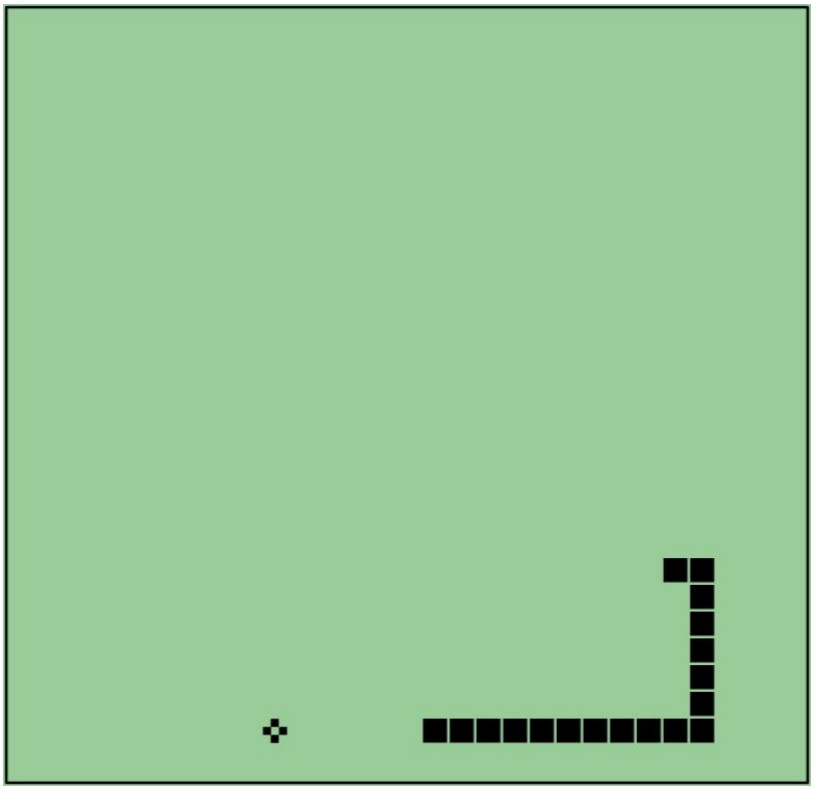
DSAP期末競賽

自動尋路貪吃蛇

時間 2020/06/03 00:00 ~ 2020/06/30 23:59

貪吃蛇是一個經典的小遊戲，玩家須以上下左右操縱細長的蛇蛇在地圖間穿梭，讓蛇蛇可以吃到得分的點點，吃到點點會增加身體的長度，同時還要避免撞到牆壁跟自己的身體，否則遊戲就會結束。



參考遊戲連結：
http://cn.game-game.com/70454/

在期末競賽中，你需要以 C++ 實作一個會**自動尋找移動路徑**的貪吃蛇 2.0，而且遊戲規則會有所不同。

你一樣需要讓你的蛇蛇可以在地圖中穿梭、吃更多的點點來取得高分並讓身體變長，而地圖中除了會有正常的圍牆與障礙物之外，還可能會同時出現兩個以上的得分點，而每個得分點所具有的分數將會有所不同，沒有被吃掉的得分點甚至會在你吃了一個或數個其他得分點後消失不見。另外，就像一般的貪吃蛇遊戲一樣，在吃到點點後貪吃蛇會立刻變長一個單位（也就是尾巴多一節）。

你需要在**有限的步數**之內取得最高的分數，這會需要最短路徑的搜尋以及對於得分點的取捨。上吧，為了你的期末分數，開吃吧蛇蛇！

任務

請撰寫一個 **`Snake`** Class，此 Class 至少須具備以下成員：

- 一個 private member，名為 **`position`**
  - 資料型別為 **`queue<tuple<int, int>>`**
  - 代表其當下的位置
- 一個 **`constructor`**
  - 用來初始化貪吃蛇的位置
- 一個 public function，名為 **`nextPosition`**
  - input 為當前的地圖
  - output 為下一時間單位貪吃蛇的新位置

除了以上的要求以外，你可以在 **`Snake`** 中新增任何你需要的東西，你也可以使用**任何方法**。

在期末競賽中，你會需要繳交下列兩個檔案：

- **`Snake`** Class 的 header 檔（Snake.h）

```cpp
#include <queue>
#include <tuple>
#include <vector>

// Add anything else you need

using namespace std;

class Snake {
private:
    queue<tuple<int, int>> position;

    // Add anything else you need

public:
    // Don't edit interface
    Snake(queue<tuple<int, int>> startPosition);
    queue<tuple<int, int>> nextPosition(vector<vector<int>> map);

    // Add anything else you need
};

