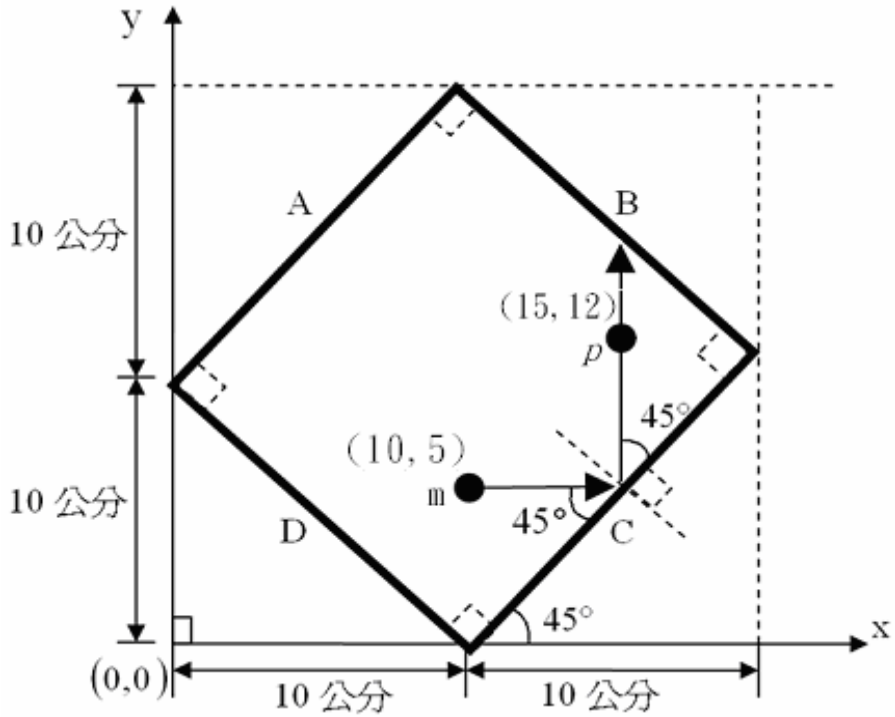
台北市九十三學年度高級中學
資訊學科能力競賽　程式設計實作測驗卷

說明：
1. 作答時間 3 小時。請經常做備份存檔到你的硬碟或磁片中，以免因斷電或故障而致資料毀損。
2. 本程式題目共有 5 題，每題 20 分，滿分為 100 分。
3. 本測驗所有題目，一律使用鍵盤做為輸入的設備，一律使用螢幕做為輸出的設備。
4. 本測驗所有題目，允許之執行時間均為 60 秒。超過 60 秒以上才輸出答案者不予計分。
5. 本測驗你所作答之所有題目之原始程式碼均必須儲存到主辦單位所準備之三吋半磁片中，以做為評審之參考。若未存入磁片中，則該題不計分。

第一題：分子的運動問題

問題敘述

分子運動是物理學中一個無法用肉眼觀察但卻可以用數學分析的有趣問題。本題假設有一個分子，m，被放在一個正方形的框子中，只以二維平面的方向移動。給定分子有一個初始運動速度為 6 公分/每秒，即使在碰到框子的邊緣反射後，分子的運動速度都不會改變。其中分子的大小與重量我們忽略不考慮。當分子碰到框子的邊緣時會依入射時的相同角度反射出去，並且繼續以 6 公分/每秒的速度移動，如下圖所示。



假設正方形框子以上圖的方式放在二維平面座標系中，框子的四個邊分別稱之為 A、B、C、D，而這四個邊的厚度我們忽略不考慮。其中 C 邊與水平座標 x 軸成 45° 夾角(角度單位：度)，因此框子的每一邊長均為($\frac{10}{\sin 45°}$)公分。其中 C 邊與 D 邊的交點為(10,0)。一開始時，分子 m 的座標(x,y)為(10,5)（座標長度單位：公分)；運動速度為 6 公分/每秒；在第 0 秒時的行進方向為往右，與水平座標 x 軸平行。請問在第 t 秒時，分子 m 所在的座標為何?

