

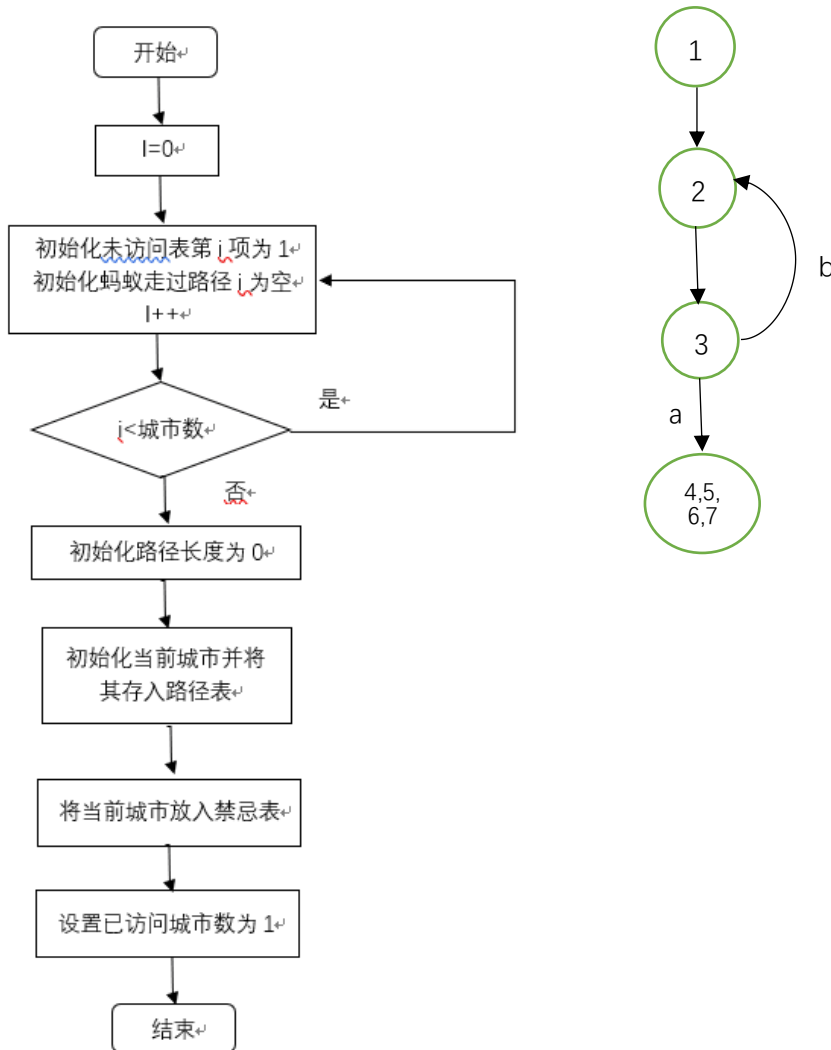
小火龙宅急便测试文档

在测试过程中，我们画出了路径规划的控制流图，同时，采用等价类划分法对整个程序进行测试，结果如下。

程序均使用 c 语言写成，经过测试，该系统完成了预定目标，能够很好地满足需求，虽然形式较为粗陋，有待改进，但是已经有了物流管理系统的雏形。

一， 路径规划部分（系统核心部分）的控制流图

一.初始化



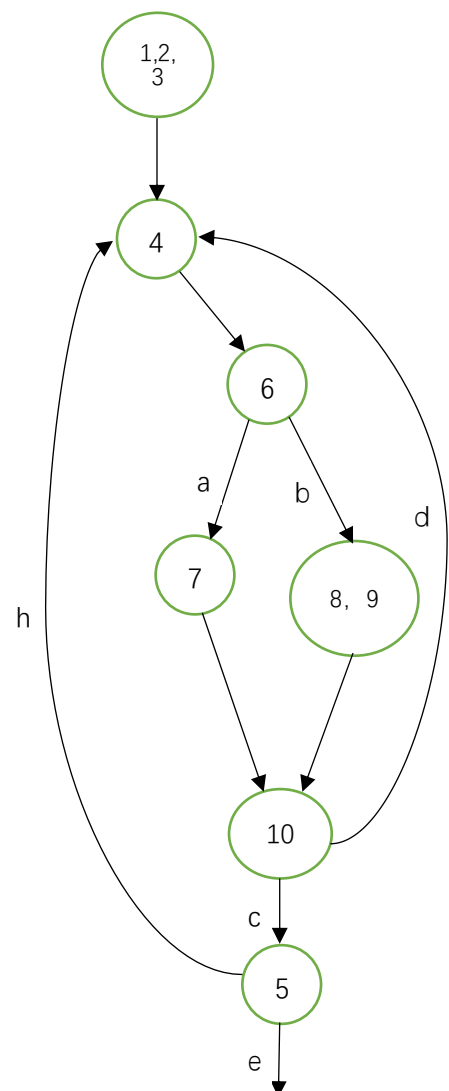
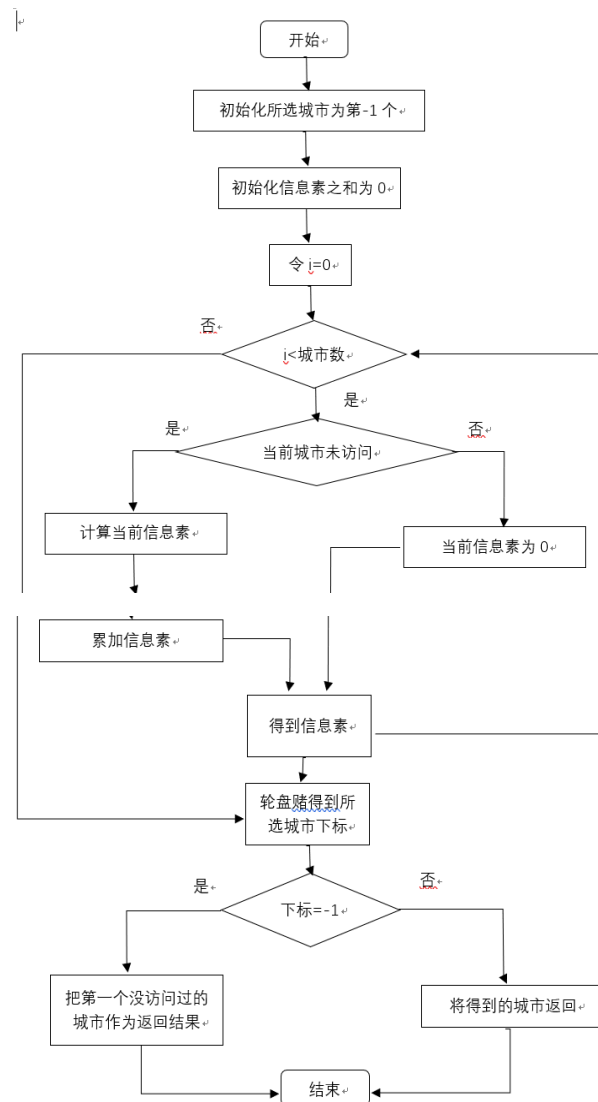
环路复杂度 $V(G)=E-N+2=P+1=4-4+2=1+1=2$

基本路径集:

Path1: 1->2->3->2->3->4,5,6,7

测试用例: 取城市数为 2

二.选择下一城市



环路复杂度 $V(G)=E-N+2=P+1=14-11+2=4+1=5$

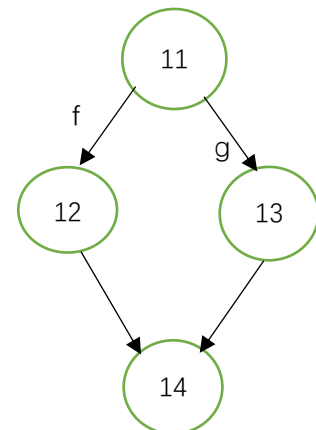
基本路径集:

Path1: 1,2,3->4->6->7->10->5->11->12->14

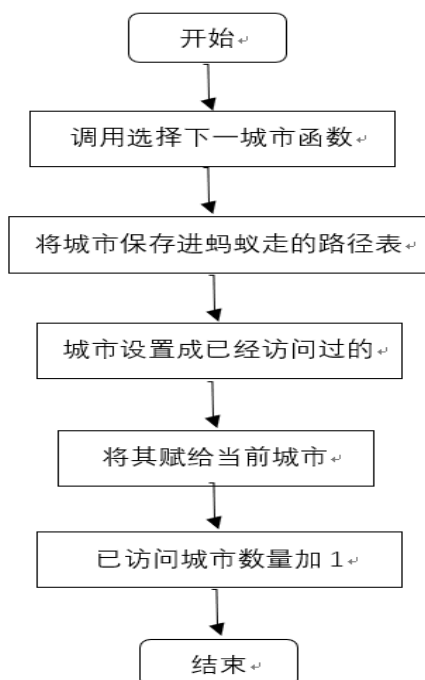
Path2: 1,2,3->4->6->8,9->10->4->6->7->10->5->11->13->14

