國立屏東大學 資訊工程系 程式設計(一)

Turnin作業4

- 範圍：ch12 ~ ch13-5
- Turnin Code: c.hw4
- Due Date: 2026/04/22 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 圓面積及判斷點是否在圓內

在二維座標系中，圓形可以用「圓心座標」與「半徑」完整定義，而判斷點是否在圓內則是幾何運算的基礎，因此，本題需請同學完成一個 C 語言程式讓使用者先輸入 **結構體 (circle_t)** 的資料，其中包含 一個圓的 **中心座標(center)**，以及該圓的 **半徑(radius)**，輸入完後計算圓面積並回傳該數值，接著使用者輸入另一點的座標後判斷該點是否在圓內。

實作注意

- 計算圓面積時 π 請以 **3.14** 計算
- 計算浮點數時請以 **double** 型態處理

請參考以下 main.c 的實作：

```
#include <stdio.h>

#include "circle.h"

int main()
{
    point_t p = {0, 0};
    circle_t c = {{0, 0}, 0};

    printf("Enter the center of the circle (x y): ");
    scanf(" %lf %lf", &c.center.x, &c.center.y);
    printf("Enter the radius of the circle: ");
    scanf(" %lf", &c.radius);

    printf("Area of the circle: %.2lf\n", circleArea(c));

    printf("Enter a point to check (x y): ");
    scanf(" %lf %lf", &p.x, &p.y);
    if (isPointInCircle(p, c))
    {
        printf("The point is inside the circle.\n");
    }
    else
    {
        printf("The point is outside the circle.\n");
    }

    return 0;
}
```

另外，為了讓同學完成該程式，本題提供下面兩個公式及標頭檔 **point.h**

圓面積公式： $A = \pi r^2$

兩點直線距離的公式： $D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

```
typedef struct
{
    double x;
    double y;
} point_t;
```

本題需請同學定義並實作下列函式和結構體：

- 結構體：
  - **circle_t**
- 函式：

- **circleArea()**
- **isPointInCircle()**（此處 **l** 不是 **小寫的 L**，是 **大寫的 i**）

完成後，將內容放入以下檔案：

- **circle.c**：此文件需包含上述 **函式** 的實作。
- **circle.h**：此文件需包含上述 **函式** 及 **結構體** 的定義。

另外，本題目使用以下 Makefile 進行批改：

```
all: main.c circle.o
    gcc main.c circle.o -lm
circle.o: circle.c point.h
    gcc -c circle.c
clean:
    rm -rf *.o *~ *.out
```

本題的執行結果可參考如下：

```
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲the▲center▲of▲the▲circle▲(x▲y):▲0▲0↵
Enter▲the▲radius▲of▲the▲circle:▲3↵
Area▲of▲the▲circle:▲28.26↵
Enter▲a▲point▲to▲check▲(x▲y):▲1▲1↵
The▲point▲is▲inside▲the▲circle.↵
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲the▲center▲of▲the▲circle▲(x▲y):▲0▲0↵
Enter▲the▲radius▲of▲the▲circle:▲3↵
Area▲of▲the▲circle:▲28.26↵
Enter▲a▲point▲to▲check▲(x▲y):▲5▲5↵
The▲point▲is▲outside▲the▲circle.↵
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲the▲center▲of▲the▲circle▲(x▲y):▲-1▲2↵
Enter▲the▲radius▲of▲the▲circle:▲8↵
Area▲of▲the▲circle:▲200.96↵
Enter▲a▲point▲to▲check▲(x▲y):▲5▲7↵
The▲point▲is▲inside▲the▲circle.↵
[3:23▲user@ws▲p1]▲./a.out↵
Enter▲the▲center▲of▲the▲circle▲(x▲y):▲-300▲124↵
Enter▲the▲radius▲of▲the▲circle:▲60↵
Area▲of▲the▲circle:▲11304.00↵
Enter▲a▲point▲to▲check▲(x▲y):▲1▲1↵
The▲point▲is▲outside▲the▲circle.↵
[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（**point.h**、**main.c**及**Makefile** 則不需繳交）：
  - **circle.c**
  - **circle.h**

P2 學生分數記錄器

本題需請同學完成一個簡單的學生分數記錄器，該程式首先需要請使用者輸入某一名單的 **學生人數 (n)**，接著依序填入 **結構體 (stu_t)** 的資料，其中包含 **學生的名字(name)** 及對應的 **分數 (score)**，最後將已輸入的學生資料根據分數 **降冪排列 (由大到小)**，並計算全部已輸入學生的 **平均成績**。

另外，在輸入時：

- 若名字中含有 **英文字母或空白之外的字元** 時，需輸出“ **Invalid name!** ”並立即返回主程式。
- 若成績範圍不在 $0 \le score \le 100$ 時，需輸出“ **Invalid score!** ”並立即返回主程式。

實作注意

- 批改時，學生名字長度 **不超過 20 個字** 且 **至少有 1 個字**
- 計算浮點數時請以 **double** 型態處理

請參考以下 **main.c** 的實作：