舉例來說：當輸入 t = 2 時，輸出結果為圖中 p 點的座標(15,12)。

程式存檔

請將此題解題之原始程式碼儲存到主辦單位所準備之三吋半磁片之檔案 prog_1 中，即名稱為"A:\prog_1.c"、 "A:\prog_1.pas"等。

條件限制

1. t 為介於 0 至 20000 間之整數。
2. 在任意時間中，分子 m 的座標位置(x,y)一定落在正方形的框子內。

輸入格式

一律使用鍵盤輸入。輸入的資料為 t。

輸出格式

一律使用螢幕輸出。請輸出在第 t 秒時分子 m 的座標(x,y)。

範例

輸入範例一	**輸入範例二**	**輸入範例三**	**輸入範例四**
0	2	12	20000
輸出範例一	**輸出範例二**	**輸出範例三**	**輸出範例四**
(10,5)	(15,12)	(5,8)	(10,5)

第二題：Burrows-Wheeler 轉換

問題敘述

許多壓縮法則並不會直接壓縮輸入的訊息，他們通常在執行壓縮前會對輸入的訊息做一些前置處理，能讓此訊息有效率地壓縮。以下說明一種前置處理的方式，稱之為 Burrows-Wheeler (BW)轉換。

假設輸入一個字串，S，其包含的字母個數為 N，則 BW 轉換的第一個步驟便是依據下列的兩個式子另外產生 N 個新字串，分別為 $S_1, S_2, ..., S_N$ 。

1. 字串 S_1 等於 S；
2. 字串 S_k 向左位移一個字母，第一個字母放到最後一個位置，即可得字串 S_{k+1}，其中 $1 \le k \le N$。

以下為 BW 轉換的第一個步驟之實例，假設輸入一個字串 S="WHEELER"，其中字母個數 $N = 7$，則產生的新字串如下所示：

列＼欄		1	2	3	4	5	6	7
1	S_1 =	W	H	E	E	L	E	R
2	S_2 =	H	E	E	L	E	R	W
3	S_3 =	E	E	L	E	R	W	H
4	S_4 =	E	L	E	R	W	H	E
5	S_5 =	L	E	R	W	H	E	E
6	S_6 =	E	R	W	H	E	E	L
7	S_7 =	R	W	H	E	E	L	E

在 BW 轉換之第二個步驟中我們將這 N 個新字串排序。首先依據每個字串之第一個字母(從最左邊算起)排序，字母在排序時的大小順序為 A > B > C > … > Z。若有某些字串第一個字母相同，則這些字串間的排序依照第二個字母。若仍相同，則依照第三個字母，依此類推到第 N 個字母為止。承續第一步驟的實例，其排序結果如下所示：

列 \ 欄		1	2	3	4	5	6	7
1	S_3 =	E	E	L	E	R	W	H
2	S_4 =	E	L	E	R	W	H	E
3	S_6 =	E	R	W	H	E	E	L
4	S_2 =	H	E	E	L	E	R	W
5	S_5 =	L	E	R	W	H	E	E
6	S_7 =	R	W	H	E	E	L	E
7	S_1 =	W	H	E	E	L	E	R

其中 S_3 、S_4 及 S_6 字串中的第一個字母皆相同，因此這三個字串的排序順序由第二個字母比較而得。

在 BW 轉換完成後，我們需要的結果為兩個：

1. 排序後第 N 欄形成的字串；即為依序抓出每列的最後一個字母形成需要的字串，所以在本例中最後一欄形成的字串為 "HELWEER"；
2. S_2 所在的列位置；在本例中為 4（列數從 1 開始計算，非從 0 開始）。

本題即是請你撰寫上述 Burrows-Wheeler 轉換的程式。

程式存檔

請將此題解題之原始程式碼儲存到主辦單位所準備之三吋半磁片之檔案 prog_2 中，即名稱為"A:\prog_2.c"、 "A:\prog_2.pas"等。

條件限制

輸入字串及產生的新字串中僅限於大寫的英文字母。

輸入格式

1. 一律使用鍵盤輸入。
2. 輸入兩行，第一行為字串的字母個數 N ，$2 \le N \le 20$；第二行為字串 S。

輸出格式

輸出答案共有兩行，第一行為第二步驟後的第 N 欄字母，第二行為第二步驟後 S_2 所在的列位置。注意列數從 1 開始計算，非從 0 開始。

範例

輸入範例一	**輸入範例二**	**輸入範例三**
7 WHEELER	6 MYTEST	5 NANNY
輸出範例一	**輸出範例二**	**輸出範例三**
HELWEER 4	TTEYSM 6	NYANN 1

第三題：最大子矩陣

問題敘述

給定一個由整數所組成的正方形矩陣，其大小為$N \times N$（亦即列、行數均為N）。令 W 為一矩形的框子，其大小為 a x b（其中 1 ≤ a ≤ N，1≤ b ≤ N），則 W 所形成之子矩陣為將 W 置於該正方形矩陣內時，W 所框出的小矩陣。由於 W 的大小及位置均可變動，因此可在給定的正方形矩陣中形成許多的子矩陣。請寫出一程式能找出一子矩陣，該矩陣中元素的加總值是所有子矩陣中最大的，稱為總和值最大的子矩陣；由於總和值最大的子矩陣可能有許多個，程式只要輸出總和值即可。

例如一個正方形矩陣如下：

$$\begin{bmatrix} 15 & -49 & -9 & 50 & 26 \\ 17 & 83 & -72 & -91 & 2 \\ 18 & -100 & -52 & 60 & 93 \\ 19 & 14 & -32 & -29 & 81 \\ 20 & -31 & -12 & 101 & 69 \end{bmatrix},$$

則框線標示出來的就是總和值最大的子矩陣，總和值為 60 + 93 + (-29) + 81 + 101 + 69 = 375。

程式計算如下：

首先必須依照輸入的兩個數值自動產生方形矩陣。

令正方型矩陣中第 i 列第 j 行的元素為 a_{ij} $(1 \le i \le N ; 1 \le j \le N)$，則正方形矩陣為

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1N} \\ a_{21} & \ddots & & & \vdots \\ a_{31} & & \ddots & & \vdots \\ \vdots & & & a_{ij} & \vdots \\ \vdots & & & \vdots & \vdots \\ a_{N1} & \cdots & \cdots & \cdots & a_{NN} \end{bmatrix},$$

其產生方式如下：給予 N 和 a_{11}，矩陣中的元素可根據下列式子自動產生。

$$\begin{cases} \text{if } \ j = 1 \text{ ，} a_{i1} = a_{11} + i \text{ ；} \\ \text{if } \ j > 1 \text{ ，} a_{ij} = \left[\left(a_{i\,j-1} \times 17 \right) \bmod \ 103 \right] \times (-1)^{(i+j)}; \end{cases}$$

（此處 mod 定義為: a mod b 表示 a 除以 b 之後取餘數，餘數可能為負值 ）

再依據上述計算產生出的 $N \times N$ 矩陣求出所有子矩陣中最大的總和值作為程式輸出。

程式存檔

請將此題解題之原始程式碼儲存到主辦單位所準備之三吋半磁片之檔案 prog_3 中，即名稱為"A:\prog_3.c"、 "A:\prog_3.pas"等。

條件限制

程式執行時間 60 秒以內。

輸入格式

1. 一律使用鍵盤輸入。
2. 輸入的第一個數字為矩陣的大小 N $(2 \le N \le 20)$；第二個數字為 a_{11}，以空格分開。

範例

輸入範例一	**輸入範例二**	**輸入範例三**	**輸入範例四**
5 15	4 35	4 -85	5 13
輸出範例一	**輸出範例二**	**輸出範例三**	**輸出範例四**
375	149	178	185

第四題： 辣椒醬蛋糕問題

問題敘述

台北市政府為了獎勵資訊能力競賽的選手，在比賽之後的午餐時間中，特別請來著名的「北政」蛋糕公司，烘焙出一種棋盤狀的蛋糕，其尺寸有大有小。但在水平及垂直格線處有凹溝，因此你可以很容易地將蛋糕一刀切成上下或左右兩塊，而吃掉其中一塊。在這次比賽中，你已經花費了很多力氣在撰寫程式，所以你已經很餓了，實在想趕快能吃一塊大一點的蛋糕來止餓。

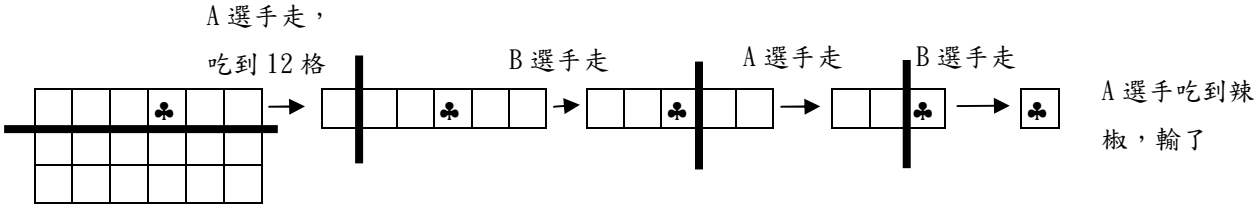
然而，為了考驗選手的智慧，「北政」蛋糕公司精心設計了一個難題。他們在其中一格蛋糕上面塗滿了辣椒醬，不幸吃到這格蛋糕的人將會辣到一輩子難忘！

考驗時以兩個選手(A 和 B)為一組，一開始有一塊很大的 n*m 的矩形蛋糕，這兩個選手輪流將蛋糕切成兩塊(A 選手先切及吃)，而吃掉其中不含辣椒醬的那一塊。但是到了最後，「吃了塗滿了辣椒醬的那一格蛋糕」的人就輸了這個遊戲。由於你實在是很餓了，所以你極力想要避免輸了這個遊戲，但是一開始就想吃愈大塊的蛋糕愈好。舉例來說，下圖是一塊 3*6 的蛋糕，其中標示「♣」的格子表示該格子塗滿了辣椒醬。如果 A 選手選擇走上方「過程一」的路，吃下面兩列(row)共有 12 格的蛋糕，接著 B 選手就會吃最左邊一欄(column)蛋糕，最後 A 選手就會輸了這個遊戲。反過來說，如果 A 選手選擇走中間「過程二」的路，吃左邊三欄(column)共有 9 格的蛋糕，接著 B 選手不管怎麼走，最後 A 選手一定可以贏了這個遊戲。因此，此例子一個必勝的走法是垂直地切一刀，切在從左邊算起第 3 欄及第 4 欄之間，而且其第一次可吃到 9 格的蛋糕。

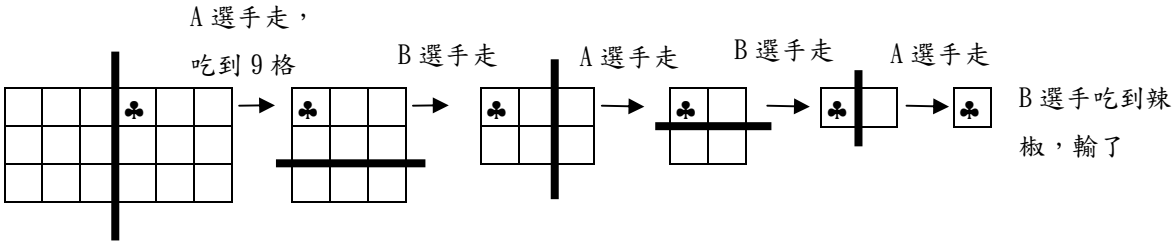
另外一種必勝的走法是 A 選手選擇走下方「過程三」的路，吃右邊一欄(column)共有 3 格的蛋糕，接著 B 選手不管怎麼走，最後 A 選手一定可以贏了這個遊戲。因此，此「過程三」也是一個必勝的走法，即垂直地切一刀，切在從左邊算起第 5 欄及第 6 欄之間，而且其第一次可吃到 3 格的蛋糕。但是因為你想第一次就吃較多的蛋糕，因此 A 選手應選擇「過程二」的走法較好。

