

**算法期末大作业**

学 系： 信息科学学院

专业班级： 20级计算机科学与技术1班

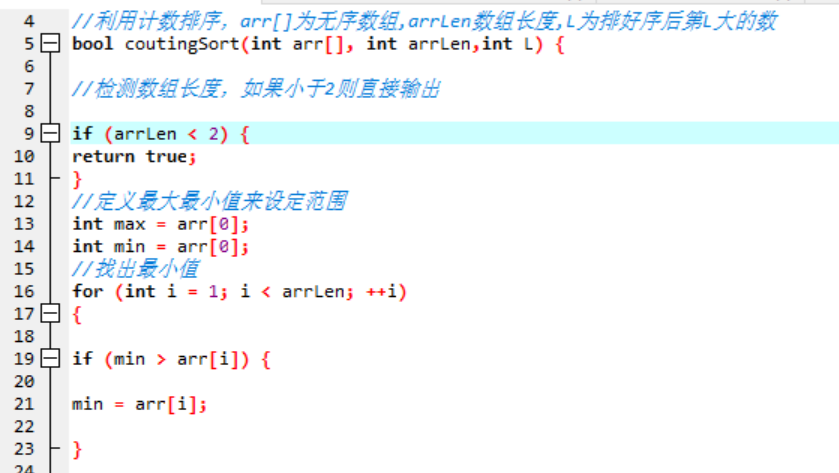
学 号： 20052050

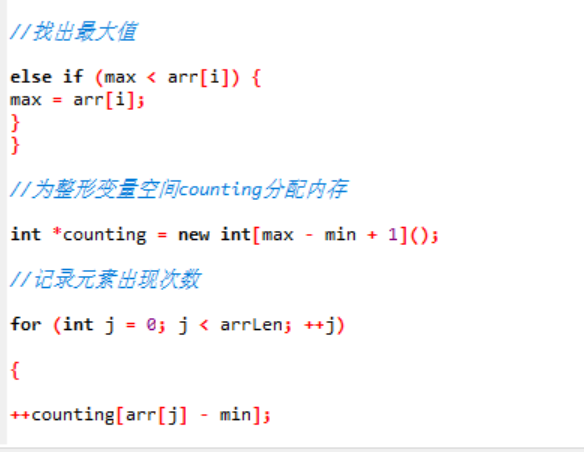
学生姓名： 杨伟城

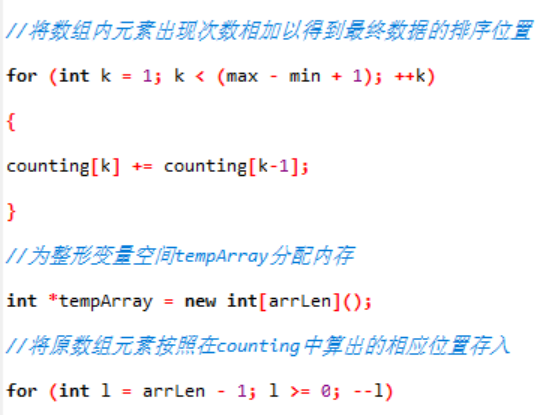
二〇二一 年 十二 月

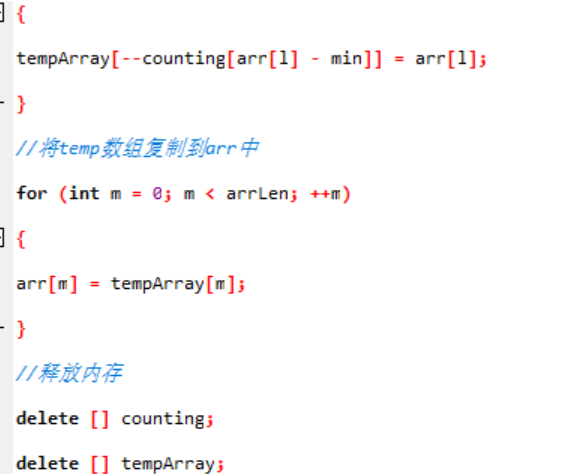
## 要求分别用𝑂(𝑛)时间复杂度的算法找到未排序数组中第K大的数据

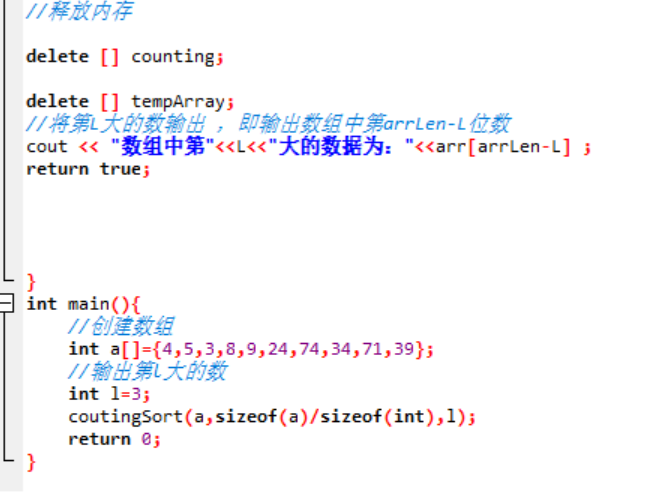
### 代码实现与截图:

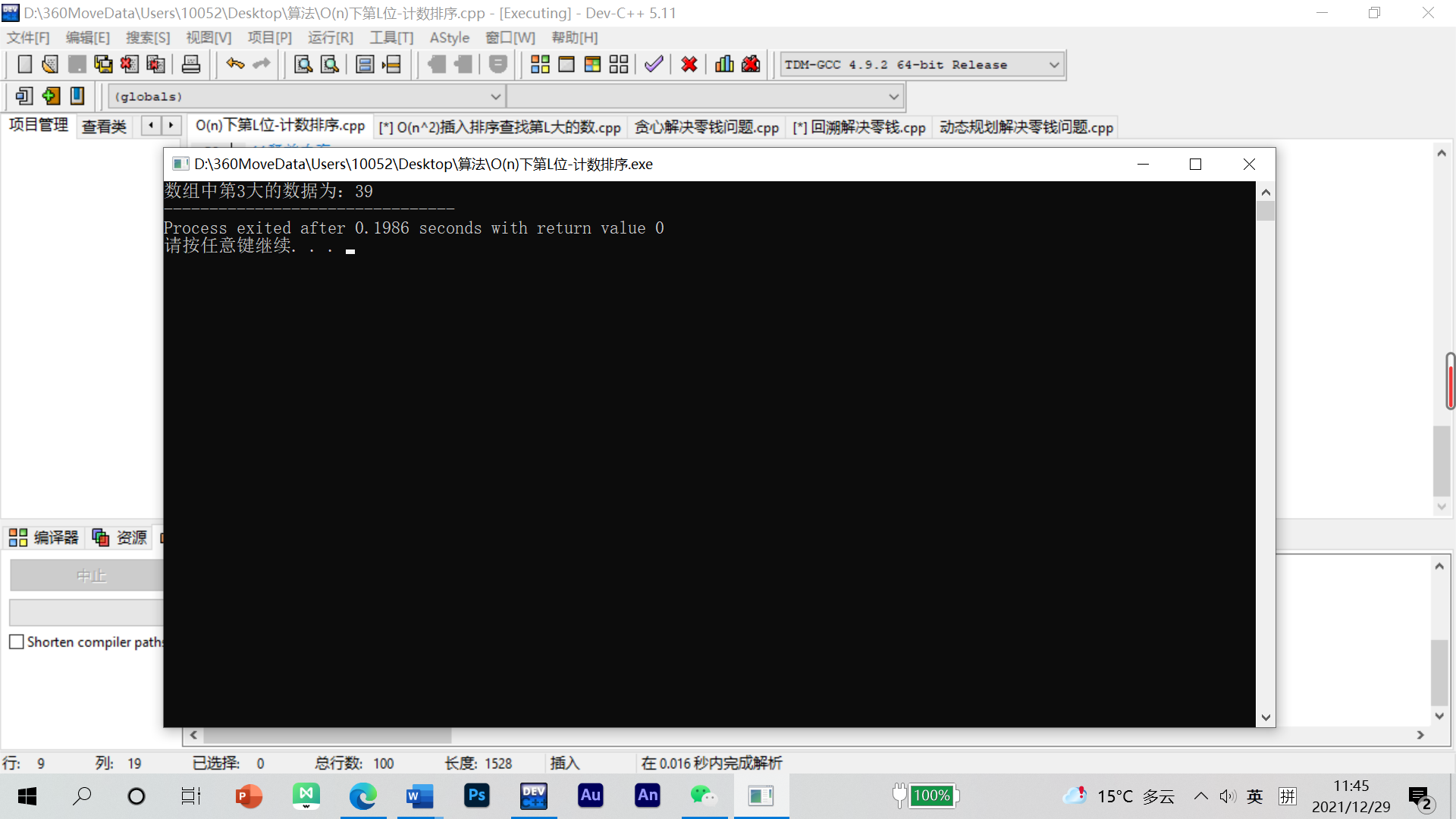






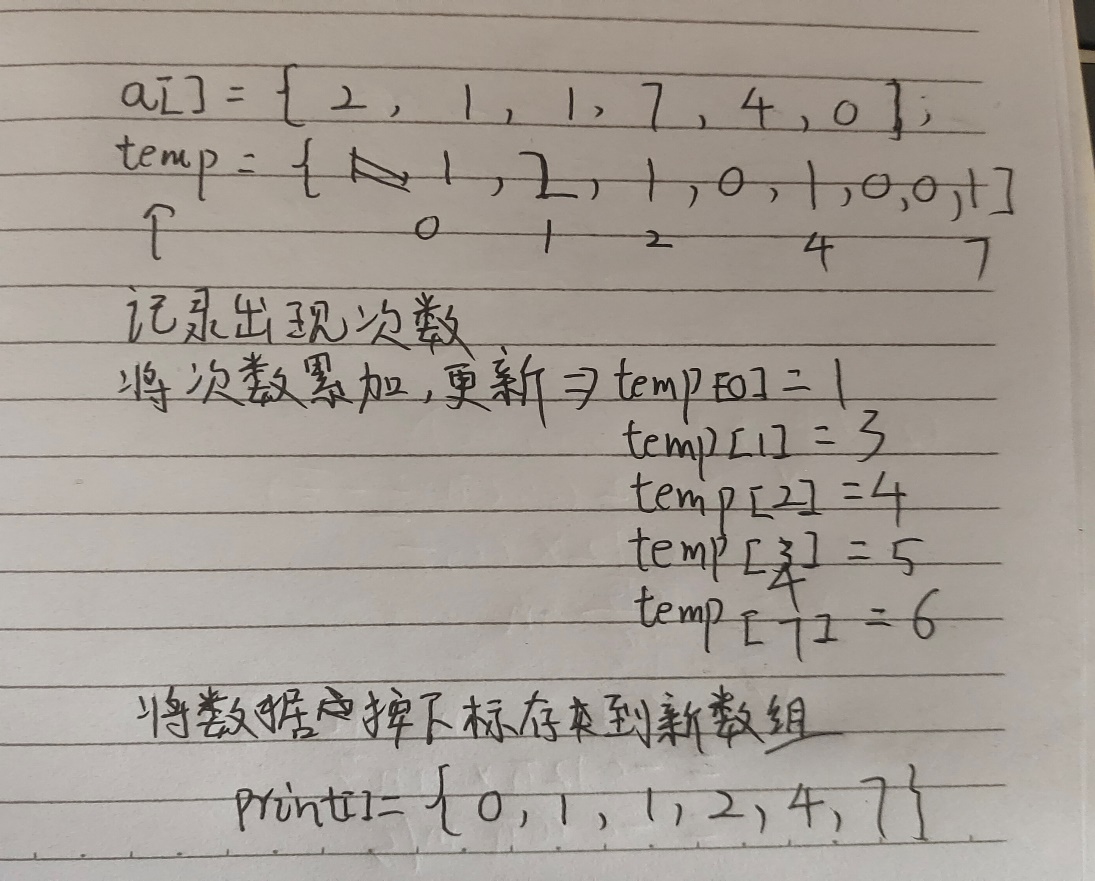






### 图解：

想法：将未排序数组利用计数排序化为有序数组后，将（数组长度-K）位数输出，即数组中第K大的数。



### 时间复杂度分析

**时间复杂度：O（n）**

解释：其对n个输入进行排序，但其排序是通过最大值与最小值之间的范k围来实现的，所以时间复杂度未为O（n+k）