```
#include <stdio.h>

#include "stu.h"

void stuPrint(stu_t *list, int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("%s: %lf\n", list[i].name, list[i].score);
    }
}

int main()
{
    int n = 0;
    printf("Number of students: ");
    scanf(" %d", &n);
```

```
    stu_t stus[n];
    for (int i = 0; i < n; i++)
    {
        printf("Student %d:\n", i + 1);
        int stat = stuInput(&stus[i]);
        if (stat != 0)
        {
            return stat;
        }
    }

    stuSort(stus, n);
    printf("\nStudents (Sorted by score, in descending order):\n");
    stuPrint(stus, n);

    printf("\nAverage score: %lf\n", stuAverage(stus, n));

    return 0;
}
```

因此，本題需請同學定義並實作下列函式：

- **stuInput()**
- **stuSort()**
- **stuAverage()**

完成後，將內容放入以下檔案：

- **stu.c**：此文件需包含上述 **函式** 的實作。
- **stu.h**：此文件需包含上述 **函式** 的定義。

另外，本題目使用以下 Makefile 進行批改：

```
all: main.c stu.o
    gcc main.c stu.o
stu.o: stu.c
    gcc -c stu.c
clean:
    rm -rf *.o *~ *.out
```

本題的執行結果可參考如下：

```
[3:23▲user@ws▲p2]▲./a.out↵
Number▲of▲students:▲3↵
```

```
↵
Student▲1:↵
Name:▲Jun▲Wu↵
Score:▲98↵
Student▲2:↵
Name:▲Markus▲Danfol↵
Score:▲61↵
Student▲3:↵
Name:▲Dennis▲Black↵
Score:▲89↵
↵
Students▲(Sorted▲by▲score,▲in▲descending▲order):↵
Jun▲Wu:▲98.00↵
Dennis▲Black:▲89.00↵
Markus▲Danfol:▲61.00↵
↵
Average▲score:▲82.666667↵
[3:23▲user@ws▲p2]▲./a.out↵
Number▲of▲students:▲1↵
↵
Student▲1:↵
Name:▲Test↵
Score:▲101↵
Invalid▲score!↵
[3:23▲user@ws▲p2]▲./a.out↵
Number▲of▲students:▲1↵
↵
Student▲1:↵
Name:▲Test↵
Score:▲-1↵
Invalid▲score!↵
[3:23▲user@ws▲p2]▲./a.out↵
Number▲of▲students:▲1↵
↵
Student▲1:↵
Name:▲T3st↵
Invalid▲name!↵
[3:23▲user@ws▲p2]▲./a.out↵
Number▲of▲students:▲1↵
↵
Student▲1:↵
Name:▲Tes$t↵
Invalid▲name!↵
[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 (**main.c** 及 **Makefile** 則不需繳交)：

- **stu.c**
- **stu.h**

P3 雷阿斗的紀念品商店（續・課本 12-29 到 12-37 頁，程式演練 21）

雷阿斗看完你提供的測試版後，希望身為開發者的你再幫他為系統加入關於 **馬克杯(mug)** 的品項以及 一系列馬克杯上的圖案，有 **海灘 (beach)、山 (mountain)、及 村莊(village)**三個圖案。

因此，請同學到系計中上，路徑 `/home/stu/public/cbook/Projects/21` 下取得以下檔案後：

- **ProductInfo.c**
- **ProductInfo.h**
- **SouvenirShopMain.c**
- **Makefile**

將圖案 (Pattern) 的列舉加入到 **ProductInfo.h**中，並加以修改 **ProductInfo.c** 內容，以符合雷阿斗要求，請參考以下 **SouvenirShopMain.c** 的實作：

```
#include <stdio.h>
#include "ProductInfo.h"

int main()
{
  Product products[10];
  int i=0,d;
  int id;
  boolean quit=false;
  char cmd;
  while(!quit)
  {
    printf("command?");
    cmd=getchar();
    switch(cmd)
    {
      case 'h':
        showHelp();
        getchar();
        break;
      case 'i':
        products[i++]=getAProductInfo();
        break;
      case 'l':
        showAllProductInfo(products, i);
        getchar();
```

```
        break;
      case 's':
        printf("ID?");
        scanf(" %d", &id);
        searchProduct(products, i, id);
        getchar();
        break;
      case 'q':
        quit=true;
        break;
      case 10:
        printf("Wrong command!\n");
        break;
      default:
        printf("Wrong command!\n");
        getchar();
        break;
    }
  }
}
```

因此，本題需請同學定義並實作下列列舉：

- **Pattern**

完成後，將修改後的內容放入以下檔案：

- **ProductInfo.c**：此文件需包含修改後函式的實作。
- **ProductInfo.h**：此文件需包含上述新增的列舉以及其他原有的函式、列舉、及結構體的定義。