测试用例: 1、取城市数为 1

2、取城市数为 2



三. 移动



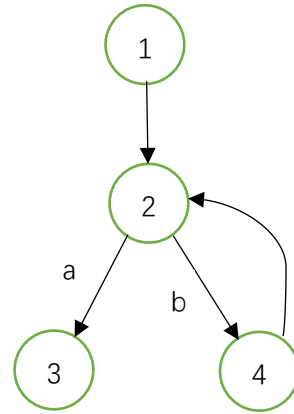
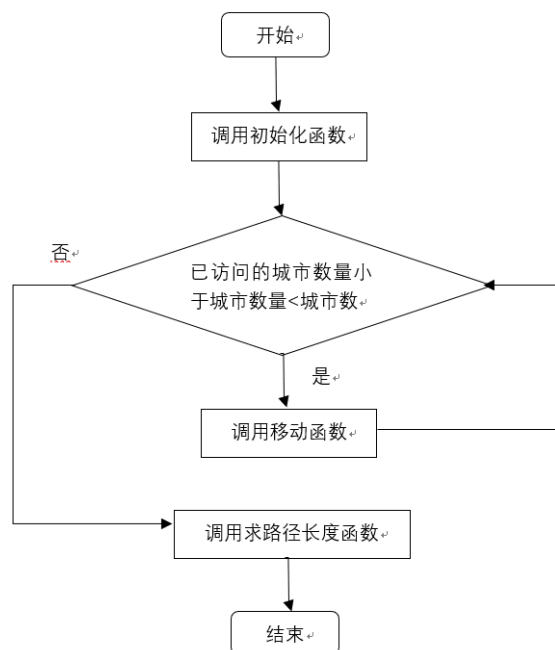
环路复杂度 $V(G)=E-N+2=P+1=4-5+2=0+1=1$

基本路径集:

Path1: 1->2->3->4->5

测试用例: 直接运行, 查看结果是否有改变: 已访问城市数量是否+1、路径表是否更新。

四. 搜索



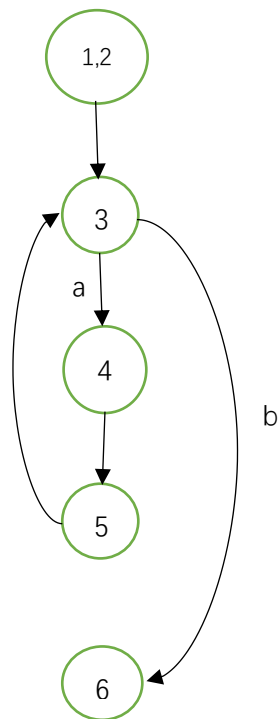
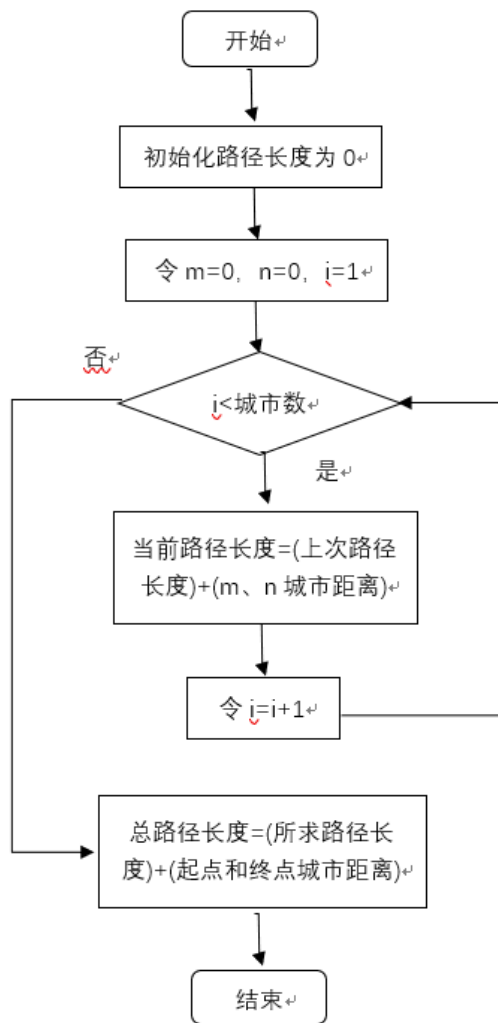
环路复杂度 $V(G) = E - N + 2 = P + 1 = 4 - 4 + 2 = 1 + 1 = 2$

基本路径集:

Path1: 1->2->4->2->3

测试用例: 取城市数=2, 直接运行, 查看相关数据量是否改变: 已访问城市数量是否+1、路径表是否更新、是否进入路径长度函数。

五. 计算路径长度



环路复杂度 $V(G)=E-N+2=P+1=6-5+2=2+1=3$

基本路径集:

Path1: 1,2->3->6

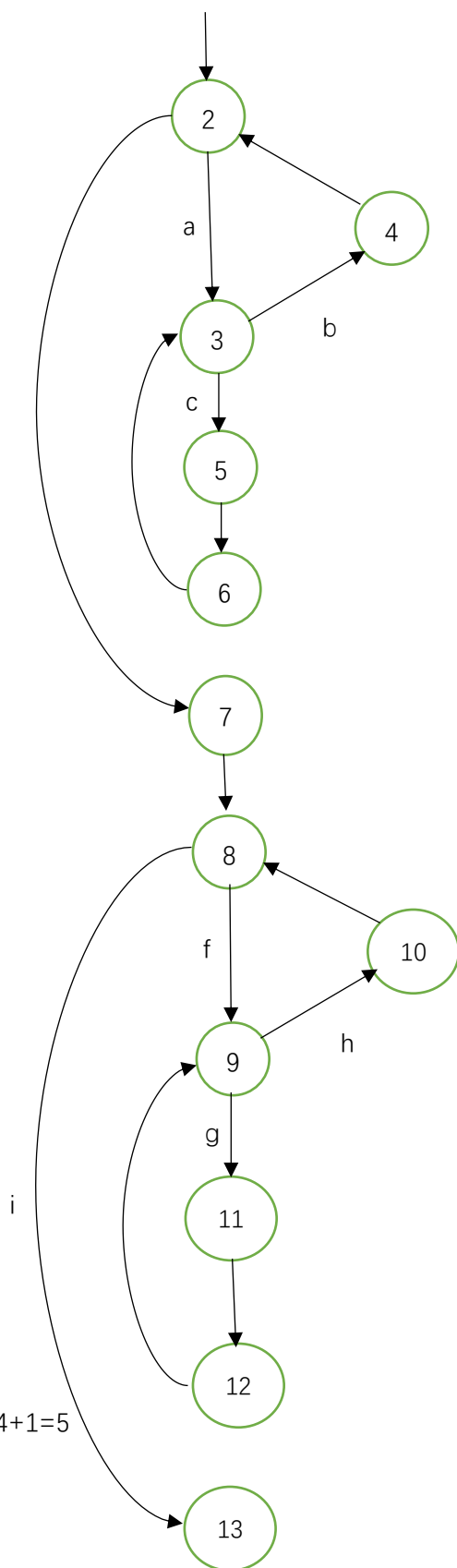
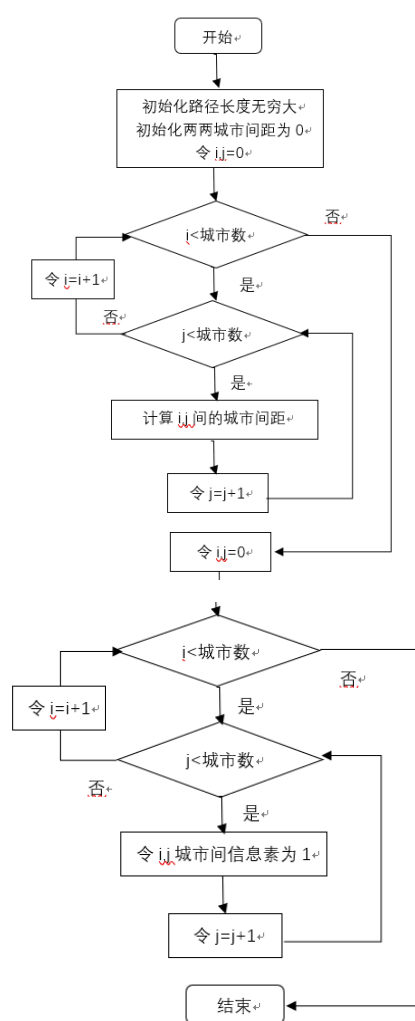
Path2: 1,2->3->4->5->6

测试用例: 1、取城市数为 1

2、取城市数为 2

六.初始化数据





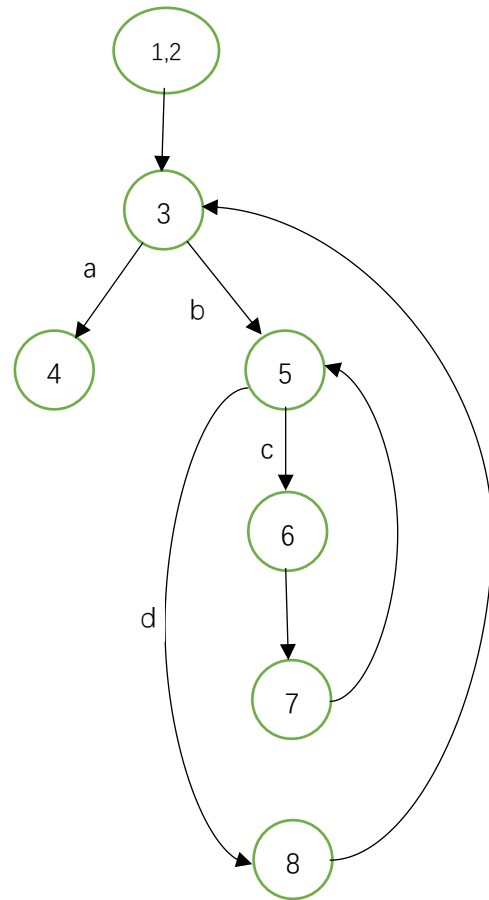
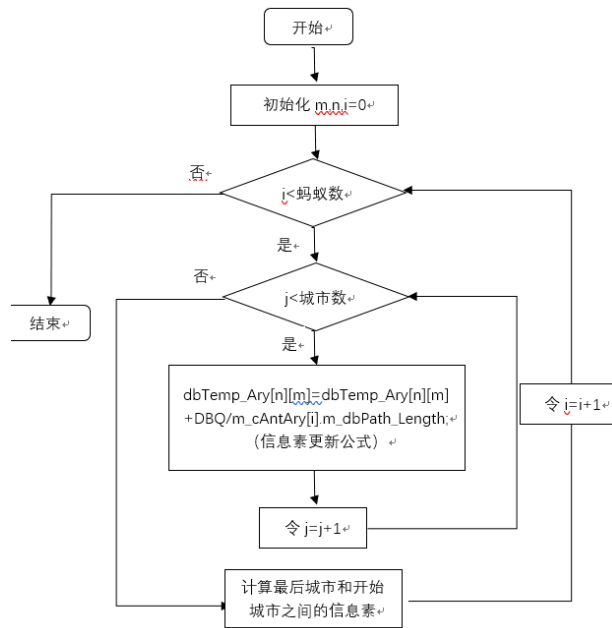
环路复杂度 $V(G)=E-N+2=P+1=16-13+2=4+1=5$

测试用例：取城市数为 1

基本路径集：

Path1: 1->2->3->5->6->3->4->2->7->8->9->11->12->9->10->8->13

七.更新信息素



环路复杂度 $V(G) = E - N + 2 = P + 1 = 8 - 7 + 2 = 2 + 1 = 3$

基本路径集:

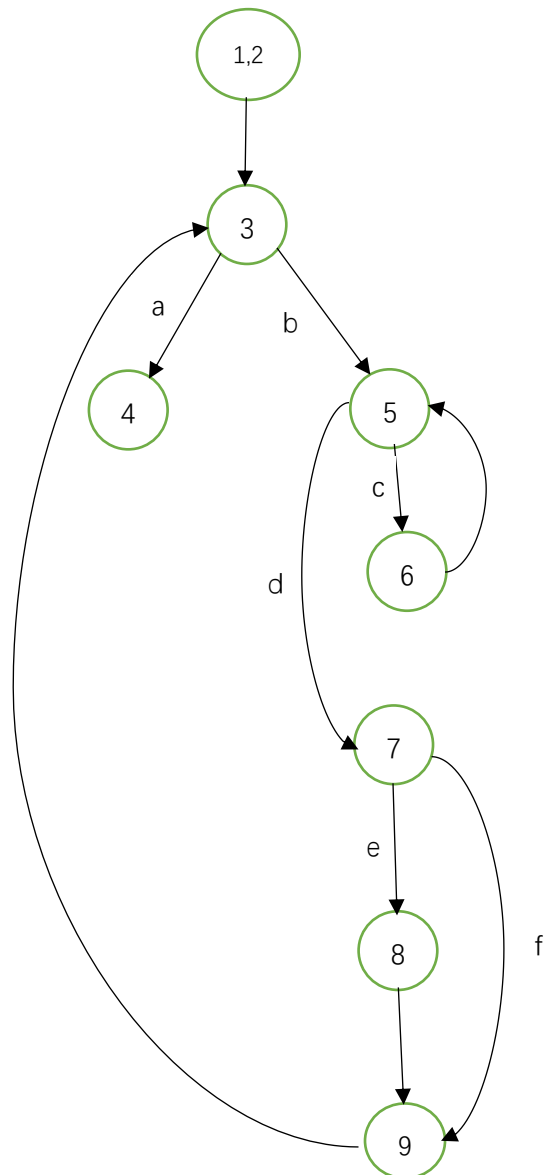
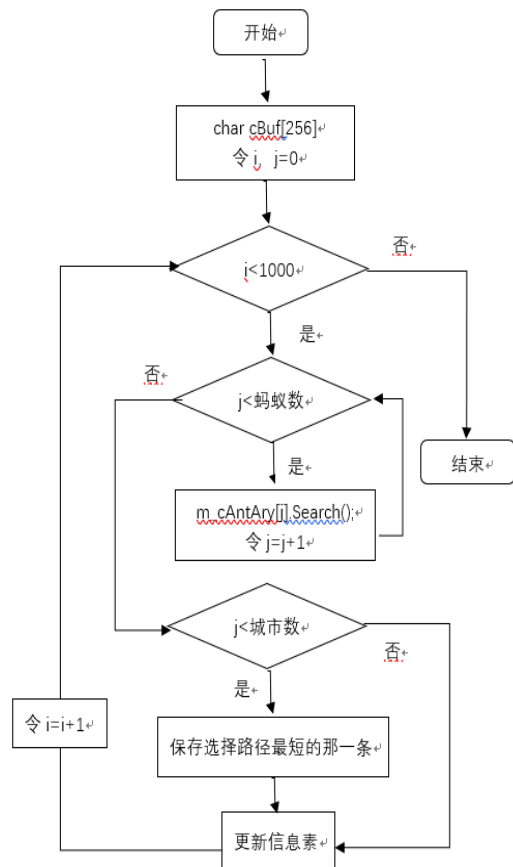
Path1: 1,2->3->4

Path2: 1,2->3->5->6->7->5->8->3->4

测试用例: 1、取蚂蚁数为 0, 城市数为 1

2、取蚂蚁数为 1, 城市数为 1

八.选择最优路径



环路复杂度 $V(G)=E-N+2=P+1=11-8+2=4+1=5$

基本路径集:

Path1: 1,2->3->4

Path2: 1,2->3->5->7->8->9->3->4

Path3: 1,2->3->5->6->5->7->9->3->4

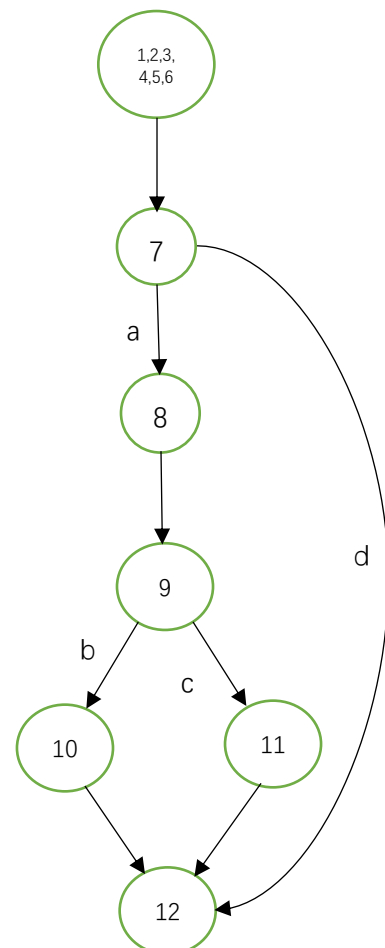
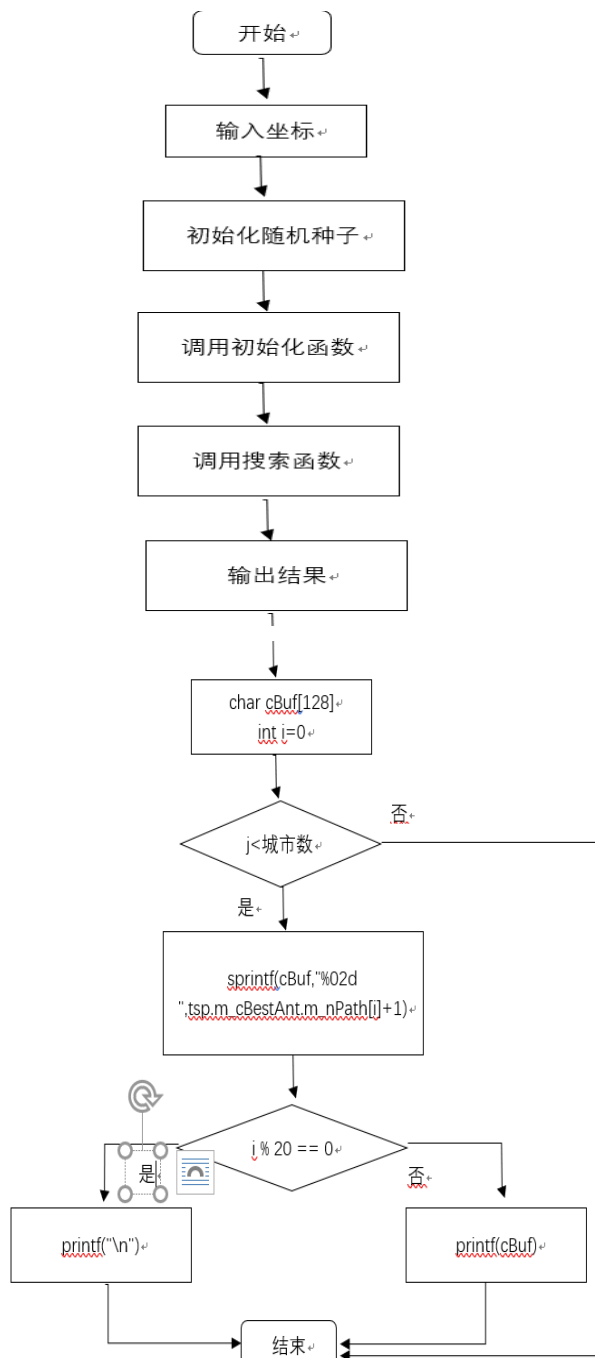
测试用例: 1、取迭代次数为 1, 蚂蚁数为 1, 城市数为 1

2、取迭代次数为 2, 蚂蚁数为 0, 城市数为 1

3、取迭代次数为 2, 蚂蚁数为 1, 城市数为 1

($i < 1000$ 是迭代次数, 测试时需进行修改)

九.主函数



环路复杂度 $V(G)=E-N+2=P+1=8-7+2=2+1=3$

基本路径集:

Path1: 1,2,3,4,5,6->7->12

Path2: 1,2,3,4,5,6->7->8->9->10->12

Path3: 1,2,3,4,5,6->7->8->9->11->12

测试用例: 1、取城市数为 1

2、取城市数为 2

3、取城市数为 6

最终测试结果:

在经过一系列的测试后，我们设置蚂蚁数为 34，迭代次数为 1000，城市数量取 5（实际中城市数量肯定大于 5，但为了测试方便，我们缩小了规划的城市范围），输入城市横坐标和纵坐标后得到如下结果：

```
请输入城市横坐标：
1
4
13
15
67
34
请输入城市纵坐标：
24
46
89
69
34
16

The best tour is :
01 02 04 03 05

Press any key to exit!
```

从结果可知，该代码可以较好地运算出最好的路径规划，但由于我们采用的蚁群算法，虽然算法正确率高，较其他算法更易于收敛于全局最优点，但算法本身具有一定的局限性，在大量的数据面前可能存在运行时间长的问题，这与我们快速得出路径规划的预想相违。同时，由于代码有 400 多行，算法在一定程度上是可以优化的，采取更为简单的程序表达。因为代码需要我们手动输入城市坐标，一定程度上也影响了我们的速度，所以优化时会考虑更为方便的输入方式（如读取地图选择城市编号）。

除此之外，我们在代码设计时有许多方面没有考虑到，所以测试时出现了不少问题，比如当输入不合要求时程序不能运行等，看来优化程序以及测试都是任重而道远啊。

二， 等价类划分法测试系统：

1.主函数：

输入条件	有效等价类	无效等价类
1<=X<=10	X = 1 到 10(1)	X<1(2),X>10 (3)

测试用例：

X=0, x=11, x=1

```
请选择数字进行操作
1. 建立库存信息
2. 显示所有商品的库存信息
3. 购物车
4. 结算
5. 查看销售记录
6. 查看缺货情况
7. 查看购买记录
8. 添加用户地址
9. 寻找最短配送路径
10. 退出
请选择对应数字
0

输入错误，请重新输入：
11

输入错误，请重新输入：
1

请依次输入货物信息：
-----
品名：
```

