

中山大学数据科学与计算机学院本科生实验报告

(2016 学年春季学期)

课程名称: Algorithm design

任课教师: 张子臻

年级	1501	专业(方向)	移动信息工程
学号	15352015	姓名	曹广杰
电话	13727022190	Email	1553118845@qq.com
开始日期	2017/3/25	完成日期	2017/3/25

1. 实验题目



2. 实验目的

- 1) 给定某一个点, 笛卡尔坐标系表示。起点在原点, 按要求的操作计算给定时间内距离目的地的距离;
- 2) 操作方式如下: 给出次数 n , 每操作一次, 操作权限少一次, 用完即止。
操作有两种: 转向 和 前进 10 个单位。
- 3) 计算最后距目的地的距离;

3. 程序设计

储存输入数据以便后续比较

DFS 划定所有可到达区域

探索新的点即更新数值

详细描述如下:

1. 输入均为整数, 用整形类型存储即可。Min 处在百万之内, 不必使用 long long;
2. 用尽给出的所有次数, 不断更新距离目的地的值。
 - 1) 用尽给定的所有次数。即展示出所有的可能的情况, 每次新的情况出现都更新一次记录值。在展示所有情况的过程中, 表示的其实就是对于一切情况的讨论。因为我们需要找到能够到达的距离目的地最近的点, 而这一个过程简化而言就是方向与距离。方向受限, 距离取整, 这样的话, 即便是人工作图也免不了计算的困扰。在这种情况下, 对于输入的点坐标的讨论就得不偿

失了。所以此处，使用搜索结构是非常必要的。

笔者在这里使用 Dfs，实现对于所有情况的考虑与讨论，计划实现出所有的能够到达的点。继而
对每一个点都进行一次运算，用运算结果更新最小的数据记录值。;

- 2) 不断更新距离目的地的距离。这里由于输入的目的地值不一定是整数，或者即便是整数，
由于勾股定理的计算最后也会使得计算值是浮点数类型，这里使用双浮点数对记录值进行储存。
3. 在主函数内部，只需要调用一个搜索函数（这里是 Dfs），然后直接把更新后的数值输出即可（笔者
已经将更新记录值的操作放在 Dfs 内部了，每到达一个点都会更新一遍）。;

4. 程序运行与测试

```
2
24 5.0 5.0
0.502525
9 7.0 17.0
0.100505

-----
Process exited after 2.187 seconds with return value 0
请按任意键继续. . .
```

5. 实验总结与心得

总结如下：

- 1) 对距离关系的处理，使用距离的平方并在最后采取开方运算，减少运算时间节约运算资源；
- 2) 由于计算尺度产生的搜索必要性。正如之前所说，因为我们需要找到能够到达的距离目的地最近的点，而这一个过程简化而言就是方向与距离。方向受限，距离取整，这样的话，即便是人工作图也免不了计算的困扰。在这种情况下，对于输入的点坐标的讨论就得不偿失了。所以此处，使用搜索结构是非常必要的。很多时候，搜索的操作也是可以避免的，但是为了避免这种操作进行的简化效率并不高；
- 3) 更新数据的测试。数据更新的过程中，应该是每出现一个新的位置点，记录值就实现一次更新。之后进行次数的检查，是否韩哟偶权限进行新的位置点选取。那么，实现顺序就是：
到达新的位置点=>更新记录值=>检查权限=>使用权限，执行操作
那么，更新记录值理应在检查权限之后，因为即使不进行新一步的操作，记录值也是要更新的。但是这里如果这么实现，Sicily 会报错，很显然这是检查样例出了问题，是系统的错误。
- 4) Dfs 各个方位的遍历行为。因为每一个位置点都会有不同的方向选择，这里，使用顺时针设计与逆时针设计都是没有问题的。因为最后都要找到同一个最近的点，但是事实并不是，在 Sicily 的编译器中显示，顺时针的显示结果与逆时针的显示结果是不同的。因为……我不知道因为什么，Dfs 遍历对于每一种情况产生的概率应该是一致的。在这里只有逆时针设计才可以通过，可能是一个隐藏的 Bug；

望 Ta（师兄师姐们），及时改正。

作图：



smie15352015
where there is will, there is a way.
[Logout](#)

Home Problems Contests Courses Ranklist Submit Setting Status Discuss <->										
1000	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000+
First		Previous	1	Next	Last		Search: <input type="text"/>			
Status	ID	Title			Accepted		Submissions		Ratio	
	1707	Repair Depots			80		236		33.9%	
	1708	Crazy Circuits			4		9		44.44%	
✓	1709	PropBot			248		770		32.21%	

附录、提交文件清单

15352015-caogj-1709-v0;

1709.cpp;