****

算法设计与分析

实验报告(回溯)

**学生姓名 杨凯楠**

**学 号** 8208201004

**专业班级 信息安全2002班**

**指导教师** 石峰

**学 院** 计算机学院

**完成时间** 2021年11月21日

1. **实验目的：**熟练掌握回溯算法
2. **实验内容**

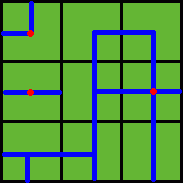
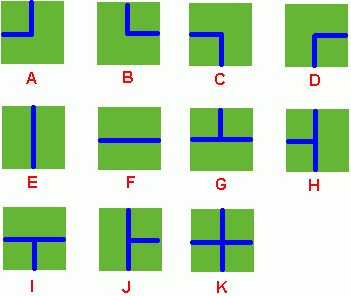
1.装载问题

有两艘船，载重量分别是c1、 c2，n个集装箱，重量是wi (i=1…n)，且所有集装箱的总重量不超过c1+c2。确定是否有可能将所有集装箱全部装入两艘船。

提示：求出不超过c1的最大值max，若总重量－max < c2则能装入到两艘船。

2.农场灌溉问题（ZOJ2412）

一农场由图所示的十一种小方块组成，蓝色线条为灌溉渠。若相邻两块的灌溉渠相连则只需一口水井灌溉。给出若干由字母表示的最大不超过50×50具体由(m，n)表示，的农场图，编程求出最小需要打的井数。每个测例的输出占一行。当M=N=-1时结束程序。



**Sample Input**

2 2

DK

HF

3 3

ADC

FJK

IHE

-1 -1

**Sample Output**

2

3

1. **具体设计**

**1.装载问题**

有两艘船，载重量分别是c1、 c2，n个集装箱，重量是wi (i=1…n)，且所有集装箱的总重量不超过c1+c2。确定是否有可能将所有集装箱全部装入两艘船。

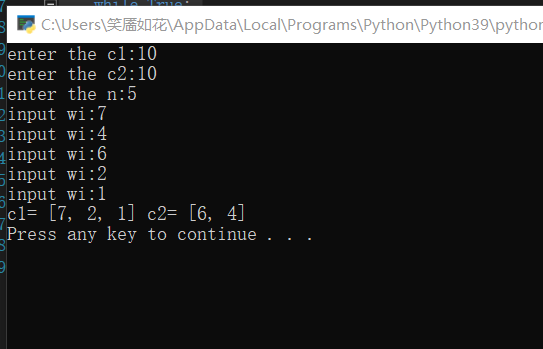
提示：求出不超过c1的最大值max，若总重量－max < c2则能装入到两艘船。

**分析**：本题较为简单，通过递归，采用深度优先遍历，边遍历边生成树在遍历过程中随时查看是否已经到达死节点。最终寻找出最优解。

**详细设计：**

1. **def** set():
2. c1=int(input("enter the c1:"))
3. c2=int(input("enter the c2:"))
4. n=int (input("enter the n:"))
5. a=[]
6. maxw=0
7. **for** i **in** range(n):
8. a.append(int(input("input wi:")))
9. maxw+=a[i]
10. **if** maxw>c1+c2:                     #当输入错误的数据返回FALSE，并在主函数中重新输入数据
11. **return** False,False,False,False
12. **return** c1,c2,n,a

15. **def** solution(c1,c2,w):
16. w.sort(reverse=True)
17. temp\_c=tot=0
18. a=[]
19. b=[]
20. **for** i **in** w:
21. temp\_c+=i
22. tot+=i
23. **if** temp\_c<=c1:
24. a.append(i)
25. **else**:
26. temp\_c-=i
27. b.append(i)
28. **if** tot-temp\_c<=c2:
29. **return** True,a,b
30. **else**:
31. **return** False,False,False
33. **def** main():
34. **while** True:
35. c1,c2,n,w=set()
36. **if** c1:
37. **break**
38. **print**("wrong data!")
39. flag,soc1,soc2=solution(c1,c2,w)
40. **if** flag:
41. **print**("c1=",soc1,"c2=",soc2)
42. **else**:
43. **print**("wrong")
44. main()

**实现：**

调试分析：由此可见在该输入条件下是可以装载进去的。有一个小技巧是，先对一只船做最大容量装载，然后用总重量减去最大容量判断第二只船能不能装下剩下的集装箱。

**2.农场灌溉问题（ZOJ2412）**

一农场由图所示的十一种小方块组成，蓝色线条为灌溉渠。若相邻两块的灌溉渠相连则只需一口水井灌溉。给出若干由字母表示的最大不超过50×50具体由(m，n)表示，的农场图，编程求出最小需要打的井数。每个测例的输出占一行。当M=N=-1时结束程序。

