國立屏東大學 資訊工程系 程式設計(一)

# Turnin作業7

- chapter 7
- Turnin Code: c.hw7
- Due Date: 2026/05/27 23:59 **Hard Deadline**

---

## 繳交方式說明

本次作業繳交將以資料夾的形式繳交，需要為每一題建立一個資料夾（資料夾名稱為該題題目前方之代號，第一題為□p1□□第二題為□p2□□餘以此類推），

繳交說明可參考作業3：**連結**

**任何未依照正確繳交格式的檔案將以 0 分計算**

本文使用「 」及「 」代表「空白字元」與□Enter 換行字元」，並且將使用者輸入的部份使用灰階方式顯示。另外，題目的執行結果中，如果出現「(」、「)」、「:」、「;」、「.」與「,」等符號，皆為英文半形！

---

## p1 動態字串輸入

在日常生活當中，我們使用者所接觸的交互式介面通常不會限制使用者輸入訊息的長度，比如我們跟朋友用 LINE 聊天時，想打多少字就想打多少，而不會被系統提示說我們只能在每一次輸入的訊息中，只能打15或20個字等限制的情況。

因此，本題需請同學完成一個C語言程式，讓使用者輸入一個 **長度不限** 的字串，並將使用者輸入的訊息重新顯示給使用者看。

### 實作注意

1. 同學在實作中可以嘗試先以 **一個字元的方式** 依序讀取輸入的內容，接著再檢查目前存放的空間是否已滿，如果滿了再將當前的空間增大。

2. 本次測試檔輸入的 **內容長度 $\gt 1$**

請參考以下 main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "dynInput.h"

int main()
{
    unsigned curr_cap = 1;
    char* str = (char*)calloc(curr_cap, sizeof(char));

    printf("Enter a string: ");
    str = getInput(str, curr_cap);

    printf("You entered:\n[%s]\n", str);

    free(str);
    return 0;
}
```

在本題，同學需要實作以下函式：

- **getInput()**

待同學實作完畢後，請函式定義與實作分別放入以下檔案後並繳交：

- **dynInput.c**：含有上述函式的實作
- **dynInput.h**：含有上述函式的定義

本題將以下 Makefile 進行批改：

```
all: main.c dynInput.o
    gcc main.c dynInput.o
dynInput.o: dynInput.c
    gcc -c dynInput.c
clean:
    rm -rf *.*~ *~ a.* *.o *.o~
```

請參考以下輸出範例：

```
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲a▲string:▲May▲the▲Force▲be▲with▲you.↵
You▲entered:↵
[May▲the▲Force▲be▲with▲you.]↵
```

```
[3:23▲user@ws▲p1]
```

```
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲a▲string:▲It’s▲over,▲Anakin.▲I▲have▲the▲high▲ground.↵
You▲entered:↵
[It’s▲over,▲Anakin.▲I▲have▲the▲high▲ground.]↵
[3:23▲user@ws▲p1]
```

```
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲a▲string:▲THE▲ROPE,▲ALONG▲WITH▲THE▲STICK,▲ARE▲TWO▲OF▲MANKINDS▲OLDEST▲TO
OLS.▲THE▲STICK▲TO▲KEEP▲THE▲BAD▲AWAY.▲THE▲ROPE▲USED▲TO▲BRING▲THE▲GOOD▲TOWA
RD▲US.▲THEY▲WERE▲OUR▲FIRST▲FRIENDS,▲OF▲OUR▲OWN▲INVENTION.▲WHEREVER▲THERE▲
WERE▲PEOPLE,▲THERE▲WERE▲THE▲ROPE▲AND▲THE▲STICK.▲-▲from▲Kobo▲Abe▲"Nawa"↵
You▲entered:↵
[THE▲ROPE,▲ALONG▲WITH▲THE▲STICK,▲ARE▲TWO▲OF▲MANKINDS▲OLDEST▲TOOLS.▲THE▲STICK▲TO
▲KEEP▲THE▲BAD▲AWAY.▲THE▲ROPE▲USED▲TO▲BRING▲THE▲GOOD▲TOWARD▲US.▲THEY▲WERE▲OUR
▲FIRST▲FRIENDS,▲OF▲OUR▲OWN▲INVENTION.▲WHEREVER▲THERE▲WERE▲PEOPLE,▲THERE▲WERE▲T
HE▲ROPE▲AND▲THE▲STICK.▲-▲from▲Kobo▲Abe▲"Nawa"]↵
[3:23▲user@ws▲p1]
```

- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。
- 本題將以上述提供的 Makefile 進行編譯，請同學作答時 **“以題目提供的文件為主”**。
- 本題應繳交檔案如下（**main.c**及**Makefile** 則不需繳交）：
  - **dynInput.c**
  - **dynInput.h**

---

# p2 簡易泛型鍵值對管理器

本題需請同學完成一個簡易泛型鍵值對 (Key-Value) 的管理器。該程式在執行後會提供一個輸入介面，讓使用者可以輸入以下操作：

- 選項 **(a)**：新增一筆資料
- 選項 **(d)**：移除一筆資料
- 選項 **(l)**：列出目前以存放的資料
- 選項 **(h)**：提示操作選單
- 選項 **(q)**：退出程式

為了實現以上操作，本體提供 **標頭檔 define.h** 來幫助同學完成本次作業，請參考以下內容：

```
#define MIN_VSTRING_LEN 2
#define GROWTH_RATE 2
#define SHRINK_RATE 0.5

typedef enum
{
    INT,
    DOUBLE,
    STRING
} type_t;

typedef struct
{
    type_t  type;
    char    key[32];
    void*   value;
} kv_pair_t;