```

- **`Snake`** Class 的實作檔（Snake.cpp）

```cpp
#include "Snake.h"

// Add anything else you need

Snake::Snake(queue<tuple<int, int>> startPosition) {
    // Implement by yourself
}

queue<tuple<int, int>> Snake::nextPosition(vector<vector<int>> map) {
    // Implement by yourself
}

// Add anything else you need

```

地圖說明

地圖是二維的 **`vector`**（**`vector<vector<int>>`**），代表目前的地圖資訊。

數值說明：

`-1`：障礙物，撞上去（蛇頭的位置在障礙物上）即遊戲結束

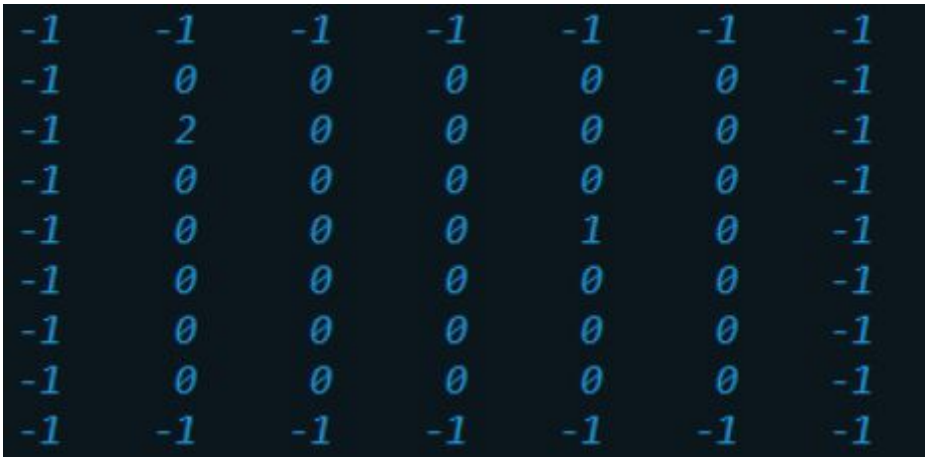
`0`：空格，可移動的空間

`1`：得分點，吃到即得 1 分

`2`：得分點，吃到即得 2 分

（數值大於等於 **`1`** 的均為得分點）

地圖範例示意圖（實際地圖不一定是這個大小）：



以上圖為例，此範例地圖會存在一個 **vector<vector<int>>** 中：

```
vector<vector<int>> map = {
    {-1, -1, -1, -1, -1, -1, -1},
    {-1,  0,  0,  0,  0,  0, -1},
    {-1,  2,  0,  0,  0,  0, -1},
    {-1,  0,  0,  0,  0,  0, -1},
    {-1,  0,  0,  0,  1,  0, -1},
    {-1,  0,  0,  0,  0,  0, -1},
    {-1,  0,  0,  0,  0,  0, -1},
    {-1,  0,  0,  0,  0,  0, -1},
    {-1, -1, -1, -1, -1, -1, -1}
};
```

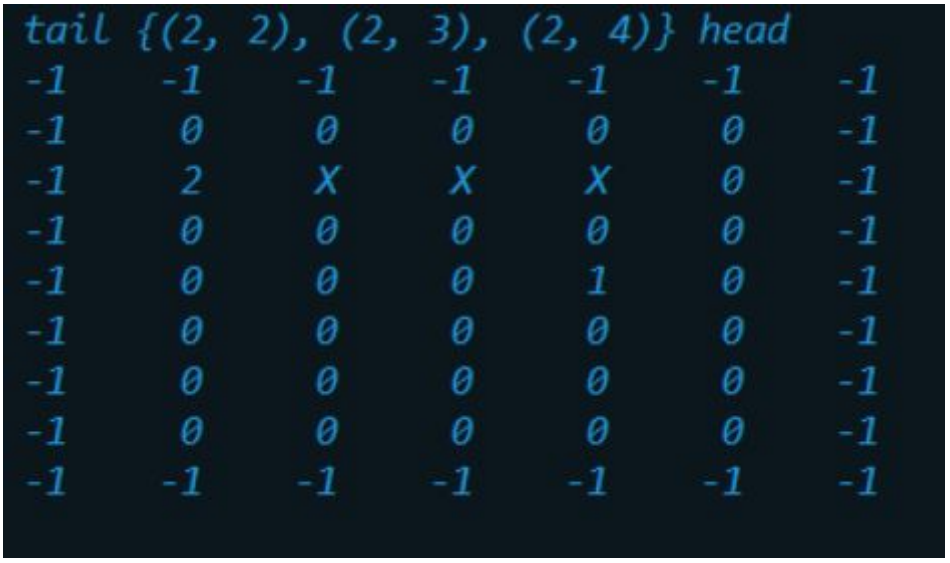
- 地圖高為 9（map.size() = 9）
- 地圖寬為 7（map[0].size() = 7）
- 在 map[2][1] 的位置有一個 2 分的得分點（map[2][1] = 2）
- 在 map[4][4] 的位置有一個 1 分的得分點（map[4][4] = 1）

