LAB 07

GA mechanism for optimization process: The following is a sequence of steps of GA mechanism when used for optimization of problems.

Step 1: Generate the initial population randomly.
Step 2: Select the initial solution with best fitness values.
Step 3: Recombine the selected solutions using mutation and crossover operators.
Step 4: Insert an offspring into the population.
Step 5: Now, if the stop condition is met, return the solution with their best fitness value. Else go to step 2.

```
import random
from deap import base, creator, tools


def eval_func(individual):
    target_sum = 15
    return len(individual) - abs(sum(individual) - target_sum),


def create_toolbox(num_bits):
    creator.create("FitnessMax", base.Fitness, weights=(1.0,))
    creator.create("Individual", list, fitness=creator.FitnessMax)

    toolbox = base.Toolbox()

    toolbox.register("attr_bool", random.randint, 0, 1)
    toolbox.register("individual", tools.initRepeat,
                     creator.Individual, toolbox.attr_bool, num_bits)
    toolbox.register("population", tools.initRepeat, list, toolbox.individual)

    toolbox.register("evaluate", eval_func)

    toolbox.register("mate", tools.cxTwoPoint)

    toolbox.register("mutate", tools.mutFlipBit, indpb=0.05)

    toolbox.register("select", tools.selTournament, tournsize=3)

    return toolbox


if __name__ == "__main__":
    num_bits = 45
    toolbox = create_toolbox(num_bits)
    random.seed(7)
```

```
population = toolbox.population(n=500)
probab_crossing, probab_mutating = 0.5, 0.2
num_generations = 10
print('\nEvolution process starts')
fitnesses = list(map(toolbox.evaluate, population))

for ind, fit in zip(population, fitnesses):
    ind.fitness.values = fit
print('\nEvaluated', len(population), 'individuals')

for g in range(num_generations):
    print("\n- Generation", g)
    offspring = toolbox.select(population, len(population))
    offspring = list(map(toolbox.clone, offspring))

    for child1, child2 in zip(offspring[::2], offspring[1::2]):
        if random.random() < probab_crossing:
            toolbox.mate(child1, child2)
            del child1.fitness.values
            del child2.fitness.values

    for mutant in offspring:
        if random.random() < probab_mutating:
            toolbox.mutate(mutant)
            del mutant.fitness.values

    invalid_ind = [ind for ind in offspring if not ind.fitness.valid]
    fitnesses = list(map(toolbox.evaluate, invalid_ind))

    for ind, fit in zip(invalid_ind, fitnesses):
        ind.fitness.values = fit
    print('Evaluated', len(invalid_ind), 'individuals')

    population[:] = offspring
    fits = [ind.fitness.values[0] for ind in population]
    length = len(population)
    mean = sum(fits) / length
    sum2 = sum(x*x for x in fits)
    std = abs(sum2 / length - mean**2)**0.5
    print('Min =', min(fits), ', Max =', max(fits))
    print('Average =', round(mean, 2),
          ', Standard deviation =', round(std, 2))

print("\nEvolution ends")
```

```
    best_ind = tools.selBest(population, 1)[0]
    print('\nBest individual:\n', best_ind)
    print('\nNumber of ones:', sum(best_ind))
```

OUTPUT:

```
PS C:\Users\amasw\Downloads\HTML> & C:/Users/amasw/AppData/Local/Programs/Python/Python311/python.exe c:/Users/amasw/Downloads/GeneticAlgo.py

Evolution process starts

Evaluated 500 individuals

- Generation 0
Evaluated 295 individuals
Min = 32.0 , Max = 45.0
Average = 40.29 , Standard deviation = 2.61

- Generation 1
Evaluated 292 individuals
Min = 34.0 , Max = 45.0
Average = 42.35 , Standard deviation = 1.91

- Generation 2
Evaluated 277 individuals
Min = 37.0 , Max = 45.0
Average = 43.39 , Standard deviation = 1.46

- Generation 3
Evaluated 321 individuals
Min = 39.0 , Max = 45.0
Average = 43.79 , Standard deviation = 1.18

- Generation 4
Evaluated 303 individuals
Min = 39.0 , Max = 45.0
Average = 44.0 , Standard deviation = 1.19

- Generation 5
Evaluated 292 individuals
Min = 40.0 , Max = 45.0
Average = 44.07 , Standard deviation = 1.1
```

```
- Generation 4
Evaluated 303 individuals
Min = 39.0 , Max = 45.0
Average = 44.0 , Standard deviation = 1.19

- Generation 5
Evaluated 292 individuals
Min = 40.0 , Max = 45.0
Average = 44.07 , Standard deviation = 1.1

- Generation 6
Evaluated 283 individuals
Min = 39.0 , Max = 45.0
Average = 44.22 , Standard deviation = 1.11

- Generation 7
Evaluated 292 individuals
Min = 40.0 , Max = 45.0
Average = 44.15 , Standard deviation = 1.14

- Generation 8
Evaluated 296 individuals
Min = 39.0 , Max = 45.0
Average = 44.08 , Standard deviation = 1.27

- Generation 9
Evaluated 299 individuals
Min = 40.0 , Max = 45.0
Average = 44.12 , Standard deviation = 1.11

Evolution ends

Best individual:
 [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1]

Number of ones: 15
PS C:\Users\amasw\Downloads\HTML>
```