**Practical 10A**

import random

**# Number of individuals in each generation**

POPULATION\_SIZE = 100

**# Valid genes**

GENES = '''abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOP

QRSTUVWXYZ 1234567890, .-;:\_!"#%&/()=?@${[]}'''

**# Target string to be generated**

TARGET = "I am InEvitable"

class Individual(object):

    '''Class representing individual in population'''

def \_\_init\_\_(self,chromosome):

        self.chromosome = chromosome

        self.fitness = self.cal\_fitness()

    @classmethod

    def mutated\_genes(self):

        '''create random genes for mutation'''

        global GENES

        gene = random.choice(GENES)

        return gene

    @classmethod

    def create\_gnome(self):

        '''create chromosome or string of genes'''

        global TARGET

        gnome\_len = len(TARGET)

        return [self.mutated\_genes() for \_ in range(gnome\_len)]

    def mate(self, par2):

        '''Perform mating and produce new offspring'''

**# chromosome for offspring**

        child\_chromosome = []

        for gp1, gp2 in zip(self.chromosome, par2.chromosome):

            prob = random.random()

            if prob < 0.45:

                child\_chromosome.append(gp1)

            elif prob < 0.90:

                child\_chromosome.append(gp2)

            else:

                child\_chromosome.append(self.mutated\_genes())

        return Individual(child\_chromosome)

    def cal\_fitness(self):

        '''Calculate fittness score, it is the number of

        characters in string which differ from target string.'''

        global TARGET

        fitness = 0

        for gs, gt in zip(self.chromosome, TARGET):

            if gs != gt: fitness+= 1

        return fitness

**# Driver code**

def main():

    global POPULATION\_SIZE

    generation = 1

    found = False

    population = []

    for \_ in range(POPULATION\_SIZE):

                gnome = Individual.create\_gnome()

                population.append(Individual(gnome))

    while not found:

        population = sorted(population, key = lambda x:x.fitness)

        if population[0].fitness <= 0:

            found = True

            break

        new\_generation = []

        s = int((10\*POPULATION\_SIZE)/100)

        new\_generation.extend(population[:s])

        s = int((90\*POPULATION\_SIZE)/100)

        for \_ in range(s):

            parent1 = random.choice(population[:50])

            parent2 = random.choice(population[:50])

            child = parent1.mate(parent2)

            new\_generation.append(child)

        population = new\_generation

        print("Generation: {}\tString: {}\tFitness: {}".format(generation, "".join(population[0].chromosome), population[0].fitness))

        generation += 1

    print("Generation: {}\tString: {}\tFitness: {}".format(generation, "".join(population[0].chromosome), population[0].fitness))

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**Output:**

Generation: 1 String: g %5 Xu}DX0M}ys Fitness: 13

Generation: 2 String: g %5 Xu}DX0M}ys Fitness: 13

Generation: 3 String: WEQJ1:nd} t?b4p Fitness: 12

Generation: 4 String: 0JaNRIjz1rWwbl& Fitness: 11

Generation: 5 String: IGK.CIByuMtwgle Fitness: 10

Generation: 6 String: IJaNRIBy1Mtagle Fitness: 8

Generation: 7 String: IJaN IBlD}tagle Fitness: 7

Generation: 8 String: IJaN IBlD}tagle Fitness: 7

Generation: 9 String: IJaN IBlD}tagle Fitness: 7

Generation: 10 String: IPaNkInAv;Dable Fitness: 6

Generation: 11 String: IPaNkInAv;Dable Fitness: 6

Generation: 12 String: IPaNkInAv;Dable Fitness: 6

Generation: 13 String: IJaN InADitable Fitness: 4

Generation: 14 String: IJaN InADitable Fitness: 4

Generation: 15 String: IJaN InADitable Fitness: 4

Generation: 16 String: IJaN InADitable Fitness: 4

Generation: 17 String: I aj In44itable Fitness: 3

Generation: 18 String: I a) InHvitable Fitness: 2

Generation: 19 String: I a) InHvitable Fitness: 2

Generation: 20 String: I a) InHvitable Fitness: 2

Generation: 21 String: I a) InHvitable Fitness: 2

Generation: 22 String: I a) InHvitable Fitness: 2

Generation: 23 String: I a) InHvitable Fitness: 2

Generation: 24 String: I a) InHvitable Fitness: 2

Generation: 25 String: I am In(vitable Fitness: 1

Generation: 26 String: I am In(vitable Fitness: 1

Generation: 27 String: I am In(vitable Fitness: 1

Generation: 28 String: I am In(vitable Fitness: 1

Generation: 29 String: I am In(vitable Fitness: 1

Generation: 30 String: I am In(vitable Fitness: 1

Generation: 31 String: I am In(vitable Fitness: 1

Generation: 32 String: I am InEvitable Fitness: 0