CS Data Structure

Fall 2019

Program Exercise #2

*Submit Due Date: 2019/10/4 (Fri.) 23:59:00 p.m.

Grading

=======

1. 得分 100: 當日資結實習課(2019/10/1 12:00 p.m.)結束前完成 demo，且正確回答助教的提問。
2. 得分 90: 以 iLearn 系統最後修改時間為基準，2019/10/4 (Fri.) 23:59:00 p.m. 上傳繳交，於隔週實習課才完成 demo 以及正確回答助教提問。

* 隔週實習課沒有完成 demo 者，一律以 0 分計。

Problem: Random walking

在一個 n x m 的格子裡一隻蟑螂被放在一個預定的起始點 (start_i, start_j) 它可以在不超出四面的牆的範圍的條件下任意的向八個方向移動 問:它最少需要幾次移動才能把每個格子最少走過一次

作法:

=====

先用一個 n x m 的 array (一維或二維或任何可用的方法) 叫 count 用來記錄每個格子所被走過的次數 (起始位置也算1)

起始位置若也算 1, 會使移動次數會比 count 的記錄少 1

為了方便, 統一用"存在某個格子次數"來做統計

而這個 count array 的每一個格子是先被 initialize 成 0 代表尚未走過

以後每走到一個格子, 就在該格子的 count 加 1

移動方法:

=========

bug 的位置是 (ibug, jbug)

八個方向由 0 ~ 7 來表示

用 random 來決定下一個移動方向為何

bug 的新位置可由此算出:

(ibug + imove[k], jbug + jmove[k])

k 是方向 (0 ~ 7)

imove[0] = -1, jmove[0] = 1

...

imove[7] = -1, jmove[7] = 0

因此畫在 xy 軸上所對應的方向為:

6 7 0
5 1
4 3 2
如果新的位置會跑到牆裡 (boundary)
就重新取一個方向
直到找到一個允許的方向 (legal move)
同學可自行決定是否要像課本一樣在 array 裡包一個外牆
或是用一些 if 來測試是否碰到牆
很多同學提到課本的移動方向和實際的矩陣 index 方向不符
其實這只是人的主觀上的差異, 跑出來的結果並不會有差異
如果不喜歡課本的用法, 也可以自行定義自己的方向
只要是合乎邏輯且易懂就可
但最好是用 comments 說明一下所用的方法(和方向)

程式結束:
=========
當每個格子都至少被走過一次之後
就可以輸出結果了

程式需求:
=========
a) 2 < n <= 40, 2 <= m <= 20 (超過就給錯誤訊息)
b) 程式碼要能compile成 xxxxxxx_program2.exe xxxxxxx 是學號
c) 程式要能從 command-line 讀 input
   格式為: xxxxxxx_program2.exe n m start_i start_j
   例: bug 在 39 x 19 的格子裡, 起始位置為 (2, 5)
   M0645505_PE2.exe 39 19 2 5
d) 為了避免程式跑太久(或是跑個不停)
   迴圈(iteration)的次數最大限定為50,000次
   如果超過此次數就停止, 輸出結果
e) 程式要能夠用 Dev-C++ 5.11 或以上的版本 compile
f) 有人問可不可以用 C++ 跟 C 混寫, 可以。但課本是用C, 所以盡量統一用C
g) 程式的簡潔程度也會占一些評分標準, 如空行, indentation之類的
i) comments 也占一些分數, 請同學用 comments 說明程式碼
   如果程式跑不出來,但有寫合理的 comments 說明想要做的方法會斟酌給分數(但當然不會是滿分)
   如果程式跑不出來又沒有說明(comments) 那當然就是零分了
   程式跑出來了但是沒有寫comments會扣一點分數 (不寫comments是很不好的習慣!!!)

j) 課本的 20 20 10 10 和 39 19 1 1 只是兩種"範例"測試資料
助教在評分的時候並不一定會用這兩種!

Output

=======

輸出結果儲存至 xxxxxx_PE2.csv 檔，其中數字間以逗號 ","作分隔。(xxxxxx： 學號)
如此可不必考量輸出矩陣大小和數字對齊的問題！

xxxxxx_ PE2.csv的內容如下：

The total number of moves = xxx

a ... b

. . .

. . .

. . .

c ... d

xxx 是總移動次數 (count array 的總和) 之後是輸出 count 矩陣