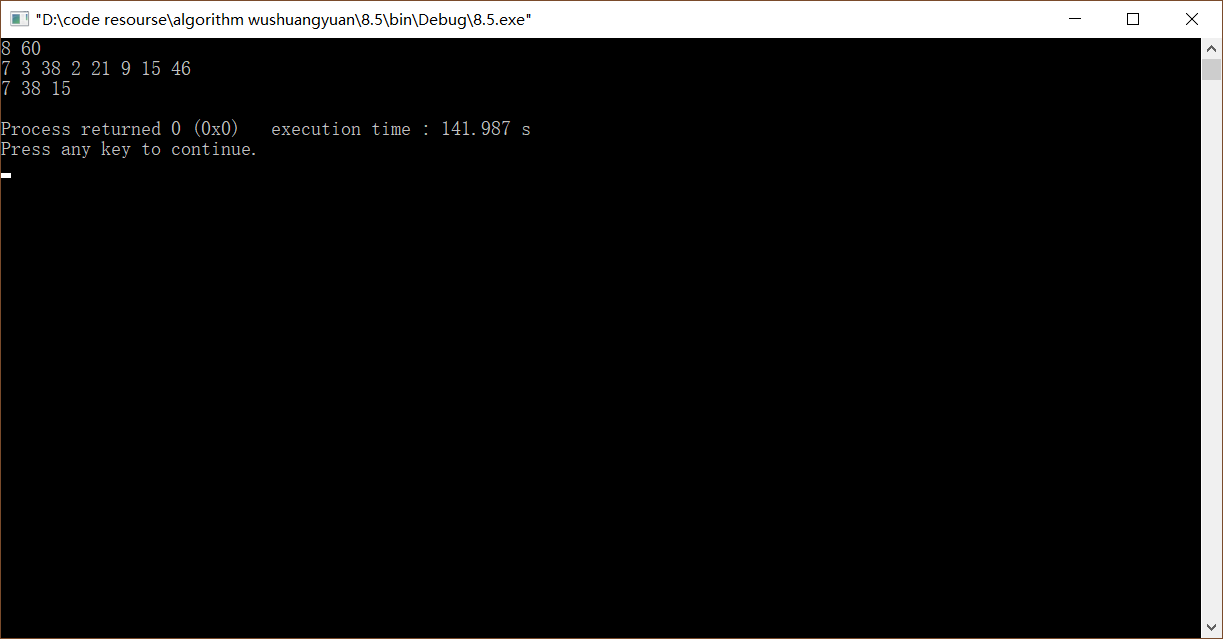
习题8 5.

该算法遍历每一个可行的解，最终得到答案。设置标志b，每次搜索的初始阶段都检查b是否为真。当搜索得到答案序列时，将b置为真，接下来的搜索一律跳过不执行。当满足加深搜索的条件时，将当前数据压栈，递归深入一层，当不满足条件时，递归退出，并且清除之前压栈的元素，在此处实现回溯。程序运行截图如下。



输入n=8，y=6，数组7 3 38 2 21 9 15 46，得到一个答案序列7 38 15.

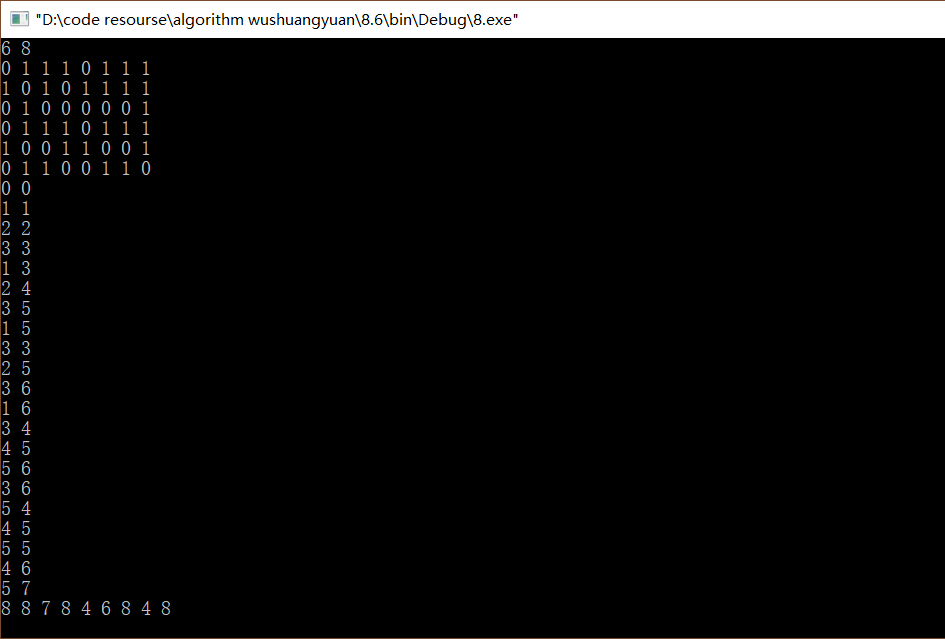
习题8 6.

编写回溯程序，实现找到迷宫中的一条路径（不一定最短）。设置了八个前进方向向量，按数字编号，依次遍历尝试每一个可能的前进方向，并给出一个可行解。

用整型栈保存前进信息，每次尝试一步移动策略，如果最终能达到一个可行解，该策略就是答案的一步，如果最终不能得到可行解，就在尝试过所有可能性后数据出栈，继续尝试其他可能的移动。

程序截图如下。最短路线如绿线标号，该算法解如红线标识。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 01 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 02 | 1 | 4 | 1 | 1 | 1 | 1 |
| 0 | 1 | 03 | 0 | 5 | 6 | 0 | 1 |
| 0 | 1 | 1 | 1 | 07 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 08 | 09 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 010 |



显然，如果想得到最优路径，只需要遍历所有可行解，每次记录移动长度，并保存其中的路径最短值即可。