松鼠代码实验报告：

分析：

相较于普通的单目标的松鼠算法，多目标的松鼠算法主要是将单纯的比较一个函数的值改为了比较帕累托等级和距离值，从单个的最优解和3个次优解，改为了5个最优解和9个次优解（这里的数字可以改动，也许经过改动后效果会更加的明显），，也正因为这个我们将NSGA中的快速非支配排序与距离值计算的方法引入，利用这两个算法得到我们需要的帕累托等级值和距离值。

1. 快速非支配排序：与NSGA中的快速非支配排序的思想相同，就是对整个种群中的所有地个体进行一次遍历，得到他们的支配个体数组和被支配次数，然后再对被支配次数为0的个体进行遍历（也就是帕累托等级为1的个体），将他们支配个体数组中的所有个体被支配次数减1（相当于在整个种群中删去帕累托等级为1的个体），然后在得到新种群中帕累托等级为1的个体（同时标记这些个体的帕累托等级为2），循环遍历上述操作，直到所有个体都分级
2. 距离值的计算：距离值的计算同NSGA 对种群的一个函数进行升序排序，同时最大值和最小值设为无穷大（就相当于我们用偏好于边界值，这样的话我们就会向边界发散），然后对于其他值，距离值计算：

*distance*[i]=(*FuncVal*[i+1][0]-*FuncVal*[i-1][0])/(*FuncVal*[*N*-1][0]-*FuncVal*[0][0]);

第二个函数在原本距离值的基础上进行相同操作。

实现注意点：

1. 因为代码是在数组的基础上实现的，所以在具体的操作中应该注意到数组也要随着排序的变化而发生变化
2. ZDT1的函数计算方式

实验结果：

X y

0.982194473899958 0.06180523810365847 0

0.975647546667674 0.14105693218989956 1

0.758883606069131 0.2556540748192761 2

0.9715348783914177 0.20440963090206182 3

0.531160738306665 0.484706573335647 4

8.260507289650776E-4 1.4087810010981907 5

0.40736136681586166 0.535834215399515 6

0.27497071861376715 0.665455703630794 7

0.14741415032358607 0.920612272100098 8

0.31482741139163545 0.6478507644511071 9

0.005200743647814393 1.1526810075140876 10

0.4061626361471724 0.6305219450270124 11

0.14545899508582 0.9440653635587082 12

0.7846896071636816 0.2299520612638579 13

0.21832429798718306 0.7360934447976931 14

0.06949396088541107 1.0008581341975378 15

0.24991615735537942 0.7258918079008634 16

0.11497229986828303 0.9902377627347241 17

0.16940780797210847 0.8204149854066144 18

0.2146985593736364 0.7448883003002703 19

0.5900371768227264 0.36256715702127057 20

0.12191830377161472 0.9464395083530188 21

0.038274210256099436 1.0499953959501533 22

0.063642294607263 1.0471944786244407 23

0.06828061596819593 1.0433300394578473 24

0.04346872367277976 1.0484282205387576 25

