**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）**

{{content0}}

**三、程序代码（可附算法）**

{{code0}}

**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-踩方格Description: <p>有一个方格矩阵，矩阵边界在无穷远处。我们做如下假设：<br>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；<br>b.    走过的格子立即塌陷无法再走第二次；<br>c.    只能向北、东、西三个方向走；<br>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。<br></p> 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: <pre>2</pre> SampleOutput: <pre>7</pre> Title:B-棋盘问题Description:<div class="ptx" lang="en-US">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。</div>Input:<div class="ptx" lang="en-US">输入含有多组测试数据。
<br>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
<br>当为-1 -1时表示输入结束。
<br>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）。
<br></div>Output:<div class="ptx" lang="en-US">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）。</div>SampleInput: <pre class="sio">2 1#..#4 4...#..#..#..#...-1 -1</pre>SampleOutput:<pre class="sio">21</pre>Title:C-迷宫问题Description: <p><span style="font-family: 'Times New Roman', Times, serif">定义一个二维数组： </span><br style="font-family: 'Times New Roman', Times, serif; white-space: normal;"></p> <pre style="overflow-x: auto;">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,};</pre> <p><br style="font-family: 'Times New Roman', Times, serif; white-space: normal;"><span style="font-family: 'Times New Roman', Times, serif">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。</span></p> <p><br></p>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: <pre>0 1 0 0 00 1 0 1 00 0 0 0 00 1 1 1 00 0 0 1 0</pre>SampleOutput: <pre>(0, 0)(1, 0)(2, 0)(2, 1)(2, 2)(2, 3)(2, 4)(3, 4)(4, 4)</pre>Title:D-Dungeon MasterDescription:[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](https://imgchr.com/i/BuF0OO)[![BuF0OO.jpg](https://s1.ax1x.com/2020/10/26/BuF0OO.jpg)](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:<div class="ptx" lang="en-US"><p>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。</ p> <p>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。</ p> <p>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。</p></div>Input:<div class="ptx" lang="en-US">第1行：两个以空格分隔的整数： M 和 N <br>第2 .. M +1行：第 i +1行描述了网格中第i行（ N </ ）的颜色（从左到右） i>以空格分隔的整数，黑色为1，白色为0。</div>Output:<div class="ptx" lang="en-US">第1 .. <i> M </ i>行：每行包含<i> N </ i>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。</div>SampleInput: <pre class="sio">4 41 0 0 10 1 1 00 1 1 01 0 0 1</pre>SampleOutput:<pre class="sio">0 0 0 01 0 0 11 0 0 10 0 0 0</pre>Title: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: <pre>26190</pre>SampleOutput: <pre>10100100100100100100111111111111111111</pre>Title: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。<br>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。<br>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。<br>例如：1033 -> 8179 <br>1033 <br>1733 <br>3733 <br>3739 <br>3779 <br>8779 <br>8179<br>最少变换了6次。<br>Input:第一行输入整数T，表示样例数。 (T <= 100) <br>每个样例输入两个四位的素数a，b。(没有前导零) <br>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: <pre class="sio">31033 81791373 80171033 1033</pre>SampleOutput:<pre class="sio">670</pre>Title:H-Oil DepositsDescription:<p> <span><span><span>GeoSurvComp地质调查公司负责探测地下石油储藏。</span></span></span> 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。</p>Input:<p> 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<=<span><span><span>100，如果m</span></span></span> =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'\*'<span><span><span>，代表没有油，要么是'@'，表示有油。</span></span></span></p>Output:<p> 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）。</p>SampleInput: <div> <div>  </div> <div> <div> 1 1 </div> </div> <div>\* </div> <div>3 5 </div> <div>\*@\*@\* </div> <div>\*\*@\*\* </div> <div>\*@\*@\* </div> <div>1 8 </div> <div>@@\*\*\*\*@\* </div> <div>5 5  </div> <div>\*\*\*\*@ </div> <div>\*@@\*@ </div> <div>\*@\*\*@ </div> <div>@@@\*@ </div> <div>@@\*\*@ </div> <div>0 0  </div> <div>  </div> </div>SampleOutput:<pre></pre> <div> <div>0 </div> </div> <div> 1</div> <div> 2</div> <div> 2</div>Title:I-非常可乐Description:<script type='text/x-mathjax-config'>MathJax.Hub.Config({tex2jax: { inlineMath: [['$','$']] } }); </script>
<script type='text/javascript' src='https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.1/MathJax.js?config=TeX-AMS-MML\_HTMLorMML'></script>
<script type='text/javascript'>setTimeout(function(){MathJax.Hub.Queue(['Typeset', MathJax.Hub, 'left\_view']);}, 2000);</script>
<div class="panel\_content">
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"。
</div>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: <pre>7 4 34 1 30 0 0</pre>SampleOutput:<pre>NO3</pre>Title: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-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）**

{{content2}}

**三、程序代码（可附算法）**

{{code2}}

**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）**

{{content3}}

**三、程序代码（可附算法）**

{{code2}}

**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）**

{{content4}}

**三、程序代码（可附算法）**

{{code4}}

**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）**

{{content5}}

**三、程序代码（可附算法）**

{{code5}}

**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）**