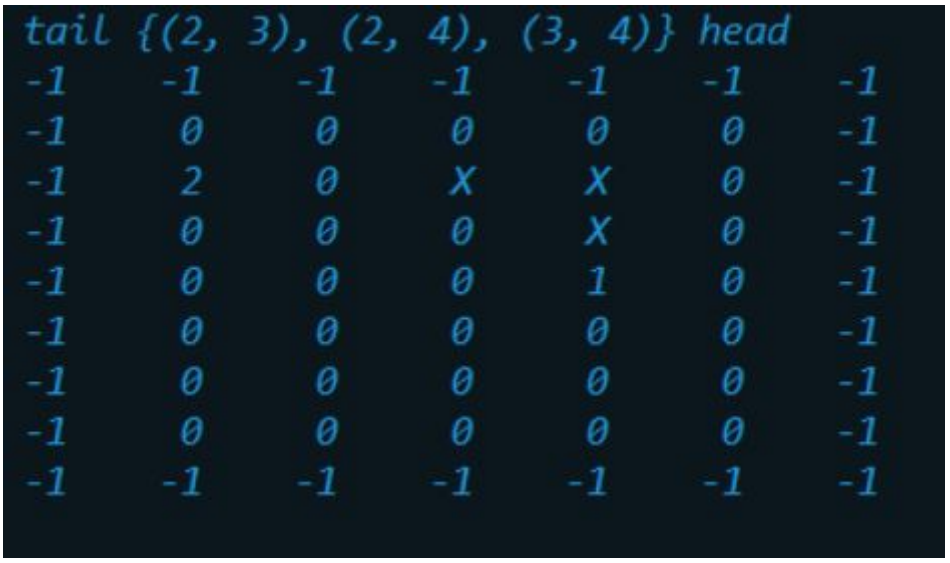
貪吃蛇位置說明

貪吃蛇的位置資訊是一個 **queue<tuple<int, int>>**。

- **queue** 的最後一個 element 代表蛇頭的位置（**queue.back()**）
- **queue** 的第一個 element 代表蛇尾的位置（**queue.front()**）
- 每經過一單位的時間，貪吃蛇會向上、下、左或右移動一格，也就是會 **push** 一個 **tuple** 到尾端並 **pop** 最前端的 **tuple**，而 push 進去的 **tuple** 跟前一個 **tuple** 只能相差一個單位距離

