**TASK 1:**

Using Genetic Algorithms to guess a password given the number of correct letters in the guess. Build a mutation engine

**CODE:**

*# Owned*

\_\_author\_\_ *=* "Qaiser Abbas"

\_\_copyright\_\_ *=* "Copyright 2020, Artificial Intelligence lab-10"

\_\_email\_\_ *=* "qaiserabbas889@yahoo.com"

*#===============================================================*

*# {code}*

*import* random

*import* datetime

*import* sys

*import* time

geneSet *=* 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!. '

password *=* "Genetic Algorithm by Qaiser!"

*def* *generate\_parent*(length):

    gene *=* []

*while* len(gene) *<* length:

        sample\_size *=* min(length *-* len(gene), len(geneSet))

*# appends list with values ==> ['X','B'].extend('a','y') -> ['X','B','a','y']*

        gene.extend(random.sample(geneSet, sample\_size))

*# converts list to str ==> ['a', 'b', ' ', 'C', 'N', 'x'] -> ab CNx*

*return* ''.join(gene)

*def* *get\_fitness*(guess):

*return* sum(1 *for* expected, actual *in* zip(password, guess) *if* expected *==* actual)

*def* *mutate*(parent):

    index *=* random.randrange(0, len(parent))

    child\_genes *=* list(parent)

    new\_gene, alternate *=* random.sample(geneSet, 2)

    child\_genes[index] *=* alternate *if* new\_gene *==* child\_genes[index] *else* new\_gene

*return* ''.join(child\_genes)

*def* *display*(guess):

    time\_diff *=* datetime.datetime.now() *-* startTime

    fitness *=* get\_fitness(guess)

*# print('{0}\t{1}\t{2}'.format(guess, fitness, time\_diff))*

    time.sleep(.001)

    sys.stdout.write('\rGeneration #' *+* '\t' *+* guess)

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

    random.seed(4)

    startTime *=* datetime.datetime.now()

    best\_parent *=* generate\_parent(len(password))

    best\_fitness *=* get\_fitness(best\_parent)

    display(best\_parent)

*while* True:

        child *=* mutate(best\_parent)

        child\_fitness *=* get\_fitness(child)

        display(child)

*if* best\_fitness *>=* child\_fitness:

*continue*

*if* child\_fitness *>=* len(best\_parent):

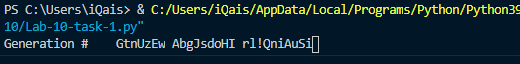
*break*

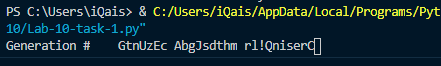
        best\_fitness *=* child\_fitness

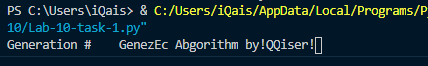
        best\_parent *=* child

**OUTPUT:**

**Generating Password:**







**Password Generated:**

