**异常检测算法:KNN检测算法**

**相关的文献链接（请按住Ctrl键不放，然后鼠标单击下面这段文字来查看对应的文档内容）**

[(49条消息) 【机器学习】一文详解异常检测算法：KNN\_knn异常检测\_Python数据挖掘的博客-CSDN博客](https://blog.csdn.net/qq_34160248/article/details/124527999)

**请先掌握本文前面的第一大点——图解**[KNN](https://so.csdn.net/so/search?q=KNN&spm=1001.2101.3001.7020)**异常检算法，后面的可看也可不看，毕竟本人也没看懂，哈哈……**

**根据该文中所提到的KNN检测算法思路可知我们总体思路就是先要计算每个样本点与其它样本点之间的距离，然后再将这些距离值进行从小到大升序操作，取前面K个样本距离的均值作为异常值再将其与阈值进行比较从而判断出这个数据是否为我们要检测出的那些异常点（异常数据）。**

**假如这里的k=3，可以先用殴式距离计算出每个点与其它点之间的距离，然后将这些距离从小到大排序（升序）后，取前3个距离值的均值作为最后的异常值，这里就以上面链接中文档中的数据为例**

**7 7 9 3**

5 4 5 6

8 6 9 3

9 9 7 7

5 5 5 5

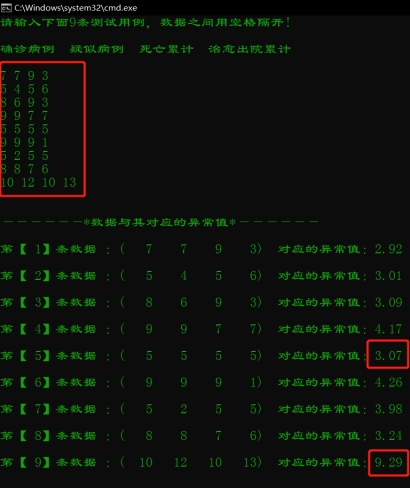
9 9 9 1

5 2 5 5

8 8 7 6

10 12 10 13

**咳咳，由于本人对python的学习不是了解的很多，所以还是用C++来实现这个过程较妥当。参考代码在后面，由本人亲自从头写到尾，利用C++面向对象的特性完成，过程简单易懂，如果有什么不对的地方，编者水平有限，还请凉解。我的大体思路：首先用创建的Point类来存储每个数据点，然后再用Distance类来计算该点与其它点之间的距离，然后再配合vector容器将其存储并进行升序，最后取前面三个距离的均值作为异常值存入对应的对象中，之后再根据每个对象的异常值与阈值比较加上另外两个条件来判断它是否为异常数据，至少这里说的另外两个条件后面有提到，由于本人的数学知识有限，不知道如何去确定这个阈值，网上的资料本人太笨也没看懂，但最后我还是根据它们最后的异常值规律来设置阈值。首先我们先验证我的这个程序对那个KNN文档中的示例是否正确，如图所示：**





**可以看出于是与文档中的结果是一致的，在这里我们可以看出假如设定阈值为5，再用异常值与阈值比较，异常值若比阈值要大则是异常数据，否则就不是异常数据，于是就可以得出异常数据是第9条数据。**

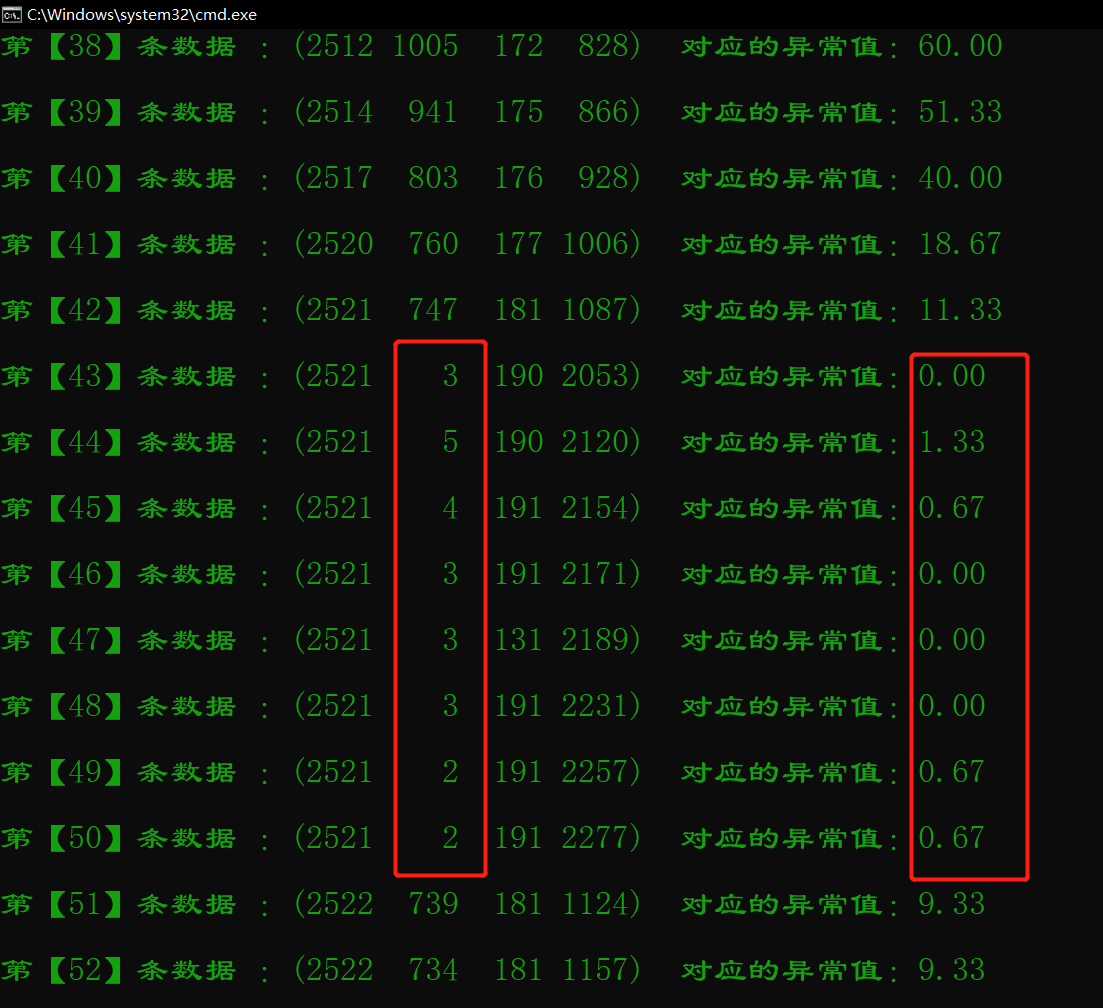
**于是乎进行对原来那个流感数据进行分析,发现可以找出如下几种是异常数据的情况：**