如果你是 A 選手，第一步你要如何走才能必勝而且第一步你就吃到最多的蛋糕？此題你的任務即是要找出其中一個必勝的走法而且第一步就吃到最多的蛋糕。如果沒有必勝的走法，那麼請輸出"No winning strategy."。

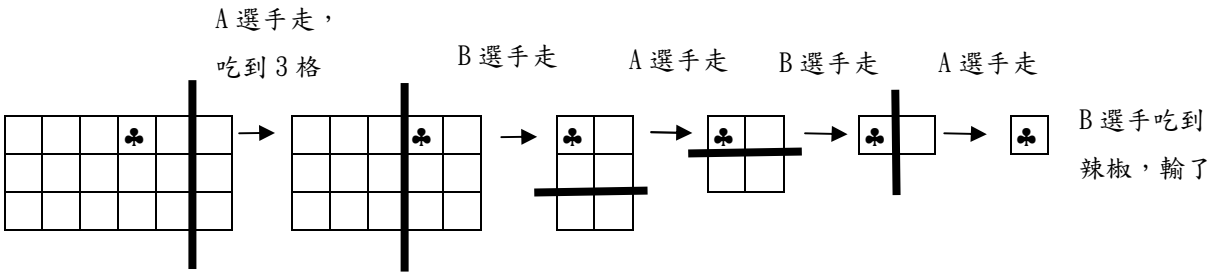
過程一：



過程二：



過程三：



程式存檔

請將此題解題之原始程式碼儲存到主辦單位所準備之三吋半磁片之檔案 prog4 中，即名稱為"A:\prog_4.c"、 "A:\prog_4.pas"等。

條件限制

矩形的大小不超過 1000×1000。也就是說，$1 \le n \le 1000$，$1 \le m \le 1000$。

輸入格式

一律使用鍵盤輸入四個正整數 n、m、x 和 y，中間以一個空白分開。n 為矩形的列數，m 為矩形的欄數，x 為塗滿了辣椒醬的那一格蛋糕的列座標位置，y 為塗滿了辣椒醬的那一格蛋糕的欄座標位置，$1 \le x \le n \le 1000$，$1 \le y \le m \le 1000$。

輸出格式

一律在螢幕上輸出。如果沒有必勝的走法，那麼請輸出"No winning strategy."即可。如果有必勝的走法(可能不只一種)，請輸出第一步吃到最多蛋糕的一種必勝的走法即可。也就是只輸出三個資料：切蛋糕的方向 d、位置 k 及吃到蛋糕的格數，中間以一個空白分開。其中 d="horizontal"，表示水平地切一刀，切在第 k 列及第 k+1 列之間。另一種狀況 d="vertical"，表示垂直地切一刀，切在第 k 欄及第 k+1 欄之間。

輸入範例一	**輸入範例二**	**輸入範例三**	**輸入範例四**
3 6 1 4	2 2 2 1	1 4 1 2	2 3 2 2
輸出範例一	**輸出範例二**	**輸出範例三**	**輸出範例四**
vertical 3 9	No winning strategy.	vertical 3 1	horizontal 1 3

