**Genetic Algorithm**

**Aim :** Create a program to depict the working of genetic algorithm.

**Knapsack Problem**

#knapsack

import random

n=int(input("Enter the number of elements in knapsack : "))

child=list()

w=[0,0,0,0]

v=[0,0,0,0]

best=[]

weight=[int(y) for y in input("Enter weight of items : ").split()]

value=[int(y) for y in input("Enter values of items : ").split()]

capacity=int(input("Knapsack capacity : "))

generation=int(input("Enter number of generation : "))

#n bits are needed, 2^n values

#x=random.randint(0,1)

parent=[[0,1,1,0],[0,1,0,1],[1,1,0,1],[1,1,1,1]]

#for j in range(pow(2,n)):

#for i in range(n):

#x[j].append(random.randint(0,1))

def fitness(n):

w=[0]\*len(parent)

v=[0]\*len(parent)

for j in range(len(parent)):

for i in range(n):

w[j]=w[j] + parent[j][i]\*weight[i]

v[j]=v[j] + parent[j][i]\*value[i]

if w[j]<=capacity:

continue

else:

v[j]=0

print("\n Parent",parent)

print(" weight and value",w,v)

fit1=sorted(v)[-1]

fit2=sorted(v)[-2]

m=v.index(fit1)

n=v.index(fit2)

best[0:3]=parent[m].copy(),w[m],v[m]

child[0:2]=parent[m].copy(),parent[n].copy()

print(" child", child)

child[0][-1],child[1][-1]=child[1][-1],child[0][-1]

print("\n crossed" , child)

child[0][0]=0 if child[0][0]==1 else 1

child[1][0]=0 if child[1][0]==1 else 1

print(" mutated", child)

parent.extend([child[0],child[1]])

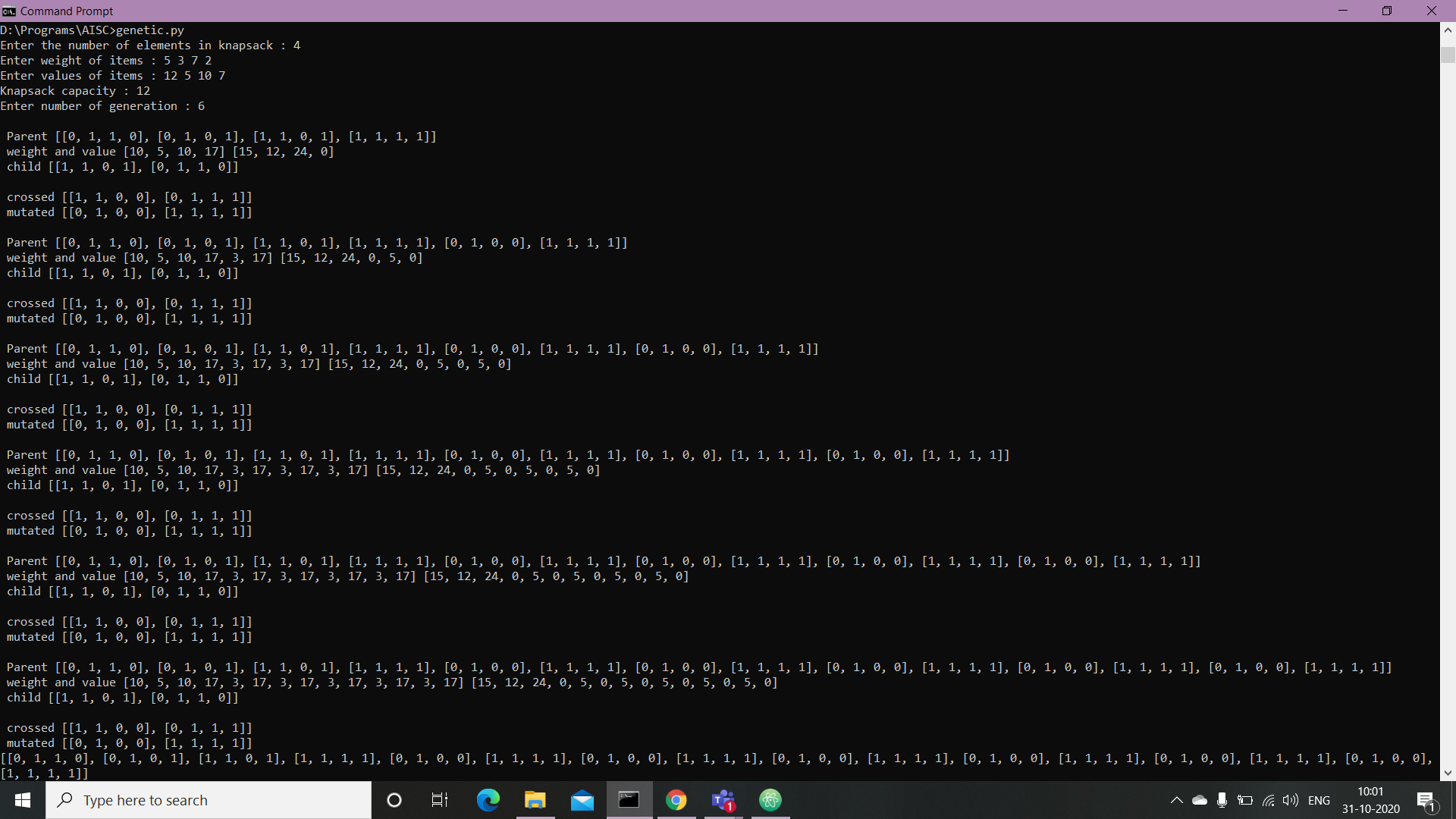
for i in range(generation):

