

NAME:	Arnav Hoskote
UID:	2021300044
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	08
DATE OF PERFORMANCE	17/04/2023
DATE OF SUBMISSION	24/04/2023
AIM:	To use backtracking to solve 15 puzzle problem.
PROBLEM STATEMENT 1:	15 puzzle problem
ALGORITHM and THEORY:	<p>1.If N is odd, then puzzle instance is solvable if number of inversions is even in the input state.</p> <p>2.If N is even, puzzle instance is solvable if</p> <ul style="list-style-type: none"> the blank is on an even row counting from the bottom (second-last, fourth-last, etc.) and number of inversions is odd. the blank is on an odd row counting from the bottom (last, third-last, fifth-last, etc.) and number of inversions is even. <p>3.For all other cases, the puzzle instance is not solvable.</p>

PROGRAM:

```
#include<stdio.h>

int m=0,n=4;

int cal(int temp[10][10],int t[10][10])
{
    int i,j,m=0;
    for(i=0;i < n;i++)
        for(j=0;j < n;j++)
        {
            if(temp[i][j]!=t[i][j])
                m++;
        }
    return m;
}

int check(int a[10][10],int t[10][10])
{
    int i,j,f=1;
    for(i=0;i < n;i++)
        for(j=0;j < n;j++)
            if(a[i][j]!=t[i][j])
                f=0;
    return f;
}

void main()
{
    int p,i,j,n=4,a[10][10],t[10][10],temp[10][10],r[10][10];
    int m=0,x=0,y=0,d=1000,dmin=0,l=0;
    printf("\nEnter the matrix to be solved,space with zero :\n");
    for(i=0;i < n;i++)
```

```

        for(j=0;j < n;j++)
            scanf("%d",&a[i][j]);

printf("\nEnter the target matrix,space with zero :\n");
for(i=0;i < n;i++)
    for(j=0;j < n;j++)
        scanf("%d",&t[i][j]);

printf("\nEnter Matrix is :\n");
for(i=0;i < n;i++)
{
    for(j=0;j < n;j++)
        printf("%d\t",a[i][j]);
    printf("\n");
}

printf("\nTarget Matrix is :\n");
for(i=0;i < n;i++)
{
    for(j=0;j < n;j++)
        printf("%d\t",t[i][j]);
    printf("\n");
}

while(!(check(a,t)))
{
    l++;
    d=1000;
    for(i=0;i < n;i++)
        for(j=0;j < n;j++)
        {
            if(a[i][j]==0)
            {
                x=i;

```

```

        y=j;
    }
}

//To move upwards
for(i=0;i < n;i++)
    for(j=0;j < n;j++)
        temp[i][j]=a[i][j];

if(x!=0)
{
    p=temp[x][y];
    temp[x][y]=temp[x-1][y];
    temp[x-1][y]=p;
}
m=cal(temp,t);
dmin=l+m;
if(dmin < d)
{
    d=dmin;
    for(i=0;i < n;i++)
        for(j=0;j < n;j++)
            r[i][j]=temp[i][j];
}

//To move downwards
for(i=0;i < n;i++)
    for(j=0;j < n;j++)
        temp[i][j]=a[i][j];
if(x!=n-1)
{
    p=temp[x][y];
    temp[x][y]=temp[x+1][y];
    temp[x+1][y]=p;
}

```

```
}  
m=cal(temp,t);  
dmin=l+m;  
if(dmin < d)  
{  
    d=dmin;  
    for(i=0;i < n;i++)  
        for(j=0;j < n;j++)  
            r[i][j]=temp[i][j];  
}
```

```
//To move right side  
for(i=0;i < n;i++)  
    for(j=0;j < n;j++)  
        temp[i][j]=a[i][j];  
if(y!=n-1)  
{  
    p=temp[x][y];  
    temp[x][y]=temp[x][y+1];  
    temp[x][y+1]=p;  
}  
m=cal(temp,t);  
dmin=l+m;  
if(dmin < d)  
{  
    d=dmin;  
    for(i=0;i < n;i++)  
        for(j=0;j < n;j++)  
            r[i][j]=temp[i][j];  
}
```

```
//To move left  
for(i=0;i < n;i++)  
    for(j=0;j < n;j++)
```

```

        temp[i][j]=a[i][j];
    if(y!=0)
    {
        p=temp[x][y];
        temp[x][y]=temp[x][y-1];
        temp[x][y-1]=p;
    }
    m=cal(temp,t);
    dmin=l+m;
    if(dmin < d)
    {
        d=dmin;
        for(i=0;i < n;i++)
            for(j=0;j < n;j++)
                r[i][j]=temp[i][j];
    }

    printf("\nCalculated Intermediate Matrix Value :\n");
    for(i=0;i < n;i++)
    {
        for(j=0;j < n;j++)
            printf("%d\t",r[i][j]);
        printf("\n");
    }
    for(i=0;i < n;i++)
        for(j=0;j < n;j++)
        {
            a[i][j]=r[i][j];
            temp[i][j]=0;
        }
    printf("Minimum cost : %d\n",d);
}
}

```

OUTPUT:

```
students@students-HP-280-G3-MT: ~  
students@students-HP-280-G3-MT:~$ ./a.out  
Enter the matrix to be solved,space with zero :  
1 2 3 4  
5 6 0 8  
9 10 7 11  
13 14 15 12  
  
Enter the target matrix,space with zero :  
1 2 3 4  
5 6 7 8  
9 10 11 12  
13 14 15 0  
  
Entered Matrix is :  
1 2 3 4  
5 6 0 8  
9 10 7 11  
13 14 15 12  
  
Target Matrix is :  
1 2 3 4  
5 6 7 8  
9 10 11 12  
13 14 15 0  
  
Calculated Intermediate Matrix Value :  
1 2 3 4  
5 6 7 8  
9 10 0 11  
13 14 15 12  
Minimum cost : 4  
  
Calculated Intermediate Matrix Value :  
1 2 3 4  
5 6 7 8  
9 10 11 0  
13 14 15 12  
Minimum cost : 4  
  
Calculated Intermediate Matrix Value :  
1 2 3 4  
5 6 7 8  
9 10 11 12  
13 14 15 0  
Minimum cost : 3  
students@students-HP-280-G3-MT:~$
```

CONCLUSION:

By performing above experiment I have understood 15 puzzle problem and I have been able to rearrange the puzzle.