typedef struct
{
    kv_pair_t*  records;
    unsigned    size;
    unsigned    capacity;
} database;
```

在提供的標頭檔中，有以下兩個結構體：

- **data 結構體**
  - **key (鍵)**：資料的名稱，長度 **不超過 31 個字元**
  - **type (型態)**：資料的型態，型態需使用題目提供的 **type_t** 列舉。
  - **value (值)**：指向資料所存放的值的指標。此處 **必須以動態記憶配置** 來實作，指向的數值有以下：
    - 型態 **整數** 的資料。
    - 型態 **浮點數** 的資料。
    - 型態 **字串** 的資料，長度 **不限（可參考 p1**
- **database 結構體**
  - **records**：指向一個 **kv_pair_t** 結構體型態的動態配置陣列的指標。
  - **size**：記錄目前實際存放多少資料
  - **capacity**：*記錄目前向系統要求多少 data 結構體大小

從以上兩個結構體的內容可以看出本題所有操作都是建立在該兩個結構體上，因此本題需請同學完成以下函式：

- **dbAdd()**：處理增加一筆資料至陣列
- **dbDelete()**：負責處理從陣列中刪除一筆資料
- **dbPrint()**：負責處理顯示陣列中的一筆資料
- **dbFree()**：釋放在 **database結構體** 中 **records指標** 以及其中所有存放的 **kv_pair_t結構體** 中

**value指標** 動態配置過的空間。

---

在實作各項操作的函式時，請依照以下 **左邊編號順序** 進行錯誤判斷與處理（換行與排版請參考下方的輸出範例）：

- 在實作 **dbAdd()** 函式時：
  1. 若輸入的鍵 (key) 中包含 **英文字母** 及 **數字 以外** 的字元時，需輸出“ **invalid key** ”並且立即返回主程式。
  2. 若輸入的鍵 (key) 已經在陣列中時，需輸出“ **data already exists** ”並且立即返回主程式
  3. 若輸入的型態 (type) 不是 **int**、**double** 或 **string** 時，需輸出“ **unknown type** ”並且立即返回主程式。
  4. 若輸入的型態為 **string** 時，請以 **MIN_VSTRING_LEN(定義在define.h) 的大小** 進行初始的動態配置。
  5. 若以上使用者所輸入的資料皆正確後，需檢查 **當前的 size 是否 $\ge$ 當前的 capacity**？
     如果是，請以 **capacity $\times$ GROWTH_RATE(定義在define.h)** 擴增當前容量。
  6. 輸入的 **型態(type)** 皆為 **小寫**。
- 在實作 **dbDelete()** 函式時：
  1. 若當前陣列中沒有任何資料 **（即 數量(size) $\lt 1$**）時，需輸出“ **no data** ”並且立即返回主程式。
  2. 若 **無法找到** 使用者所輸入的資料時，需輸出“ **data not found** ”並且立即返回主程式。
  3. 若 **成功刪除** 使用者所輸入的資料後，需檢查當前的 **capacity $\times$ SHRINK_RATE(定義在define.h) 是否 $\ge$ 刪除後的 size**？
     如果是，請以 **capacity $\times$ SHRINK_RATE** 縮減當前容量。

### 實作注意

- 處理浮點數運算時，請以 **double** 型態處理。
- 批改時：
  - 在輸入鍵值時，測試輸入 **不小於 1 個字元**。
  - 在輸入資料的字串型態時，測試輸入 **不小於 1 個字元**。

請參考下面 main.c：

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "database.h"

void dbList(const database* db)
{
    printf("size(%u);cap(%u)\n", db->size, db->capacity);
    if (db->size < 1)
    {
        printf("no data\n");
```

```
        return;
    }

    for (unsigned i = 0; i < db->size; ++i)
    {
        printf("%u. ", i);
        dbPrint(&db->records[i]);
        printf("\n");
    }
}

void dbInit(database* db)
{
    db->records = (kv_pair_t*)malloc(2 * sizeof(kv_pair_t));
    db->size = 0;
    db->capacity = 2;
}

void dbHelp()
{
    printf("usage:\n");
    printf("  a)dd: add a new record\n");
    printf("  d)elete: delete an record\n");
    printf("  l)ist: list existing records\n");
    printf("  q)uit: exit the program\n");
}

int main()
{
    database db;
    char choice = 0;
    dbInit(&db);

    printf("[type 'h' for help]\n");
    do
    {
        printf("> ");
        scanf(" %c", &choice);
        switch (choice)
        {
            case 'a':
                printf("\n");
                dbAdd(&db);
                printf("\n");
                break;
            case 'd':
                printf("\n");
                dbDelete(&db);
                printf("\n");
                break;
            case 'l':
```

```
                printf("\n");
                dbList(&db);
                printf("\n");
                break;
            case 'q':
                dbFree(&db);
                break;
            case 'h':
                printf("\n");
                dbHelp();
                printf("\n");
                break;
            default:
                printf("\n");
                printf("unknown choice\n");
                printf("\n");
        }
    } while (choice != 'q');

    dbFree(&db);

    return 0;
}
```

待同學實作完畢後以下函式後：

- **dbAdd()**
- **dbDelete()**
- **dbPrint()**
- **dbFree()**

請函式定義與實作分別放入以下檔案後並繳交：

- **database.c**：含有上述函式的實作
- **database.h**：含有上述函式的定義

本題將以下面的 Makefile 進行批改：

```
all: main.c database.o
    gcc main.c database.o
database.o: database.c define.h
    gcc -c database.c
clean:
    rm -rf *.*~ *~ a.* *.o *.o~
```