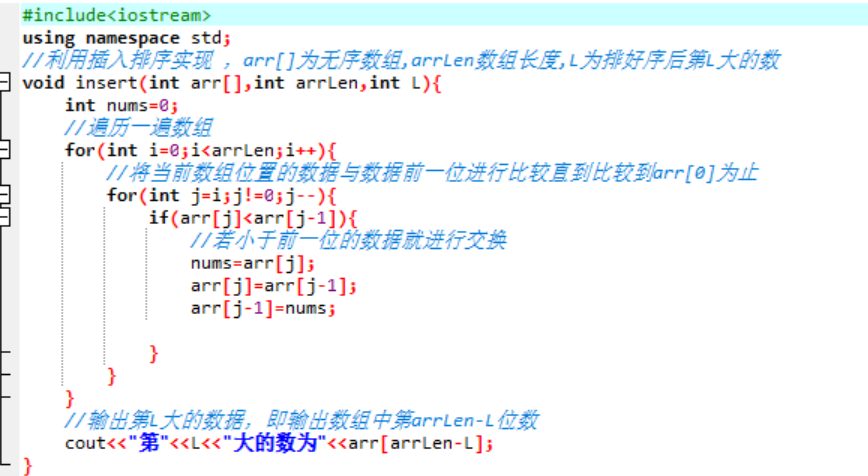
### 空间复杂度分析

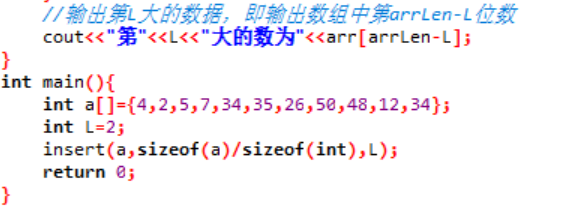
**空间复杂度：O（n）**

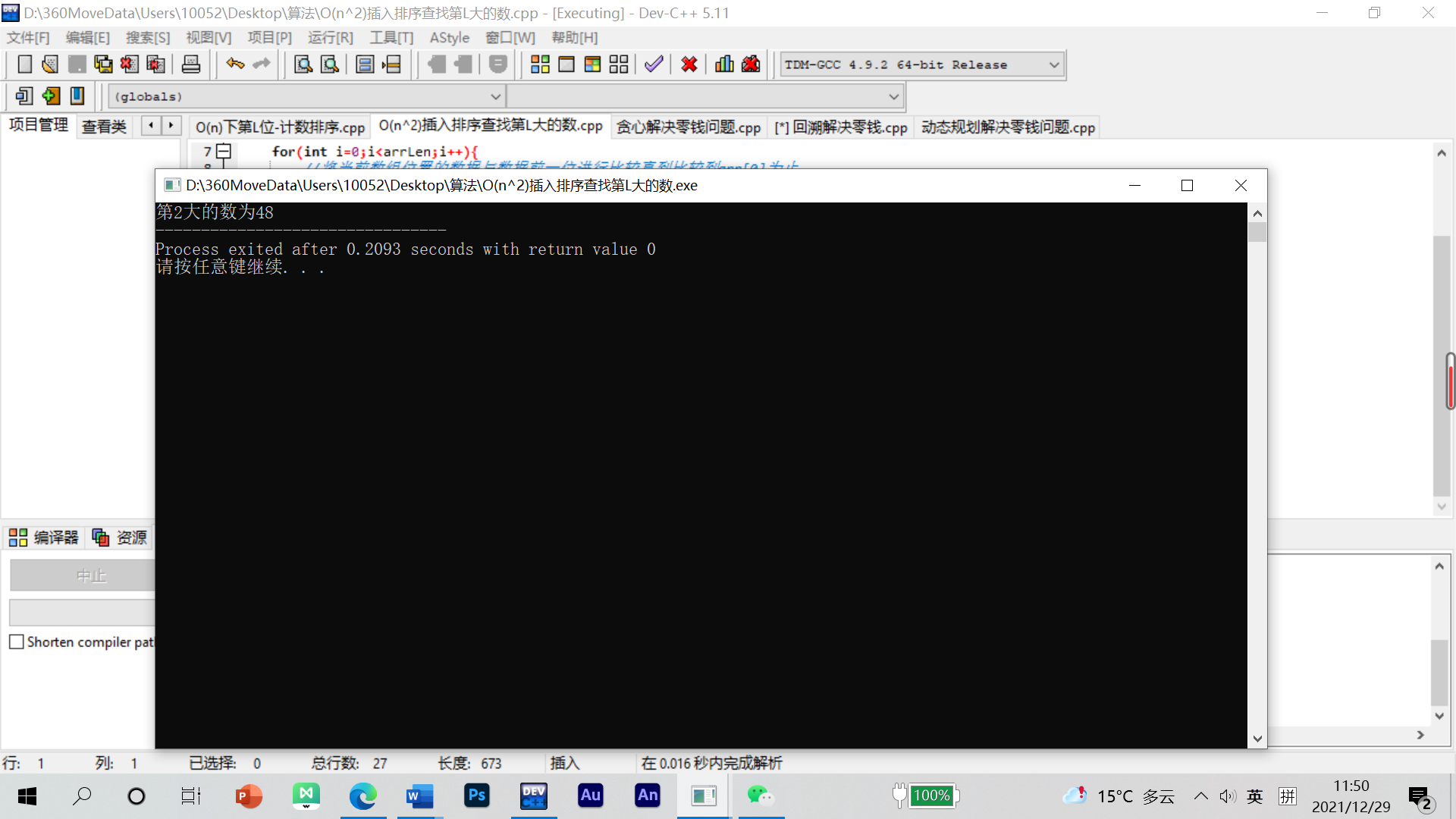
解释：在进行排序的过程中会创建新的数组来记录范围K内数字的出现个数与新数组来存放排列好的数据。所以空间复杂度为 O（n+k）

## 二、 要求分别用𝑂(𝑛^2)时间复杂度的算法找到未排序数组中第K大的数据

### 1、代码实现与运行截图:

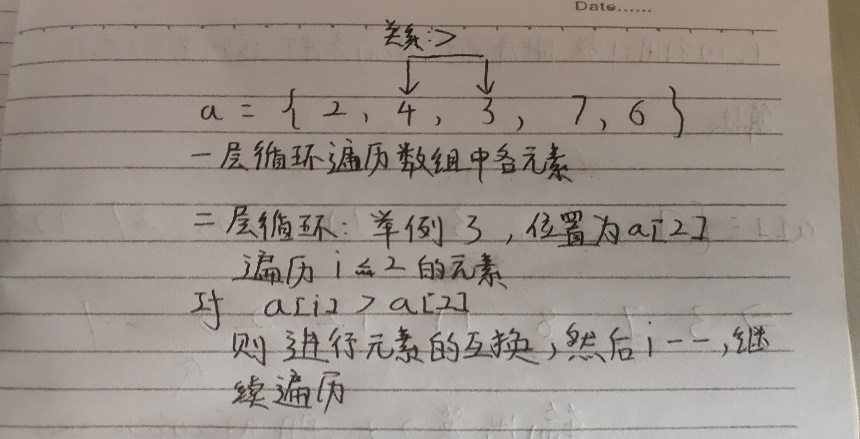






### 2、图解：

想法：通过嵌套循环遍历数组，然后将数组位置的数据与数据前一位进行比较直到比较，若小于则交换。



### 3、时间复杂度分析

**时间复杂度：O（n^2）**

解释：无论如何都会遍历整个数组，并且因为循环函数嵌套，所以时间复杂度为O（n\*n）

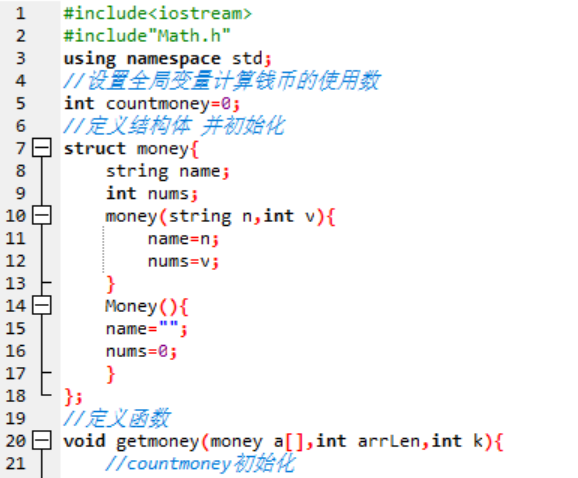
### 4、空间复杂度分析

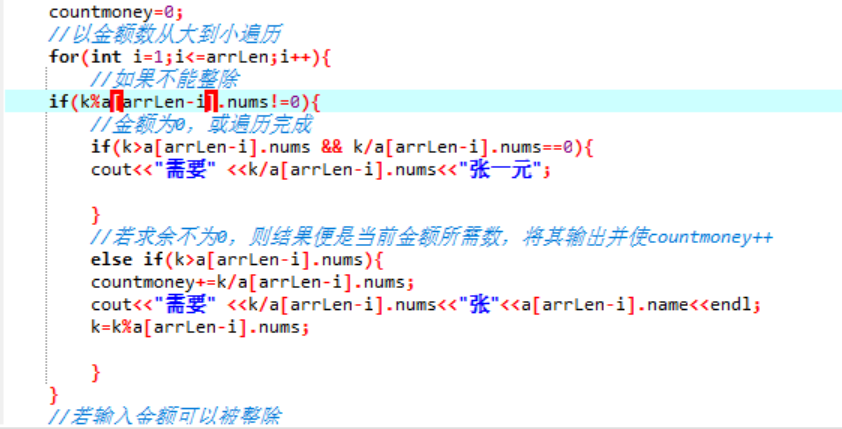
**空间复杂度：O（1）**

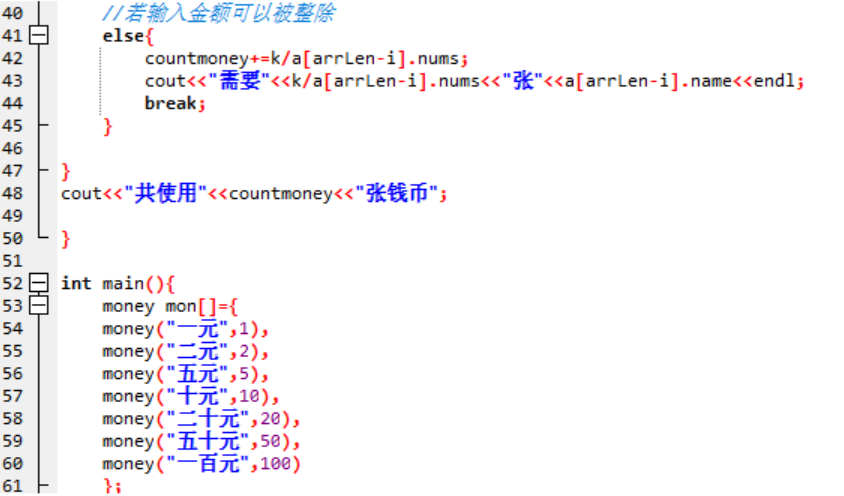
解释：遍历过程并没有新空间的创建，所以空间复杂度为O（1）；

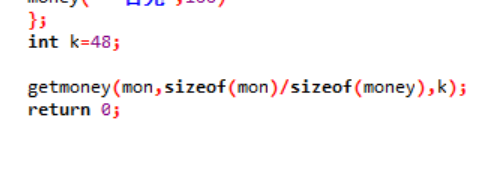
## 二、使用贪心算法实现下列问题，假设有1元、2元、5元、10元、20元、50元、100元面额的纸币，它们的张数不限。现在要用这些钱能够恰好支付K元，请问最少需要多少张纸币？如果不能恰好支付，返回不能支付的提示