2 . 建立库存函数

输入条件	有效等价类	无效的等价类
文本或者数字	文本、数字 (1)	回车符 (2)

测试用例：

- (1) 输入文本和文字：成功建立库存函数。
- (2) 输入回车符：程序一直提醒输入相应信息，无法结束循环。

3.显示库存函数

货品	品名	单价	库存量
001	衣服	150.00	50
货品	品名	单价	库存量
002	食品	150.00	50
货品	品名	单价	库存量
003	日用品	150.00	50
货品	品名	单价	库存量
004	工具	150.00	50
货品	品名	单价	库存量
005	其他	50.00	50

4.购物车函数

输入条件	有效等价类	无效等价类
1<=x<=3	X=1~3 (1)	X<0,X>3

测试用例：x = 0， 4， 1

```
请选择操作
-----
1. 显示当前购物列表
2. 添加商品
3. 退出
-----

0
输入错误, 请重新输入:4
输入错误, 请重新输入:1
购物车为空

请选择操作
-----
1. 显示当前购物列表
2. 添加商品
3. 退出
```

5.添加商品函数

输入条件	有效等价类	无效等价类
商品存在库存里	商品存在库存里 (1)	商品不在库存里 (2)
商品数量充足	商品不在库存里 (3)	商品数量不足 (4)

测试用例：

① (1) (3)

```
输入想购买物品的名称或货号：食品
已经找到想购买的物品：
-----
货号      品名      单价      库存量
002        食品        150.00      50
-----

请输入想购买的数量：12

是否购买?(Y/N) y
是否继续购物?(Y/N) y
```

② (1) (4)

```

输入想购买物品的名称或货号：衣服
已经找到想购买的物品：
-----
货号      品名      单价      库存量
001        衣服        150.00        50
-----
请输入想购买的数量：200
库存不足
是否继续购物?(Y/N) y

```

③ (2) (3)

```

输入想购买物品的名称或货号：飞船
未找到所需物品
是否继续购物?(Y/N) y

```

④ (2) (4)

```

输入想购买物品的名称或货号：模型
未找到所需物品
是否继续购物?(Y/N) n

```

6.查看购物车的物品函数（没有无效条件类测试）

(1) 有物品添加情况

```

-----
货号      品名      单价      数量
          002        食品  150.00        12

```

(2) 无物品添加情况

```

1
购物车为空

```

7.付款函数

输入条件	等效条件类	无效条件类
实付金额>=需付金额	实付>=需付（1）	实付<需付（2）

测试用例：

（1）

```

以下是购物清单：
-----
货号    品名    单价    数量
      002          食品 150.00      12
总计 1800.00
输入实付金额：1800
实付： 1800.00 找零： 0.00

```

（2）

```

以下是购物清单：
-----
货号    品名    单价    数量
      002          食品 150.00      12
总计 1800.00
输入实付金额：1000
金额不足

```

8.查看已购买记录

输入条件	有效条件类	无效条件类
地址补全	补全（1）	未补全（2）

测试用例：

（1）

货号	品名	单价	数量
002	食品	150.00	12

本月总共的销售额为：1800.00

(2)

购物清单			
货号	品名	单价	数量
002	食品	150.00	12

本月您总共购买了：1800.00

您的收货地址尚未补全噢！

9.补全地址信息（没有无效条件类测试）

(1) 未确认地址

```

请输入您的家庭地址：
xidian1haolou
这是您家庭地址： xidian1haolou
是否确认？（Y/N） :n
是否继续填写？

```

(2) 确认地址


```
请输入您的家庭地址：
xidian
这是您家庭地址： xidian
是否确认？（Y/N）：y
成功！
```

10.查看缺货情况（没有无效条件类测试）

```
-----
货号      品名      单价      数量
          001              衣服 150.00          150
-----
是否想要补货？（Y/N） y
输入想要补货的商品货号和数量：
001
150
```

11.查看销售记录

```
-----
货号      品名      单价      数量
          002              食品 150.00          12
-----
本月总共的销售额为： 1800.00
```

12.查看最短路径配送

输入条件	有效条件类	无效条件类
数字	输入数字（1）	输入非数字（2）

（1）

请输入城市横坐标:

3

2

5

6

3

3

请输入城市纵坐标:

6

4

5

3

8

3

最短距离的配送顺序为:

01 05 03 04 02

(2)

请输入城市横坐标:

S

请输入城市纵坐标:

最短距离的配送顺序为:

01 05 02 04 03

系统会随机挑选一条配送路径