In [1]:
```
from deap import algorithms, tools,base, creator
import random
import matplotlib.pyplot as plt
```

In [2]:
```
# Step 1: Define the problem (Maximizing the sum of bits
```

In [3]:
```

creator.create("FitnessMax", base.Fitness, weights=(1.0,))


creator.create("Individual", list, fitness=creator.FitnessMax)
```

In [4]:
```
# Step 2: Set up the toolbox
```

In [5]:
```
toolbox = base.Toolbox()

toolbox.register('attr_int', random.randint, 0,1)

toolbox.register('individual',tools.initRepeat,creator.Individual,toolbox.attr_int,n = 100)

toolbox.register('population', tools.initRepeat, list, toolbox.individual)


```

In [6]:
```
# Step 3: Define the evaluation function
```

In [7]:
```
def evalOneMax(individual):
    return sum(individual),
```

In [8]:
```
toolbox.register("evaluate", evalOneMax)
```

In [9]:
```
#register genetic operators
```

In [11]:

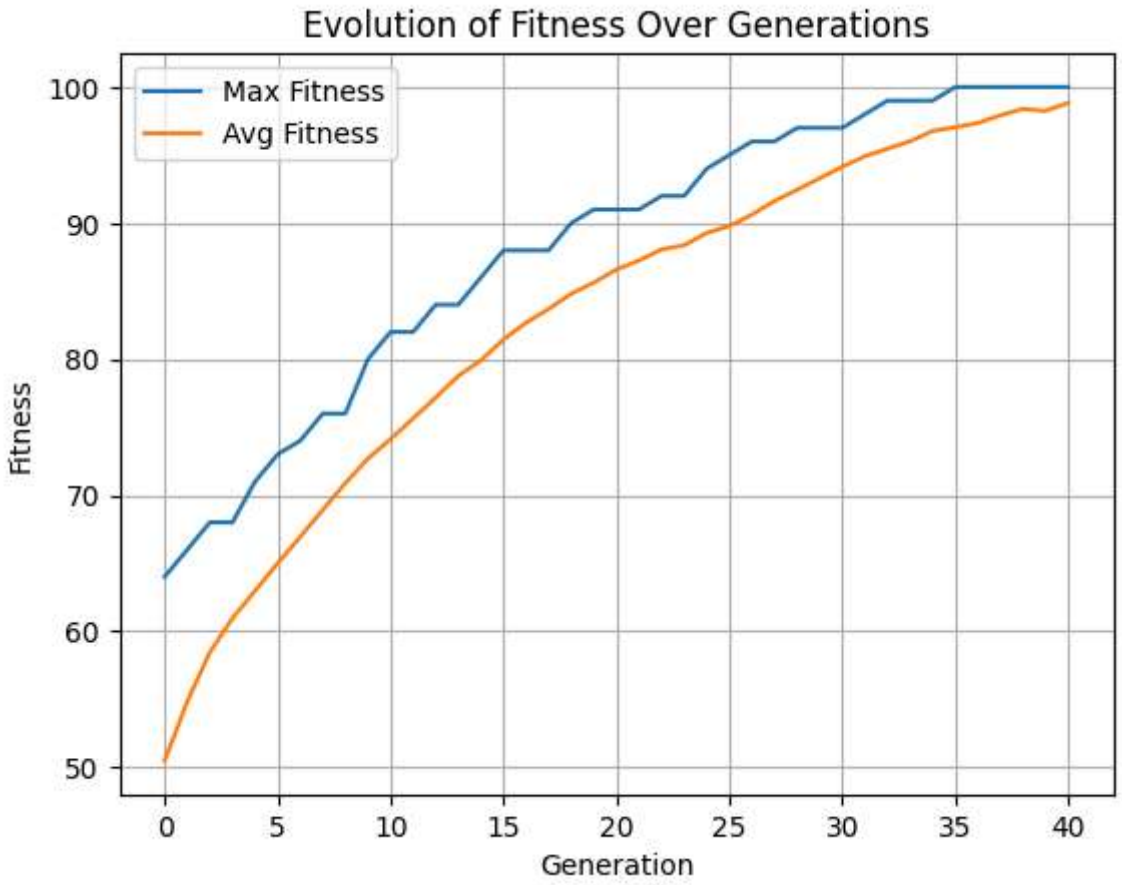
```
toolbox.register("mate", tools.cxTwoPoint) # crossover

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

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

In [26]:
```
def main():
    random.seed(64)

    pop = toolbox.population(n = 300)

    stats =tools.Statistics(lambda ind : ind.fitness.values)
    stats.register("avg", lambda x: sum(v[0] for v in x) / len(x))
    stats.register("min", lambda x : min(v[0] for v in x))
    stats.register("max", lambda x: max(v[0] for v in x))


    pop, logbook = algorithms.eaSimple(pop, toolbox,
                                       cxpb = 0.5,
                                       mutpb = 0.2,
                                       ngen = 40,
                                       stats = stats,
                                       verbose = True)

    best_ind = tools.selBest(pop, 1)[0]
    print ("best Individual :", best_ind)
    print("fitness of best individual:", best_ind.fitness.values[0])


    gen = logbook.select('gen')
    max_fitness_values = logbook.select("max")

    avg_fitness_values = logbook.select("avg")
    plt.plot(gen, max_fitness_values, label='Max Fitness')
    plt.plot(gen, avg_fitness_values, label='Avg Fitness')
    plt.xlabel('Generation')
    plt.ylabel('Fitness')
    plt.title('Evolution of Fitness Over Generations')
    plt.legend()
    plt.grid()
    plt.show()
```

In [27]:
```
if __name__ == "__main__":
    main()
```

```
gen     nevals  avg     min     max
0       300     50.4933 34      64
1       181     54.8333 44      66
2       191     58.4567 47      68
3       199     60.9533 52      68
4       167     62.96   47      71
5       175     64.99   57      73
6       168     66.9333 58      74
7       187     68.9167 59      76
8       171     70.8867 62      76
9       155     72.69   62      80
10      171     74.1233 64      82
11      191     75.64   65      82
12      171     77.18   69      84
13      173     78.7667 69      84
14      185     79.9067 72      86
15      205     81.4433 72      88
16      163     82.6767 74      88
17      175     83.6833 76      88
18      181     84.8067 74      90
19      179     85.6233 74      91
20      178     86.58   78      91
21      173     87.2533 78      91
22      155     88.06   79      92
23      187     88.37   80      92
24      184     89.2767 82      94
25      198     89.7767 80      95
26      185     90.6233 80      96
27      160     91.62   82      96
28      182     92.45   83      97
29      171     93.2933 84      97
30      184     94.1433 84      97
31      161     94.91   85      98
32      181     95.4633 85      99
33      177     96.02   88      99
34      182     96.7733 88      99
35      177     97.0433 86      100
36      161     97.3567 88      100
37      178     97.9167 90      100
38      176     98.4    87      100
39      202     98.2467 88      100
40      180     98.8333 90      100
best Individual : [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
fitness of best individual: 100.0
```



In [ ]: