معصومه پاسبانی 99243022

تمرین کامپیوتری دوم

همانطور که در الگوریتم genetic میدانیم، باید ژنهای ورودی یا همان دو والد را با هم ترکیب کنیم و دو فرزند جدید تولید کنیم. که این همان روش بازترکیبی یا crossover است. پس در این متد، یک عدد رندوم تولید می کنیم تا دو والد را از ایندکس رندوم تولید شده، جدا کرده و با ترکیب آنها، دو فرزند جدید بسازیم.

```
def crossover(generated_num, parent1, parent2):
    rand = random.randint(0, 99)
    child1 = []
    child2 = []
    child1.append(None)
    child2.append(None)
    child1 += parent1[0:rand] + parent2[rand:101]
    child2 += parent2[0:rand] + parent1[rand:101]
    generated_num.append(child1)
    generated_num.append(child2)
    generated_num.remove(parent1)
    generated_num.remove(parent2)
    return generated_num
```

حاال به سراغ الگوریتم میرویم :

همانطور که میدانیم، روشهای مختلفی در این الگوریتم برای بازترکیب کردن ژنها وجود دارد. ساده ترین روش آن، انتخاب رندوم دو والد و ترکیب کردن آنها با هم است. اما راه بهتر، انتخاب یک ژن خوب و یک ژن بد است.

```
def high_rate(binary_num, input):
    nums_true = 0
    for item in input:
        if (item[0] > 0 and binary_num[item[0]] == 1) \
                or (item[1] > 0 and binary_num[item[1]] == 1) \
                or (item[2] > 0 and binary_num[item[2]] == 1) \
                or (item[0] < 0 and binary_num[-item[0]] == 0) \
                or (item[1] < 0 and binary_num[-item[1]] == 0) \
                or (item[2] < 0 and binary_num[-item[2]] == 0):
            nums_true += 1

    return nums_true
```

ژنهای ورودی را با تابع sort، از کوچک به بزرگ مرتب میکنیم. سپس نصف ژنهای ورودی را به عنوان "ژن خوب" و نصف دیگر را به عنوان "ژن بد" در نظر میگیریم تا ترکیب این دو، ژنهای بهتری را بدهد. از بین این ژنها، یکی از ژن خوب و یکی از ژن بد به صورت رندوم انتخاب میکنیم تا روی این دو crossover را انجام دهیم. این عملیات را چندین بار تکرار می کنیم.

```
def genetic(random_numbers, input):
    for i in range(1000):
        random_numbers = map(lambda tp: tp[1], sort(random_numbers, input))
        random_numbers = list(random_numbers)
        bad_genes = random_numbers[0:10]
        good_genes = random_numbers[40:50]
        good_rand = random.randint(0, 9)
        bad_rand = random.randint(0, 9)
        random_numbers = crossover(random_numbers, good_genes[good_rand], bad_genes[bad_rand])
    return random_numbers
```

حاال به تعداد دلخواهی رشتههای باینری 100 بیتی تولید میکنیم که همان جمعیت اولیه هستند. این مقادیر به کمک تابعgenetic ، جمعیت جدید را تولید می کنیم.