Laporan Tugas Algoritma genetic  
  
Menggunakan Python

Berikut source code nya :

import random

a\_batas\_atas = 30

b\_batas\_atas = 10

c\_batas\_atas = 10

d\_batas\_atas = 10

jumlah\_populasi = 6

NUM\_GENERATIONS = 50

CROSSOVER\_PROBABILITY = 0.8

MUTATION\_PROBABILITY = 0.1

NUM\_ELITES = 5

def evaluasi\_kromosom(kromosom):

a, b, c, d = kromosom

return abs((a + 4\*b + 2\*c + 3\*d) - 30)

def inisialisasi\_populasi():

populasi = []

for \_ in range(jumlah\_populasi):

kromosom = [random.randint(0, a\_batas\_atas),

random.randint(0, b\_batas\_atas),

random.randint(0, c\_batas\_atas),

random.randint(0, d\_batas\_atas)]

populasi.append(kromosom)

return populasi

def select\_parents(populasi):

total\_fitness = sum(1 / (1 + evaluasi\_kromosom(kromosom)) for kromosom in populasi)

probabilities = [(1 / (1 + evaluasi\_kromosom(kromosom))) / total\_fitness for kromosom in populasi]

parents = []

for \_ in range(jumlah\_populasi):

parent = random.choices(populasi, probabilities)[0]

parents.append(parent)

return parents

def crossover(parent1, parent2):

if random.random() < CROSSOVER\_PROBABILITY:

crossover\_point = random.randint(1, len(parent1) - 1)

child1 = parent1[:crossover\_point] + parent2[crossover\_point:]

child2 = parent2[:crossover\_point] + parent1[crossover\_point:]

return child1, child2

else:

return parent1, parent2

def mutate(kromosom):

mutated\_kromosom = kromosom.copy()

for i in range(len(mutated\_kromosom)):

if random.random() < MUTATION\_PROBABILITY:

mutated\_kromosom[i] = random.randint(0, [a\_batas\_atas, b\_batas\_atas, c\_batas\_atas, d\_batas\_atas][i])

return mutated\_kromosom

def select\_new\_generation(populasi, parents, elites):

new\_generation = elites.copy()

while len(new\_generation) < jumlah\_populasi:

parent1, parent2 = random.sample(parents, 2)

child1, child2 = crossover(parent1, parent2)

new\_generation.extend([mutate(child1), mutate(child2)])

return new\_generation[:jumlah\_populasi]

populasi = inisialisasi\_populasi()

evaluasi\_populasi = [(kromosom, evaluasi\_kromosom(kromosom)) for kromosom in populasi]

for \_ in range(NUM\_GENERATIONS):

evaluasi\_populasi.sort(key=lambda x: x[1])

elites = [kromosom for kromosom, \_ in evaluasi\_populasi[:NUM\_ELITES]]

parents = select\_parents([kromosom for kromosom, \_ in evaluasi\_populasi])

populasi = select\_new\_generation(populasi, parents, elites)

evaluasi\_populasi = [(kromosom, evaluasi\_kromosom(kromosom)) for kromosom in populasi]

best\_kromosom, best\_fitness = min(evaluasi\_populasi, key=lambda x: x[1])

print("Solusi terbaik:")

print("Nilai a =", best\_kromosom[0])

print("Nilai b =", best\_kromosom[1])

print("Nilai c =", best\_kromosom[2])

print("Nilai d =", best\_kromosom[3])

print("Fitness terbaik:", best\_fitness)

Hasil Akhir berupa nilai dari tiap variable yakni a, b, c, dan d serta nilai Fitnessnya.