另外，本題目使用以下 Makefile 進行批改：

```
all: SouvenirShopMain.c ProductInfo.o
    cc SouvenirShopMain.c ProductInfo.o

ProductInfo.o: ProductInfo.c ProductInfo.h
    cc ProductInfo.c -c
clean:
    rm -f *.c~ *.h~ *~ *.o a.out
```

本題的執行結果可參考如下：

```
[3:23▲user@ws▲p3]▲./a.out↵
command?h↵
i:▲insert▲a▲new▲product.↵
l:▲list▲all▲products.↵
```

```
s:▲search▲for▲a▲product.↵
q:▲quit.↵
command?i↵
ID=?10↵
price=?12.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?m↵
Pattern▲(b▲for▲beach,▲m▲for▲mountain,▲v▲for▲village)?b↵
command?l↵
ID:10▲price=12.59▲Mug(▲beach▲)↵
command?i↵
ID=?11↵
price=?11.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?m↵
Pattern▲(b▲for▲beach,▲m▲for▲mountain,▲v▲for▲village)?v↵
command?s↵
ID?11↵
ID:11▲price=11.59▲Mug(▲village▲)↵
command?q↵
[3:23▲user@ws▲p3]▲./a.out↵
command?h↵
i:▲insert▲a▲new▲product.↵
l:▲list▲all▲products.↵
s:▲search▲for▲a▲product.↵
q:▲quit.↵
command?i↵
ID=?10↵
price=?12.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?k↵
Material▲(c▲for▲copper,▲s▲for▲steel,▲w▲for▲wood,▲p▲for▲plastic▲)?c↵
command?l↵
ID:10▲price=12.59▲Keychain(▲copper▲)↵
command?i↵
ID=?15↵
price=?11.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?t↵
Size▲(XS,▲S,▲M,▲L,▲XL,▲XXL)?M↵
command?s↵
ID?12↵
Product▲not▲found!↵
command?s↵
ID?11↵
Product▲not▲found!↵
command?s↵
ID?15↵
ID:15▲price=11.59▲T▲Shirt(▲M▲)↵
command?q↵
[3:23▲user@ws▲p3]▲./a.out↵
command?H↵
Wrong▲command!↵
```

```
command?i↵
ID=?124↵
price=?33.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?b↵
Author?Jun▲Wu↵
command?l↵
ID:124▲price=33.59▲Book(▲Jun▲Wu▲)↵
command?i↵
ID=?11↵
price=?11.59↵
type▲(b▲for▲book,▲k▲for▲keychain,▲t▲for▲T-shirt,▲m▲for▲mug)=?m↵
Pattern▲(b▲for▲beach,▲m▲for▲mountain,▲v▲for▲village)?m↵
command?l↵
ID:124▲price=33.59▲Book(▲Jun▲Wu▲)↵
ID:11▲price=11.59▲Mug(▲mountain▲)↵
command?q↵
[3:23▲user@ws▲p3]
```

- 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（**SouvenirShopMain.c**、**Makefile** 則不需繳交）：
  - **ProductInfo.c**
  - **ProductInfo.h**

P4 Dell的熟食餐館 (Dell's Deli)

Dell在舊金山新開了一家自己的熟食餐館，由於餐點美味且選擇多樣，開幕不久便吸引了大量顧客，生意絡繹不絕。然而，隨著點餐人潮急遽增加，傳統人工手寫的記帳與點餐方式已經無法應付繁忙的客流量，有時甚至會出現漏單或算錯金額的窘境。

為了解決這個問題，本題需請同學幫 Dell 完成一個的數位訂餐系統的測試版。該程式在執行後，需提供一個互動式的指令介面，讓使用者可以持續進行操作：輸入 **'m'** 顯示菜單、輸入 **'a'** 新增餐點、輸入 **'r'** 移除餐點、輸入 **'l'** 檢視訂單，以及輸入 **'q'** 進行結帳處理。程式背後需透過一系列對應的函式來實現這些功能。

為了順利處理訂單記錄，**同學需在 deli.h 檔案中自行定義相關結構體**，用以記錄 **單一餐點 (item)** 與 **整筆訂單（order）**，具體規範如下：

- **單一餐點 (item)**，裡面需包含以下欄位：
  - 一個能記錄餐點 **名稱**，其長度 **不超過 19 個字元** 的字串。
  - 一個對應 **價格** 的整數。
- **整筆訂單 (order)**，裡面需包含以下欄位：
  - 一個長度為 **MAX_ORDER_ITEMS** 的餐點陣列（用以儲存上述的單一餐點）。
  - 一個用來記錄當前訂單中實際餐點數量的整數。