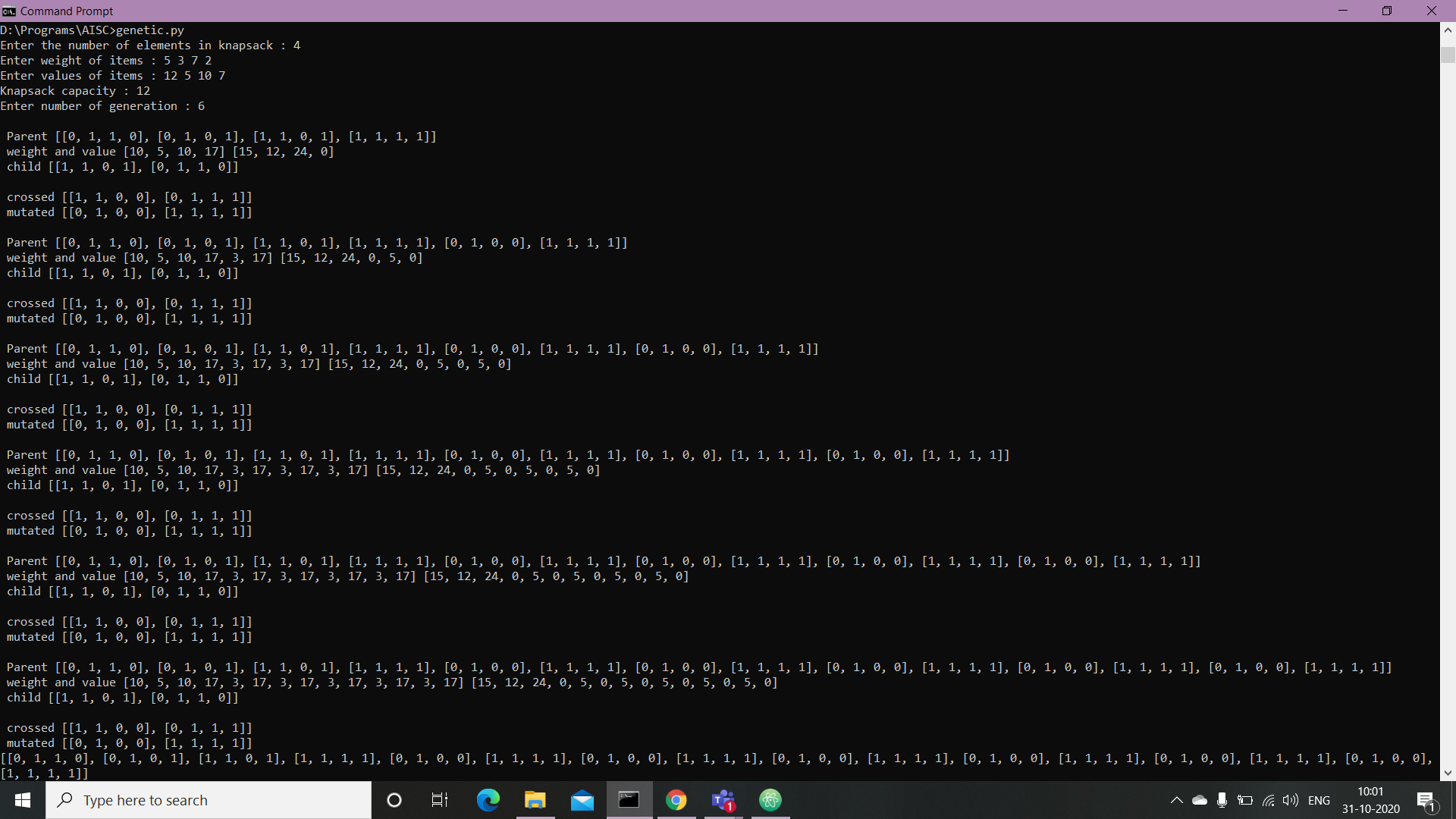
fitness(n)