**第一：如果死亡累计和治愈出院累计的数据比前一些数中最大的那个数要小，说明它是异常数据，因为累计的话字面意思就是累加，也就是只会增加，不会减少，于是这两个地方可以用这种方法来检测它是否为异常的数据。**

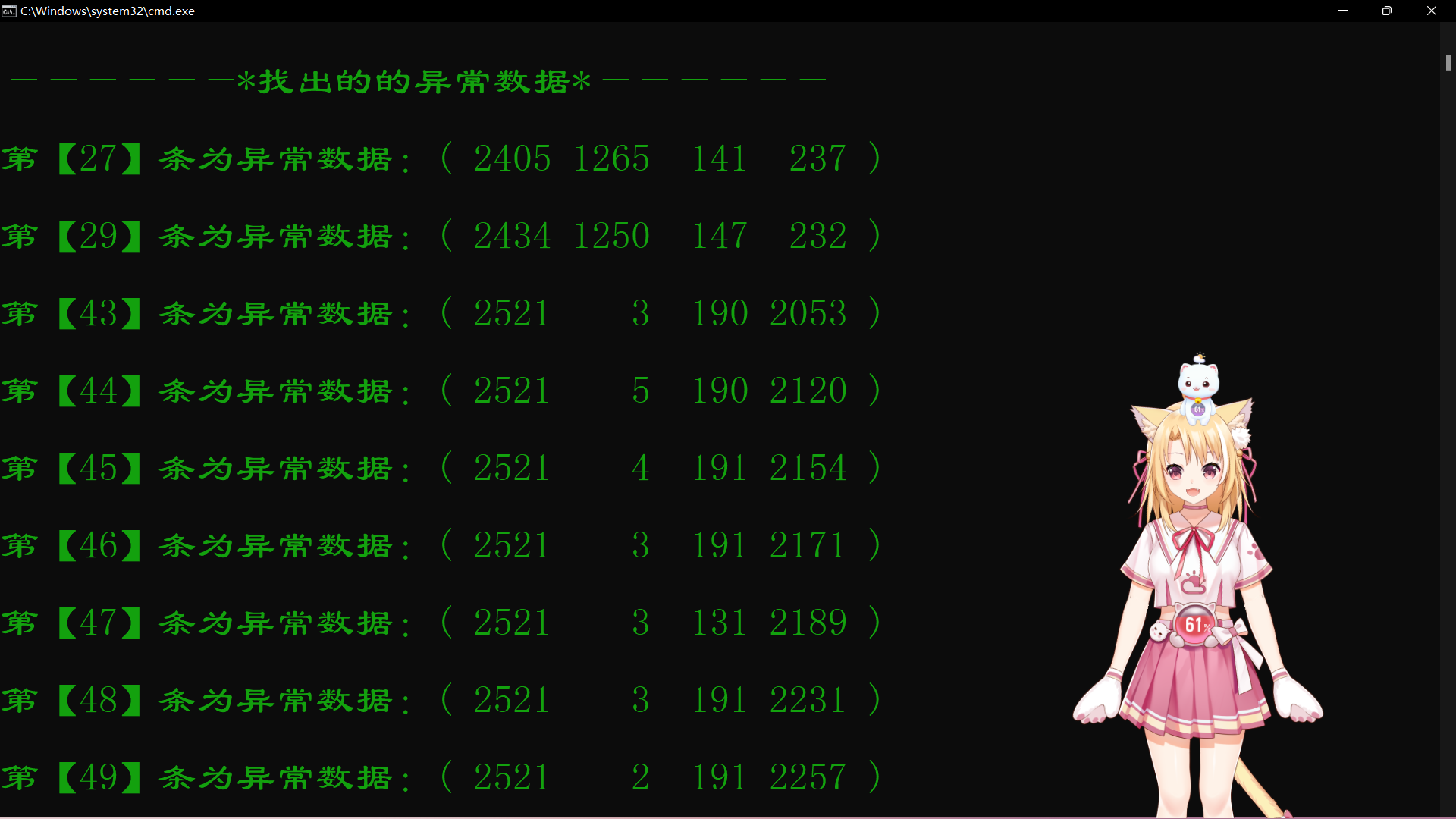
**例如：这里面的第28行和第30行明显治愈出院累计出现倒退，所以是属于异常的数据。**