在實作各項操作時，請依照以下 **左邊編號順序** 進行錯誤判斷與處理（**括號中的內容** 請依據提示替換，換行與排版等輸出請參考下方的輸出範例）：

- 當輸入選項 **(a)** 時：
  1. 若新增物品後會超出 **MAX_ORDER_ITEMS** 所定義的數值時，需輸出“ **You cannot add more than (MAX_ORDER_ITEMS 的數值) items to your order!** ”並且立即返回主程式。
  2. 若使用者輸入的物品名稱 **無法被找到** 時，需輸出“ **(使用者輸入的物品名稱) is not on the menu!** ”並且立即返回主程式。
- 當輸入選項 **(r)** 時：
  1. 若當前的訂單中 **沒有物品** 時，需輸出“ **You have no items to remove from your order!** ”並且立即返回主程式。
  2. 若使用者輸入的物品編號 **超出範圍** 時，需輸出“ **That number is out of range! Please enter a number between 1 and (當前訂單的總數).** ”並且立即返回主程式。
- 當輸入選項 **(l)** 時：
  1. 若當前的訂單中 **沒有物品** 時，需輸出“ **Your order is empty, try order some!** ”並且立即返回主程式。
- 當輸入選項 **(q)** 時：
  1. 若當前的訂單中沒有物品時，則結束程式。
  2. 若使用者輸入的付款金額 **小於** 0 時，需輸出“ **You cannot enter a negative payment!** ”並讓使用者重新輸入。
  3. 若使用者輸入的付款金額 **小於** 應付金額或剩餘金額時，需先輸出“ **That is not enough!** “，接著輸出“ **You still need to pay $(剩餘應付金額).** “，並讓使用者輸入剩餘金額。
  4. 若使用者輸入的付款金額 **大於** 應付金額或剩餘金額時，需先輸出“ **Here is your order.** “，接著輸出“ **And the change: $(多餘的找零金額)**“
  5. 若使用者輸入的付款金額 **剛好等於** 應付金額或剩餘金額時，需輸出“ **Here is your order.**“

實作注意

- 計算浮點數時請以 **double** 型態處理。
- 批改時，**MAX_ORDER_ITEMS** 將會採用 **不同數值**，本題 **define.h** 中提供的大小 3 限制僅為了測試方便，請務必使用常數名稱進行陣列大小與邊界判斷。

請參考以下 **define.h** 與 **main.c** 的實作：

```
#define MAX_ORDER_ITEMS 3

typedef enum
{
    Unknown = -1,
    Sandwich = 8,
    Pizza = 12,
    Burger = 10,
```

```
    Chipotle = 11,
    Steak = 18,
    Cola = 3,
    Tea = 2,
    Milk = 4,
    Water = 1
} food;
```

```
#include <stdio.h>
#include <ctype.h>

#include "deli.h"

void deliShowChoices()
{
    printf("> (m) Show menu\n");
    printf("> (a) Add item to order\n");
    printf("> (r) Remove item from order\n");
    printf("> (l) List items in order\n");
    printf("> (q) Checkout and quit\n");
}

int main() {
    order myOrder = {0};
    char choice = 0;

    printf("Welcome to Dell's Deli!\n");
    printf("How can we help you today?\n");

    do
    {
        printf("\n");
        deliShowChoices();
        printf("-- Choice: ");
        scanf(" %c", &choice);
        switch (tolower(choice))
        {
            case 'm':
                printf("\n");
                deliShowMenu();
                break;
            case 'a':
                printf("\n");
                deliAddItem(&myOrder);
                break;
            case 'r':
                printf("\n");
                deliRemoveItem(&myOrder);
```

```
                break;
            case 'l':
                printf("\n");
                deliListOrder(&myOrder);
                break;
            case 'q':
                printf("\n");
                deliCheckout(&myOrder);
                printf("Thank you for visiting Dell's Deli!\n");
                break;
            default:
                printf("\nInvalid choice! Please try again.\n");
        }
    } while (choice != 'q' && choice != 'Q');

    return 0;
}
```

因此，本題需請同學定義並實作下列函式和結構體：

- **結構體**:
  - **item**
  - **order**
- **函式**:
  - **deliShowMenu()**
  - **deliAddItem()**
  - **deliRemoveItem()**
  - **deliListOrder()**
  - **deliCheckout()**

完成後，將內容放入以下檔案：

- **deli.c**：此文件需包含上述 **函式** 的實作。
- **deli.h**：此文件需包含上述 **函式** 及 **結構體** 的定義。

另外，本題目使用以下 Makefile 進行批改：

```
all: main.c deli.o
    gcc main.c deli.o
deli.o: deli.c
    gcc -c deli.c
clean:
    rm -rf *.o *~ *.out
```