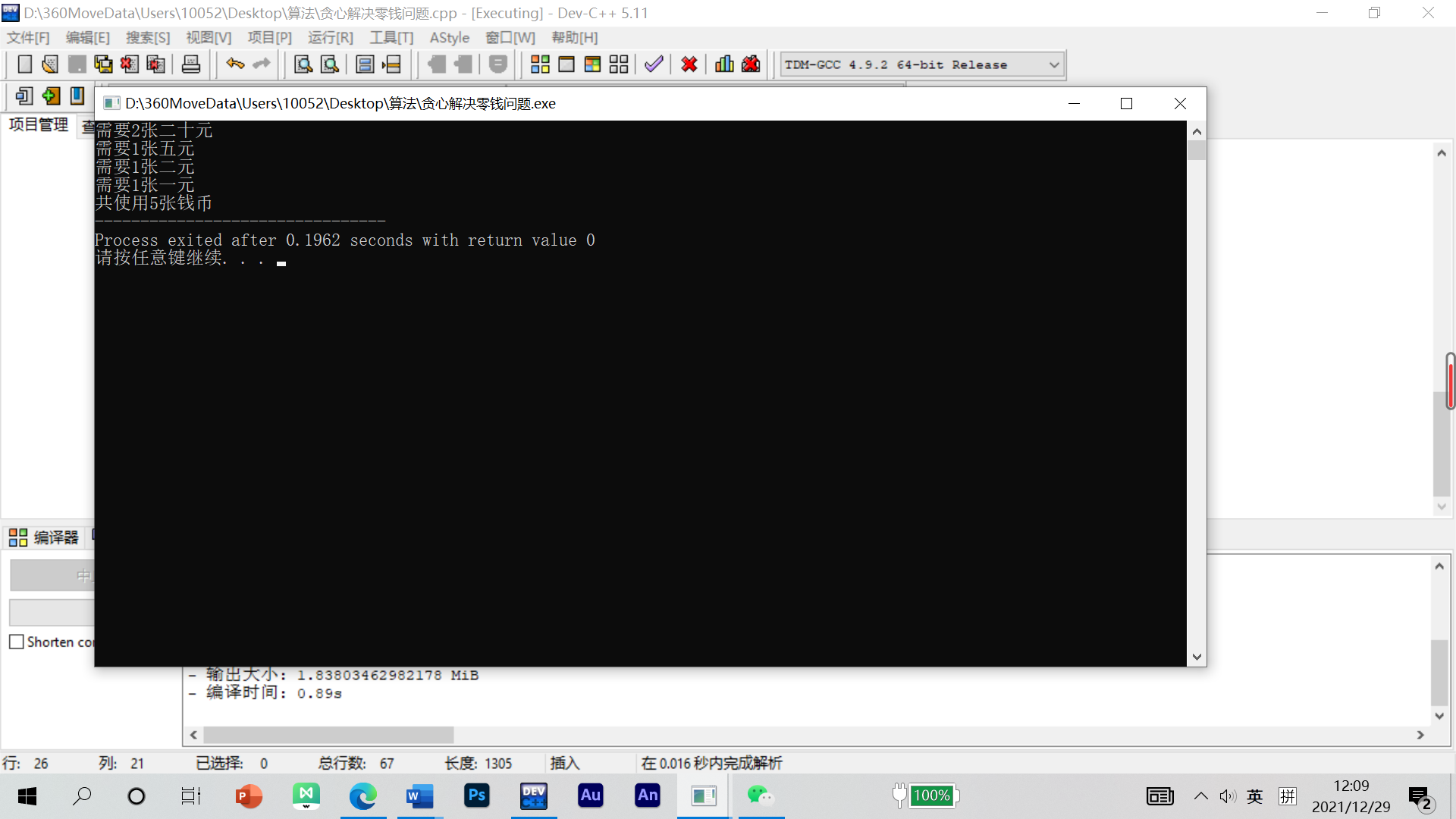
### 1、代码实现与截图:





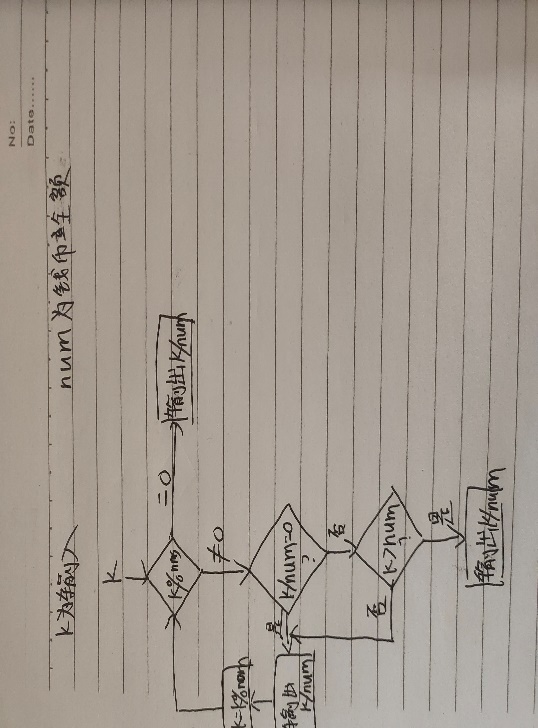






### 2、图解：

思路：**将金额从大到小遍历，令金额k对当前钱币数额num求余，如果结果为0，则输出k/num,否则令k/num,若结果！=0，再判断k是否大于num,是的话输出，若k/num结果为0，或k小于等于num，则输出k/num，并令k=k%num,继续循环。**