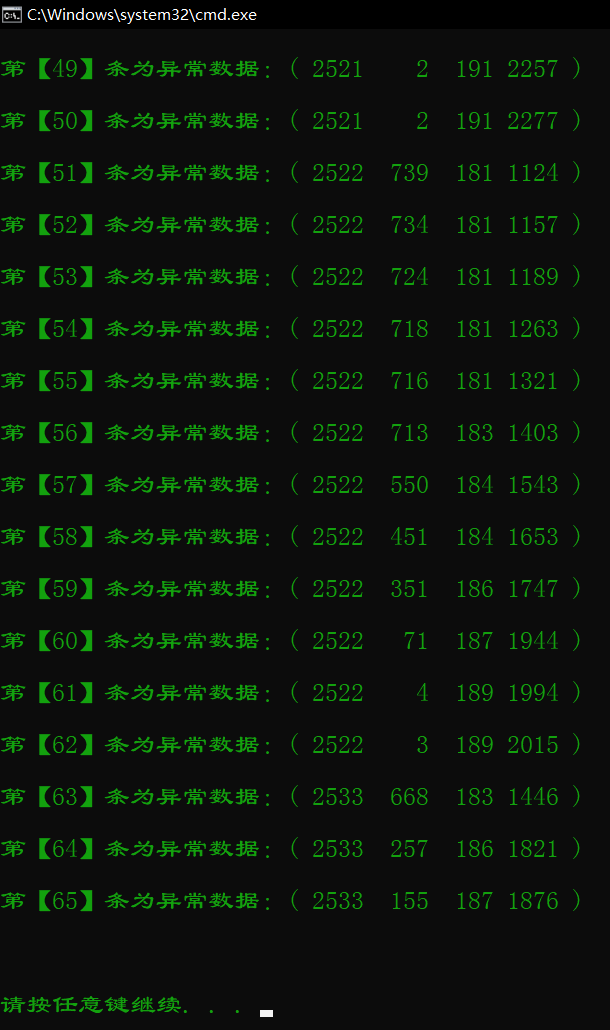


**第二：由于确诊病例会受到死亡人数和治愈出院人数的增多而减少，通过观察文档中的确诊数据还是比较正常的，后面的值可能是由于到了疫情稳定的情况数据变得平稳，但后面的几天肯定会有所下滑。而最有异常的情况就是疑似病例这列数据了，此时我利用KNN算法单独对它来计算它的异常值，发现结果如图所示：**