本題範例輸出如下：

```
[3:23▲user@ws▲p2]▲./a.out↵
[type▲'h'▲for▲help]↵
>▲h↵
↵
usage:↵
▲▲a)dd:▲add▲a▲new▲record↵
▲▲d)elete:▲delete▲an▲record↵
▲▲l)ist:▲list▲existing▲records↵
▲▲q)uit:▲exit▲the▲program↵
↵
>▲D↵
↵
unknown▲choice↵
↵
>▲d↵
↵
no▲data↵
↵
>▲l↵
↵
size(0);cap(2)↵
no▲data↵
↵
>▲q↵
[3:23▲user@ws▲p2]
```

```
[3:23▲user@ws▲p2]▲./a.out↵
[type▲'h'▲for▲help]↵
>▲a↵
↵
key:▲k▲1↵
invalid▲key↵
↵
>▲a↵
↵
key:▲k@1↵
invalid▲key↵
↵
>▲a↵
↵
key:▲k1↵
type▲(int,▲double,▲string):▲int↵
value:▲114514↵
↵
>▲l↵
↵
```

```
size(1);cap(2)↵
0.▲k1(int):▲[114514]↵
↵
>▲a↵
↵
key:▲k1↵
data▲already▲exists↵
↵
>▲a↵
↵
key:▲k2↵
type▲(int,▲double,▲string):▲float↵
unknown▲type↵
↵
>▲a↵
↵
key:▲k2↵
type▲(int,▲double,▲string):▲double↵
value:▲114514.114514↵
↵
>▲a↵
↵
key:▲k3↵
type▲(int,▲double,▲string):▲string↵
value:▲This▲is▲a▲test▲message▲that▲checks▲if▲dynamic▲input▲is▲working▲or▲no
t,▲if▲this▲message▲is▲not▲getting▲chopped▲off▲or▲displaying▲some▲visual▲e
rrors,▲then▲IT▲WORKS▲:).↵
↵
>▲l↵
↵
size(3);cap(4)↵
0.▲k1(int):▲[114514]↵
1.▲k2(double):▲[114514.114514]↵
2.▲k3(string):▲[This▲is▲a▲test▲message▲that▲checks▲if▲dynamic▲input▲is▲working▲or▲not,▲if▲this▲
message▲is▲not▲getting▲chopped▲off▲or▲displaying▲some▲visual▲errors,▲then▲IT▲WORKS▲:).]↵
↵
>▲q↵
[3:23▲user@ws▲p2]
```

```
[3:23▲user@ws▲p2]▲./a.out↵
[type▲'h'▲for▲help]↵
>▲a↵
↵
key:▲k1↵
type▲(int,▲double,▲string):▲int↵
value:▲114514↵
```

```
↵
>▲a↵
↵
key:▲k2↵
type▲(int,▲double,▲string):▲double↵
value:▲114514.114514↵
↵
>▲a↵
↵
key:▲k3↵
type▲(int,▲double,▲string):▲string↵
value:▲A▲test▲string▲that▲says▲“Hello▲World!”!↵
↵
>▲l↵
↵
size(3);cap(4)↵
0.▲k1(int):▲[114514]↵
1.▲k2(double):▲[114514.114514]↵
2.▲k3(string):▲[A▲test▲string▲that▲says▲“Hello▲World!”!]↵
↵
>▲d↵
↵
key:▲k▲1↵
data▲not▲found↵
↵
>▲d↵
↵
key:▲k2↵
↵
>▲l↵
↵
size(2);cap(2)↵
0.▲k1(int):▲[114514]↵
1.▲k3(string):▲[A▲test▲string▲that▲says▲“Hello▲World!”!]↵
↵
>▲d↵
↵
key:▲k1↵
↵
>▲l↵
↵
size(1);cap(1)↵
0.▲k3(string):▲[A▲test▲string▲that▲says▲“Hello▲World!”!]↵
↵
>▲q↵
[3:23▲user@ws▲p2]
```

- 本題相關的程式碼路徑已註明於檔名右側，同學們可以透過路徑複製到自己的家目錄。

- 本題將以上述提供的 Makefile 進行編譯，請同學作答時 **“ 以題目提供的文件為主 ”**。
- 本題應繳交檔案如下（**define.h**、**main.c**及**Makefile** 則不需繳交）：
  - **database.c**
  - **database.h**

---

# p3 五子棋-最佳落子位點判斷

經過了前幾週的作業對各種棋型（如活三死四等）的練習，同學已經具備了分析棋盤棋型的能力。本題為五子棋系列的綜合實踐題：將模擬真實比賽對弈環境，給定目前的棋盤狀態與落子顏色後， 請同學參考五子棋 AI 程式競賽 - 進階篇權重分析的方法，找出棋盤上目前「最佳」的落子位置（最佳落子位置是由以下權重分析規則決定）。