****

### 时间复杂度分析

**时间复杂度：O（n）**

解释：仅有一层for循环作为主题。

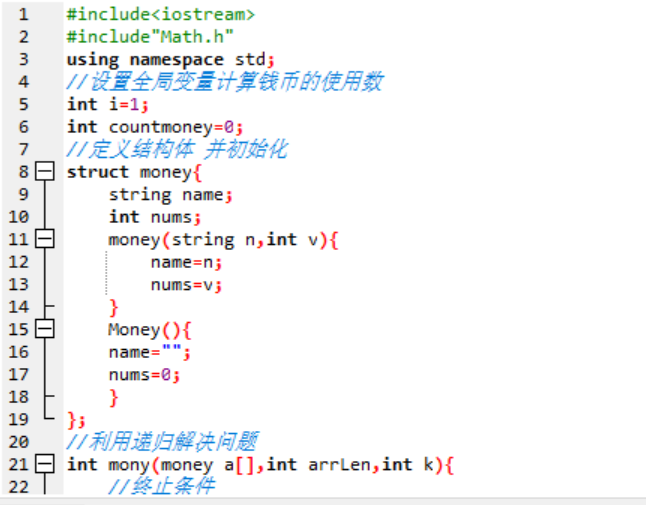
### 4、空间复杂度分析

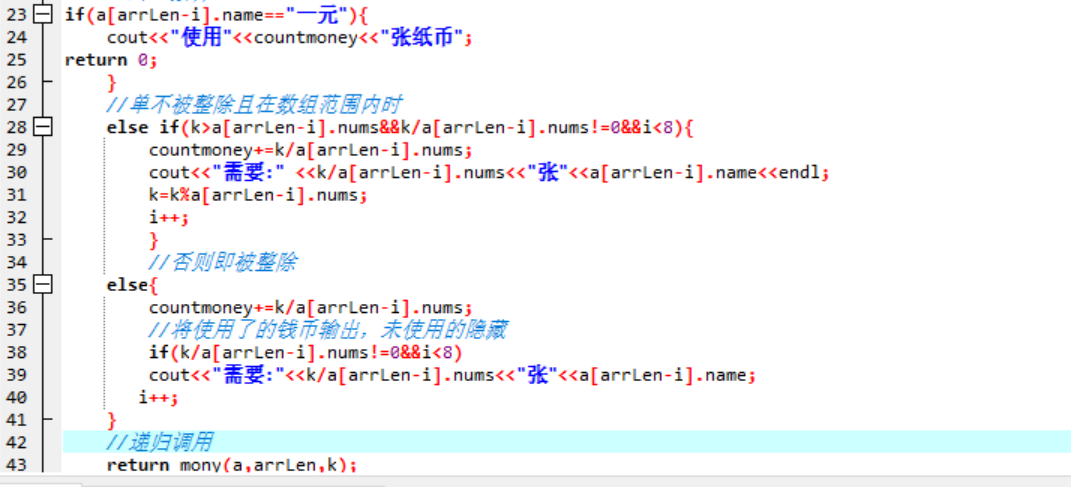
**空间复杂度：O（1）**

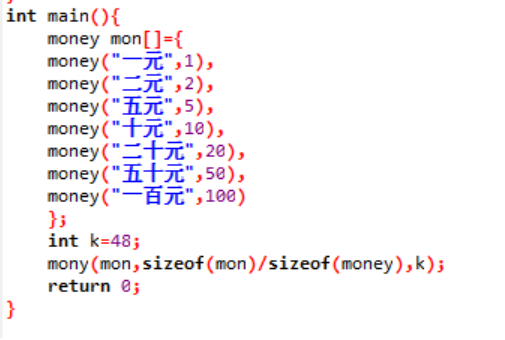
解释：遍历过程并没有新空间的创建，所以空间复杂度为O（1）

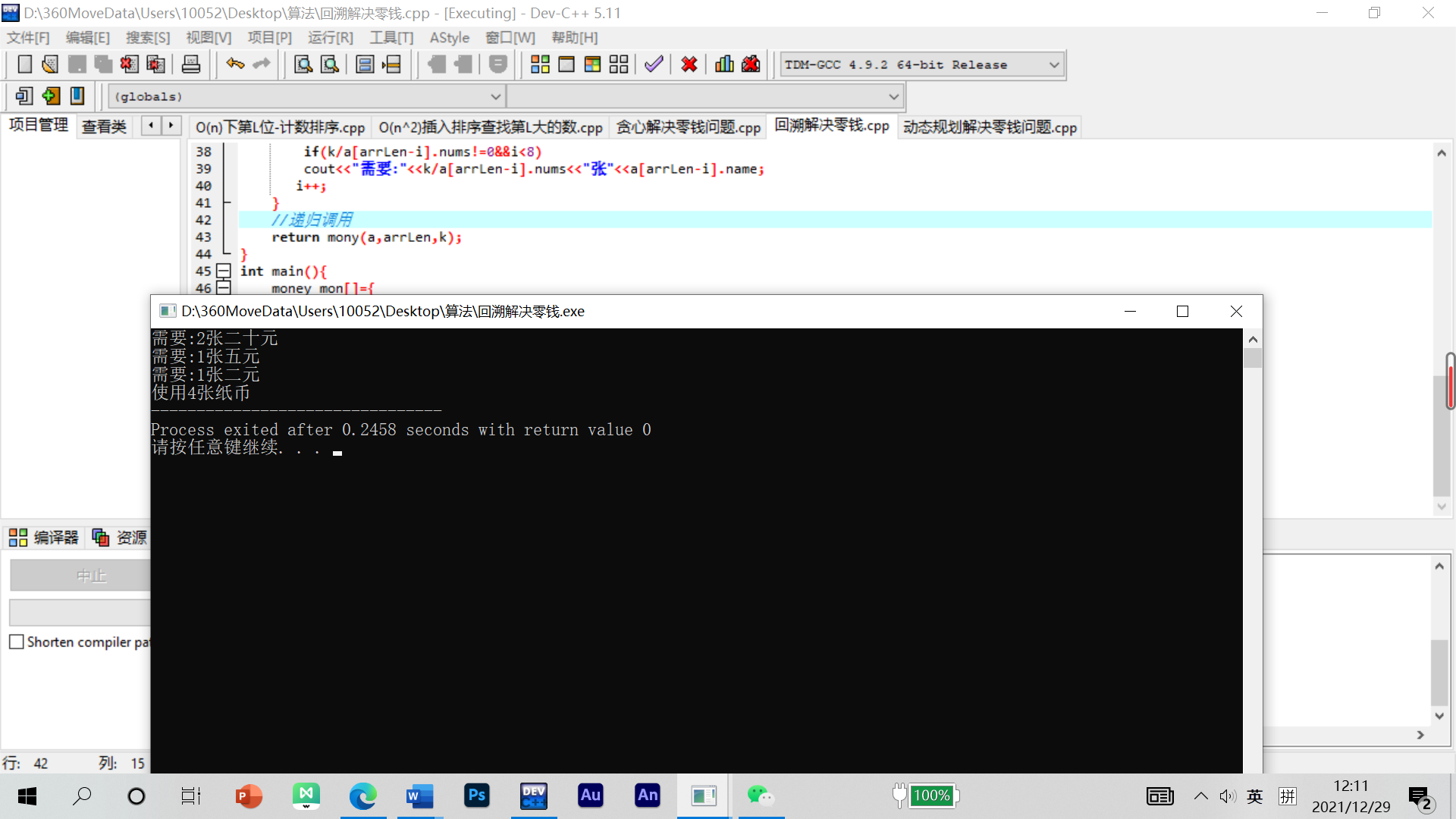
## 三、使用回溯实现下列问题，假设有1元、2元、5元、10元、20元、50元、100元面额的纸币，它们的张数不限。现在要用这些钱能够恰好支付K元，请问最少需要多少张纸币？如果不能恰好支付，返回不能支付的提示