**可以看到异常数据所对应的异常值很小，几乎在[0,2)之间，于是我将阈值设置为2，假设异常值在[0,阈值)之间的就是异常数据，尽管与文中说的有点相反，但是头笨请凉解，最后通过这两个条件进行数据的检测，最后跑出来的代码如图所示：**

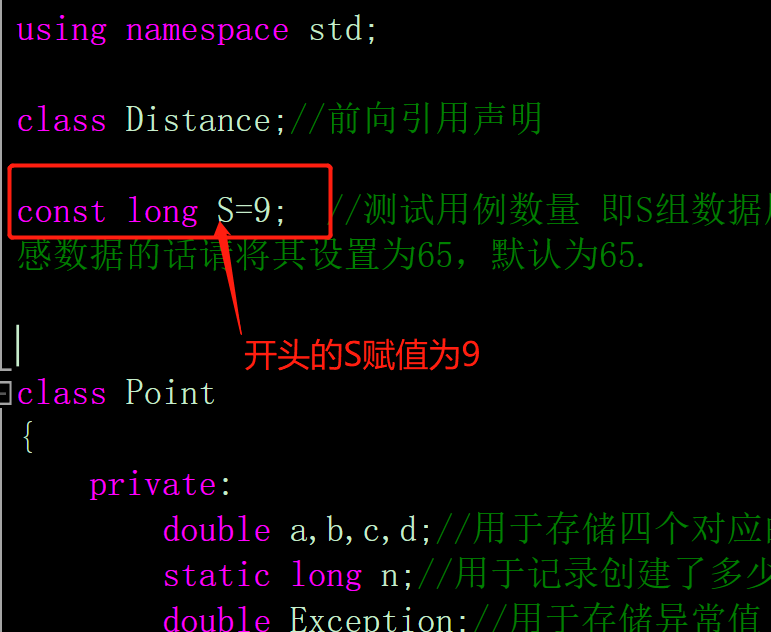


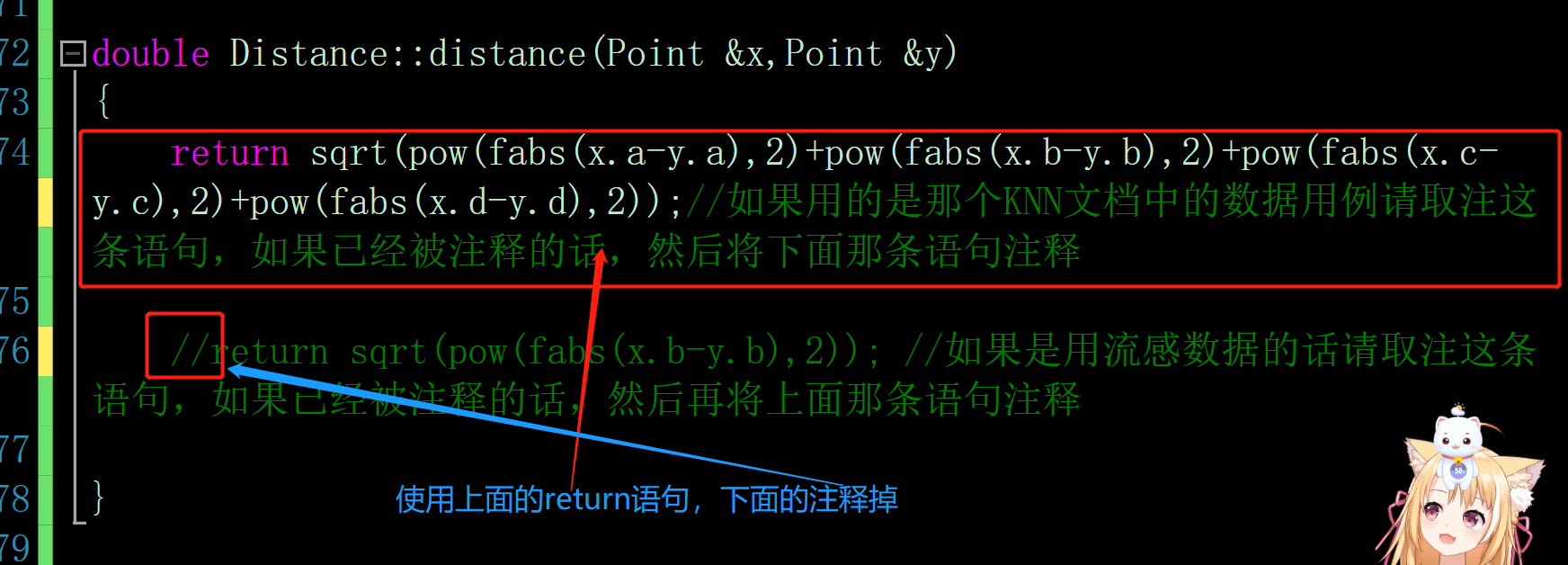


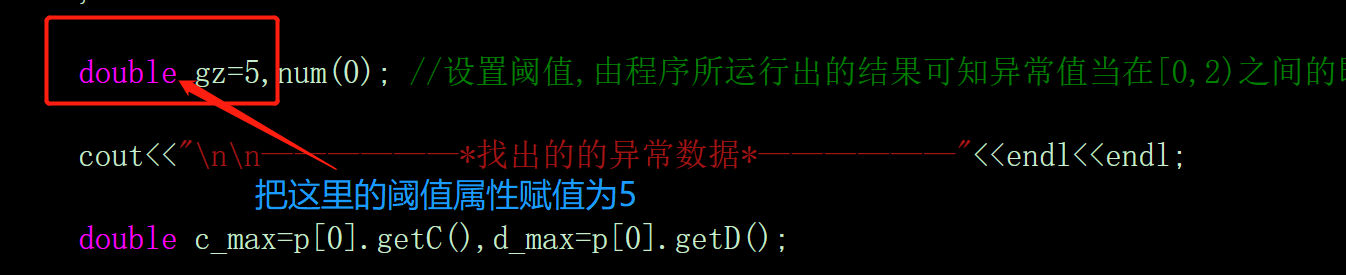
**即最后的这些点就是我们要排除掉的异常数据了。**

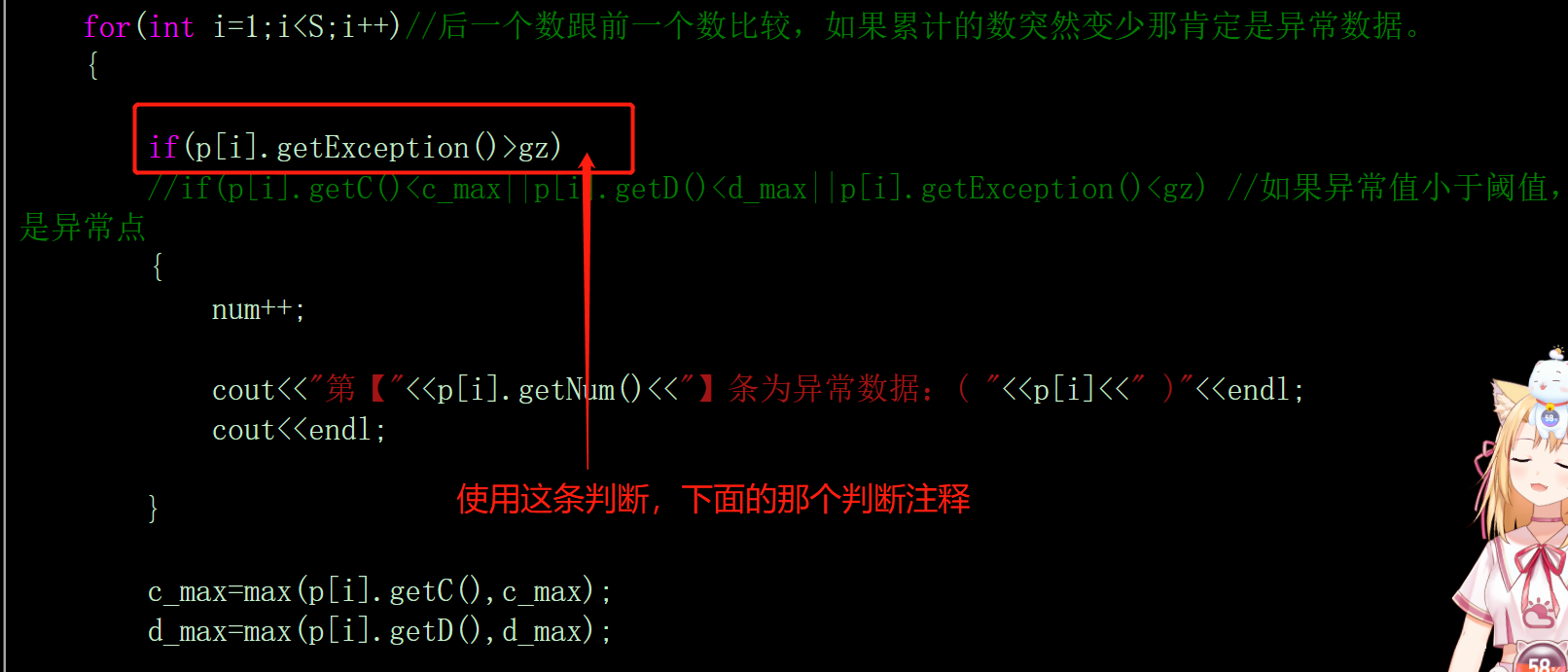
**取注就是把前面的“//”给去掉，注释就是在前面加上“//”,即单行注释：**