權重分析規則：

1. **活四 / 死四 / 洞四 / 洞死四：10000 分**
   - 落子後能在該方向形成連五（直接獲勝）。
2. **活三 / 洞三：1000 分**
   - 落子後能形成「活四」的點。
3. **死三 / 洞死三：800 分**
   - 落子後能形成「死四」的點。
4. **活二 / 洞二：100 分**
   - 落子後能在該方向形成「活三」。
5. **死二：80 分**
   - 落子後能在該方向形成「死三」。
6. **活一：10 分**
7. **死一：1 分**

- 若棋型為我方，則 **分數*1.3**。

- 常見錯誤：`.1.1.1.`棋型為死三而不是洞三。

可參考以下 main.c , define.h 的內容，讀取命令列參數與棋盤：

```
typedef enum {Black=49, White=48, None=0} Player;
```

```
#include <stdio.h>
#include <string.h>
#include "chess.h"

int main(int argc, char *argv[])
```

```
{
    char chessboard[19][19];
    Player player;

    for(int i=0; i<19; i++)
        for(int j=0; j<19; j++)
            scanf(" %c", &chessboard[i][j]);

    player = (strcmp(argv[1], "Black") == 0) ? Black : White;
    findBestMove(chessboard, player);
}
```

本題的執行結果可參考如下：

```
[3:23 user@ws p3] ./a.out Black < testfile_1.cb↵
J,▲11↵
[3:23 user@ws p3] ./a.out White < testfile_2.cb↵
P,▲▲8↵
[3:23 user@ws p3] ./a.out Black < testfile_3.cb↵
N,▲15↵
[3:23 user@ws p3]
```

本題測試檔路徑：

- /home/stu/public/c2026s/c.hw7/p3/testfile_1.cb
- /home/stu/public/c2026s/c.hw7/p3/testfile_2.cb
- /home/stu/public/c2026s/c.hw7/p3/testfile_3.cb

- 建議參考前兩週作業所寫的五子棋作業。
- argv[1] 為你本次程式執行所持的棋子，“Black” 為黑，“White” 為白。
- 本題的所有測試檔案均經過設計，**只會出現一組最高分的最佳落子點**。
- 本題需使用提供的 **main.c** 來實作，不限制其他 Turnin 所繳交的檔名，但必須繳交 Makefile 檔案並將可執行檔命名為 a.out。

- 本題批改時將採**人工批改**。
- 請勿使用**AI撰寫**，若是助教發現疑似使用AI撰寫，將會約談同學。
- 約談過程中，若對本人繳交之代碼內容回答模糊、邏輯不通，將以 **0 分**計算。

---

# p4 五子棋-第一次預賽

在上個禮拜作業7的第6題中，同學已經完成亂數版的五子棋並上傳至平台進行初步參賽，而這次功課需要請同學將上一題實作的內容修改至自己的參賽版本，讓自己的五子棋具有一定棋力。

同學修改完後，請將修改完的五子棋重新上傳至平台：國立屏東大學資工系電腦五子棋 AI 競賽平台上(同樣使用gomoku2026為turnin code)預計將會於2026/05/28於舉行第一次預賽，所有參賽同學的作品將進行兩兩對奕比試10場，其中5場先手，5場後手，勝負記點採用:勝者2分、平手1分、敗者0分（每局獲勝規則為:黑子先行，五子連線為勝利）。預賽完成後依最終每位參賽同學所得到的分數給定此次作業配分：

- 前 **8%**：本題分數 **$\times 8$**
- 前 **20%**：本題分數 **$\times 4$**
- 前 **50%**：本題分數 **$\times 2$**
- 前 **95%**：本題分數 **$\times 1$**
- **未上傳者** 或 **編譯錯誤** 將以 **0分** 計算

- 本題只要 **上傳至參賽平台** 並 **檢查是否能夠由平台順利完成編譯** 即完成本題目，**不需繳交至 turnin**

- 請勿過度使用**AI撰寫**，若是助教發現疑似使用AI撰寫，將會約談同學。
- 約談過程中，若對本人繳交之代碼內容回答模糊、邏輯不通，將會取消同學分數，並以0分計。