本題的執行結果可參考如下：

```
[3:23▲user@ws▲p4]▲./a.out↵
Welcome▲to▲Dell's▲Deli!↵
How▲can▲we▲help▲you▲today?↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲b↵
↵
Invalid▲choice!▲Please▲try▲again.↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲L↵
↵
Your▲order▲is▲empty,▲try▲order▲some!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲M↵
↵
Here▲is▲our▲menu:↵
>▲Sandwich▲-▲\$8↵
>▲Pizza▲-▲\$12↵
>▲Burger▲-▲\$10↵
>▲Chipotle▲-▲\$11↵
>▲Steak▲-▲\$18↵
>▲Cola▲-▲\$3↵
>▲Tea▲-▲\$2↵
>▲Milk▲-▲\$4↵
>▲Water▲-▲\$1↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲q↵
↵
Thank▲you▲for▲visiting▲Dell's▲Deli!↵
[3:23▲user@ws▲p4]
```

```
[3:23▲user@ws▲p4]▲./a.out↵
Welcome▲to▲Dell's▲Deli!↵
How▲can▲we▲help▲you▲today?↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲m↵
↵
Here▲is▲our▲menu:↵
>▲Sandwich▲-▲\$8↵
>▲Pizza▲-▲\$12↵
>▲Burger▲-▲\$10↵
>▲Chipotle▲-▲\$11↵
>▲Steak▲-▲\$18↵
>▲Cola▲-▲\$3↵
>▲Tea▲-▲\$2↵
>▲Milk▲-▲\$4↵
>▲Water▲-▲\$1↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲Sandwich↵
Okay,▲one▲Sandwich▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲q↵
↵
That▲will▲be▲\$8▲in▲total.↵
--▲Payment:▲8↵
↵
Here▲is▲your▲order.↵
Thank▲you▲for▲visiting▲Dell's▲Deli!↵
[3:23▲user@ws▲p4]
```

```
[3:23 user@ws p4] ./a.out
Welcome to Dell's Deli!
How can we help you today?

>(m) Show menu
>(a) Add item to order
>(r) Remove item from order
>(l) List items in order
>(q) Checkout and quit
--Choice: m

Here is our menu:
> Sandwich - \$8
> Pizza - \$12
> Burger - \$10
> Chipotle - \$11
> Steak - \$18
> Cola - \$3
> Tea - \$2
> Milk - \$4
> Water - \$1

>(m) Show menu
>(a) Add item to order
>(r) Remove item from order
>(l) List items in order
>(q) Checkout and quit
--Choice: A

Enter the name of the food to add
--Name: SteaK
Okay, one Steak coming right up!

>(m) Show menu
>(a) Add item to order
>(r) Remove item from order
>(l) List items in order
>(q) Checkout and quit
--Choice: a

Enter the name of the food to add
--Name: cola
Okay, one Cola coming right up!

>(m) Show menu
>(a) Add item to order
>(r) Remove item from order
>(l) List items in order
>(q) Checkout and quit
```

```
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲wATER↵
Okay,▲one▲Water▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
You▲cannot▲add▲more▲than▲3▲items▲to▲your▲order!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲Q↵
↵
That▲will▲be▲\$22▲in▲total.↵
--▲Payment:▲50↵
↵
Here▲is▲your▲order.↵
And▲the▲change:▲\$28↵
Thank▲you▲for▲visiting▲Dell's▲Deli!↵
[3:23▲user@ws▲p4]
```

```
[3:23▲user@ws▲p4]▲./a.out↵
Welcome▲to▲Dell's▲Deli!↵
How▲can▲we▲help▲you▲today?↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲m↵
↵
Here▲is▲our▲menu:↵
>▲Sandwich▲-▲\$8↵
>▲Pizza▲-▲\$12↵
>▲Burger▲-▲\$10↵
>▲Chipotle▲-▲\$11↵
```

```
>▲Steak▲-▲$18↵
>▲Cola▲-▲$3↵
>▲Tea▲-▲$2↵
>▲Milk▲-▲$4↵
>▲Water▲-▲$1↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲Pizza↵
Okay,▲one▲Pizza▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲colA↵
Okay,▲one▲Cola▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲Chipotla↵
Chipotla▲is▲not▲on▲the▲menu!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲Chipotle↵
Okay,▲one▲Chipotle▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
```

```
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲Q↵
↵
That▲will▲be▲\$26▲in▲total.↵