**使用后面的代码前请注意参数的调整，如果是用KNN文档中的数据的话，有四个地方需要保证：**

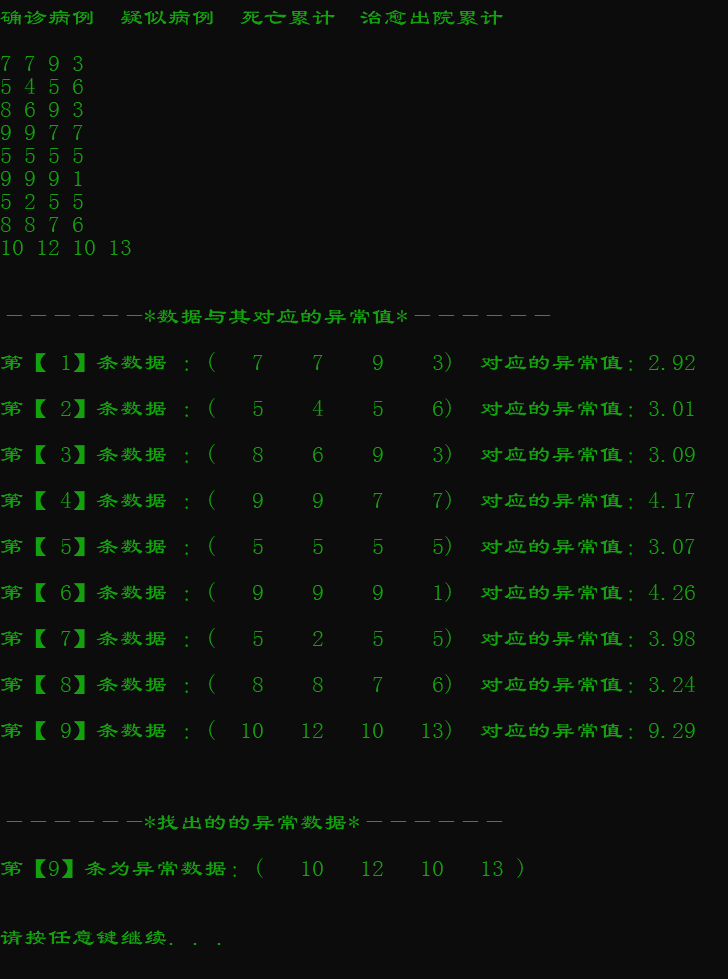




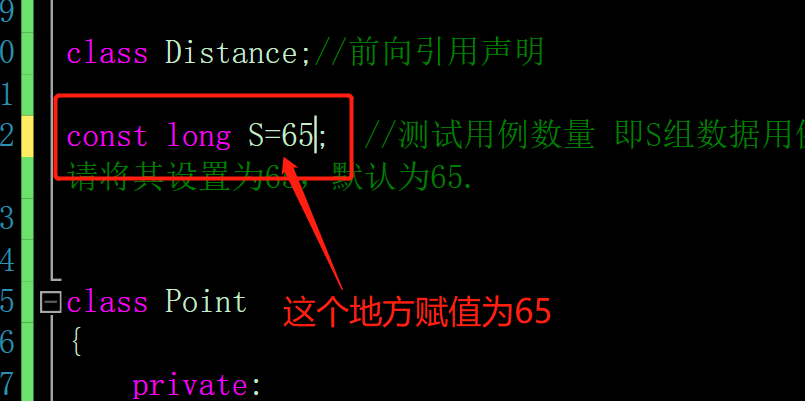


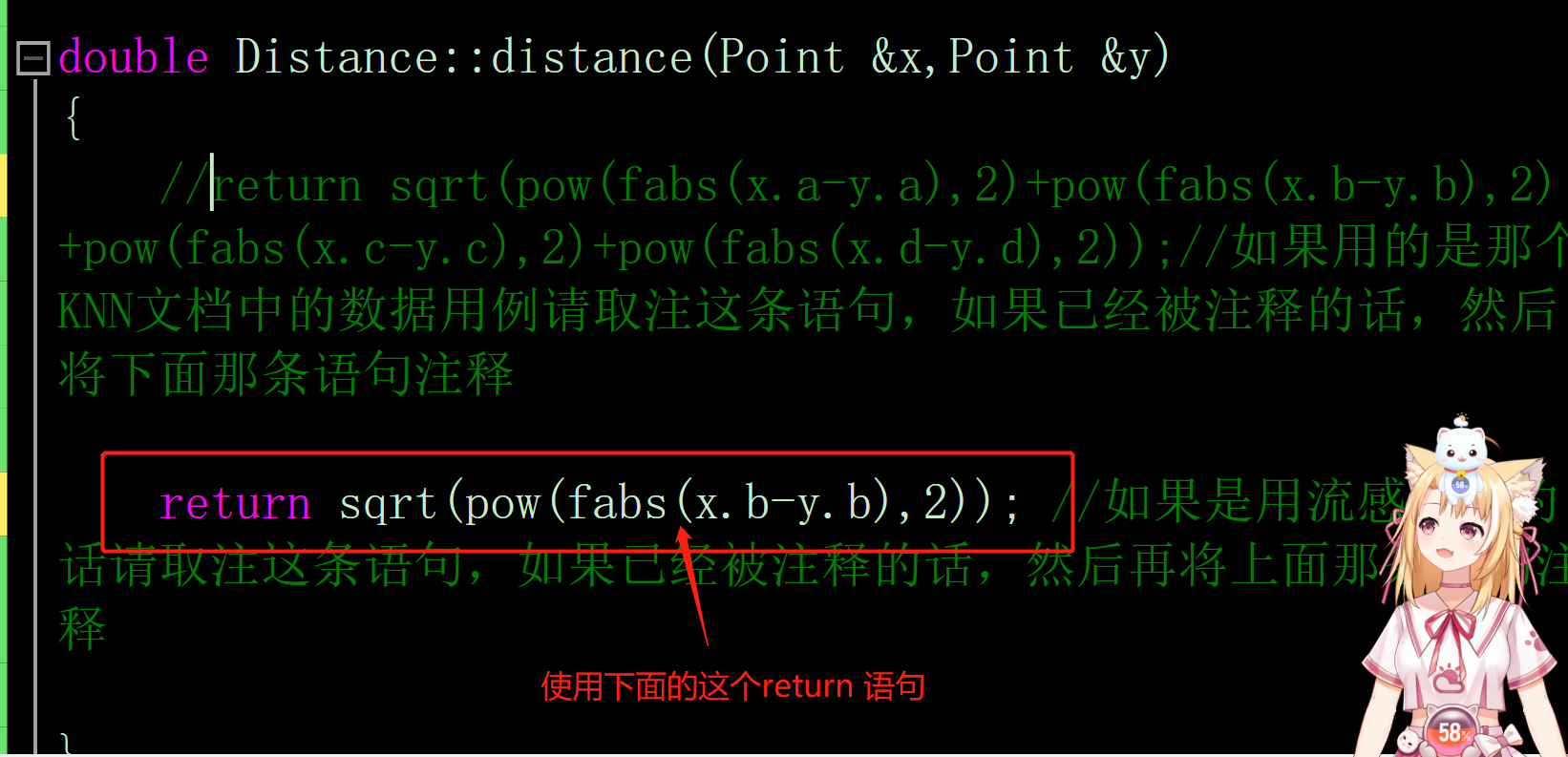


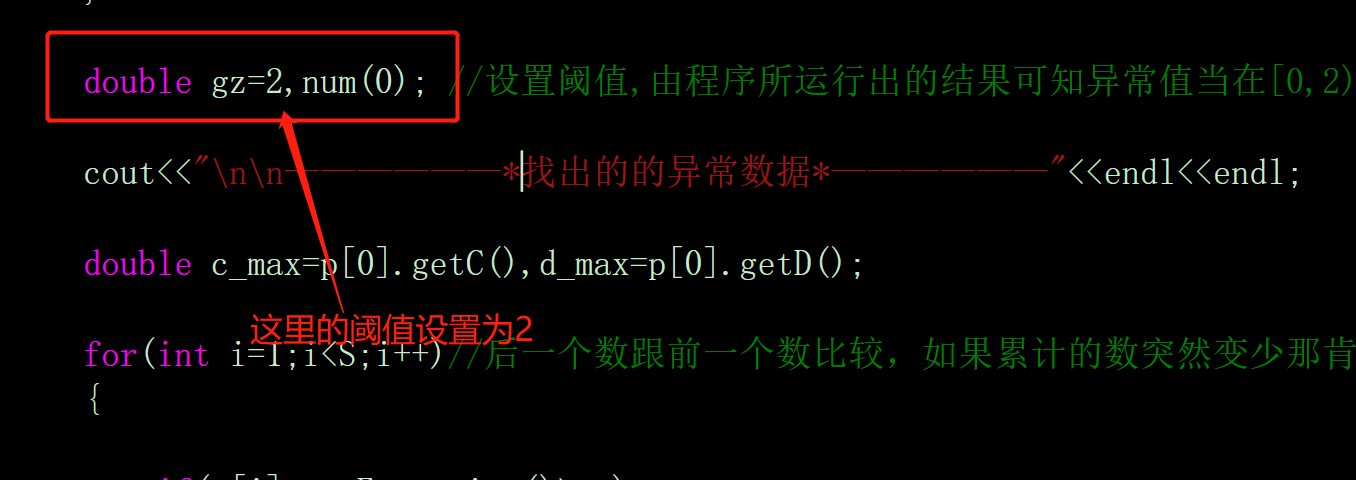
