Subject - SCOA

```
import random
best=-100000
populations =([[random.randint(0,1) for x in range(6)] for i in range(4)])
print(type(populations))
parents=[]
new_populations = []
print(populations)
```

```
 <class 'list'>
 [[0, 0, 0, 0, 1, 0], [0, 0, 1, 1, 1, 1], [0, 1, 1, 1, 1, 0], [1, 0, 1, 1, 1, 1]]
```

```
def fitness_score() :
    global populations,best
    fit_value = []
    fit_score=[]
    for i in range(4) :
        chromosome_value=0

        for j in range(5,0,-1) :
            chromosome_value += populations[i][j]*(2**(5-j))
        chromosome_value = -1*chromosome_value if populations[i][0]==1 else chromosome_val
        print(chromosome_value)
        fit_value.append(-(chromosome_value**2) + 5 )
    print(fit_value)
    fit_value, populations = zip(*sorted(zip(fit_value, populations) , reverse = True))
    best= fit_value[0]

fitness_score()
```

```
 2
 15
 30
 -15
 [1, -220, -895, -220]
```

```
def selectparent():
    global parents
    #global populations , parents
    parents=populations[0:2]
    print(type(parents))
    print(parents)
selectparent()
```

```
 <class 'tuple'>
 ([0, 0, 0, 0, 1, 0], [1, 0, 1, 1, 1, 1])
```

```
def crossover() :
    global parents
```

```
    cross_point = random.randint(0,5)
    parents=parents + tuple([(parents[0][0:cross_point +1] +parents[1][cross_point+1:6])])
    parents =parents+ tuple([(parents[1][0:cross_point +1] +parents[0][cross_point+1:6])])

    print(parents)


crossover()
```

```
 ([0, 0, 0, 0, 1, 0], [1, 0, 1, 1, 1, 1], [0, 0, 1, 1, 1, 1], [1, 0, 0, 0, 1, 0])
```

```
def mutation() :
    global populations, parents
    mute = random.randint(0,49)
    if mute == 20 :
        x=random.randint(0,3)
        y = random.randint(0,5)
        parents[x][y] = 1-parents[x][y]
    populations = parents
    print(populations)
mutation()
```

```
 ([0, 0, 0, 0, 1, 0], [1, 0, 1, 1, 1, 1], [0, 0, 1, 1, 1, 1], [1, 0, 0, 0, 1, 0])
```

```
for i in range(1000) :
    fitness_score()
    selectparent()
    crossover()
    mutation()
print("best score :")
print(best)
print("sequence........")
print(populations[0])
```

```
 (⌊1, 0, 0, 0, 0, 0⌋, ⌊1, 0, 0, 0, 0, 0⌋, ⌊1, 0, 0, 0, 0, 0⌋, ⌊1, 0, 0, 0, 0, 0⌋)
 0
 0
 0
 0
 [5, 5, 5, 5]
 <class 'tuple'>
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 0
 0
 0
 0
 [5, 5, 5, 5]
 <class 'tuple'>
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 ([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
 0
 0

 0
```

```
0
[5, 5, 5, 5]
<class 'tuple'>
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
0
0
0
0
[5, 5, 5, 5]
<class 'tuple'>
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
0
0
0
0
[5, 5, 5, 5]
<class 'tuple'>
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
0
0
0
0
[5, 5, 5, 5]
<class 'tuple'>
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
([1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0])
best score :
5
sequence........
[1, 0, 0, 0, 0, 0]
```