--▲Payment:▲12↵
↵
That▲is▲not▲enough!↵
You▲still▲need▲to▲pay▲\$14.↵
--▲Payment:▲3↵
↵
That▲is▲not▲enough!↵
You▲still▲need▲to▲pay▲\$11.↵
--▲Payment:▲15↵
↵
Here▲is▲your▲order.↵
And▲the▲change:▲\$4↵
Thank▲you▲for▲visiting▲Dell's▲Deli!↵
[3:23▲user@ws▲p4]
```

```
[3:23▲user@ws▲p4]▲./a.out↵
Welcome▲to▲Dell's▲Deli!↵
How▲can▲we▲help▲you▲today?↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲m↵
↵
Here▲is▲our▲menu:↵
>▲Sandwich▲-▲\$8↵
>▲Pizza▲-▲\$12↵
>▲Burger▲-▲\$10↵
>▲Chipotle▲-▲\$11↵
>▲Steak▲-▲\$18↵
>▲Cola▲-▲\$3↵
>▲Tea▲-▲\$2↵
>▲Milk▲-▲\$4↵
>▲Water▲-▲\$1↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
```

```
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲SteaK↵
Okay,▲one▲Steak▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲cola↵
Okay,▲one▲Cola▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲wATER↵
Okay,▲one▲Water▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲l↵
↵
Here▲are▲the▲items▲in▲your▲order:↵
>▲#1▲Steak▲-▲\$18↵
>▲#2▲Cola▲-▲\$3↵
>▲#3▲Water▲-▲\$1↵
Total:▲\$22↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
```

```
Enter▲the▲item▲number▲to▲remove▲from▲your▲order↵
--▲Number:▲-1↵
That▲number▲is▲out▲of▲range!▲Please▲enter▲a▲number▲between▲1▲and▲3.↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
Enter▲the▲item▲number▲to▲remove▲from▲your▲order↵
--▲Number:▲4↵
That▲number▲is▲out▲of▲range!▲Please▲enter▲a▲number▲between▲1▲and▲3.↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
Enter▲the▲item▲number▲to▲remove▲from▲your▲order↵
--▲Number:▲3↵
Okay,▲and▲...▲done!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲l↵
↵
Here▲are▲the▲items▲in▲your▲order:↵
>▲#1▲Steak▲-▲\$18↵
>▲#2▲Cola▲-▲\$3↵
Total:▲\$21↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
Enter▲the▲item▲number▲to▲remove▲from▲your▲order↵
--▲Number:▲1↵
Okay,▲and▲...▲done!↵
↵
```

```
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲l↵
↵
Here▲are▲the▲items▲in▲your▲order:↵
>▲#1▲Cola▲-▲\$3↵
Total:▲\$3↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
Enter▲the▲item▲number▲to▲remove▲from▲your▲order↵
--▲Number:▲1↵
Okay,▲and▲...▲done!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲l↵
↵
Your▲order▲is▲empty,▲try▲order▲some!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲r↵
↵
You▲have▲no▲items▲to▲remove▲from▲your▲order!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲m↵
↵
Here▲is▲our▲menu:↵
>▲Sandwich▲-▲\$8↵
>▲Pizza▲-▲\$12↵
>▲Burger▲-▲\$10↵
```

```
>▲Chipotle▲-▲\$11↵
>▲Steak▲-▲\$18↵
>▲Cola▲-▲\$3↵
>▲Tea▲-▲\$2↵
>▲Milk▲-▲\$4↵
>▲Water▲-▲\$1↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲SteaK↵
Okay,▲one▲Steak▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲a↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲cola↵
Okay,▲one▲Cola▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲A↵
↵
Enter▲the▲name▲of▲the▲food▲to▲add↵
--▲Name:▲wATER↵
Okay,▲one▲Water▲coming▲right▲up!↵
↵
>▲(m)▲Show▲menu↵
>▲(a)▲Add▲item▲to▲order↵
>▲(r)▲Remove▲item▲from▲order↵
>▲(l)▲List▲items▲in▲order↵
>▲(q)▲Checkout▲and▲quit↵
--▲Choice:▲q↵
↵
That▲will▲be▲\$22▲in▲total.↵
--▲Payment:▲18↵
```

```
↵
That▲is▲not▲enough!↵
You▲still▲need▲to▲pay▲\$4.↵
--▲Payment:▲3↵
↵
That▲is▲not▲enough!↵
You▲still▲need▲to▲pay▲\$1.↵
--▲Payment:▲1↵
↵
Here▲is▲your▲order.↵
Thank▲you▲for▲visiting▲Dell's▲Deli!↵
[3:23▲user@ws▲p4]
```

- 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**、**Makefile**及**define.h** 則不需繳交）：
  - **deli.c**
  - **deli.h**

P5 五子棋：活三死四判斷

五子棋是一項非常有趣的棋奕遊戲，由分持白子與黑子的雙方在縱橫19道的棋盤上輪流落子。在五子棋的對弈中，除了落下的每一枚棋子外，對棋盤的精確判斷更是勝負的關鍵！

其中，「**活三**」與「**死四**」是進攻與防守時最常見的判斷指標。為了簡化問題，本題將針對棋盤中其中某一行進行判斷，識別該行是否具有活三與死四的棋型。請參考以下說明：

- 在棋盤中：
  - 字元'**1**' 代表黑子。
  - 字元'**0**' 代表白子。
  - 字元'**.**' 代表未落子處。
- **活三 (LIVE_THREE)**：
  - 連續 **3** 顆同色棋子。
  - 條件：兩端必須同時為空位 ('**.**')。
  - 範例：`. 1 1 1 .`。
- **死四 (DEAD_FOUR)**：
  - 連續 **4** 顆同色棋子。
  - 條件：一端是空位 (**.**)，且另一端為異色棋子或棋盤邊界。
  - 範例：`0 1 1 1 1 .`。
- 同學需自行於 **chess.h** 定義以下結構體：
  - **chess_record_t**：
    - start：一個對應 起始位置 的整數。
    - end：一個對應 結束位置 的整數。
    - color：一個對應 棋子顏色 的整數。
    - type：表示該資料的棋型，型別需使用系統提供的 pattern_t 列舉（請參考下方的

define.h□□

因此同學需完成以下函式的實作：

- **analyzeChess()**□
  - 負責巡檢棋盤行，識別符合的棋型並存入紀錄陣列，並更新count□記錄陣列的大小）。

1. 為避免重複偵測，當識別出一個棋型區間後，掃描指標應跳至該區間的後方繼續偵測。
   - 例如：若棋盤為 `. 1 1 1 . 0`，當偵測到索引 **[1]~[3]** 符合黑子活三時，下一波偵測應跳過該區間從索引 **[4]** 開始，而非從 **[2]** 重新開始。
2. 本題不需考慮洞三、洞死四等其他棋型。
3. 本題輸出需依照起始座標□start□的索引值，由小到大依序呈現。

完成後，同學應繳交下列檔案：

- **chess.c**：此文件需包含analyzeChess()函式的實作。
- **chess.h**：此文件需包含結構體定義與函式宣告。

請參考以下 main.c, define.h 的內容：

```
#define MAX_BOARD_SIZE 19
#define MAX_RECORDS 4

typedef enum
{
    LIVE_THREE,
    DEAD_FOUR
} pattern_t;
```

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

void chessPrint(chess_record_t* records, int count) {
    if (count == 0) {
        printf("No LIVE_THREE or DEAD_FOUR detected.\n");
        return;
    }

    int liveThreeCount = 0;
    int deadFourCount = 0;

    for (int i = 0; i < count; i++) {
        if (records[i].type == LIVE_THREE) liveThreeCount++;
        else if (records[i].type == DEAD_FOUR) deadFourCount++;
```

```
    }

    printf("LIVE_THREE: %d, DEAD_FOUR: %d\n", liveThreeCount,
deadFourCount);

    for (int i = 0; i < count; i++) {
        printf("%d ~ %d: %s %s\n",
            records[i].start + 1,
            records[i].end + 1,
            (records[i].color == 1) ? "Black" : "White",
            (records[i].type == LIVE_THREE) ? "LIVE_THREE" : "DEAD_FOUR"
        );
    }
}

int main()
{
    char chessboard[MAX_BOARD_SIZE];
    for (int i = 0; i < MAX_BOARD_SIZE; i++)
    {
        scanf(" %c", &chessboard[i]);
    }

    char* p = &chessboard[0];
    int count = 0;
    chess_record_t records[MAX_RECORDS];

    analyzeChess(p, records, &count);
    chessPrint(records, count);
}
```

另外本題將使用以下 Makefile 進行編譯：

```
all: main.c chess.o
    gcc main.c chess.o
chess.o: chess.c
    gcc -c chess.c
clean:
    rm -rf *.o *~ *.out
```

本題的執行結果可參考如下：

```
 [3:23▲user@ws▲p5]▲./a.out↵
 .▲1▲1▲1▲.▲0▲0▲0▲0▲.▲1▲1▲1▲1▲1▲.▲0▲0▲0↵
 LIVE_THREE:▲1,▲DEAD_FOUR:▲0↵
 2▲~▲4:▲Black▲LIVE_THREE↵
 [3:23▲user@ws▲p5]▲./a.out↵
```