**运行效果：**

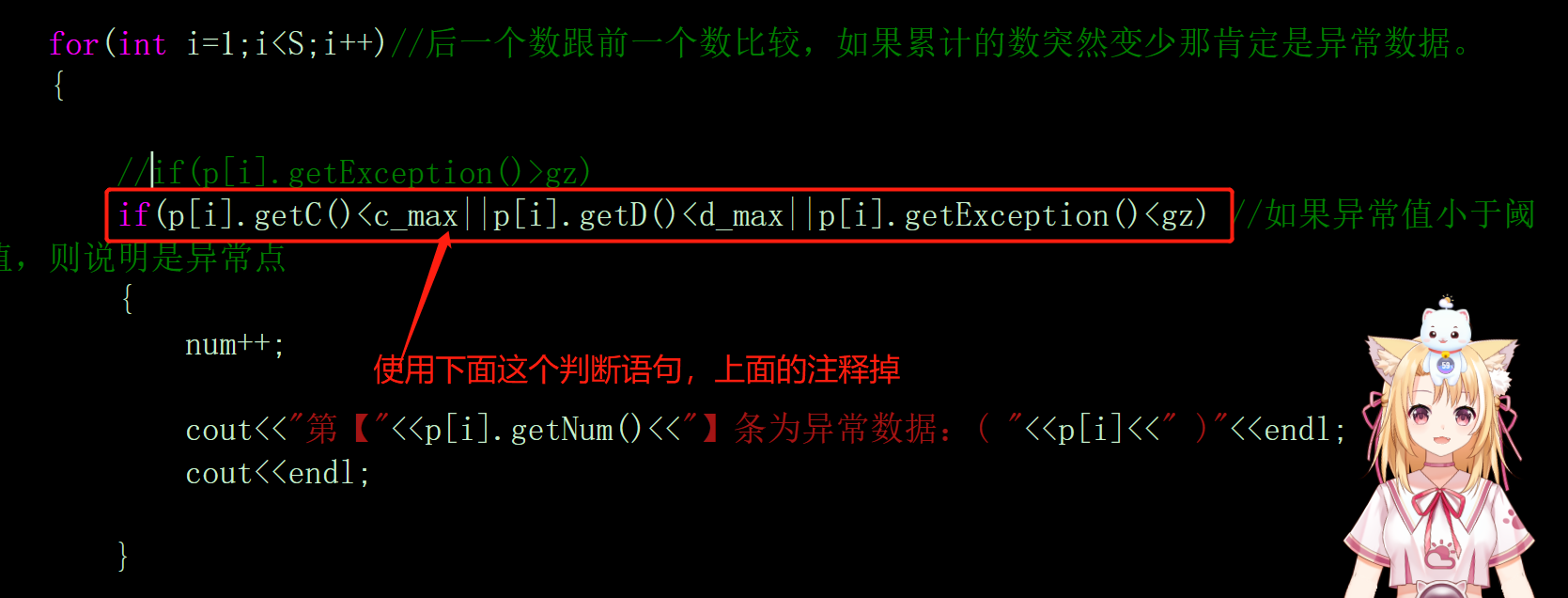


**同理，如果是使用的流感数据的话同样有四个要注意的地方：**

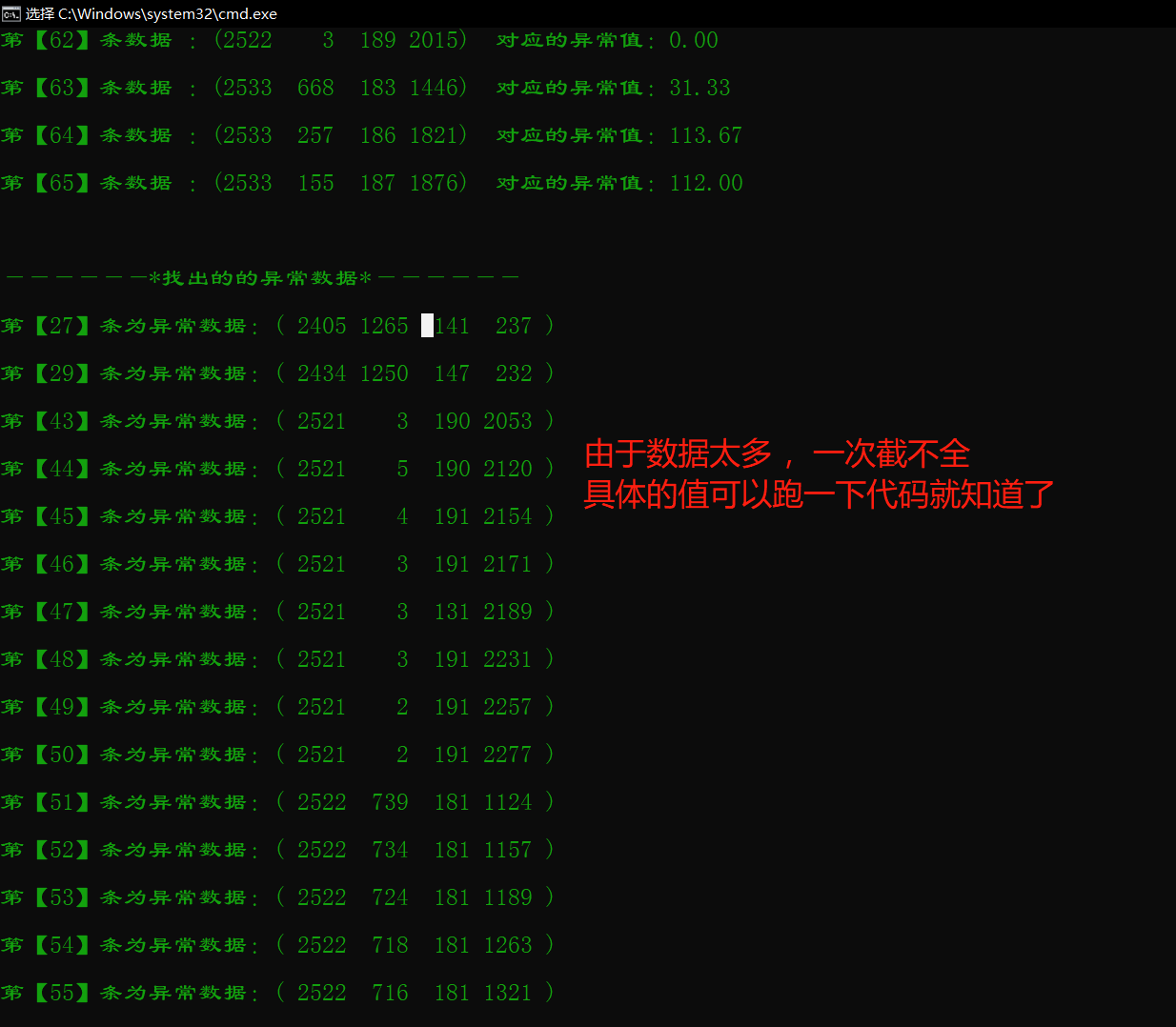








**运行效果：**



**代码实现如下：**

#include<iostream>

#include<math.h>

#include<algorithm>

#include<vector>

#include<stdlib.h>

#include<iomanip>

using namespace std;

class Distance;//前向引用声明

const long S=65; //测试用例数量 即S组数据用例 如果是KNN文档中的数据请将其设置为 9 ，如果是流感数据的话请将其设置为65，默认为65.

