計算機程式與應用 EC1011301

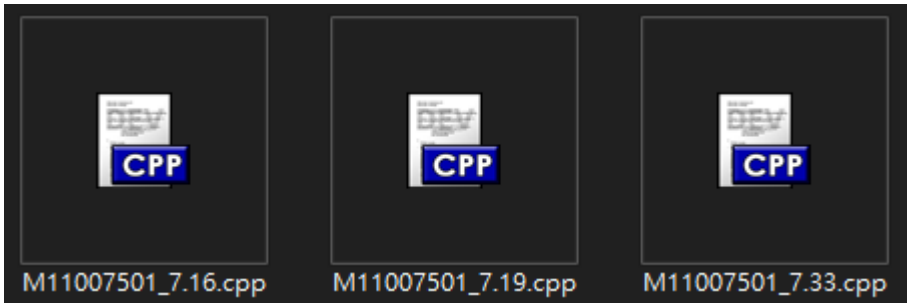
黃意婷 教授

Ch7 作業說明

- ✓ 作業繳交請依各題目各別存檔
- ✓ 作業繳交檔案格式區分為 CPP 檔與 PDF 檔，擇一繳交，區別如下

CPP 檔	PDF 檔
作業程式能順利執行 以 C++ 程式碼繳交	作業程式無法執行 以文字說明繳交
檔案命名格式：學號_題號.cpp	檔案命名格式：學號_題號.pdf

- ✓ 檔案命名格式說明
  - ➢ 學號：如 M11007501
  - ➢ 題號：本次作業題號分別為 7.16, 7.19 與 7.33
  - ➢ 正確格式範例：



- ✓ 若有任何問題歡迎寄信至 M11007501@gapps.ntust.edu.tw 與 M11007513@gapps,ntust.edu.tw 進行詢問，Moodle 內訊息容易被略過請以寄信為主，謝謝！

7.16 題目

7.16 *(Dice Rolling)* Write a program that simulates the rolling of two dice. The sum of the two values should then be calculated. [*Note:* Each die can show an integer value from 1 to 6, so the sum of the two values will vary from 2 to 12, with 7 being the most frequent sum and 2 and 12 being the least frequent sums.] Figure 7.22 shows the 36 possible combinations of the two dice. Your program should roll the two dice 36,000,000 times. Use a one-dimensional `array` to tally the numbers of times each possible sum appears. Print the results in a tabular format. Also, determine if the totals are reasonable (i.e., there are six ways to roll a 7, so approximately one-sixth of all the rolls should be 7).

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Fig. 7.22 | The 36 possible outcomes of rolling two dice.

7.19 題目

7.19 (*Converting `vector` Example of Section 7.10 to `array`*) Convert the `vector` example of Fig. 7.21 to use `arrays`. Eliminate any `vector`-only features.

7.33 題目

script equals the ending subscript.

7.33 (*Maze Traversal*) The grid of hashes (#) and dots (.) in Fig. 7.27 is a two-dimensional built-in array representation of a maze. In the two-dimensional built-in array, the hashes represent the walls of the maze and the dots represent squares in the possible paths through the maze. Moves can be made only to a location in the built-in array that contains a dot.

There is a simple algorithm for walking through a maze that guarantees finding the exit (assuming that there is an exit). If there is not an exit, you'll arrive at the starting location again. Place your right hand on the wall to your right and begin walking forward. Never remove your hand from the wall. If the maze turns to the right, you follow the wall to the right. As long as you do not remove your hand from the wall, eventually you'll arrive at the exit of the maze. There may be a shorter path than the one you've taken, but you are guaranteed to get out of the maze if you follow the algorithm.

```
# # # # # # # # # # # # #
# . . . # . . . . . . . #
. . # . # . # # # # . # .
# # # . # . . . . # . # .
# . . . . # # # . # . . .
# # # # . # . # . # . # .
# . . # . # . # . # . # .
# # . # . # . # . # . # .
# . . . . . . . . # . # .
# # # # # # . # # # . # .
# . . . . . . # . . . # .
# # # # # # # # # # # # #
```

Fig. 7.27 | Two-dimensional built-in array representation of a maze.