# AI P&T HONOURS LAB INTERNAL II

**Name : Shaik Mohammed Sameer Ht.No : 1602-20-737-168**

## Question:

**Solve One-max puzzle with Evolutionary computing. it should work for any size of bit vector.**

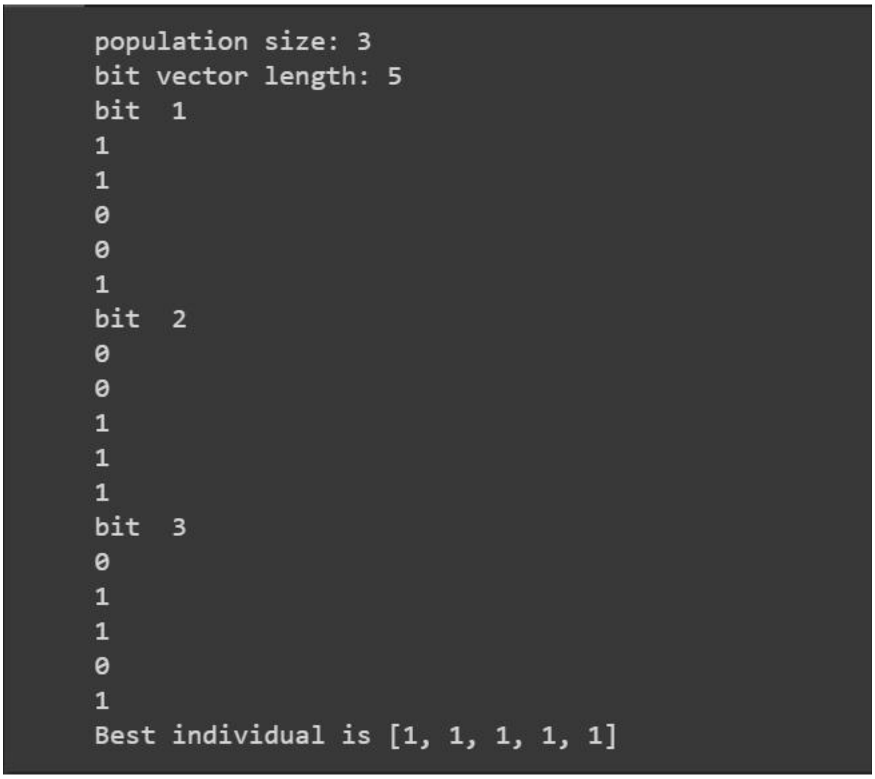
**You can choose single objective or multi-objective, your own mating, mutation and selection operators**

## CODE:

```
import random
POP_SIZE = int(input("population size: "))
BIT_VECTOR_LEN = int(input("bit vector length: "))
MUTATION_PROB = 0.01
NUM_ITERATIONS = 100
population = []
for i in range(POP_SIZE):
  bit_vector = []
  print("bit  %s "%(i+1))
  for j in range(BIT_VECTOR_LEN):
    bit_vector.append(int(input()))
  population.append(bit_vector)
for i in range(NUM_ITERATIONS):
  fitness = []
  for j in range(POP_SIZE):
    fitness.append((sum(population[j]), j))
  fitness.sort(reverse=True)

  new_population = []
  for j in range(POP_SIZE):
    parent1 = population[fitness[j][1]]
    parent2 = population[fitness[(j + 1) % POP_SIZE][1]]
    # Crossover
    crossover_point = random.randint(0, BIT_VECTOR_LEN - 1)
    child = parent1[:crossover_point] + parent2[crossover_point:]
    # Mutate
    for k in range(BIT_VECTOR_LEN):
      if random.random() < MUTATION_PROB:
        child[k] = 1 - child[k]
    # Add child to new population
    new_population.append(child)
  # Replace old population with new population
  population = new_population
fittest = max(population, key=sum)
print("Best individual is %s "%fittest)
```

# OUTPUT:



```
population size: 3
bit vector length: 5
bit  1
1
1
0
0
1
bit  2
0
0
1
1
1
bit  3
0
1
1
0
1
Best individual is [1, 1, 1, 1, 1]
```