class Point

{

private:

double a,b,c,d;//用于存储四个对应的数据

static long n;//用于记录创建了多少个对象

double Exception;//用于存储异常值 Exception

long num;//记录数

public:

friend class Distance;//设置为友元类

Point(){}

Point(double x,double y,double s,double n):a(x),b(y),c(s),d(n){n++;}

friend istream& operator>> (istream &in ,Point &x);

friend ostream& operator<< (ostream &in ,Point &x);

static double size(){return n;}

double getC(){return c;}

double getD(){return d;}

double getException() {return Exception;}

double setException(double x) { Exception=x; return x;}

long getNum(){return num;}

long setNum(long n){num=n; return num;}

};

long Point::n=0;

istream& operator>> (istream &in ,Point & x)

{

Point::n++;

in>>x.a>>x.b>>x.c>>x.d;

return in;

}

ostream& operator<< (ostream &ou ,Point & x)

{

ou<<setiosflags(ios::right);//设置输出格式

return ou;

}

class Distance

{

private:

Point x,y;

public:

Distance(){}

Distance(Point &a,Point &b):x(a),y(b){}

double distance(Point&,Point &);//计算两点之间的距离

};

double Distance::distance(Point &x,Point &y)

{

//return sqrt(pow(fabs(x.a-y.a),2)+pow(fabs(x.b-y.b),2)+pow(fabs(x.c-y.c),2)+pow(fabs(x.d-y.d),2));//如果用的是那个KNN文档中的数据用例请取注这条语句，如果已经被注释的话，然后将下面那条语句注释

return sqrt(pow(fabs(x.b-y.b),2)); //如果是用流感数据的话请取注这条语句，如果已经被注释的话，然后再将上面那条语句注释

}

void start()

{

cout<<"确诊病例 疑似病例 死亡累计 治愈出院累计\n\n";

}

void findException()

{

Point p[S];

long n=0; //用于配合goto语句进行计算每个点的异常值

start();

for(int i=0;i<S;i++)

cin>>p[i];//已进行输入符的重载

vector<double> mean;//用于计算平均值 mean

Distance ds[9];//用于存每个点与p[n]之间的距离

go\_on:

//计算每个点与p[n]之间的距离存入容器mean中，自身与自身不进行计算

for(int i=0;i<S;i++)

if(i==n) continue;//自身与自身不进行计算

else mean.push\_back(ds[i].distance(p[i],p[n]));

sort(mean.begin(),mean.end());//将所有的距离值进行升序

p[n++].setException((mean[0]+mean[1]+mean[2])/3.0); //记录异常值，假设knn算法用k=3的情况，存取到对应的对象中去

mean.clear(); //清除所有平均值，重新计算下一个点的异常值

if(n<=S-1) goto go\_on; //如果每个点还没遍历完成则继续循环。

cout<<endl<<endl;

cout<<"——————\*数据与其对应的异常值\*——————\n"<<endl;

for(int i=0;i<S;i++)//遍历所有的异常值

{

printf("第【%2d】条数据 ：",p[i].setNum(i+1));

cout<<"("<<p[i]<<")";

printf("对应的异常值：%.2lf\n\n",p[i].getException());

}

double gz=2,num(0); //设置阈值,由程序所运行出的结果可知异常值当在[0,2)之间的即为异常数据

double c\_max=p[0].getC(),d\_max=p[0].getD();

for(int i=1;i<S;i++)//后一个数跟前一个数比较，如果累计的数突然变少那肯定是异常数据。

{

//if(p[i].getException()>gz)

if(p[i].getC()<c\_max||p[i].getD()<d\_max||p[i].getException()<gz) //如果异常值小于阈值，则说明是异常点

{

num++;

cout<<endl;

}

c\_max=max(p[i].getC(),c\_max);

d\_max=max(p[i].getD(),d\_max);

}

if(!num) cout<<"未找到异常点！\n";

cout<<endl<<endl;

system("pause");

}

int main()

{

findException();

return 0;

}

/\*测试用例

表中的流感数据：

339 402 18 13

482 610 25 23

588 669 28 32

693 782 35 55

774 863 39 65

877 953 42 73

988 1093 48 76

1114 1255 56 78

1199 1275 59 78

1347 1358 66 83

1440 1408 75 90

1553 1415 82 100

1636 1468 91 109

1741 1493 96 115

1803 1537 100 118

1897 1510 103 121

1960 1523 107 134

2049 1514 110 141

2136 1486 112 152

2177 1425 114 168

2227 1397 116 175

2265 1411 120 186

2304 1378 129 208

2347 1338 134 244

2370 1308 139 252

2388 1317 140 257

2405 1265 141 237

2420 1250 145 307

2434 1250 147 232

2437 1249 150 349

2444 1225 154 395

2444 1221 156 447

2456 1205 158 528

2465 1179 160 582

2490 1134 163 667

2499 1105 167 704

2504 1069 168 747

2512 1005 172 828

2514 941 175 866

2517 803 176 928

2520 760 177 1006

2521 747 181 1087

2521 3 190 2053

2521 5 190 2120

2521 4 191 2154

2521 3 191 2171

2521 3 131 2189

2521 3 191 2231

2521 2 191 2257

2521 2 191 2277

2522 739 181 1124

2522 734 181 1157

2522 724 181 1189

2522 718 181 1263

2522 716 181 1321

2522 713 183 1403

2522 550 184 1543

2522 451 184 1653

2522 351 186 1747

2522 71 187 1944

2522 4 189 1994

2522 3 189 2015

2533 668 183 1446

2533 257 186 1821

2533 155 187 1876

KNN文档中的测试数据：

7 7 9 3

5 4 5 6

8 6 9 3

9 9 7 7

5 5 5 5

9 9 9 1

5 2 5 5

8 8 7 6

10 12 10 13

\*/

**好的，大一小白的思路就写到这了，感谢您的观看…………**