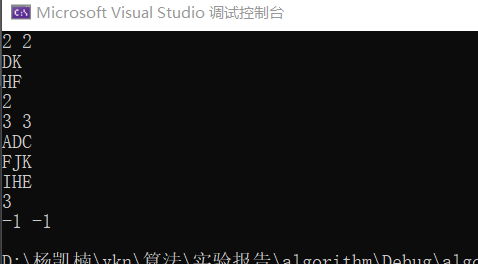
**分析：**本题实际上就是图的连通度的问题，采用深度优先遍历，遍历一遍就可得出连通度。在将农田抽象化时，采用了把农田看成一个点，连通的水渠看成无向边的方法。初始化农田时，只对单块农田进行初始化，当他上下左右伸展出水渠时就定义为二维数组的1，否则为0。两点之间存在无向边的条件是两点相邻的地方都伸展出了水渠。

**详细设计**：

1. #include<stdio.h>
2. #include<iostream>
3. #include<vector>
4. using namespace std;
5. void set\_canal(int a[][4]) {
6. a[0][0] = a[0][1] = 1;
7. a[1][1] = a[1][2] = 1;
8. a[2][0] = a[2][3] = 1;
9. a[3][2] = a[3][3] = 1;
10. a[4][1] = a[4][3] = 1;
11. a[5][0] = a[5][2] = 1;
12. a[6][0] = a[6][1] = a[6][2] = 1;
13. a[7][0] = a[7][1] = a[7][3] = 1;
14. a[8][0] = a[8][2] = a[8][3] = 1;
15. a[9][1] = a[9][2] = a[9][3] = 1;
16. a[10][0] = a[10][1] = a[10][2] = a[10][3] = 1;
17. }
18. void set\_canal\_now(vector<vector<int> >& a,int row,int col) {
19. char b;
20. scanf\_s("%c", &b, 1);
21. **for** (int i = 0; i < row; i++) {
22. **for** (int j = 0; j < col; j++) {
23. scanf\_s("%c", &b, 1);
24. a[i][j] = b - 65;
25. }
26. scanf\_s("%c",&b,1);
27. }
29. }
30. //a里面存的是字母对应的数字
31. void count\_recursion(vector<vector<int> >& a, int row, int b[][4], int col, vector<vector<int> >& flag,int i,int j) {
32. **if** (flag[i][j] == 0&&(i>=0&&i<row)&&(0<=j&&j<col)) {
33. flag[i][j] = 1;
34. **if**(i<row-1)
35. **if**(b[a[i][j]][3]==1&&b[a[i+1][j]][1]==1)
36. count\_recursion(a, row, b, col, flag, i+1, j);
37. **if**(i>0)
38. **if**(b[a[i][j]][1]==1&&b[a[i-1][j]][3]==1)
39. count\_recursion(a, row, b, col, flag, i -1, j);
40. **if**(j>0)
41. **if** (b[a[i][j]][0] == 1 && b[a[i][j-1]][2] == 1)
42. count\_recursion(a, row, b, col, flag, i , j-1);
43. **if**(j<col-1)
44. **if** (b[a[i][j]][2] == 1 && b[a[i][j+1]][0] == 1)
45. count\_recursion(a, row, b, col, flag, i , j+1);
46. }

49. }
50. int countl(vector<vector<int> >& a, int row, int col, int b[][4]) {
51. int i, j,count=0;
52. vector<vector<int> > flag(row, vector<int>(col));
53. **for** (i = 0; i < row; i++)
54. **for** (j = 0; j < col; j++)
55. flag[i][j] = 0;
56. **for** (i = 0; i < row; i++) {
57. **for** (j = 0; j < col; j++) {
58. **if** (flag[i][j] == 0) {
59. count++;
60. count\_recursion(a, row, b, col, flag, i, j);
61. }
62. }
63. }
64. **return** count;
65. }
66. int main(void) {
67. int a[11][4] = { 0 };
68. set\_canal(a);
69. int row, col;
70. cin >> row >> col;
71. **while** (row != -1 && col != -1) {
72. vector<vector<int> > b(row, vector<int>(col));
73. set\_canal\_now(b, row, col);
74. int count\_jing = countl(b, row, col, a);
75. cout << count\_jing << endl;
76. cin >> row >> col;
77. }
78. **return** 0;
79. }

**实现**：



**调试分析**：测试结果显示没有问题。在调试过程中，我用容器vector库创建了一个二维空间来存储农田信息，每个空间单元为一个农田种类。

1. **实验心得**

本次试验充分体现了用算法解决实际问题的特点，当拿到一个具体问题是，需要迅速将其抽象化，判断适用什么数据结构，采用什么算法，可以将其等效为哪一种已经学过的问题。可以说本次实验非常具有实践意义。经过本次实验，我掌握了回溯的算法，深刻体会到了算法的妙处，开始进一步尝试用算法问题解决生活中的问题，感觉收获颇丰而又愈发感觉自身知识的欠缺，今后将继续努力。