Assignment

Aim : Genetic Algorithm

Code :

import numpy as np

import pandas as pd

import os

import random

#initialize population

best = -100000

populations =([[random.randint(0,1) for x in range(6)] for i in range(4)])

print(type(populations))

parents=[]

new\_populations = []

print(populations)

#fitness score calculation ............

def fitness\_score() :

global populations,best

fit\_value = []

fit\_score=[]

for i in range(4) :

chromosome\_value=0

for j in range(5,0,-1) :

chromosome\_value += populations[i][j]\*(2\*\*(5-j))

chromosome\_value = -1\*chromosome\_value if populations[i][0]==1 else chromosome\_value

print(chromosome\_value)

fit\_value.append(-(chromosome\_value\*\*2) + 5 )

print(fit\_value)

fit\_value, populations = zip(\*sorted(zip(fit\_value, populations) , reverse = True))

best= fit\_value[0]

#print(type(populations))

#selecting parents....

def selectparent():

global parents

#global populations , parents

parents=populations[0:2]

print(type(parents))

print(parents)

#single-point crossover .........

def crossover() :

global parents

cross\_point = random.randint(0,5)

parents=parents + tuple([(parents[0][0:cross\_point +1] +parents[1][cross\_point+1:6])])

parents =parents+ tuple([(parents[1][0:cross\_point +1] +parents[0][cross\_point+1:6])])

print(parents)

def mutation() :

global populations, parents

mute = random.randint(0,49)

if mute == 20 :

x=random.randint(0,3)

y = random.randint(0,5)

parents[x][y] = 1-parents[x][y]

populations = parents

print(populations)

#fitness\_score()

#selectparent()

#crossover()

#mutation()

for i in range(5) :

fitness\_score()

selectparent()

crossover()

mutation()

print("best score :")

print(best)

print("sequence........")

print(populations[0])

Output:

