**安 庆 师 范 大 学**

**计算机与信息 学院实验报告**

2020- 2021 学年 2 学期

课程名称： 算法设计与分析

指导教师： 江克勤

学号姓名： 071219130

学 院： 计算机与信息学院

专 业： 计算机科学与技术

年级班级： 2019 级 2 班

**安庆师范大学学生实验守则**

一、应按照课程教学计划安排，准时上课，不得迟到早退。

二、实验前应认真阅读实验指导书，明确实验目的、原理、步骤，预习有关理论知识。

三、遵守实验室规章制度，不得喧哗和打闹，不准抽烟、随地吐痰和乱丢纸屑等杂物；有净化要求的实验室，进室须换鞋。

四、严格遵守仪器设备操作规程，爱护仪器设备，节约使用材料，服从实验教师和技术人员的指导。未经许可，不得动用与实验无关的仪器设备及其他物品。

五、实验中不得擅自离开操作岗位，应仔细观察，认真记录，不得抄袭他人实验数据。

六、实验时须注意安全，杜绝发生人为安全事故。遇到问题，应及时向指导教师报告。

七、实验完毕，应清理实验现场。经指导教师检查仪器设备和实验记录后，方可离开。

八、实验后应认真完成实验报告，包括数据处理、图表绘制、结果分析等。实验报告应在规定时间内交给指导教师批阅。

九、实验过程中，因不慎造成实验设备损坏，应填写仪器损坏登记表，由指导教师根据情况进行处理。

十、因违反操作规程、擅自动用与实验无关的仪器设备、私自拆卸实验设备等不当行为，造成损失和事故的，须做出书面检查，视情节轻重和认错态度，按规定予以赔偿。

十一、进入开放实验室做实验时，应遵守《安庆师范大学关于实验室向学生开放的暂行规定》（校教字〔2018〕6号）。

目 录

|  |  |  |
| --- | --- | --- |
| **序 号** | **实验名称** | **页码** |
| 1 | 模拟与枚举 |  |
| 2 | 递归与分治（一） |  |
| 3 | 递归与分治（二） |  |
| 4 | 贪心算法 |  |
| 5 | 动态规划 |  |
| 6 | 搜索算法 |  |

# 《算法设计与分析》实验报告1

实验名称：  **模拟与枚举**

实验日期： 2021年3月15日-3月28日

实验地点： A608

1. **实验目的**

**二、实验内容（编程题）**

Title:A-简单的整数划分问题Description: 将正整数n 表示成一系列正整数之和，n=n1+n2+…+nk, 其中n1>=n2>=…>=nk>=1 ，k>=1 。正整数n 的这种表示称为正整数n 的划分。正整数n 的不同的划分个数称为正整数n 的划分数。 Input: 标准的输入包含若干组测试数据。每组测试数据是一个整数N(0 < N <= 50)。 Output: 对于每组测试数据，输出N的划分数。 SampleInput: 5 SampleOutput: 7 Title:B-复杂的整数划分问题Description: 将正整数n 表示成一系列正整数之和，n=n1+n2+…+nk, 其中n1>=n2>=…>=nk>=1 ，k>=1 。正整数n 的这种表示称为正整数n 的划分。Input: 标准的输入包含若干组测试数据。每组测试数据是一行输入数据,包括两个整数N 和 K。 (0 < N <= 50, 0 < K <= N)Output: 对于每组测试数据，输出以下三行数据: 第一行: N划分成K个正整数之和的划分数目 第二行: N划分成若干个不同正整数之和的划分数目 第三行: N划分成若干个奇正整数之和的划分数目SampleInput: 5 2SampleOutput: 233Title:C-Coin ChangeDescription:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
Suppose there are 5 types of coins: 50-cent, 25-cent, 10-cent, 5-cent, and 1-cent. We want to make changes with these coins for a given amount of money.
For example, if we have 11 cents, then we can make changes with one 10-cent coin and one 1-cent coin, or two 5-cent coins and one 1-cent coin, or one 5-cent coin and six 1-cent coins, or eleven 1-cent coins. So there are four ways of making changes for 11 cents with the above coins. Note that we count that there is one way of making change for zero cent.
Write a program to find the total number of different ways of making changes for any amount of money in cents. Your program should be able to handle up to 100 coins.
Input:The input file contains any number of lines, each one consisting of a number ( ≤250 ) for the amount of money in cents.Output:For each input line, output a line containing the number of different ways of making changes with the above 5 types of coins.SampleInput: 1126SampleOutput:413Title:D-最大报销额Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
现有一笔经费可以报销一定额度的发票。允许报销的发票类型包括买图书（A类）、文具（B类）、差旅（C类），要求每张发票的总额不得超过1000元，每张发票上，单项物品的价值不得超过600元。现请你编写程序，在给出的一堆发票中找出可以报销的、不超过给定额度的最大报销额。
Input:测试输入包含若干测试用例。每个测试用例的第1行包含两个正数 Q 和 N，其中 Q 是给定的报销额度，N（<=30）是发票张数。随后是 N 行输入，每行的格式为：
m Type\_1:price\_1 Type\_2:price\_2 ... Type\_m:price\_m
其中正整数 m 是这张发票上所开物品的件数，Type\_i 和 price\_i 是第 i 项物品的种类和价值。物品种类用一个大写英文字母表示。当N为0时，全部输入结束，相应的结果不要输出。
Output:对每个测试用例输出1行，即可以报销的最大数额，精确到小数点后2位。
SampleInput: 200.00 32 A:23.50 B:100.001 C:650.003 A:59.99 A:120.00 X:10.001200.00 22 B:600.00 A:400.001 C:200.501200.50 32 B:600.00 A:400.001 C:200.501 A:100.00100.00 0SampleOutput:123.501000.001200.50Title:E-FATEDescription:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
最近xhd正在玩一款叫做FATE的游戏，为了得到极品装备，xhd在不停的杀怪做任务。久而久之xhd开始对杀怪产生的厌恶感，但又不得不通过杀怪来升完这最后一级。现在的问题是，xhd升掉最后一级还需n的经验值，xhd还留有m的忍耐度，每杀一个怪xhd会得到相应的经验，并减掉相应的忍耐度。当忍耐度降到0或者0以下时，xhd就不会玩这游戏。xhd还说了他最多只杀s只怪。请问他能升掉这最后一级吗？
Input:输入数据有多组，对于每组数据第一行输入n，m，k，s(0 < n,m,k,s < 100)四个正整数。分别表示还需的经验值，保留的忍耐度，怪的种数和最多的杀怪数。接下来输入k行数据。每行数据输入两个正整数a，b(0 < a,b < 20)；分别表示杀掉一只这种怪xhd会得到的经验值和会减掉的忍耐度。(每种怪都有无数个)Output:输出升完这级还能保留的最大忍耐度，如果无法升完这级输出-1。SampleInput: 10 10 1 101 110 10 1 91 19 10 2 101 12 2SampleOutput:0-11Title:F-最少拦截系统Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } }); setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000); 某国为了防御敌国的导弹袭击,发展出一种导弹拦截系统.但是这种导弹拦截系统有一个缺陷:虽然它的第一发炮弹能够到达任意的高度,但是以后每一发炮弹都不能超过前一发的高度.某天,雷达捕捉到敌国的导弹来袭.由于该系统还在试用阶段,所以只有一套系统,因此有可能不能拦截所有的导弹. 怎么办呢?多搞几套系统呗!你说说倒蛮容易,成本呢?成本是个大问题啊.所以俺就到这里来求救了,请帮助计算一下最少需要多少套拦截系统. Input:输入若干组数据.每组数据包括:导弹总个数(正整数),导弹依此飞来的高度(雷达给出的高度数据是不大于30000的正整数,用空格分隔)Output:对应每组数据输出拦截所有导弹最少要配备多少套这种导弹拦截系统.SampleInput: 8 389 207 155 300 299 170 158 65SampleOutput:2Title:G-Common SubsequenceDescription: 　　郭哥与瑞瑞在玩一个游戏。
　　他们先各自写下一串字符，然后互相展示。展示过后，他们再从自己写的那串字符中依次挑出若干字符（保持原有顺序不变），组成新的一串。他们希望自己新组成的字符串与对方新组成的完全相同，并且尽可能长。
　　例如，郭哥写下abcde，瑞瑞写下aeiou，然后郭哥挑出自己那串里的第1和第5个字符组成新串ae，瑞瑞挑出自己那串中的第1、2个字符，也组成字符串ae。ae就是他们能共同挑出的最长串。
　　现在，郭哥和瑞瑞分别写出了自己的字符串，请帮他们算一下他们能共同挑出组成的字符串最长有多长。Input: 　　输入包含多组数据，处理至文件结尾。
　　每组数据占一行，包括以空格分隔的两个字符串，分别是郭哥和瑞瑞写下的字符串。两个字符串长度都在1000以内。Output: 　　对于每组输入，输出一个整数，即他们能共同挑出组成的字符串的最大长度。SampleInput: abcfbc abfcabprogramming contest abcd mnpSampleOutput:420Title:H-Eating TogetherDescription:FJ的奶牛们在吃晚饭时很傻。他们把自己组织成三组（方便编号为1, 2和3），坚持一起用餐。当他们在谷仓排队进入喂食区时，麻烦就开始了。每头奶牛都随身带着一张小卡片，小卡片上刻的是Di（1≤Di≤3）表示她属于哪一组。所有的N（1≤N≤30000）头奶牛排队吃饭，但他们并不能按卡片上的分组站好。FJ的工作并不是那么难。他只是沿着牛的路线走下去，把旧的号码标出来，换上一个新的。通过这样做，他创造了一群奶牛，比如111222333或333222111，奶牛的就餐组按他们的晚餐卡片按升序或降序排列。FJ就像任何人一样懒惰。他很好奇：怎样他才能进行适当的分组，使得他只要修改最少次数的数字？由于奶牛们已经很长时间没有吃到饭了，所以“哞哞”的声音到处都是，FJ只好更换卡号，而不能重新排列已经排好队的奶牛。Input:第1行：一个整数：n；第2~n+1行：第i-1行描述第i个奶牛目前分组。 Output:一个整数，表示必须做出的最小变化数，以便以升序或降序排序最终序列。 SampleInput: 513211SampleOutput:1Title:I-命运Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
穿过幽谷意味着离大魔王lemon已经无限接近了！
可谁能想到，yifenfei在斩杀了一些虾兵蟹将后，却再次面临命运大迷宫的考验，这是魔王lemon设下的又一个机关。要知道，不论何人，若在迷宫中被困1小时以上，则必死无疑！
可怜的yifenfei为了去救MM，义无返顾地跳进了迷宫。让我们一起帮帮执着的他吧！
命运大迷宫可以看成是一个两维的方格阵列，如下图所示：
yifenfei一开始在左上角，目的当然是到达右下角的大魔王所在地。迷宫的每一个格子都受到幸运女神眷恋或者痛苦魔王的诅咒，所以每个格子都对应一个值，走到那里便自动得到了对应的值。
现在规定yifenfei只能向右或者向下走，向下一次只能走一格。但是如果向右走，则每次可以走一格或者走到该行的列数是当前所在列数倍数的格子，即：如果当前格子是（x,y），下一步可以是（x+1,y），(x,y+1)或者(x,y\*k) 其中k>1。
为了能够最大把握的消灭魔王lemon，yifenfei希望能够在这个命运大迷宫中得到最大的幸运值。
Input:输入数据首先是一个整数C，表示测试数据的组数。
每组测试数据的第一行是两个整数n,m，分别表示行数和列数(1<=n<=20,10<=m<=1000)；
接着是n行数据，每行包含m个整数，表示n行m列的格子对应的幸运值K ( |k|<100 )。
Output:请对应每组测试数据输出一个整数，表示yifenfei可以得到的最大幸运值。SampleInput: 13 89 10 10 10 10 -10 10 1010 -11 -1 0 2 11 10 -20-11 -11 10 11 2 10 -10 -10SampleOutput:52Title:J-免费馅饼Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
都说天上不会掉馅饼，但有一天gameboy正走在回家的小径上，忽然天上掉下大把大把的馅饼。说来gameboy的人品实在是太好了，这馅饼别处都不掉，就掉落在他身旁的10米范围内。馅饼如果掉在了地上当然就不能吃了，所以gameboy马上卸下身上的背包去接。但由于小径两侧都不能站人，所以他只能在小径上接。由于gameboy平时老呆在房间里玩游戏，虽然在游戏中是个身手敏捷的高手，但在现实中运动神经特别迟钝，每秒种只有在移动不超过一米的范围内接住坠落的馅饼。现在给这条小径如图标上坐标：
为了使问题简化，假设在接下来的一段时间里，馅饼都掉落在0-10这11个位置。开始时gameboy站在5这个位置，因此在第一秒，他只能接到4,5,6这三个位置中其中一个位置上的馅饼。问gameboy最多可能接到多少个馅饼？（假设他的背包可以容纳无穷多个馅饼）
Input:输入数据有多组。每组数据的第一行为以正整数n(0<n<100000)，表示有n个馅饼掉在这条小径上。在结下来的n行中，每行有两个整数x,T(0<T<100000),表示在第T秒有一个馅饼掉在x点上。同一秒钟在同一点上可能掉下多个馅饼。n=0时输入结束。
Output:每一组输入数据对应一行输出。输出一个整数m，表示gameboy最多可能接到m个馅饼。
提示：本题的输入数据量比较大，建议用scanf读入，用cin可能会超时。
SampleInput: 65 14 16 17 27 28 30SampleOutput:4Title:K-Cow Roller CoasterDescription:The cows are building a roller coaster! They want your help to design as fun a roller coaster as possible, while keeping to the budget.The roller coaster will be built on a long linear stretch of land of length L (1 ≤ L ≤ 1,000). The roller coaster comprises a collection of some of the N (1 ≤ N ≤ 10,000) different interchangable components. Each component i has a fixed length Wi (1 ≤ Wi ≤ L). Due to varying terrain, each component i can be only built starting at location Xi (0 ≤ Xi ≤ L - Wi). The cows want to string together various roller coaster components starting at 0 and ending at L so that the end of each component (except the last) is the start of the next component.Each component i has a "fun rating" Fi (1 ≤ Fi ≤ 1,000,000) and a cost Ci (1 ≤ Ci ≤ 1000). The total fun of the roller coster is the sum of the fun from each component used; the total cost is likewise the sum of the costs of each component used. The cows' total budget is B (1 ≤ B ≤ 1000). Help the cows determine the most fun roller coaster that they can build with their budget.Input:Line 1: Three space-separated integers: L, N and B.
Lines 2..N+1: Line i+1 contains four space-separated integers, respectively: Xi, Wi, Fi, and Ci.Output:Line 1: A single integer that is the maximum fun value that a roller-coaster can have while staying within the budget and meeting all the other constraints. If it is not possible to build a roller-coaster within budget, output -1.SampleInput: 5 6 100 2 20 62 3 5 60 1 2 11 1 1 31 2 5 43 2 10 2SampleOutput:17Title:L-悼念512汶川大地震遇难同胞——珍惜现在，感恩生活Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
急！灾区的食物依然短缺！
为了挽救灾区同胞的生命，心系灾区同胞的你准备自己采购一些粮食支援灾区，现在假设你一共有资金n元，而市场有m种大米，每种大米都是袋装产品，其价格不等，并且只能整袋购买。
请问：你用有限的资金最多能采购多少公斤粮食呢？
后记：
人生是一个充满了变数的生命过程，天灾、人祸、病痛是我们生命历程中不可预知的威胁。
月有阴晴圆缺，人有旦夕祸福，未来对于我们而言是一个未知数。那么，我们要做的就应该是珍惜现在，感恩生活——
感谢父母，他们给予我们生命，抚养我们成人；
感谢老师，他们授给我们知识，教我们做人
感谢朋友，他们让我们感受到世界的温暖；
感谢对手，他们令我们不断进取、努力。
同样，我们也要感谢痛苦与艰辛带给我们的财富～
Input:输入数据首先包含一个正整数C，表示有C组测试用例，每组测试用例的第一行是两个整数n和m(1<=n<=100, 1<=m<=100),分别表示经费的金额和大米的种类，然后是m行数据，每行包含3个数p，h和c(1<=p<=20,1<=h<=200,1<=c<=20)，分别表示每袋的价格、每袋的重量以及对应种类大米的袋数。Output:对于每组测试数据，请输出能够购买大米的最多重量，你可以假设经费买不光所有的大米，并且经费你可以不用完。每个实例的输出占一行。SampleInput: 18 22 100 44 100 2SampleOutput:400

**三、程序代码（可附算法）**

Title: A-简单的整数划分问题Code: #include<iostream>#include<stack>#include<string>#include<cstring>#include<vector>using namespace std;int dp[105];int main(){ int n; dp[0] = 1; for(int i = 1; i <= 50; i++) for(int j = 1; j <= 50; j++){ if(j-i >= 0) dp[j] += dp[j-i]; } while(cin >> n){ cout << dp[n] << endl; } return 0;} Title: B-复杂的整数划分问题Code: #include <iostream>#include <cstring>using namespace std;#define N 51int dp1[N][N]; //N划分成K个正整数之和的划分数目。int dp2[N][N]; //N划分成若干个不同正整数之和的划分数目。int dp3[N][N]; //N划分成若干个可相同的正整数之和的划分数目。int f[N][N]; //N划分成K个奇正整数之和的划分数目。int g[N][N]; //N划分成K个偶正整数之和的划分数目。void initDivideInt() { memset(dp1, 0, sizeof(dp1)); //dp[n][k]=dp[n-k][k]+dp[n-1][k-1] memset(dp2, 0, sizeof(dp2)); //dp[n][m]=dp[n][m-1]+dp[n-m][m-1] memset(dp3, 0, sizeof(dp3)); //dp[n][m]=dp[n][m-1]+dp[n-m][m] for (int i = 1; i < N; i++) { for (int j = 1; j < N; j++) { if (i < j) { dp1[i][j] = 0; dp2[i][j] = dp2[i][i]; dp3[i][j] = dp3[i][i]; } else if (i == j) { dp1[i][j] = 1; dp2[i][j] = dp2[i][j - 1] + 1; dp3[i][j] = dp3[i][j - 1] + 1; } else { dp1[i][j] = dp1[i - j][j] + dp1[i - 1][j - 1]; dp2[i][j] = dp2[i][j - 1] + dp2[i - j][j - 1]; dp3[i][j] = dp3[i][j - 1] + dp3[i - j][j]; } } }}void initDivideOdd() { f[0][0] = 1; g[0][0] = 1; for (int i = 1; i < N; i++) { for (int j = 1; j <= i; j++) { g[i][j] = f[i - j][j]; f[i][j] = f[i - 1][j - 1] + g[i - j][j]; } }}int main() { int n, k; initDivideInt(); initDivideOdd(); while (cin >> n >> k) { cout << dp1[n][k] << endl; cout << dp2[n][n] << endl; int sum = 0; for (int i = 0; i <= n; i++) { sum += f[n][i]; } cout << sum << endl; } return 0;}Title: C-Coin ChangeCode: #include<stdio.h>int main(){ int a,b,c,d,e,count,n; while(scanf("%d",&n)!=EOF) { count=0; for(a=0;a<=n;a++) for(b=0;5\*b<=n-a;b++) for(c=0;10\*c<=n-a-5\*b;c++) for(d=0;25\*d<=n-a-5\*b-10\*c;d++) { e=n-a-5\*b-10\*c-25\*d; if(e%50==0&&a+b+c+d+e/50<=100)count++; } printf("%d\r
",count); }}Title: D-最大报销额Code: #include<iostream>#include<cstdio>#include<cstring>#include<algorithm>#include<cmath>#include<iomanip>using namespace std;int cost[35];int dp[3000300];int main(){ //freopen("in.txt","r",stdin); double q; int n; while(cin>>q>>n&&n){ memset(dp,0,sizeof(dp)); int index = 1; while(n--){ double t,sum = 0; double A = 0,B = 0,C = 0; bool flag = 0; int cnt; cin>>cnt; while(cnt--){ char type,pass; double money; cin>>type>>pass>>money; if(money > 600) flag = 1; if(type == 'A') A += money; else if(type == 'B' ) B += money; else if(type == 'C') C += money; else flag = 1; sum = A + B + C; if(sum > 1000 || A > 600 || B > 600 || C > 600) { flag = 1; } } if(!flag) cost[index++] = sum\*100; } // for(int i = 1; i < index; i++)// cout<<cost[i]<<" "; int qq = (int)(q\*100); for(int i = 1; i < index; i++) for(int j = qq; j >= cost[i]; j--) dp[j] = max(dp[j],dp[j-cost[i]]+cost[i]); cout<<fixed<<setprecision(2)<<dp[qq]/100.00<<endl; } return 0;} Title: E-FATECode: #include<stdio.h>#include<string.h>#include<algorithm>using namespace std;struct node{ int j,r;}mon[1010];int dp[1010][1010];int cmp(node x,node y){ return x.r<y.r;}int main(){ int n,m,a,b; while(scanf("%d%d%d%d",&n,&m,&a,&b)!=EOF)//经验 忍耐 种族 杀怪数 { memset(dp,0,sizeof(dp)); for(int i=1; i<=a; i++) scanf("%d%d",&mon[i].j,&mon[i].r); int maxx,flag=0; for(int i=1; i<=a; i++) { for(int j=1; j<=b; j++) { for(int k=mon[i].r; k<=m; k++) { dp[j][k]=max(dp[j][k],dp[j-1][k-mon[i].r]+mon[i].j); } } } for(int j=1;j<=b;j++) for(int i=0; i<=m; i++) if(dp[j][i]>=n) { flag=1; maxx=m-i; break; } if(flag==0) printf("-1\r
"); else printf("%d\r
",maxx); } return 0;}Title: F-最少拦截系统Code: #include<iostream>using namespace std;const int MAXN = 10000;int n,high[MAXN];int LIS(){ int ans = 1; int dp[MAXN]; dp[1] = 1; for(int i = 2; i <= n; i++){ int maxx = 0; for(int j = 1; j < i; j++) if(dp[j] > maxx && high[j] < high[i]) maxx = dp[j]; dp[i] = maxx + 1; if(dp[i] > ans) ans = dp[i]; } return ans;}int main(){ while(cin >> n){ for(int i = 1; i <= n; i++) cin >> high[i]; cout << LIS() << endl; } return 0;}Title: G-Common SubsequenceCode: #include<bits/stdc++.h>using namespace std;const int MAXN = 1000 + 5;char a[MAXN];char b[MAXN]; int dp[MAXN][MAXN];int main(){ while(cin >> a+1 >> b+1){ int len1 = strlen(a+1); int len2 = strlen(b+1); int maxx = max(len1, len2); for(int i = 0; i <= maxx; i++){ dp[i][0] = 0; dp[0][i] = 0; } for(int i = 1; i <= len1; i++) for(int j = 1; j <= len2; j++){ if(a[i] == b[j]) dp[i][j] = dp[i-1][j-1] + 1; else dp[i][j] = max(dp[i-1][j], dp[i][j-1]); } cout << dp[len1][len2] << endl; } return 0;}Title: H-Eating TogetherCode: #include<cstdio>#include<cstring>#include<iostream>#include<algorithm>#include<queue>#include<stack>#include<vector>#include<cstdlib> using namespace std;const int maxn = 300000;int A[maxn],f1[maxn],f2[maxn];int main(){ int n;scanf("%d",&n); for(int i = 1;i <= n;i++) scanf("%d",&A[i]); int cnt = 1;f1[cnt] = A[1]; for(int i = 2;i <= n;i++){ if(A[i] >= f1[cnt]){ f1[++cnt] = A[i]; }else \*upper\_bound(f1+1,f1+cnt+1,A[i]) = A[i]; } int ans = cnt; cnt = 1;f2[cnt] = A[1]; for(int i = 2;i <= n;i++){ if(A[i] <= f2[cnt]) f2[++cnt] = A[i]; else \*upper\_bound(f2+1,f2+1+cnt,A[i],greater<int>()) = A[i]; } printf("%d\r
",n-max(ans,cnt)); return 0;} Title: I-命运Code: #include<iostream>#include<algorithm>using namespace std;int n,m,C;const int MAXN = 1005;int dp[MAXN][MAXN];int a[MAXN][MAXN];int main(){ cin >> C; while(C--){ cin >> n >> m; for(int i = 1; i <= n; i++) for(int j = 1; j <= m; j++) cin >> a[i][j]; for(int i = 0; i <= n; i++) dp[i][0] = -1e8; for(int i = 0; i <= m; i++) dp[0][i] = -1e8; dp[1][1] = a[1][1]; for(int i = 1; i <= n; i++){ for(int j = 1; j <= m; j++){ if(i == 1 && j == 1) continue; dp[i][j] = max(dp[i-1][j], dp[i][j-1]) + a[i][j]; for(int k = 1; k < j; k ++){ if(j % k == 0) dp[i][j] = max(dp[i][j], dp[i][k] + a[i][j]); } } } cout << dp[n][m] << endl; } return 0;}Title: J-免费馅饼Code: #include<bits/stdc++.h>using namespace std;const int MAXN = 100000+5;//dp[i][j]:第i秒j位置的个数 int dp[MAXN][11];//int a[MAXN][11];int m,n;int maxx;int main(){ while(~scanf("%d",&n) && n){// for(int i = 0; i <= 10; i++)// for(int j = 1; j < MAXN; j++){// dp[j][i] = 0;// } memset(dp,0,sizeof(dp)); m = maxx = -1e3; for(int i = 1; i <= n; i++){ int x,y; scanf("%d %d",&x,&y); if(y == 1){ if(x == 4 || x == 5 || x == 6) dp[y][x]++; } else{ dp[y][x]++; } maxx = max(maxx, y); } // for(int i = 1; i <= maxx; i++){// for(int j = 4; j <=8 ;j++){// cout << i << "--" <<j<<":"<<dp[i][j]<<endl;// }// } for(int i = 2; i <= maxx; i++){ dp[i][0] = max(dp[i-1][1], dp[i-1][0]) + dp[i][0]; for(int j = 1; j <= 9; j++){ dp[i][j] = max(max(dp[i-1][j-1] ,dp[i-1][j]), dp[i-1][j+1]) + dp[i][j]; } dp[i][10] = max(dp[i-1][9], dp[i-1][10]) + dp[i][10]; } for(int i = 0; i <= 10; i++){ m = max(dp[maxx][i], m); } printf("%d\r
",m); } return 0;}Title: K-Cow Roller CoasterCode: #include<cstdio>#include<cstring>#include<iostream>#include<algorithm>using namespace std;const int maxn=10020;struct node{ int x; int w; int f; int c; bool operator <(const node &a)const { if(x==a.x) return w<a.w; return x<a.x; }}a[maxn];int l,n,b,dp[1001][1001];int main(){ while(scanf("%d%d%d",&l,&n,&b)!=EOF) { memset(dp,-1,sizeof(dp)); for(int i=0;i<n;i++) scanf("%d%d%d%d",&a[i].x,&a[i].w,&a[i].f,&a[i].c); sort(a,a+n); for(int i=0;i<=b;i++) dp[0][i]=0; for(int i=0;i<n;i++) for(int j=0;j<=b-a[i].c;j++) if(dp[a[i].x][j]!=-1) dp[a[i].x+a[i].w][j+a[i].c]=max(dp[a[i].x+a[i].w][j+a[i].c],dp[a[i].x][j]+a[i].f); printf("%d\r
",dp[l][b]); } return 0;}Title: L-悼念512汶川大地震遇难同胞——珍惜现在，感恩生活Code: #include<iostream>using namespace std;struct item{ int p; int h; int c;};item a[105];int dp[105];int main(){ int C,n,m; cin >> C; while(C--){ cin >> m >> n; for(int i = 0; i <= 104; i++) dp[i] = 0; for(int i = 0; i < n; i++) cin >> a[i].p >> a[i].h >> a[i].c; for(int i = 0; i < n; i++){ for(int j = 0; j < a[i].c; j++){ for(int k = m; k >= a[i].p; k--){ dp[k] = max(dp[k], dp[k-a[i].p]+a[i].h); } } } cout << dp[m] << endl; } return 0;}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

# 《算法设计与分析》实验报告2

实验名称：  **递归与分治（一）**

实验日期： 2021年3月26日-4月12日

实验地点：  A608

**一、实验目的**

**二、实验内容（编程题）**

Title:A-简单的整数划分问题Description: 将正整数n 表示成一系列正整数之和，n=n1+n2+…+nk, 其中n1>=n2>=…>=nk>=1 ，k>=1 。正整数n 的这种表示称为正整数n 的划分。正整数n 的不同的划分个数称为正整数n 的划分数。 Input: 标准的输入包含若干组测试数据。每组测试数据是一个整数N(0 < N <= 50)。 Output: 对于每组测试数据，输出N的划分数。 SampleInput: 5 SampleOutput: 7 Title:B-复杂的整数划分问题Description: 将正整数n 表示成一系列正整数之和，n=n1+n2+…+nk, 其中n1>=n2>=…>=nk>=1 ，k>=1 。正整数n 的这种表示称为正整数n 的划分。Input: 标准的输入包含若干组测试数据。每组测试数据是一行输入数据,包括两个整数N 和 K。 (0 < N <= 50, 0 < K <= N)Output: 对于每组测试数据，输出以下三行数据: 第一行: N划分成K个正整数之和的划分数目 第二行: N划分成若干个不同正整数之和的划分数目 第三行: N划分成若干个奇正整数之和的划分数目SampleInput: 5 2SampleOutput: 233Title:C-Coin ChangeDescription:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
Suppose there are 5 types of coins: 50-cent, 25-cent, 10-cent, 5-cent, and 1-cent. We want to make changes with these coins for a given amount of money.
For example, if we have 11 cents, then we can make changes with one 10-cent coin and one 1-cent coin, or two 5-cent coins and one 1-cent coin, or one 5-cent coin and six 1-cent coins, or eleven 1-cent coins. So there are four ways of making changes for 11 cents with the above coins. Note that we count that there is one way of making change for zero cent.
Write a program to find the total number of different ways of making changes for any amount of money in cents. Your program should be able to handle up to 100 coins.
Input:The input file contains any number of lines, each one consisting of a number ( ≤250 ) for the amount of money in cents.Output:For each input line, output a line containing the number of different ways of making changes with the above 5 types of coins.SampleInput: 1126SampleOutput:413Title:D-最大报销额Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
现有一笔经费可以报销一定额度的发票。允许报销的发票类型包括买图书（A类）、文具（B类）、差旅（C类），要求每张发票的总额不得超过1000元，每张发票上，单项物品的价值不得超过600元。现请你编写程序，在给出的一堆发票中找出可以报销的、不超过给定额度的最大报销额。
Input:测试输入包含若干测试用例。每个测试用例的第1行包含两个正数 Q 和 N，其中 Q 是给定的报销额度，N（<=30）是发票张数。随后是 N 行输入，每行的格式为：
m Type\_1:price\_1 Type\_2:price\_2 ... Type\_m:price\_m
其中正整数 m 是这张发票上所开物品的件数，Type\_i 和 price\_i 是第 i 项物品的种类和价值。物品种类用一个大写英文字母表示。当N为0时，全部输入结束，相应的结果不要输出。
Output:对每个测试用例输出1行，即可以报销的最大数额，精确到小数点后2位。
SampleInput: 200.00 32 A:23.50 B:100.001 C:650.003 A:59.99 A:120.00 X:10.001200.00 22 B:600.00 A:400.001 C:200.501200.50 32 B:600.00 A:400.001 C:200.501 A:100.00100.00 0SampleOutput:123.501000.001200.50Title:E-FATEDescription:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
最近xhd正在玩一款叫做FATE的游戏，为了得到极品装备，xhd在不停的杀怪做任务。久而久之xhd开始对杀怪产生的厌恶感，但又不得不通过杀怪来升完这最后一级。现在的问题是，xhd升掉最后一级还需n的经验值，xhd还留有m的忍耐度，每杀一个怪xhd会得到相应的经验，并减掉相应的忍耐度。当忍耐度降到0或者0以下时，xhd就不会玩这游戏。xhd还说了他最多只杀s只怪。请问他能升掉这最后一级吗？
Input:输入数据有多组，对于每组数据第一行输入n，m，k，s(0 < n,m,k,s < 100)四个正整数。分别表示还需的经验值，保留的忍耐度，怪的种数和最多的杀怪数。接下来输入k行数据。每行数据输入两个正整数a，b(0 < a,b < 20)；分别表示杀掉一只这种怪xhd会得到的经验值和会减掉的忍耐度。(每种怪都有无数个)Output:输出升完这级还能保留的最大忍耐度，如果无法升完这级输出-1。SampleInput: 10 10 1 101 110 10 1 91 19 10 2 101 12 2SampleOutput:0-11Title:F-最少拦截系统Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } }); setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000); 某国为了防御敌国的导弹袭击,发展出一种导弹拦截系统.但是这种导弹拦截系统有一个缺陷:虽然它的第一发炮弹能够到达任意的高度,但是以后每一发炮弹都不能超过前一发的高度.某天,雷达捕捉到敌国的导弹来袭.由于该系统还在试用阶段,所以只有一套系统,因此有可能不能拦截所有的导弹. 怎么办呢?多搞几套系统呗!你说说倒蛮容易,成本呢?成本是个大问题啊.所以俺就到这里来求救了,请帮助计算一下最少需要多少套拦截系统. Input:输入若干组数据.每组数据包括:导弹总个数(正整数),导弹依此飞来的高度(雷达给出的高度数据是不大于30000的正整数,用空格分隔)Output:对应每组数据输出拦截所有导弹最少要配备多少套这种导弹拦截系统.SampleInput: 8 389 207 155 300 299 170 158 65SampleOutput:2Title:G-Common SubsequenceDescription: 　　郭哥与瑞瑞在玩一个游戏。
　　他们先各自写下一串字符，然后互相展示。展示过后，他们再从自己写的那串字符中依次挑出若干字符（保持原有顺序不变），组成新的一串。他们希望自己新组成的字符串与对方新组成的完全相同，并且尽可能长。
　　例如，郭哥写下abcde，瑞瑞写下aeiou，然后郭哥挑出自己那串里的第1和第5个字符组成新串ae，瑞瑞挑出自己那串中的第1、2个字符，也组成字符串ae。ae就是他们能共同挑出的最长串。
　　现在，郭哥和瑞瑞分别写出了自己的字符串，请帮他们算一下他们能共同挑出组成的字符串最长有多长。Input: 　　输入包含多组数据，处理至文件结尾。
　　每组数据占一行，包括以空格分隔的两个字符串，分别是郭哥和瑞瑞写下的字符串。两个字符串长度都在1000以内。Output: 　　对于每组输入，输出一个整数，即他们能共同挑出组成的字符串的最大长度。SampleInput: abcfbc abfcabprogramming contest abcd mnpSampleOutput:420Title:H-Eating TogetherDescription:FJ的奶牛们在吃晚饭时很傻。他们把自己组织成三组（方便编号为1, 2和3），坚持一起用餐。当他们在谷仓排队进入喂食区时，麻烦就开始了。每头奶牛都随身带着一张小卡片，小卡片上刻的是Di（1≤Di≤3）表示她属于哪一组。所有的N（1≤N≤30000）头奶牛排队吃饭，但他们并不能按卡片上的分组站好。FJ的工作并不是那么难。他只是沿着牛的路线走下去，把旧的号码标出来，换上一个新的。通过这样做，他创造了一群奶牛，比如111222333或333222111，奶牛的就餐组按他们的晚餐卡片按升序或降序排列。FJ就像任何人一样懒惰。他很好奇：怎样他才能进行适当的分组，使得他只要修改最少次数的数字？由于奶牛们已经很长时间没有吃到饭了，所以“哞哞”的声音到处都是，FJ只好更换卡号，而不能重新排列已经排好队的奶牛。Input:第1行：一个整数：n；第2~n+1行：第i-1行描述第i个奶牛目前分组。 Output:一个整数，表示必须做出的最小变化数，以便以升序或降序排序最终序列。 SampleInput: 513211SampleOutput:1Title:I-命运Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
穿过幽谷意味着离大魔王lemon已经无限接近了！
可谁能想到，yifenfei在斩杀了一些虾兵蟹将后，却再次面临命运大迷宫的考验，这是魔王lemon设下的又一个机关。要知道，不论何人，若在迷宫中被困1小时以上，则必死无疑！
可怜的yifenfei为了去救MM，义无返顾地跳进了迷宫。让我们一起帮帮执着的他吧！
命运大迷宫可以看成是一个两维的方格阵列，如下图所示：
yifenfei一开始在左上角，目的当然是到达右下角的大魔王所在地。迷宫的每一个格子都受到幸运女神眷恋或者痛苦魔王的诅咒，所以每个格子都对应一个值，走到那里便自动得到了对应的值。
现在规定yifenfei只能向右或者向下走，向下一次只能走一格。但是如果向右走，则每次可以走一格或者走到该行的列数是当前所在列数倍数的格子，即：如果当前格子是（x,y），下一步可以是（x+1,y），(x,y+1)或者(x,y\*k) 其中k>1。
为了能够最大把握的消灭魔王lemon，yifenfei希望能够在这个命运大迷宫中得到最大的幸运值。
Input:输入数据首先是一个整数C，表示测试数据的组数。
每组测试数据的第一行是两个整数n,m，分别表示行数和列数(1<=n<=20,10<=m<=1000)；
接着是n行数据，每行包含m个整数，表示n行m列的格子对应的幸运值K ( |k|<100 )。
Output:请对应每组测试数据输出一个整数，表示yifenfei可以得到的最大幸运值。SampleInput: 13 89 10 10 10 10 -10 10 1010 -11 -1 0 2 11 10 -20-11 -11 10 11 2 10 -10 -10SampleOutput:52Title:J-免费馅饼Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
都说天上不会掉馅饼，但有一天gameboy正走在回家的小径上，忽然天上掉下大把大把的馅饼。说来gameboy的人品实在是太好了，这馅饼别处都不掉，就掉落在他身旁的10米范围内。馅饼如果掉在了地上当然就不能吃了，所以gameboy马上卸下身上的背包去接。但由于小径两侧都不能站人，所以他只能在小径上接。由于gameboy平时老呆在房间里玩游戏，虽然在游戏中是个身手敏捷的高手，但在现实中运动神经特别迟钝，每秒种只有在移动不超过一米的范围内接住坠落的馅饼。现在给这条小径如图标上坐标：
为了使问题简化，假设在接下来的一段时间里，馅饼都掉落在0-10这11个位置。开始时gameboy站在5这个位置，因此在第一秒，他只能接到4,5,6这三个位置中其中一个位置上的馅饼。问gameboy最多可能接到多少个馅饼？（假设他的背包可以容纳无穷多个馅饼）
Input:输入数据有多组。每组数据的第一行为以正整数n(0<n<100000)，表示有n个馅饼掉在这条小径上。在结下来的n行中，每行有两个整数x,T(0<T<100000),表示在第T秒有一个馅饼掉在x点上。同一秒钟在同一点上可能掉下多个馅饼。n=0时输入结束。
Output:每一组输入数据对应一行输出。输出一个整数m，表示gameboy最多可能接到m个馅饼。
提示：本题的输入数据量比较大，建议用scanf读入，用cin可能会超时。
SampleInput: 65 14 16 17 27 28 30SampleOutput:4Title:K-Cow Roller CoasterDescription:The cows are building a roller coaster! They want your help to design as fun a roller coaster as possible, while keeping to the budget.The roller coaster will be built on a long linear stretch of land of length L (1 ≤ L ≤ 1,000). The roller coaster comprises a collection of some of the N (1 ≤ N ≤ 10,000) different interchangable components. Each component i has a fixed length Wi (1 ≤ Wi ≤ L). Due to varying terrain, each component i can be only built starting at location Xi (0 ≤ Xi ≤ L - Wi). The cows want to string together various roller coaster components starting at 0 and ending at L so that the end of each component (except the last) is the start of the next component.Each component i has a "fun rating" Fi (1 ≤ Fi ≤ 1,000,000) and a cost Ci (1 ≤ Ci ≤ 1000). The total fun of the roller coster is the sum of the fun from each component used; the total cost is likewise the sum of the costs of each component used. The cows' total budget is B (1 ≤ B ≤ 1000). Help the cows determine the most fun roller coaster that they can build with their budget.Input:Line 1: Three space-separated integers: L, N and B.
Lines 2..N+1: Line i+1 contains four space-separated integers, respectively: Xi, Wi, Fi, and Ci.Output:Line 1: A single integer that is the maximum fun value that a roller-coaster can have while staying within the budget and meeting all the other constraints. If it is not possible to build a roller-coaster within budget, output -1.SampleInput: 5 6 100 2 20 62 3 5 60 1 2 11 1 1 31 2 5 43 2 10 2SampleOutput:17Title:L-悼念512汶川大地震遇难同胞——珍惜现在，感恩生活Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
急！灾区的食物依然短缺！
为了挽救灾区同胞的生命，心系灾区同胞的你准备自己采购一些粮食支援灾区，现在假设你一共有资金n元，而市场有m种大米，每种大米都是袋装产品，其价格不等，并且只能整袋购买。
请问：你用有限的资金最多能采购多少公斤粮食呢？
后记：
人生是一个充满了变数的生命过程，天灾、人祸、病痛是我们生命历程中不可预知的威胁。
月有阴晴圆缺，人有旦夕祸福，未来对于我们而言是一个未知数。那么，我们要做的就应该是珍惜现在，感恩生活——
感谢父母，他们给予我们生命，抚养我们成人；
感谢老师，他们授给我们知识，教我们做人
感谢朋友，他们让我们感受到世界的温暖；
感谢对手，他们令我们不断进取、努力。
同样，我们也要感谢痛苦与艰辛带给我们的财富～
Input:输入数据首先包含一个正整数C，表示有C组测试用例，每组测试用例的第一行是两个整数n和m(1<=n<=100, 1<=m<=100),分别表示经费的金额和大米的种类，然后是m行数据，每行包含3个数p，h和c(1<=p<=20,1<=h<=200,1<=c<=20)，分别表示每袋的价格、每袋的重量以及对应种类大米的袋数。Output:对于每组测试数据，请输出能够购买大米的最多重量，你可以假设经费买不光所有的大米，并且经费你可以不用完。每个实例的输出占一行。SampleInput: 18 22 100 44 100 2SampleOutput:400Title:A-踩方格Description: 有一个方格矩阵，矩阵边界在无穷远处。我们做如下假设：a.    每走一步时，只能从当前方格移动一格，走到某个相邻的方格上；b.    走过的格子立即塌陷无法再走第二次；c.    只能向北、东、西三个方向走；请问：如果允许在方格矩阵上走n步，共有多少种不同的方案。2种走法只要有一步不一样，即被认为是不同的方案。 Input: 允许在方格上行走的步数n(n <= 20) Output: 计算出的方案数量 SampleInput: 2 SampleOutput: 7 Title:B-棋盘问题Description:在一个给定形状的棋盘（形状可能是不规则的）上面摆放棋子，棋子没有区别。要求摆放时任意的两个棋子不能放在棋盘中的同一行或者同一列，请编程求解对于给定形状和大小的棋盘，摆放k个棋子的所有可行的摆放方案C。Input:输入含有多组测试数据。
每组数据的第一行是两个正整数，n k，用一个空格隔开，表示了将在一个n\*n的矩阵内描述棋盘，以及摆放棋子的数目。 n <= 8 , k <= n
当为-1 -1时表示输入结束。
随后的n行描述了棋盘的形状：每行有n个字符，其中 # 表示棋盘区域， . 表示空白区域（数据保证不出现多余的空白行或者空白列）。
Output:对于每一组数据，给出一行输出，输出摆放的方案数目C （数据保证C<2^31）。SampleInput: 2 1#..#4 4...#..#..#..#...-1 -1SampleOutput:21Title:C-迷宫问题Description: 定义一个二维数组： int maze[5][5] = {0, 1, 0, 0, 0,0, 1, 0, 1, 0,0, 0, 0, 0, 0,0, 1, 1, 1, 0,0, 0, 0, 1, 0,}; 它表示一个迷宫，其中的1表示墙壁，0表示可以走的路，只能横着走或竖着走，不能斜着走，要求编程序找出从左上角到右下角的最短路线。 Input: 一个5 × 5的二维数组，表示一个迷宫。数据保证有唯一解。Output: 左上角到右下角的最短路径，格式如样例所示。SampleInput: 0 1 0 0 00 1 0 1 00 0 0 0 00 1 1 1 00 0 0 1 0SampleOutput: (0, 0)(1, 0)(2, 0)(2, 1)(2, 2)(2, 3)(2, 4)(3, 4)(4, 4)Title:D-Dungeon MasterDescription:(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)(https://imgchr.com/i/BuF0OO)花花被困在一个三维的空间中,现在要寻找最短路径逃生！空间由立方体单位构成。花花每次向上下前后左右移动一个单位需要一分钟，且花花不能对角线移动。空间的四周封闭。花花的目标是走到空间的出口。是否存在逃出生天的可能性？如果存在，则需要多少时间？Input:每个空间的描述的第一行为L，R和C（皆不超过30）。L表示空间的高度，R和C分别表示每层空间的行与列的大小。随后L层，每层R行，每行C个字符。每个字符表示空间的一个单元。'#'表示不可通过单元，'.'表示空白单元。花花的起始位置在'S'，出口为'E'。每层空间后都有一个空行。L，R和C均为0时输入结束。Output:每个空间对应一行输出。如果可以逃生，则输出如下Escaped in x minute(s).x为最短脱离时间。如果无法逃生，则输出如下Trapped!SampleInput: 3 4 5S.....###..##..###.#############.####...###########.#######E1 3 3S###E####0 0 0SampleOutput:Escaped in 11 minute(s).Trapped!Title:E-FliptileDescription:农夫约翰知道，一头知识上满意的母牛是一头快乐的母牛，它将提供更多的牛奶。他为母牛安排了脑力活动，他们在其中操纵M × N 网格（1≤ M ≤15； 1≤ N ≤15）个正方形瓷砖，每个瓷砖的一面都涂成黑色，另一面则涂成白色。 正如人们所猜测的那样，当翻转单个白色瓷砖时，它会改变变黑翻转单个黑色图块时，它会变为白色。母牛在翻转砖块时会得到奖励，因此每块砖块的白色侧面都朝上。但是，母牛的蹄子相当大，当他们尝试翻转特定的砖块时，它们也会翻转所有相邻的砖块（与翻转后的砖块共享整个边缘的砖块）。由于翻转很累，奶牛希望尽量减少必须做的翻转次数。 帮助母牛确定所需的最小翻转次数，并确定要达到最小翻转次数的位置。如果有多种方法可以以最少的翻转次数来完成任务，则当将其视为字符串时，以输出中词典顺序最少的方式返回。如果无法完成任务，请用“ IMPOSSIBLE”一词打印一行。Input:第1行：两个以空格分隔的整数： M 和 N 第2 .. M +1行：第 i +1行描述了网格中第i行（ N 以空格分隔的整数，黑色为1，白色为0。Output:第1 .. M 行：每行包含 N 个以空格分隔的整数，每个整数指定翻转该特定位置的次数。SampleInput: 4 41 0 0 10 1 1 00 1 1 01 0 0 1SampleOutput:0 0 0 01 0 0 11 0 0 10 0 0 0Title:F-Find The MultipleDescription: Given a positive integer n, write a program to find out a nonzero multiple m of n whose decimal representation contains only the digits 0 and 1. You may assume that n is not greater than 200 and there is a corresponding m containing no more than 100 decimal digits.Input: The input file may contain multiple test cases. Each line contains a value of n (1 <= n <= 200). A line containing a zero terminates the input.Output: For each value of n in the input print a line containing the corresponding value of m. The decimal representation of m must not contain more than 100 digits. If there are multiple solutions for a given value of n, any one of them is acceptable.SampleInput: 26190SampleOutput: 10100100100100100100111111111111111111Title:G-Prime PathDescription:正如前文所说，eroengine突然喜爱上了质数，他对质数的爱已经到了魔怔的地步。现在他想到了一个小游戏，首先随机挑选两个四位的素数a，b。游戏规则是：a可以通过改变某一位上的数字使其变成c，但只有当c也是四位的素数时才能进行这种改变。eroengine只擅长抛出问题而不擅长解决问题，但他觉得你一定可以很轻松的算出a最少经过多少次变化使其变为b。例如：1033 -> 8179 1033 1733 3733 3739 3779 8779 8179最少变换了6次。Input:第一行输入整数T，表示样例数。 (T 每个样例输入两个四位的素数a，b。(没有前导零) Output:对于每个样例，输出最少变换次数，如果无法变换成b则输出"Impossible"。SampleInput: 31033 81791373 80171033 1033SampleOutput:670Title:H-Oil DepositsDescription: GeoSurvComp地质调查公司负责探测地下石油储藏。 GeoSurvComp现在在一块矩形区域探测石油，并把这个大区域分成了很多小块。他们通过专业设备，来分析每个小块中是否蕴藏石油。如果这些蕴藏石油的小方格相邻，那么他们被认为是同一油藏的一部分。在这块矩形区域，可能有很多油藏。你的任务是确定有多少不同的油藏。Input: 输入可能有多个矩形区域（即可能有多组测试）。每个矩形区域的起始行包含m和n，表示行和列的数量，1<=n,m<=100，如果m =0表示输入的结束，接下来是m行，每行n个字符。每个字符对应一个小方格，并且要么是'\*'，代表没有油，要么是'@'，表示有油。Output: 对于每一个矩形区域，输出油藏的数量。两个小方格是相邻的，当且仅当他们水平或者垂直或者对角线相邻（即8个方向）。SampleInput:   1 1 \* 3 5 \*@\*@\* \*\*@\*\* \*@\*@\* 1 8 @@\*\*\*\*@\* 5 5  \*\*\*\*@ \*@@\*@ \*@\*\*@ @@@\*@ @@\*\*@ 0 0    SampleOutput: 0 1 2 2Title:I-非常可乐Description:MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } });
setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);
大家一定觉的运动以后喝可乐是一件很惬意的事情，但是seeyou却不这么认为。因为每次当seeyou买了可乐以后，阿牛就要求和seeyou一起分享这一瓶可乐，而且一定要喝的和seeyou一样多。但seeyou的手中只有两个杯子，它们的容量分别是N 毫升和M 毫升 可乐的体积为S （S<101）毫升　(正好装满一瓶) ，它们三个之间可以相互倒可乐 (都是没有刻度的，且 S==N+M，101＞S＞0，N＞0，M＞0) 。聪明的ACMER你们说他们能平分吗？如果能请输出倒可乐的最少的次数，如果不能输出"NO"。
Input:三个整数 : S 可乐的体积 , N 和 M是两个杯子的容量，以"0 0 0"结束。Output:如果能平分的话请输出最少要倒的次数，否则输出"NO"。SampleInput: 7 4 34 1 30 0 0SampleOutput:NO3Title:J-Find a wayDescription:伊菲的家在乡下，而梅基的家在市中心。于是，伊菲和梅基安排在KFC见面。西安有很多KFC，他们想选择一个花费总时间最短的KFC见面。现在给你一张西安的地图，伊菲和梅基都可以上、下、左、右移动到相邻的地点，每移动一个位置花费11分钟。Input:输入包含多个测试用例。每个测试用例包括前两个整数n，m.（2<=n，m<=200）。接下来的n行，每行包含m个字符。“Y”表示伊菲的初始位置。“M”表示梅基初始位置。“#”死路；'.'可走的路。“@” KCFOutput:对于每个测试用例，输出伊菲和梅基花费的最短总时间。总有一家KFC可以让他们见面。SampleInput: 4 4Y.#@.....#..@..M4 4Y.#@.....#..@#.M5 5Y..@..#....#...@..M.#...#SampleOutput:668866

**三、程序代码（可附算法）**

Title: A-简单的整数划分问题Code: #include<iostream>#include<stack>#include<string>#include<cstring>#include<vector>using namespace std;int dp[105];int main(){ int n; dp[0] = 1; for(int i = 1; i <= 50; i++) for(int j = 1; j <= 50; j++){ if(j-i >= 0) dp[j] += dp[j-i]; } while(cin >> n){ cout << dp[n] << endl; } return 0;} Title: B-复杂的整数划分问题Code: #include <iostream>#include <cstring>using namespace std;#define N 51int dp1[N][N]; //N划分成K个正整数之和的划分数目。int dp2[N][N]; //N划分成若干个不同正整数之和的划分数目。int dp3[N][N]; //N划分成若干个可相同的正整数之和的划分数目。int f[N][N]; //N划分成K个奇正整数之和的划分数目。int g[N][N]; //N划分成K个偶正整数之和的划分数目。void initDivideInt() { memset(dp1, 0, sizeof(dp1)); //dp[n][k]=dp[n-k][k]+dp[n-1][k-1] memset(dp2, 0, sizeof(dp2)); //dp[n][m]=dp[n][m-1]+dp[n-m][m-1] memset(dp3, 0, sizeof(dp3)); //dp[n][m]=dp[n][m-1]+dp[n-m][m] for (int i = 1; i < N; i++) { for (int j = 1; j < N; j++) { if (i < j) { dp1[i][j] = 0; dp2[i][j] = dp2[i][i]; dp3[i][j] = dp3[i][i]; } else if (i == j) { dp1[i][j] = 1; dp2[i][j] = dp2[i][j - 1] + 1; dp3[i][j] = dp3[i][j - 1] + 1; } else { dp1[i][j] = dp1[i - j][j] + dp1[i - 1][j - 1]; dp2[i][j] = dp2[i][j - 1] + dp2[i - j][j - 1]; dp3[i][j] = dp3[i][j - 1] + dp3[i - j][j]; } } }}void initDivideOdd() { f[0][0] = 1; g[0][0] = 1; for (int i = 1; i < N; i++) { for (int j = 1; j <= i; j++) { g[i][j] = f[i - j][j]; f[i][j] = f[i - 1][j - 1] + g[i - j][j]; } }}int main() { int n, k; initDivideInt(); initDivideOdd(); while (cin >> n >> k) { cout << dp1[n][k] << endl; cout << dp2[n][n] << endl; int sum = 0; for (int i = 0; i <= n; i++) { sum += f[n][i]; } cout << sum << endl; } return 0;}Title: C-Coin ChangeCode: #include<stdio.h>int main(){ int a,b,c,d,e,count,n; while(scanf("%d",&n)!=EOF) { count=0; for(a=0;a<=n;a++) for(b=0;5\*b<=n-a;b++) for(c=0;10\*c<=n-a-5\*b;c++) for(d=0;25\*d<=n-a-5\*b-10\*c;d++) { e=n-a-5\*b-10\*c-25\*d; if(e%50==0&&a+b+c+d+e/50<=100)count++; } printf("%d\r
",count); }}Title: D-最大报销额Code: #include<iostream>#include<cstdio>#include<cstring>#include<algorithm>#include<cmath>#include<iomanip>using namespace std;int cost[35];int dp[3000300];int main(){ //freopen("in.txt","r",stdin); double q; int n; while(cin>>q>>n&&n){ memset(dp,0,sizeof(dp)); int index = 1; while(n--){ double t,sum = 0; double A = 0,B = 0,C = 0; bool flag = 0; int cnt; cin>>cnt; while(cnt--){ char type,pass; double money; cin>>type>>pass>>money; if(money > 600) flag = 1; if(type == 'A') A += money; else if(type == 'B' ) B += money; else if(type == 'C') C += money; else flag = 1; sum = A + B + C; if(sum > 1000 || A > 600 || B > 600 || C > 600) { flag = 1; } } if(!flag) cost[index++] = sum\*100; } // for(int i = 1; i < index; i++)// cout<<cost[i]<<" "; int qq = (int)(q\*100); for(int i = 1; i < index; i++) for(int j = qq; j >= cost[i]; j--) dp[j] = max(dp[j],dp[j-cost[i]]+cost[i]); cout<<fixed<<setprecision(2)<<dp[qq]/100.00<<endl; } return 0;} Title: E-FATECode: #include<stdio.h>#include<string.h>#include<algorithm>using namespace std;struct node{ int j,r;}mon[1010];int dp[1010][1010];int cmp(node x,node y){ return x.r<y.r;}int main(){ int n,m,a,b; while(scanf("%d%d%d%d",&n,&m,&a,&b)!=EOF)//经验 忍耐 种族 杀怪数 { memset(dp,0,sizeof(dp)); for(int i=1; i<=a; i++) scanf("%d%d",&mon[i].j,&mon[i].r); int maxx,flag=0; for(int i=1; i<=a; i++) { for(int j=1; j<=b; j++) { for(int k=mon[i].r; k<=m; k++) { dp[j][k]=max(dp[j][k],dp[j-1][k-mon[i].r]+mon[i].j); } } } for(int j=1;j<=b;j++) for(int i=0; i<=m; i++) if(dp[j][i]>=n) { flag=1; maxx=m-i; break; } if(flag==0) printf("-1\r
"); else printf("%d\r
",maxx); } return 0;}Title: F-最少拦截系统Code: #include<iostream>using namespace std;const int MAXN = 10000;int n,high[MAXN];int LIS(){ int ans = 1; int dp[MAXN]; dp[1] = 1; for(int i = 2; i <= n; i++){ int maxx = 0; for(int j = 1; j < i; j++) if(dp[j] > maxx && high[j] < high[i]) maxx = dp[j]; dp[i] = maxx + 1; if(dp[i] > ans) ans = dp[i]; } return ans;}int main(){ while(cin >> n){ for(int i = 1; i <= n; i++) cin >> high[i]; cout << LIS() << endl; } return 0;}Title: G-Common SubsequenceCode: #include<bits/stdc++.h>using namespace std;const int MAXN = 1000 + 5;char a[MAXN];char b[MAXN]; int dp[MAXN][MAXN];int main(){ while(cin >> a+1 >> b+1){ int len1 = strlen(a+1); int len2 = strlen(b+1); int maxx = max(len1, len2); for(int i = 0; i <= maxx; i++){ dp[i][0] = 0; dp[0][i] = 0; } for(int i = 1; i <= len1; i++) for(int j = 1; j <= len2; j++){ if(a[i] == b[j]) dp[i][j] = dp[i-1][j-1] + 1; else dp[i][j] = max(dp[i-1][j], dp[i][j-1]); } cout << dp[len1][len2] << endl; } return 0;}Title: H-Eating TogetherCode: #include<cstdio>#include<cstring>#include<iostream>#include<algorithm>#include<queue>#include<stack>#include<vector>#include<cstdlib> using namespace std;const int maxn = 300000;int A[maxn],f1[maxn],f2[maxn];int main(){ int n;scanf("%d",&n); for(int i = 1;i <= n;i++) scanf("%d",&A[i]); int cnt = 1;f1[cnt] = A[1]; for(int i = 2;i <= n;i++){ if(A[i] >= f1[cnt]){ f1[++cnt] = A[i]; }else \*upper\_bound(f1+1,f1+cnt+1,A[i]) = A[i]; } int ans = cnt; cnt = 1;f2[cnt] = A[1]; for(int i = 2;i <= n;i++){ if(A[i] <= f2[cnt]) f2[++cnt] = A[i]; else \*upper\_bound(f2+1,f2+1+cnt,A[i],greater<int>()) = A[i]; } printf("%d\r
",n-max(ans,cnt)); return 0;} Title: I-命运Code: #include<iostream>#include<algorithm>using namespace std;int n,m,C;const int MAXN = 1005;int dp[MAXN][MAXN];int a[MAXN][MAXN];int main(){ cin >> C; while(C--){ cin >> n >> m; for(int i = 1; i <= n; i++) for(int j = 1; j <= m; j++) cin >> a[i][j]; for(int i = 0; i <= n; i++) dp[i][0] = -1e8; for(int i = 0; i <= m; i++) dp[0][i] = -1e8; dp[1][1] = a[1][1]; for(int i = 1; i <= n; i++){ for(int j = 1; j <= m; j++){ if(i == 1 && j == 1) continue; dp[i][j] = max(dp[i-1][j], dp[i][j-1]) + a[i][j]; for(int k = 1; k < j; k ++){ if(j % k == 0) dp[i][j] = max(dp[i][j], dp[i][k] + a[i][j]); } } } cout << dp[n][m] << endl; } return 0;}Title: J-免费馅饼Code: #include<bits/stdc++.h>using namespace std;const int MAXN = 100000+5;//dp[i][j]:第i秒j位置的个数 int dp[MAXN][11];//int a[MAXN][11];int m,n;int maxx;int main(){ while(~scanf("%d",&n) && n){// for(int i = 0; i <= 10; i++)// for(int j = 1; j < MAXN; j++){// dp[j][i] = 0;// } memset(dp,0,sizeof(dp)); m = maxx = -1e3; for(int i = 1; i <= n; i++){ int x,y; scanf("%d %d",&x,&y); if(y == 1){ if(x == 4 || x == 5 || x == 6) dp[y][x]++; } else{ dp[y][x]++; } maxx = max(maxx, y); } // for(int i = 1; i <= maxx; i++){// for(int j = 4; j <=8 ;j++){// cout << i << "--" <<j<<":"<<dp[i][j]<<endl;// }// } for(int i = 2; i <= maxx; i++){ dp[i][0] = max(dp[i-1][1], dp[i-1][0]) + dp[i][0]; for(int j = 1; j <= 9; j++){ dp[i][j] = max(max(dp[i-1][j-1] ,dp[i-1][j]), dp[i-1][j+1]) + dp[i][j]; } dp[i][10] = max(dp[i-1][9], dp[i-1][10]) + dp[i][10]; } for(int i = 0; i <= 10; i++){ m = max(dp[maxx][i], m); } printf("%d\r
",m); } return 0;}Title: K-Cow Roller CoasterCode: #include<cstdio>#include<cstring>#include<iostream>#include<algorithm>using namespace std;const int maxn=10020;struct node{ int x; int w; int f; int c; bool operator <(const node &a)const { if(x==a.x) return w<a.w; return x<a.x; }}a[maxn];int l,n,b,dp[1001][1001];int main(){ while(scanf("%d%d%d",&l,&n,&b)!=EOF) { memset(dp,-1,sizeof(dp)); for(int i=0;i<n;i++) scanf("%d%d%d%d",&a[i].x,&a[i].w,&a[i].f,&a[i].c); sort(a,a+n); for(int i=0;i<=b;i++) dp[0][i]=0; for(int i=0;i<n;i++) for(int j=0;j<=b-a[i].c;j++) if(dp[a[i].x][j]!=-1) dp[a[i].x+a[i].w][j+a[i].c]=max(dp[a[i].x+a[i].w][j+a[i].c],dp[a[i].x][j]+a[i].f); printf("%d\r
",dp[l][b]); } return 0;}Title: L-悼念512汶川大地震遇难同胞——珍惜现在，感恩生活Code: #include<iostream>using namespace std;struct item{ int p; int h; int c;};item a[105];int dp[105];int main(){ int C,n,m; cin >> C; while(C--){ cin >> m >> n; for(int i = 0; i <= 104; i++) dp[i] = 0; for(int i = 0; i < n; i++) cin >> a[i].p >> a[i].h >> a[i].c; for(int i = 0; i < n; i++){ for(int j = 0; j < a[i].c; j++){ for(int k = m; k >= a[i].p; k--){ dp[k] = max(dp[k], dp[k-a[i].p]+a[i].h); } } } cout << dp[m] << endl; } return 0;}Title: A-踩方格Code: #include<iostream>#include<cstring>#include<cstdio>using namespace std;int dp[25][3]; int main(){ int n; dp[1][0] = 1; dp[1][1] = 1; dp[1][2] = 1; for(int i = 2; i <= 20; i++){ dp[i][0] += dp[i-1][0]; dp[i][0] += dp[i-1][1]; dp[i][1] += dp[i-1][0]; dp[i][1] += dp[i-1][1]; dp[i][1] += dp[i-1][2]; dp[i][2] += dp[i-1][1]; dp[i][2] += dp[i-1][2]; } cin >> n; cout << dp[n][0]+dp[n][1]+dp[n][2] << endl; return 0;}Title: B-棋盘问题Code: #include<iostream>using namespace std;const int MAXN = 15;int a[MAXN][MAXN];int vis[MAXN];int n,k,ans;void dfs(int step, int tot){ if(tot==0){ ans++; return; } //this if mast under ^ if if(step==n+1) return; for(int i = 1; i <= n; i++){ if(a[step][i] == 1 && !vis[i]){ vis[i] = 1; dfs(step+1,tot-1); vis[i] = 0; } } //line step no put dfs(step+1,tot);}int main(){ while(cin >> n >> k){ if(n == -1 || k == -1) break; ans = 0; for(int i = 1; i <= n; i++) for(int j = 1; j <= n; j++){ char t; cin >> t; t=='#'?a[i][j]=1:a[i][j]=0; } dfs(1,k); cout << ans << endl; } return 0;}Title: C-迷宫问题Code: #include<iostream>#include<cstring>#include<cstdio>using namespace std;bool map[6][6];bool book[6][6];int nextt[4][2] = {{0,1},{1,0},{0,-1},{-1,0}};typedef struct step{ int x; int y; int f; } step;step que[36];void pp(int now){ //printf("now:%d\r
",now); if(now==1) return; now = que[now].f; pp(now); printf("(%d, %d)\r
",que[now].x-1,que[now].y-1); }int main(){ int head = 1,tail = 1; for(int i = 1; i <= 5; i++) for(int j = 1; j <= 5; j++) scanf("%d",&map[i][j]); que[tail].x = 1; que[tail].y = 1; que[tail].f = 0; tail++; book[1][1] = 1; bool flag = 0; int tx,ty; while(head < tail){ for(int i = 0; i < 3; i++){ tx = que[head].x + nextt[i][0]; ty = que[head].y + nextt[i][1]; if(tx < 1||tx > 5||ty < 1||ty > 5) continue; if(map[tx][ty] == 0 && book[tx][ty] == 0){ book[tx][ty] = 1; que[tail].x = tx; que[tail].y = ty; que[tail].f = head; tail++; } if(tx == 5&&ty == 5){ flag = 1; break; } } if(flag) break; head++; } // for(int i = 1; i <= 36; i++){// printf("x:%d y:%d f:%d\r
",que[i].x,que[i].y,que[i].f);// } pp(tail-1); printf("(4, 4)\r
"); return 0;}Title: D-Dungeon MasterCode: #include<iostream>#include<cstring>#include<cstdio>using namespace std;int l,r,c;char map[35][35][35];bool book[35][35][35];int fx[6] = {-1,1,0,0,0,0};int fy[6] = {0,0,-1,1,0,0};int fz[6] = {0,0,0,0,1,-1};typedef struct step{ int x; int y; int z; int s;} step;step que[43000];step s,e;//血的教训，变量名千万不能粗心写错了，一个bug改一天；//地图不要忘记置0 //坐标不能搞错了，又改了几个小时int main(){ while(cin>>l>>r>>c&&l&&r&&c){ memset(book,0,sizeof(book)); for(int i = 1; i <= l; i++){ for(int j = 1; j <= r; j++){ for(int k = 1; k <= c; k++){ cin>>map[j][k][i]; if(map[j][k][i] == 'S'){ s.x = j; s.y = k; s.z = i; //这里的坐标气死我也 } else if(map[j][k][i] == 'E'){ e.x = j; e.y = k; e.z = i; } map[j][k][i] = map[j][k][i]==35; } } } // // for(int i = 1; i <= l; i++){// for(int j = 1; j <= r; j++){// for(int k = 1; k <= c; k++){// cout<<(int)map[j][k][i]<<" ";// }// cout<<endl;// }// } int head = 1,tail = 1; que[tail].x = s.x; que[tail].y = s.y; que[tail].z = s.z; que[tail].s = 0; tail++; book[s.x][s.y][s.z] = 1; bool flag = 0; int tx,ty,tz; while(head < tail){ for(int i = 0; i < 6; i++){ tx = que[head].x + fx[i]; ty = que[head].y + fy[i]; tz = que[head].z + fz[i]; if(tx < 1||tx > r||ty < 1||ty > c||tz < 1||tz >l) continue; if(map[tx][ty][tz] == 0 && book[tx][ty][tz] == 0){ book[tx][ty][tz] = 1; //这里的坐标气我+1 que[tail].x = tx; que[tail].y = ty; que[tail].z = tz; que[tail].s = que[head].s + 1; tail++; } if(tx == e.x&&ty == e.y&&tz == e.z){ cout<<"Escaped in "<<que[tail-1].s<<" minute(s)."<<endl; flag = 1; break; } } if(flag) break; head++; } if(!flag) cout<<"Trapped!"<<endl; } return 0;}Title: E-FliptileCode: #include<iostream>#include<cstring>#include<cstdio>using namespace std;int l,r,c;char map[35][35][35];bool book[35][35][35];int fx[6] = {-1,1,0,0,0,0};int fy[6] = {0,0,-1,1,0,0};int fz[6] = {0,0,0,0,1,-1};typedef struct step{ int x; int y; int z; int s;} step;step que[43000];step s,e;//血的教训，变量名千万不能粗心写错了，一个bug改一天；//地图不要忘记置0 //坐标不能搞错了，又改了几个小时int main(){ while(cin>>l>>r>>c&&l&&r&&c){ memset(book,0,sizeof(book)); for(int i = 1; i <= l; i++){ for(int j = 1; j <= r; j++){ for(int k = 1; k <= c; k++){ cin>>map[j][k][i]; if(map[j][k][i] == 'S'){ s.x = j; s.y = k; s.z = i; //这里的坐标气死我也 } else if(map[j][k][i] == 'E'){ e.x = j; e.y = k; e.z = i; } map[j][k][i] = map[j][k][i]==35; } } } // // for(int i = 1; i <= l; i++){// for(int j = 1; j <= r; j++){// for(int k = 1; k <= c; k++){// cout<<(int)map[j][k][i]<<" ";// }// cout<<endl;// }// } int head = 1,tail = 1; que[tail].x = s.x; que[tail].y = s.y; que[tail].z = s.z; que[tail].s = 0; tail++; book[s.x][s.y][s.z] = 1; bool flag = 0; int tx,ty,tz; while(head < tail){ for(int i = 0; i < 6; i++){ tx = que[head].x + fx[i]; ty = que[head].y + fy[i]; tz = que[head].z + fz[i]; if(tx < 1||tx > r||ty < 1||ty > c||tz < 1||tz >l) continue; if(map[tx][ty][tz] == 0 && book[tx][ty][tz] == 0){ book[tx][ty][tz] = 1; //这里的坐标气我+1 que[tail].x = tx; que[tail].y = ty; que[tail].z = tz; que[tail].s = que[head].s + 1; tail++; } if(tx == e.x&&ty == e.y&&tz == e.z){ cout<<"Escaped in "<<que[tail-1].s<<" minute(s)."<<endl; flag = 1; break; } } if(flag) break; head++; } if(!flag) cout<<"Trapped!"<<endl; } return 0;}Title: F-Find The MultipleCode: #include<iostream>#include<stdio.h>#include<queue>#include<string.h>#include<algorithm>#include<string>#include<stack>using namespace std;typedef long long ll;int n;void bfs(ll x){ queue<ll> Q; Q.push(1) ; while(!Q.empty()) { ll u = Q.front() ; Q.pop() ; if(u%n == 0) { cout<<u<<endl; return ; } Q.push(u\*10); Q.push(u\*10+1); } return ;} int main(){ while(cin>>n) { if(n==0) { break; } bfs(1); } }Title: G-Prime PathCode: #include <iostream>#include <cstdio>#include <cmath>#include <cstring>#include <algorithm>#include <string>#include <map>#include <set>#include <vector>#include <queue>#include <stack>#define inf 100000000using namespace std;typedef long long ll;int n, z, t; // n表示输入的第一个数，z表示输入的第二个数，t表示案例数；bool prime[10000]; //存素数表；int p[10000]; //记录变化次数，bfs过程用；int bfs(){ queue<int> q; q.push(n); p[n]=0; //标记n; while(!q.empty()) { int pa=q.front(); q.pop(); if(pa==z) //判断是否已经搜索到 z 了； return p[z]; int d[4]; d[0]=(pa/10)\*10; //pa去个位数的值； d[1]=pa%10+(pa/100)\*100; //pa去十位数的值； d[2]=(pa/1000)\*1000+pa%100; //pa去百位数的值； d[3]=pa%1000; //pa去千位数的值； int d2=1; for(int i=0;i<4;i++) { int j=0; if(i==3) //千位不为0处理； j=1; for(;j<10;j++) { int w=d[i]+j\*d2; if(w>=1000&&w<10000&&prime[w]==0&&p[w]==-1) { p[w]=p[pa]+1; if(w==z) return p[z]; q.push(w); } } d2=d2\*10; } } return -1;}int main(){ prime[0]=1; prime[1]=1; for(int i=2;i\*i<10000;i++) //打素数表，表中0表示素数； { if(!prime[i]) { for(int j=i\*2;j<10000;j=j+i) { prime[j]=1; } } } scanf("%d",&t); for(int o=0;o<t;o++) { memset(p,-1,sizeof(p)); //对p初始化； int za=0; //表示变化次数，输出用； scanf("%d %d",&n,&z); za=bfs(); if(za==-1) //没搜索到； { printf("Impossible\r
"); } else //搜索到了； { printf("%d\r
",za); } } return 0;}Title: H-Oil DepositsCode: #include<iostream>#include<cstring>#include<cstdio>using namespace std;bool map[101][101];bool book[101][101];int nextt[8][2] = {{0,1},{1,1},{1,0},{1,-1},{0,-1},{-1,-1},{-1,0},{-1,1}};int m,n,count;void dfs(int x, int y){ book[x][y] = 1; for(int i = 0; i < 8; i++){ int tx = x + nextt[i][0]; int ty = y + nextt[i][1]; if(tx<1 || tx>m || ty<1 || ty > n) continue; if(book[tx][ty] == 1 || map[tx][ty] == 1) continue; book[tx][ty] = 1; dfs(tx,ty); } return; }int main(){ while(scanf("%d %d",&m,&n)!=EOF&&m){ memset(book,0,sizeof(book)); char c;getchar();count = 0; for(int i = 1; i <= m; i++){ for(int j = 1; j <= n; j++){ c = getchar(); map[i][j] = c!=64; } getchar(); } for(int i = 1; i <= m; i++){ for(int j = 1; j <= n; j++){ if(map[i][j]==0&&!book[i][j]){ count++;dfs(i,j); } } } printf("%d\r
",count); } return 0;}Title: I-非常可乐Code: #include <bits/stdc++.h>using namespace std;int main(){ int a,b,c; while(cin>>a>>b>>c&&(a&&b&&c)) { a /= \_\_gcd(b,c); if(a&1) cout<<"NO"<<endl; else cout<<a-1<<endl; } return 0;}Title: J-Find a wayCode: #include<bits/stdc++.h>using namespace std;const int MAXN = 205;int n,m;int a[MAXN][MAXN];int vis[MAXN][MAXN];int vis\_t[MAXN][MAXN];int ans[MAXN][MAXN];int nxy[4][2] = {{-1, 0}, {1, 0}, {0, -1}, {0, 1}};struct node{ int x; int y; int step;};int isIn(int x, int y){ if(x > n || x < 1) return 0; if(y > m || y < 1) return 0; return 1;}int bfs(node dot, int flag){ for(int i = 1; i <= n; i++) for(int j = 1; j <= m; j++) vis\_t[i][j] = vis[i][j]; queue<node> que; dot.step = 0; que.push(dot); int step = 0; while(!que.empty()){ node t = que.front();// cout << "t:" << t.x <<" "<<t.y<<endl; if(a[t.x][t.y] == '@'){ if(flag == 1) ans[t.x][t.y] = 0; ans[t.x][t.y] += t.step;// ma[make\_pair(t.x, t.y)] += t.step;// cout <<"string:"<<t.step<<endl;// ma[ans] += step; } int flag = 0; for(int i = 0; i < 4; i++){ int tx = t.x + nxy[i][0]; int ty = t.y + nxy[i][1]; if(isIn(tx,ty)&& !vis\_t[tx][ty]){// cout <<"tx:"<<tx<<" ty:"<<ty<<" "<<step<<endl; step++; vis\_t[tx][ty] = 1; node needIn; needIn.x = tx; needIn.y = ty; needIn.step = t.step+1; que.push(needIn); } } que.pop(); }}int main(){ node yf, mj; int len = 1; while(cin >> n >> m){ memset(vis, 0, sizeof(vis)); for(int i = 1; i <= n; i++) for(int j = 1; j <= m; j++) ans[i][j] = 1e8; len = 1; for(int i = 1; i <= n; i++){ for(int j = 1; j <= m; j++){ char t; cin >> t; if(t == 'Y'){ yf.x = i; yf.y = j; vis[i][j] = 1; }else if(t == 'M'){ mj.x = i; mj.y = j; vis[i][j] = 1; }else if(t == '#'){ vis[i][j] = 1; } a[i][j] = t; } } a[yf.x][yf.y] = '#'; a[mj.x][mj.y] = '#';// cout << endl;// for(int i = 1; i <= n; i++){// for(int j = 1; j <= m; j++){// cout << ans[i][j] << " ";// }// cout << endl;// } bfs(yf,1);// cout << endl;// for(int i = 1; i <= n; i++){// for(int j = 1; j <= m; j++){// cout << ans[i][j] << " ";// }// cout << endl;// } bfs(mj,0);// cout << endl;// for(int i = 1; i <= n; i++){// for(int j = 1; j <= m; j++){// cout << ans[i][j] << " ";// }// cout << endl;// } int minn = 1e8; for(int i = 1; i <= n; i++) for(int j = 1; j <= m; j++){ minn = min(minn, ans[i][j]); } cout << minn\*11 << endl; } return 0;}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

# 《算法设计与分析》实验报告3

实验名称：  **递归与分治（二）**

实验日期： 2021年4月11日-4月30日

实验地点： A608

**一、实验目的**

**二、实验内容（编程题）**

{{content}}

**三、程序代码（可附算法）**

{{code}}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

# 《算法设计与分析》实验报告4

实验名称： **贪心算法**

实验日期： 2021年5月6日-5月24日

实验地点： A608

**一、实验目的**

**二、实验内容（编程题）**

{{content}}

**三、程序代码（可附算法）**

{{code}}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

# 《算法设计与分析》实验报告5

实验名称：  **动态规划**

实验日期： 2021年5月25日-6月15日

实验地点： A608

**一、实验目的**

1．

2．

3．

**二、实验内容（编程题）**

{{content}}

**三、程序代码（可附算法）**

{{code}}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

# 《算法设计与分析》实验报告6

实验名称：  **搜索算法**

实验日期：

实验地点： A608

**一、实验目的**

1．

2．

3．

**二、实验内容（编程题）**

{{content}}

**三、程序代码（可附算法）**

{{code}}

**四、实验结果（抓上机运行结果图）及心得体会**

**要求：运行结果图上的标题包含个人学号姓名信息。**

**五、教师评价**

**成 绩：（必填）**

**指导教师（签名）：（必填）**

**（封底页）**