### 1、代码实现与截图:



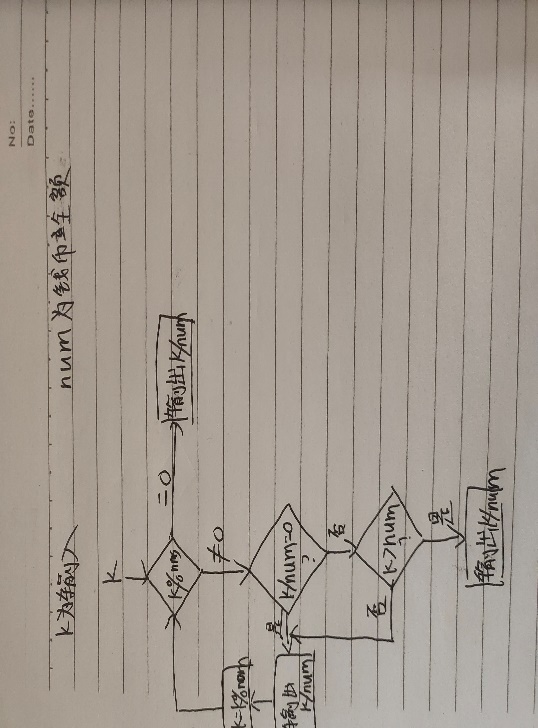






### 2、图解

**思路与贪心相似，将金额从大到小遍历，令金额k对当前钱币数额num求余，如果结果为0，则输出k/num,否则令k/num,若结果！=0，再判断k是否大于num,是的话输出，若k/num结果为0，或k小于等于num，则输出k/num，并令k=k%num,然后递归调用。**

****

### 3、时间复杂度分析

**时间复杂度：O（n）**

解释：数据规模不影响递归调用次数,与贪心一致。

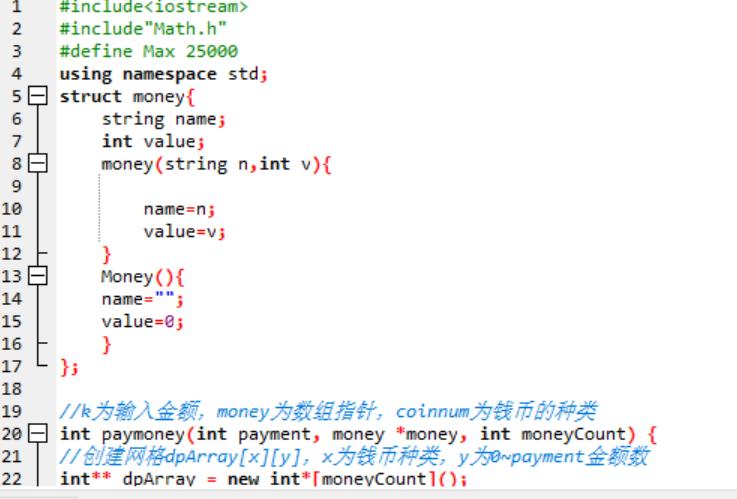
## 空间复杂度分析

**空间复杂度：O（1）**

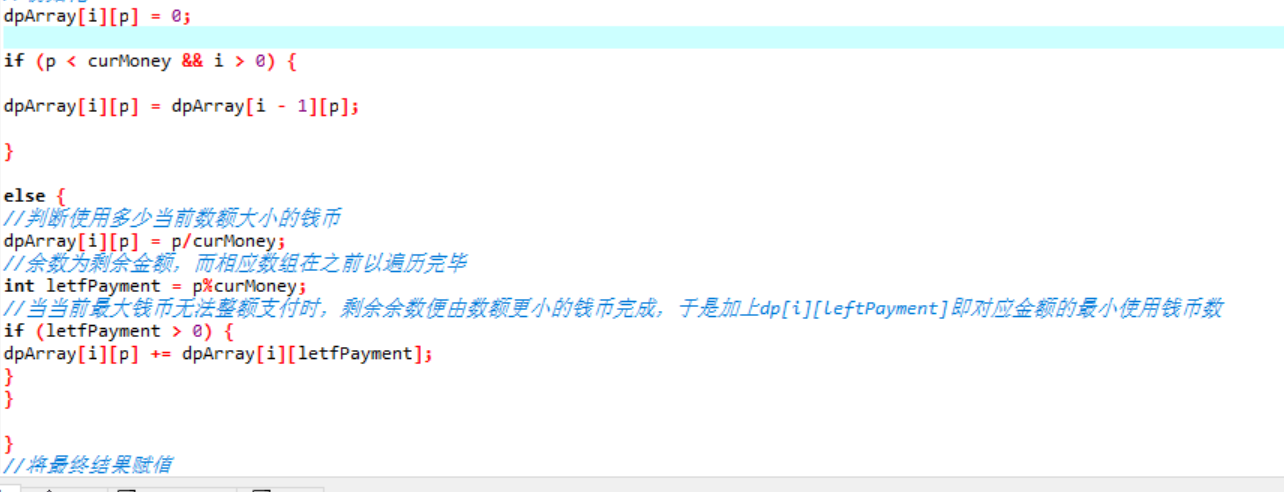
解释：递归过程并没有新空间的创建，所以空间复杂度为O（1）

## 三、使用动态规划实现下列问题，假设有1元、2元、5元、10元、20元、50元、100元面额的纸币，它们的张数不限。现在要用这些钱能够恰好支付K元，请问最少需要多少张纸币？如果不能恰好支付，返回不能支付的提示

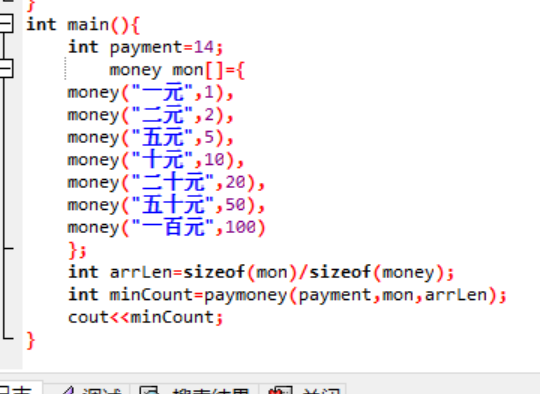
### 1、代码实现与截图:

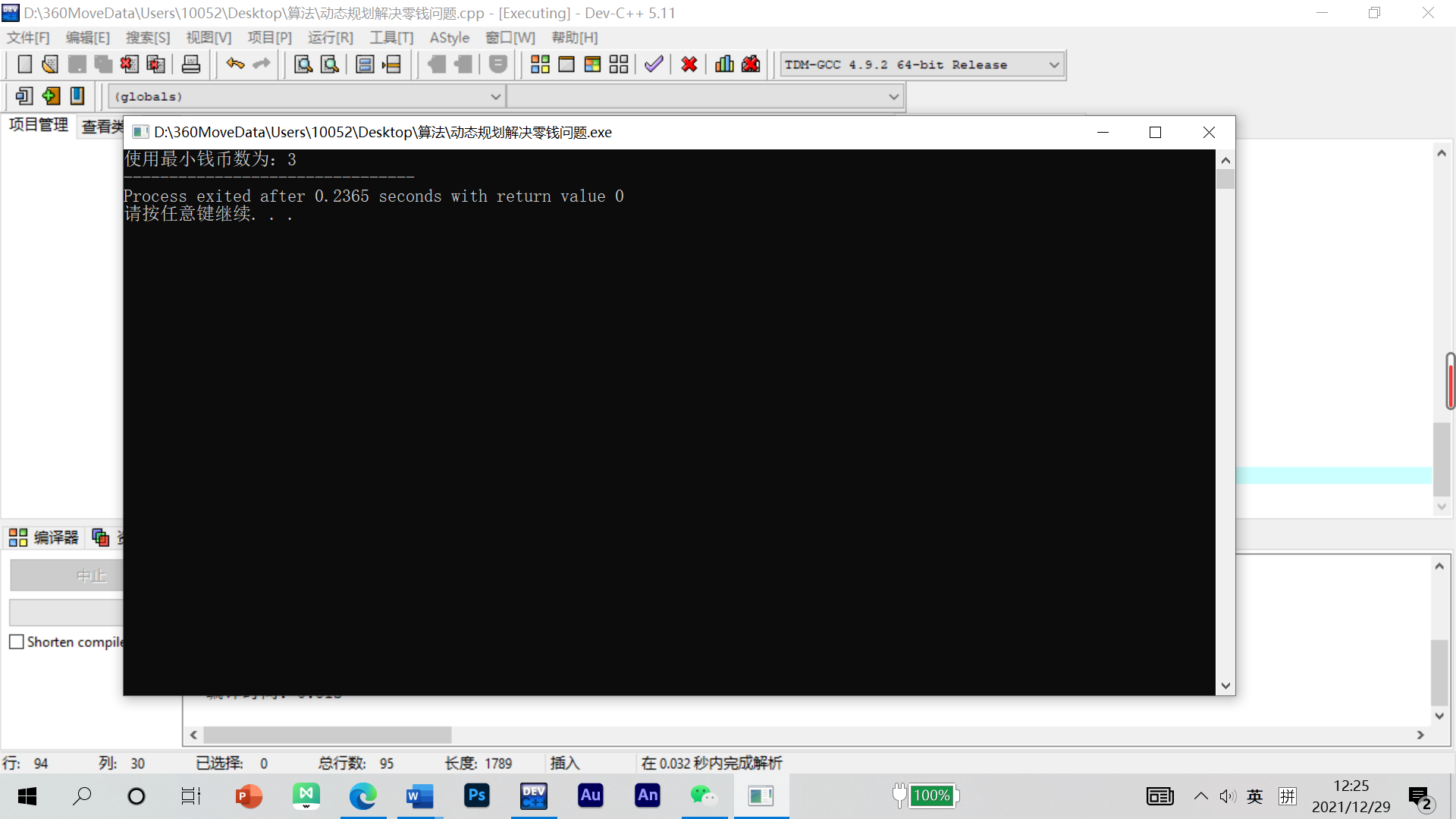






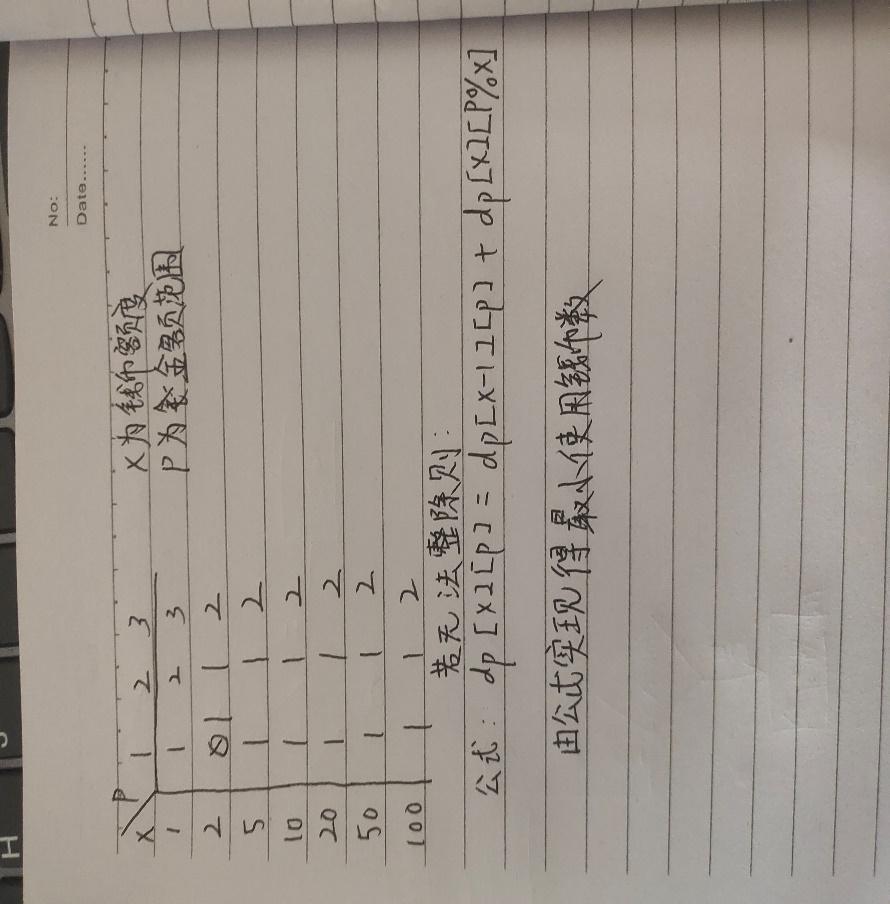






### 2、图解

**思路：令钱币数额为竖轴，金额范围为横轴，若金额可以直接被钱币整除，则直接输出钱币金额p/数额x,否则将其带入2公式dp[x][p]=dp[x-1][p]+dp[x][p%x]中得到最小使用数，最后将其相应位置输出即可**

****

### 3、时间复杂度分析

**时间复杂度：O（n^2）**

解释：方法需要两层循环来遍历每一个网格，所以时间复杂度为：O（n^2）。

### 4、空间复杂度分析

**空间复杂度：O（n）**

解释：在进行计算的过程中会创建网格来记录金额p的范围与钱币数额的关系。所以空间复杂度为 O（np）