貪吃蛇移動範例示意圖：
（蛇為圖中的 **X** 標記）

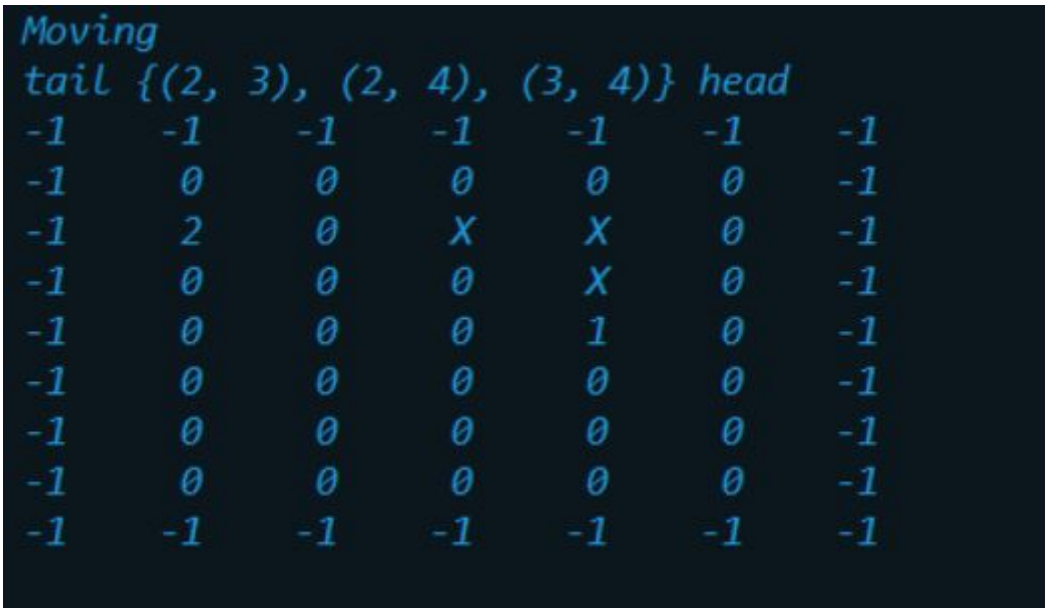


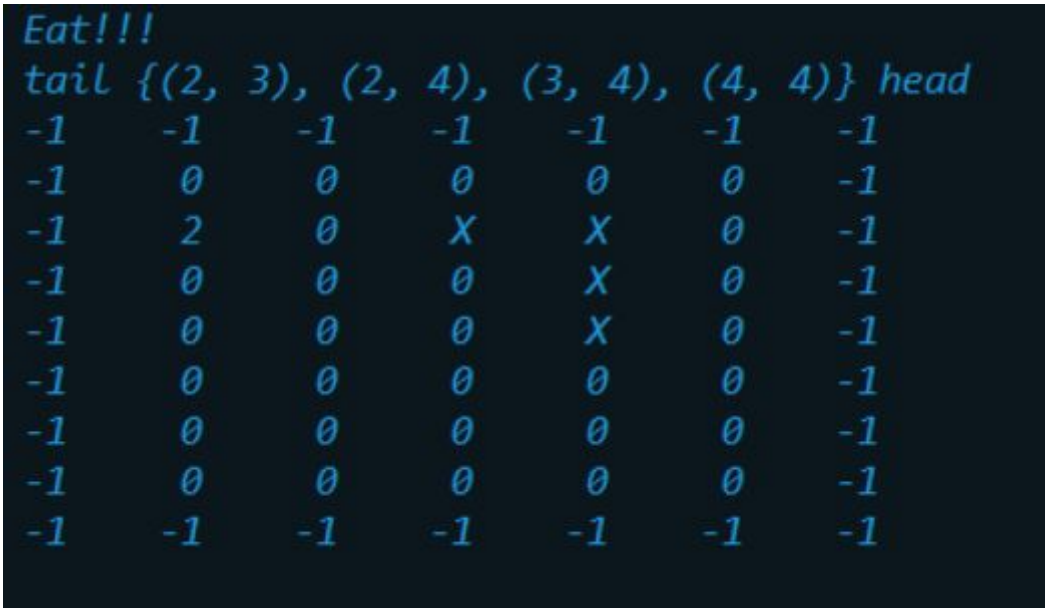


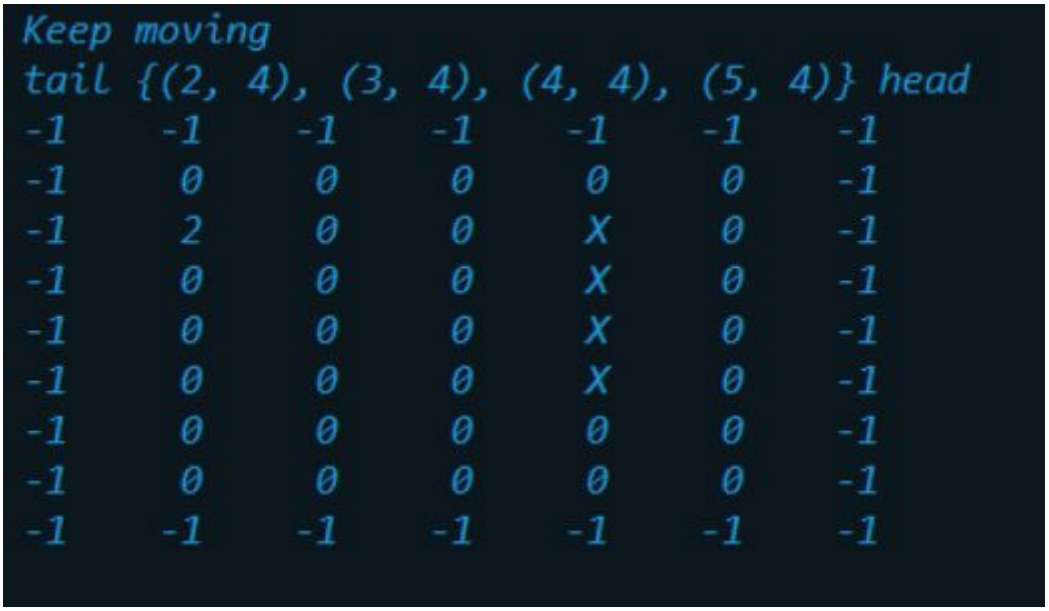
貪吃蛇進食說明

當貪吃蛇吃到一個得分點時，會立刻將身體延長一個節點（在尾巴多一節，請參考示意圖），隨後跟隨整個蛇的身體一起移動。

貪吃蛇吃得分點示意圖：
（蛇為圖中的 **x** 標記，吃得分點前的長度為 3，得分點為圖中 **1** 的標記）







批改規定

助教會使用一個批改程式來引入大家寫的 class 來進行批改，批改流程如下：

- 首先會宣告一個 **`Snake snake`** 來初始化貪吃蛇的位置
- 之後使用 for loop 重複 m 次（m 為行動步數上限），loop 內會執行以下動作：
    - 呼叫 **`snake.nextPosition(map)`** 並丟入當前的地圖資訊作為參數
    - 獲得下一單位時間貪吃蛇的位置資訊後，檢查移動是否符合規定
    - 計算貪吃蛇吃得分點所獲得的得分
- 最後算出累計得分

上傳規定

請上傳一個 zip 檔，裡面包含一個 C++ header file（Snake.h ）以及一個你實作的 Class 本體（Snake.cpp），以及其他任何你需要的程式碼。

可以參考以下教學：
http://kaiching.org/pydoing/cpp-guide/unit-13-header.html
（助教用來批改的程式就好比教學中那個包含 **`main()`** 的 .cpp 檔案）

助教會提供統一上傳的地方（NTU COOL 上傳區），**每天晚上 12 點**統一下載大家繳交的檔案、批改並公開大家的遊戲分數一次（視助教 loading 會提高批改次數）。

評分規定

- 只要你的貪吃蛇可以**移動一步**，Final Project 就可以拿到 60 分。
