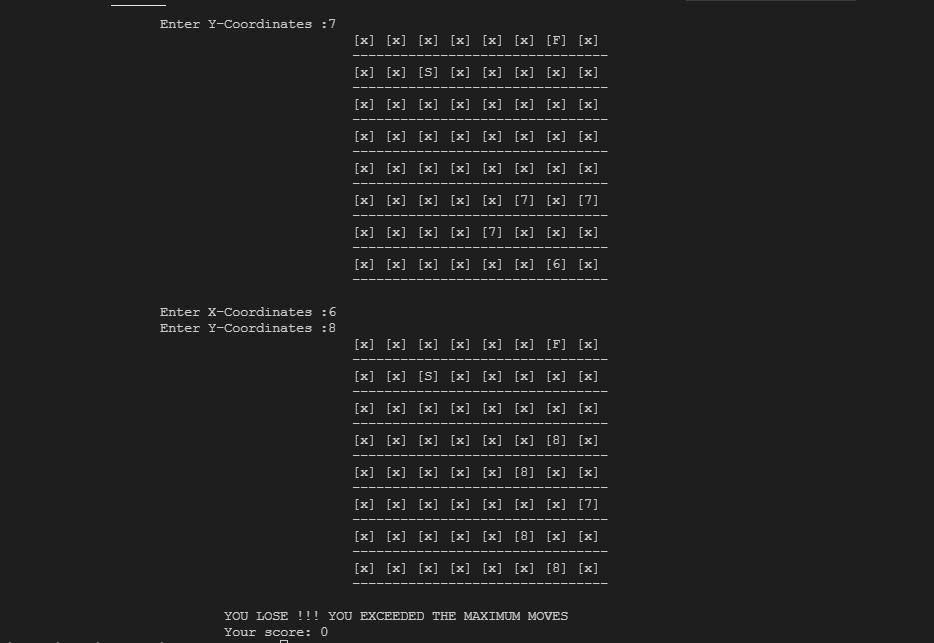


* **Case-2:** Medium level but out of moves

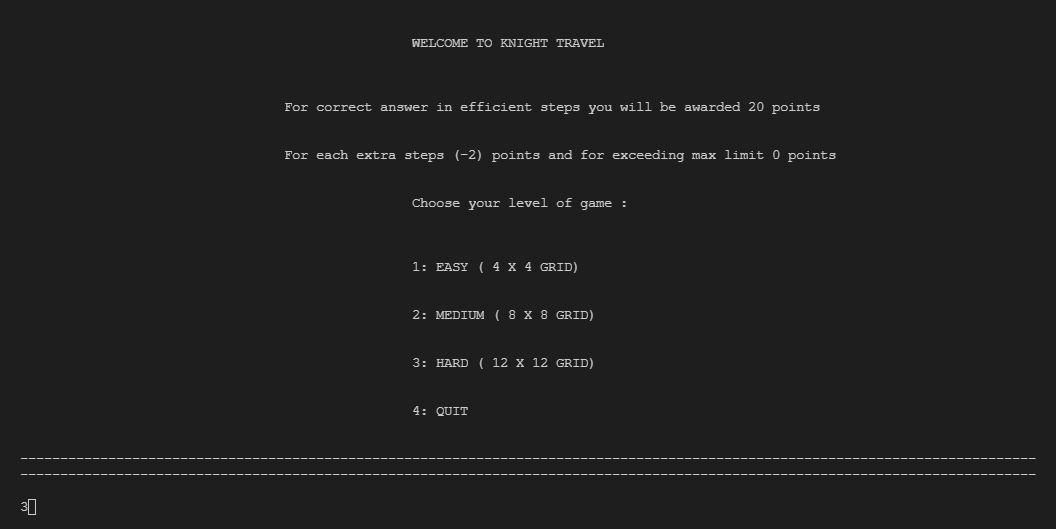


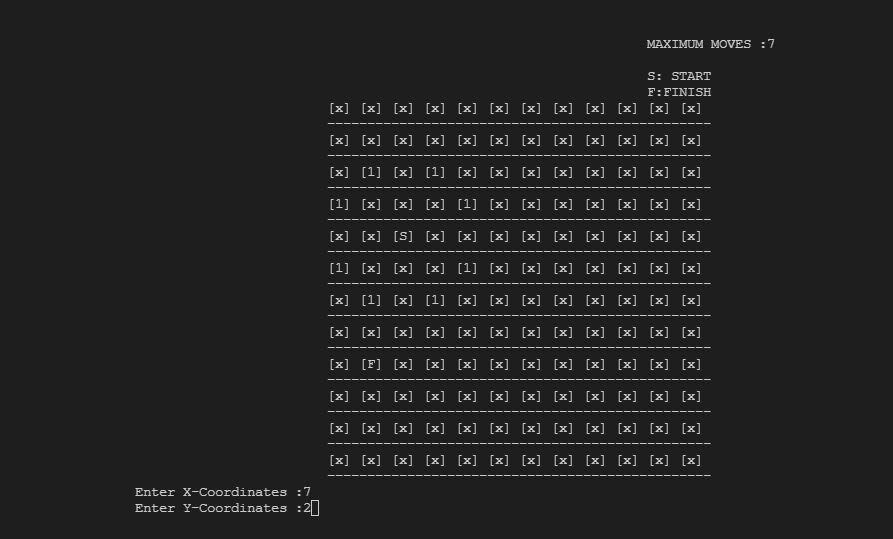






* **Case-3:** Hard level and Efficient way



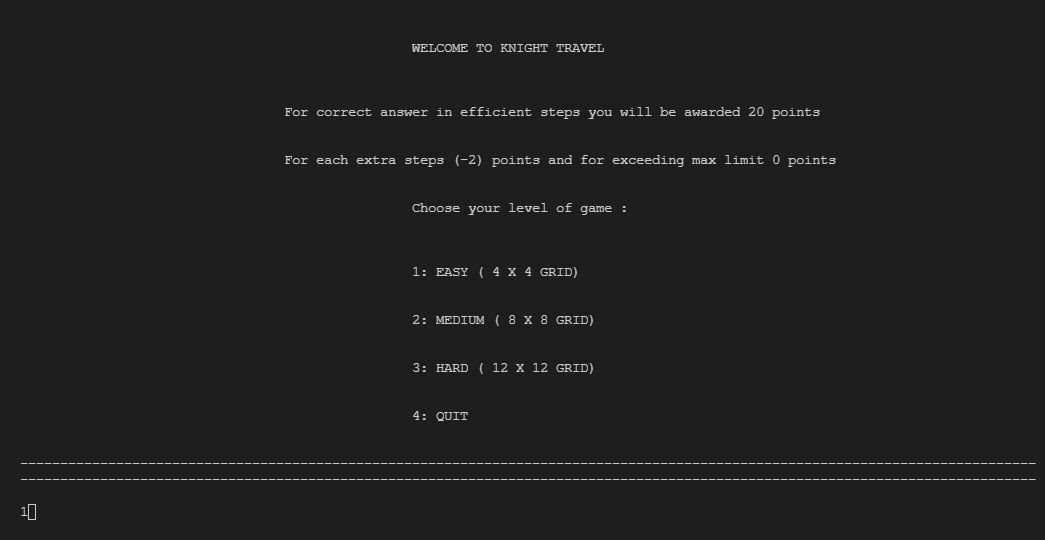






* **Case-4:** Invalid Case







* **Case-5:** Quit

