**Name: Mohammad Zain Abbas**

**Reg #: 6865**

**DE-36 (CE), Syndicate: A**

**LAB 8 JOURNEL**

**Equipment Used:** Notebook Computer, Python IDLE 3.6

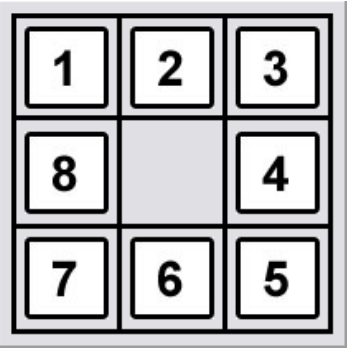
**Lab Tasks:**

An 8-slider puzzle is a combinational puzzle that challenges a player to slide pieces along

certain routes to establish a certain end-configuration. The pieces to be moved may consist

of simple shapes, or they may be imprinted with colors, patterns, sections of a larger picture,

numbers, or letters. One of such example is shown below:



**SOLUTION CODE:**

# -\*- coding: utf-8 -\*-

"""

Created on Mon Nov 20 09:22:52 2017

@author: umerm

"""

import numpy

import cv2

l=2;

m=0;

blank\_space=(l,m);

Graph=numpy.zeros((3,3));

Graph[0,::]=[4,1,3];

Graph[1,::]=[2,8,5];

Graph[2,::]=[9,7,6];

def Genetic\_Algorithm():

ch1=['up','right','up','down','down','right','left','down'];

ch2=['up','left','down','up','down','up','right','right'];

ch3=['up','up','right','up','right','up','right','right'];

ch4=['right','up','left','up','left','down','right','down'];

fitness\_ch1=0;

fitness\_ch2=0;

fitness\_ch3=0;

fitness\_ch4=0;

while(fitness\_ch1!=8 or fitness\_ch2!=8 or fitness\_ch3!=8 or fitness\_ch4!=8):

fitness\_ch1=Fitness\_Function(ch1);

fitness\_ch2=Fitness\_Function(ch2);

fitness\_ch3=Fitness\_Function(ch3);

fitness\_ch4=Fitness\_Function(ch4);

max\_fitness=max([fitness\_ch1,fitness\_ch2,fitness\_ch3,fitness\_ch4]);

temp=ch1[max\_fitness:8];

temp2=ch3[max\_fitness:8];

if(fitness\_ch1!=8):

ch1=ch1[0:max\_fitness];

ch1[max\_fitness:8]=ch2[max\_fitness:8];

if(fitness\_ch2!=8):

ch2=ch2[0:max\_fitness];

ch2[max\_fitness:8]=temp;

if(fitness\_ch3!=8):

ch3=ch3[0:max\_fitness];

ch3[max\_fitness:8]=ch4[max\_fitness:8];

if(fitness\_ch4!=8):

ch4=ch4[0:max\_fitness];

ch4[max\_fitness:8]=temp2;

fitness\_ch1=Fitness\_Function(ch1);

fitness\_ch2=Fitness\_Function(ch2);

fitness\_ch3=Fitness\_Function(ch3);

fitness\_ch4=Fitness\_Function(ch4);

min\_fitness=min(fitness\_ch1,fitness\_ch2,fitness\_ch3,fitness\_ch4);

if(fitness\_ch1==min\_fitness):

ch1=Mutate(ch1,min\_fitness);

if(fitness\_ch2==min\_fitness):

ch2=Mutate(ch2,min\_fitness);

if(fitness\_ch3==min\_fitness):

ch3=Mutate(ch3,min\_fitness);

if(fitness\_ch4==min\_fitness):

ch4=Mutate(ch4,min\_fitness);

fitness\_ch1=Fitness\_Function(ch1);

fitness\_ch2=Fitness\_Function(ch2);

fitness\_ch3=Fitness\_Function(ch3);

fitness\_ch4=Fitness\_Function(ch4);

return fitness\_ch1, fitness\_ch2,fitness\_ch3,fitness\_ch4;

def Mutate(chromosome,min\_fitness):

for k in range(min\_fitness+1):

if(k==min\_fitness):

if(chromosome[k]=='up'):

chromosome[k]='down';

break;

if(chromosome[k]=='down'):

chromosome[k]='up';

break;

if(chromosome[k]=='left'):

chromosome[k]='right';

break

if(chromosome[k]=='right'):

chromosome[k]='left';

break

return chromosome;

def Fitness\_Function(chromosome):

fitness\_value=0;

l=2;

m=0;

for i in range(3):

for j in range(3):

if(Graph[i][j]==9 ):

for k in range(len(chromosome)):

if(chromosome[k]=='up'):

l=l-1;

if(l>=0 and m>=0 and l<=2 and m<=2):

blank\_space=(l,m);

fitness\_value=fitness\_value+1;

else:

break;

elif(chromosome[k]=='down'):

l=l+1;

if(l>=0 and m>=0 and l<=2 and m<=2):

blank\_space=(l,m);

fitness\_value=fitness\_value+1;

else:

break;

elif(chromosome[k]=='right'):

m=m+1;

if(l>=0 and m>=0 and l<=2 and m<=2):

blank\_space=(l,m);

fitness\_value=fitness\_value+1;

else:

break;

elif(chromosome[k]=='left'):

m=m-1;

if(l>=0 and m>=0 and l<=2 and m<=2):

blank\_space=(l,m);

fitness\_value=fitness\_value+1;

else:

break;

return fitness\_value;

print(Genetic\_Algorithm());

**OUTPUT:**