- 最後會將所有提交程式並且可以正常遊戲（遊戲中拿到 1 分以上）的學生，在遊戲中的得分進行排序，再依常態分佈進行給分。
- 在 6/16 會公開範例程式碼，如果在範例公開前便在遊戲中得分達到 baseline（範例程式碼的分數），Final Project 最後得分（是 Final Project 成績，不是遊戲分數）會額外加 5 分。

注意事項

- **`constructor`** 與 **`nextPosition`** 的 header 不可更動，否則會造成批改程式在呼叫時產生錯誤，若造成錯誤，直接以零分計（程式不可執行）。
- 請將 Snake.h 以及 Snake.cpp 直接壓縮並上傳，壓縮檔中**請勿包含任何資料夾**。Ex:
  – r00000000.zip
  ├ Snake.h
  └ Snake.cpp
- 貪吃蛇的起始長度跟位置都有可能變化，因此一律透過 **`constructor`** 來初始化貪吃蛇的位置。
- 地圖一開始會以最簡單的規則呈現：四邊圍牆、每次出現一個得分點；
  之後會依據大家的破關程度**依序開放新關卡**，例如在吃了 10 個得分點後，路中央開始出現隨機障礙物、開始同時出現複數且分數不同的得分點、地圖會放大、開放邊界可移動（最左邊再往左走會到最右邊）等進階規則。
  開放新難度時助教會在 NTU COOL 上面公告。
- 行動步數上限 **`m`** 也會隨難度提升而調整。
- 無論吃到的得分點為幾分，貪吃蛇的長度都只會增加 1。
- 地圖只會在吃到點點的時候會有所變動（例如，在 for loop **`i=5`** 的時候吃到得分點，在 for loop **`i=6`** 時輸入 **`snake.nextPosition(map)`** 的 **`map`** 會有所不同），其中變動包含新增/刪除得分點、新增/刪除障礙物等。
- 在一般情況下，每個人的得分點出現順序以及位置都會是相同的，但如果該得分點出現的位置剛好在貪吃蛇的身體上，則會改為出現在原位置周遭的九宮格內，若再不行則出現在 25 宮格內，再不行則從整張地圖的左上角開始尋找有空格的位置。