1. Genetic Algorithm **1BM22CS251**

import numpy as np

# Parameters

population\_size = 100

generations = 50

mutation\_rate = 0.1

crossover\_rate = 0.7

range\_min = -10

range\_max = 10

# Fitness function

def fitness(x):

    return x\*\*2

# Initialization of the population

def initialize\_population(size, range\_min, range\_max):

    return np.random.uniform(range\_min, range\_max, size)

# Selection using roulette wheel selection

def select\_parents(population):

    fitness\_values = fitness(population)

    total\_fitness = np.sum(fitness\_values)

    selection\_probs = fitness\_values / total\_fitness

    selected\_indices = np.random.choice(len(population), size=2, p=selection\_probs)

    return population[selected\_indices]

# Crossover to create offspring

def crossover(parent1, parent2):

    if np.random.rand() < crossover\_rate:

        return (parent1 + parent2) / 2  # Simple average crossover

    return parent1  # No crossover occurs

# Mutation

def mutate(offspring, range\_min, range\_max):

    if np.random.rand() < mutation\_rate:

        return np.random.uniform(range\_min, range\_max)

    return offspring

# Genetic Algorithm function

def genetic\_algorithm():

    # Step 1: Initialize population

    population = initialize\_population(population\_size, range\_min, range\_max)

    for generation in range(generations):

        new\_population = []

        # Step 2: Create new population

        for \_ in range(population\_size):

            parent1, parent2 = select\_parents(population)

            child = crossover(parent1, parent2)

            child = mutate(child, range\_min, range\_max)

            new\_population.append(child)

        population = np.array(new\_population)

    # Return the best solution found

    best\_solution = population[np.argmax(fitness(population))]

    return best\_solution, fitness(best\_solution)

# Running the genetic algorithm

best\_x, best\_fitness = genetic\_algorithm()

print(f"Best x: {best\_x}, Maximum f(x): {best\_fitness}")

Output:

