第一页

问题1。房东  
“房东”是中国最流行的纸牌游戏之一。地主的目标是成为第一  
以有效的组合出他所有的牌，如果农民中有一人比地主先出所有的牌，他们就赢了。这个游戏使用54张牌包，包括两个小丑，红和黑。卡片的排列顺序由高到低:红色小丑，黑色小丑，2,A, K, Q, J,10 9 8 7 6 5 4 3在这场游戏中，西装无关紧要。我们对这个游戏做了一些简化。该表2描述了所有有效的组合。请注意，两个小丑不被视为一对，和一个四分之一的两个小丑是不允许的。你每次出的牌必须是有效的组合。你需要打出所有牌的最少次数是多少?  
输入描述  
在第一行你会收到两个整数T和n，用一个空格隔开。表示回合数。N表示每轮的牌数。下面每一行包含n个整数，用一个空格隔开。每个整数代表一张牌，如表1所示。小丑最多出现两次。除小丑外，每张牌最多出现4次。保证n≤17,T≤10。

输出描述  
你需要输出T行。每条线都有一个整数表示游戏的最小次数  
所有的卡片。

第3页  
吃豆人是一个经典的电子游戏发布于1980年。信不信由你，你已经学会了自己制作这款游戏所需的一切!  
这是一个吃豆人游戏的截图。如下图所示，游戏的几个组成部分是吃豆人、墙壁、食物、能量胶囊和幽灵。  
caDS0le  
◇得分  
我们的“吃豆人”版本没有这样的图形。令人失望的是，它由所有ASCII字符组成。吃豆人用相似的“C”表示，“#”代表“墙”，“·”代表食物，“”代表胶囊，“@”代表幽灵。(边界由所提供的模板打印，所以您不需要考虑。)  
得分:27  
剩下100种食物!  
吃豆人想要食物!(通过w / a / s / d / i控制，通过Enter确认)  
除了代码模板，我们还提供了用于Windows、MacOS和Linux的编译可执行文件。我们建议您运行可执行文件并体验游戏，以便对应该做什么有一个基本的概念。此外，当您对所编写的任何部分有任何疑问时，如果您的行为与示例程序相匹配，它(几乎)肯定会说您是安全的。  
。 在MacOS或Linux环境下，为了运行所提供的可执行文件，可能需要先在终端上执行“chmod + x Pacman MacOS”或“chmod + x PacmanLinux”命令。如果有任何问题，请到Piazza上提问。  
与实时视频游戏吃豆人不同，我们逐帧控制吃豆人。游戏会  
暂停每一帧并等待您的输入。你可以在游戏中输入“w / a / s / d”来移动吃豆人，  
第四页  
或者“I”(空闲)，让它静止不动。按“Enter”确认输入，游戏会显示下一帧。  
一个组成部分。从零开始制作游戏  
游戏由结构游戏结构运行。我们已经指定了它的一些组成部分:  
1typedef struct game {  
Char \*\*网格;  
/显示游戏的二维字符数组;int行;  
//网格的行数;int列;  
//网格的列数;int foodCount;  
/游戏中剩余食物的数量;int分数;  
/当前分数;游戏状态;1/游戏的状态，输赢、进行中或获胜。  
8}游戏;  
如果您愿意，可以随意向这个结构添加更多组件。  
尽管看起来只有一款“全局”游戏，或者游戏组件对于吃豆人(或幽灵)来说是“全局的”，但为它们声明全局变量并不是一个明智的选择。如果你选择,例如,写“(或鬼魂在B部分)在一个单独的结构,和想要吃豆子(鬼魂)了解游戏,一个很好的选择是存储游戏,一个游戏指针,作为一个成员的结构。你会了解更多关于这样的约定,当你练习——面向对象编程(oOP)在C +十。除了结构之外，您还需要编写许多函数来操作这个游戏。当在模板中实现所提供的函数原型时，您应该遵循下面的说明，或者查看注释。你不能修改函数名，或者添加/删除参数。你如果您喜欢，还可以添加更多的功能。我们不会检查任何没有提供的功能。  
由给定的行和列组成的游戏是通过调用game \* NewGame函数创建的

**第五页  
当游戏结束时，函数void EndGame (game \* game)被自动调用。在这个函数中，你应该释放任何你动态分配的内存，比如网格。在那之后，你应该自由  
参数游戏，因为它也是动态创建的。  
Wals, foods和Pacman是通过AddWal1, AddFood和AddPacman函数添加到游戏中的。艾尔这些游戏组件只能添加到一个空的单元格。特别是，如果游戏中已经有吃豆人，就不能添加吃豆人。添加任何项之后，您应该修改网格中的  
游戏结构，以确保其正确显示。  
最后，你可以写函数void MovePacman(游戏\*游戏，方向方向)来控制你的吃豆人。Direction是一个{up, down, left, right, idle}的枚举。移动你的规则《吃豆人》如下:在空闲时，吃豆人将保持静止。  
。 如果吃豆人将移动到一个空的cel，吃豆人将成功这么做;  
。 如果吃豆人将移动到食物cel，吃豆人将移动到它并吃的食物。你的分数将增加F00D分数=10。如果吃豆人吃完最后一份食物，你就赢了。你应该在  
这场比赛的状态是胜利。  
。 如果吃豆人撞到墙或边界，吃豆人会保持静止。  
在上述任何情况下，你的分数都应该在你玩了一个回合后减少1。如何玩这个游戏:你可以通过调用NewGame函数在main()函数中创建你的自定义游戏。之后，你可以添加墙壁和食物到任何特定的位置。不要忘记在游戏中添加吃豆人。  
当你的游戏准备好了，你可以调用提供的PlayGame函数。当你赢或输时，PlayGame将通过调用EndGame终止。  
如果你的游戏运行……恭喜你!你现在有一个“完整的”吃豆人游戏，在那里你可以移动，吃食物，并获胜!你可以向OJ提交A部分，但如果你想知道为什么你的游戏有点不同和无聊，让我们看看B部分。  
提交指南：当您提交代码时，您的主函数将被OJ上的一个替换。实现的函数将直接使用参数调用。因此，你绝对不能修改函数名，添加或删除参数，或者把你的代码放在main函数中。否则，你将得不到任何分数。**

**第六页**  
B部分。鬼!  
你的游戏缺少了最有趣的部分——幽灵。在这部分中，你将在游戏中添加幽灵和能量胶囊，这样游戏就会变得更具可玩性。  
对于如何为幽灵和胶囊存储数据，我们没有强加任何限制。你觉得你需要写一个结构吗，尤其是关于鬼魂的?如果是，你需要在里面存储什么?  
您的设计可以采用任何您喜欢的方式(但仍然尽量不要使用全局变量)，只要它满足  
要求如下:  
幽灵的要求:  
1. 最多有30个幽灵。  
2.通过bool AddGhost (game \* game, int r, int c, Direction Direction)函数将幽灵添加到游戏中。  
这一功能略有不同，因为幽灵可以添加到带有食物或胶囊的细胞上。鬼魂  
覆盖食物和胶囊展示，所以他们的细胞(最初&apos;。&apos;或&apos; o°)将显示在&apos; @&apos;中。然而，这些食物或胶囊必须仍然存在，当幽灵离开他们的细胞时，应该再次展示。  
。 方向定义了鬼如何移动。幽灵在水平线或垂直线上移动。  
这个函数中的参数方向是鬼的初始方向。  
3.幽灵被函数void MoveGhosts (Game \* Game)移动。  
。 这个函数将把游戏中的所有幽灵移动到它们自己的方向上。  
幽灵应该按照添加的顺序移动。  
。 如果幽灵带着食物或胶囊移动到一个细胞上，它就会在屏幕上覆盖食物或胶囊。&apos;或&apos; o)将显示在&apos; @&apos;中。然而，食物或胶囊必须仍然存在，并应再次显示时，这一幽灵离开该cel。  
如果一个鬼撞到墙，另一个鬼，或边界，它的方向就会反转，它会立即朝新的方向移动。如果那样的话，它就会撞到另一个  
墙/鬼/边界，它将停止并且不会在这个转弯中移动，它的方向会颠倒。  
为了更好地解释幽灵撞到另一个幽灵的情况，我们来看看下面的情况:  
先创建，再移动

第七页

现在有可能输掉这场比赛。根据规则，吃豆人总是先行动。如果吃豆人直接撞到鬼，吃豆人将移动到该细胞，并被杀死。你应该将游戏状态标记为失败。如果幽灵下方有食物或胶囊，吃豆人就不能吃它。如果吃豆人移动到一个鬼也试图移动的单元格，吃豆人将执行一个成功的移动，然后幽灵移动到吃豆人的单元格  
细胞。你也会输掉比赛。  
胶囊的要求:  
1. 胶囊是给吃豆人超级能力的大食物。因此，胶囊的数量  
游戏中的食物。吃豆人必须吃掉所有的食物和胶囊才能获胜。  
2. 胶囊是由函数bool AddCapsule (Game \* Game, int r, int c)添加的。像食物一样，胶囊不能添加到有墙的cel、吃豆人或幽灵上。但是，可以添加一个胶囊  
一个带有食物的cel，导致食物被升级到一个胶囊。  
3.当吃豆人吃下一个胶囊时，你的分数将增加胶囊\_分数=50，吃豆人将获得超能力的下一个胶囊持续时间=10个oves。它的超级力量是:  
． 所有鬼都会被吓到，他们的显示从“@”变成“X”。当吃豆人的超能力失效时，他们就会变回他们可爱的邪恶面孔。”