第五題：叢林冒險記

問題敘述

志明與春嬌一行人在南美亞馬遜叢林冒險旅途中得知有關於寶藏的消息，四處打聽下取得一些片斷的資訊（如下表）：

編號	路線(具有方向性)	存活率	編號	路線(具有方向性)	存活率
1	A 村落→B 村落	0.3	10	F 村落→E 村落	0.28
2	A 村落→C 村落	0.67	11	F 村落→I 村落	0.2
3	A 村落→D 村落	0.5	12	G 村落→I 村落	0.23
4	B 村落→F 村落	0.33	13	G 村落→J 村落	0.36
5	C 村落→D 村落	0.42	14	H 村落→L 村落	0.4
6	C 村落→G 村落	0.1	15	I 村落→L 村落	0.41
7	D 村落→F 村落	0.54	16	J 村落→L 村落	0.37
8	D 村落→J 村落	0.28	17	J 村落→N 村落	0.47
9	E 村落→H 村落	0.5	18	L 村落→N 村落	0.5

其中包括可以直接連接兩個村落間的路線(具有方向性)以及其對應的存活率。雖說尋寶之旅危機重重，佈滿了各式各樣的陷阱或猛獸，只有極小的存活率；但是，大家仍是一致決定勇敢地邁向尋寶之旅。身為其中一員的你，現在必須整合這些片斷資訊，為大家計算出可能的尋寶路徑中，會經過某一條路線的路徑佔所有可能路徑的存活率的比例。

舉到來說，若輸入"A I 7"，則表示想要計算“由 A 村落出發且要經過編號為 7 之 D→F 路線到達目的地 I 村落”的路徑佔所有可由 A 村落到達 I 村落路徑的存活率之比例。我們得到下列四種不同的走法：

走法 1：A 村落→C 村落→D 村落→F 村落→I 村落

其存活率為：0.67 * 0.42 * 0.54 * 0.2 = 0.0303912

走法 2：A 村落→D 村落→F 村落→I 村落

其存活率為： 0.5 * 0.54 * 0.2 = 0.054

走法 3：A 村落→B 村落→F 村落→I 村落

其存活率為： 0.3 * 0.33 * 0.2 = 0.0198

走法 4：A 村落→C 村落→G 村落→I 村落

其存活率為： 0.67 * 0.1 * 0.23 = 0.01541

因此，“A 村落出發且要經過編號 7 之 D→F 路線到達 I 村落”的存活率佔所有可能路線的存活率的比例

＝(走法 1 的存活率+走法 2 的存活率)／(走法 1 的存活率＋走法 2 的存活率+走法 3 的存活率+走法 4 的存活率)

$$=(0.0303912+0.054)/(0.0303912+0.054+0.0198+0.01541)$$

＝0.70560496

程式存檔

請將此題解題之原始程式碼儲存到主辦單位所準備之三吋半磁片之檔案 prog_5 中，即名稱為"A:\prog_5.c"、 "A:\prog_5.pas"等。

條件限制

程式執行時間 60 秒以內。

輸入格式

1. 一律使用鍵盤輸入。
2. 程式的輸入包含二個字母與一個數字（以空格分開），第一個字母表示起點之村落，第二個字母表示目的地之村落，字母所表示的村落必須存在於路線表格中；第三個數字 k 表示必經路線的編號 $(1 \le k \le 18)$。

   例如： A I 7

   表示以 A 村落為起點，I 村落為終點，中間必須經過編號為 7 之路線（D 村落→F 村落）。

輸出格式

1. 一律使用螢幕輸出。
2. 輸出存活率比例，小數點取至八位（四捨五入）。若起點到終點無路可通，或可通但路徑中不包含必經路線 k，即答案為無解，則輸出“No solution”。

範例

輸入範例一	**輸入範例二**	**輸入範例三**	**輸入範例四**
A I 7	B L 9	D L 6	G N 15
輸出範例一	**輸出範例二**	**輸出範例三**	**輸出範例四**
0.70560496	0.40579710	No solution	0.16663722