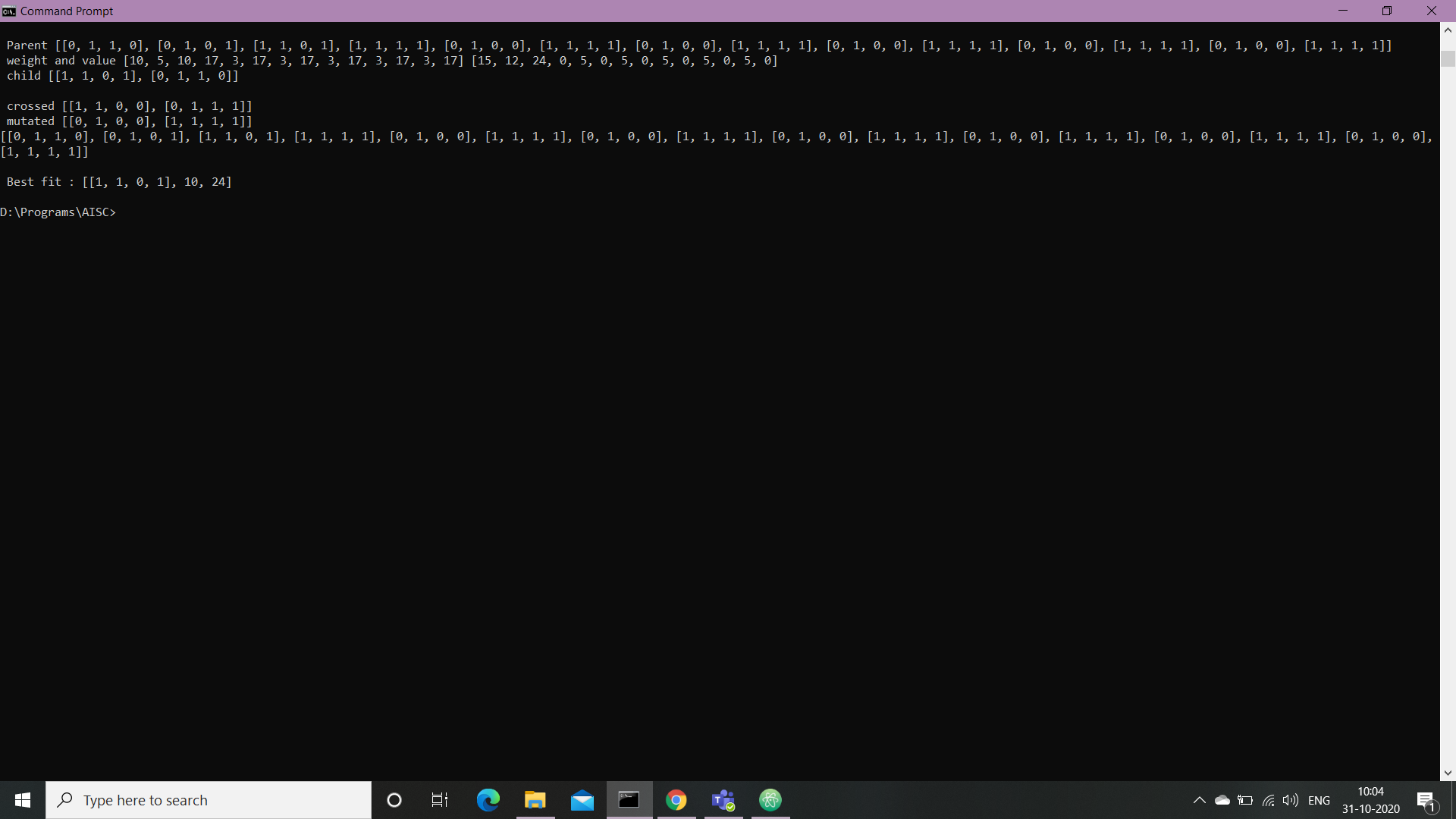
print(parent)

print("\n Best fit :", best)

**Output :**

****

****

****

**Conclusion :** Hence, we have successfully designed Knapsack Problem using Genetic algorithm.