```
1▲1▲1▲.▲.▲.▲.▲1▲0▲1▲1▲1▲0▲.▲.▲.▲.▲0▲.↵
No▲LIVE_THREE▲or▲DEAD_FOUR▲detected.↵
[3:23▲user@ws▲p5]▲./a.out↵
.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲0▲0▲0▲0↵
LIVE_THREE:▲0,▲DEAD_FOUR:▲1↵
16▲~▲19:▲White▲DEAD_FOUR↵
[3:23▲user@ws▲p5]▲./a.out↵
.▲.▲.▲0▲0▲0▲0▲1▲1▲1▲1▲.▲.▲.▲.▲.▲.▲.▲.↵
LIVE_THREE:▲0,▲DEAD_FOUR:▲2↵
4▲~▲7:▲White▲DEAD_FOUR↵
8▲~▲11:▲Black▲DEAD_FOUR↵
[3:23▲user@ws▲p5]▲./a.out↵
.▲1▲1▲1▲.▲0▲0▲0▲0▲1▲1▲1▲1▲.▲.▲0▲0▲0▲.↵
LIVE_THREE:▲2,▲DEAD_FOUR:▲2↵
2▲~▲4:▲Black▲LIVE_THREE↵
6▲~▲9:▲White▲DEAD_FOUR↵
10▲~▲13:▲Black▲DEAD_FOUR↵
16▲~▲18:▲White▲LIVE_THREE↵
[3:23▲user@ws▲p5]
```

- 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）：
  - **chess.c**
  - **chess.h**

P6 五子棋：洞三與洞死四判斷

承上題，在五子棋的對弈中，除了標準的活三與死四外，具有「間隔」的棋型（俗稱中洞、跳三、跳四）往往更具隱蔽性，對盤勢判斷同樣至關重要！本題將挑戰偵測包含空白格位在內的特殊棋型：洞三 與 洞死四。

- **洞三 (HOLE_THREE)**：
  - 連續 **3** 顆同色棋子與 1 個中洞組成。
  - 條件：有中洞，區間兩端必須同時為空位 (**.**)。
  - 範例：`. 1 . 1 1 .` 或 `. 1 1 . 1 .`。
  - **註：此題不考慮 `. 1 . 1 . 1 .` 洞三情況。**
- **洞死四 (HOLE_DEAD_FOUR)**：
  - 連續 **4**顆同色棋子與 **1** 個中洞組成。
  - 條件：有中洞，區間兩端其中一端為異色棋子或棋盤邊界。
  - 範例：`0 1 1 . 1 1 0`、`. 1 . 1 1 1 0`、`. 1 1 1 . 1 0`。
- 同學需自行於 **chess.h** 並定義與 P6 相同結構的結構體：
  - **chess_record_t**

- start：一個對應 起始位置 的整數。
- end：一個對應 結束位置 的整數。
- color：一個對應 棋子顏色 的整數。
- type：表示該資料的棋型，型別需使用系統提供的 pattern_t 列舉（請參考下方的 define.h）

因此同學需完成以下函式的實作：

- **analyzeChess()**：
  - 負責巡檢棋盤行，識別符合的棋型並存入紀錄陣列，並更新count（記錄陣列的大小）。

1. 為避免重複偵測，當識別出一個棋型區間後，掃描指標應跳至該區間的後方繼續偵測。
   - 例如：若棋盤為 `1 . 1 1 . 1 1 0 . 1`，當偵測到索引 **[2]~[6]** 符合黑子活三時，下一波偵測應跳過該區間從索引 **[7]** 開始，而非從 **[3]** 重新開始。
2. 本題不需考慮活三、死四等其他棋型。
3. 不考慮兩個洞的洞三。
4. 本題輸出輸出依照起始座標（start）的索引值，由小到大依序呈現。

完成後，同學應繳交下列檔案：

- **chess.c**：此文件需包含 analyzeChess() 函式的實作。
- **chess.h**：此文件需包含結構體定義與函式宣告。

請參考以下 main.c, define.h 的內容：

```
#define MAX_BOARD_SIZE 19
#define MAX_RECORDS 4

typedef enum
{
    HOLE_THREE,
    HOLE_DEAD_FOUR
} pattern_t;
```

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

void chessPrint(chess_record_t* records, int count) {
    if (count == 0) {
        printf("No HOLE_THREE or HOLE_DEAD_FOUR detected.\n");
        return;
    }

    int holeThreeCount = 0;
    int holeDeadFourCount = 0;
```

```
    for (int i = 0; i < count; i++) {
        if (records[i].type == HOLE_THREE) holeThreeCount++;
        else if (records[i].type == HOLE_DEAD_FOUR) holeDeadFourCount++;
    }

    printf("HOLE_THREE: %d, HOLE_DEAD_FOUR: %d\n", holeThreeCount,
holeDeadFourCount);

    for (int i = 0; i < count; i++) {
        printf("%d ~ %d: %s %s\n",
            records[i].start + 1,
            records[i].end + 1,
            (records[i].color == 1) ? "Black" : "White",
            (records[i].type == HOLE_THREE) ? "HOLE_THREE" :
"HOLE_DEAD_FOUR"
        );
    }
}

int main()
{
    char chessboard[MAX_BOARD_SIZE];
    for (int i = 0; i < MAX_BOARD_SIZE; i++)
    {
        scanf(" %c", &chessboard[i]);
    }

    char* p = &chessboard[0];
    int count = 0;
    chess_record_t records[MAX_RECORDS];

    analyzeChess(p, records, &count);
    chessPrint(records, count);
}
```

```
all: main.c chess.o
    gcc main.c chess.o
chess.o: chess.c
    gcc -c chess.c
clean:
    rm -rf *.o *~ *.out
```

本題的執行結果可參考如下：

```
 [3:23▲user@ws▲p6]▲./a.out↵
```

```
.▲1▲.▲1▲1▲.▲0▲0▲.▲0▲0▲.▲.▲.▲.▲.▲.▲.▲.↵
HOLE_THREE:▲1,▲HOLE_DEAD_FOUR:▲0↵
2▲~▲5:▲Black▲HOLE_THREE↵
[3:23▲user@ws▲p6]▲./a.out↵
.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲0▲0▲0▲.▲0↵
HOLE_THREE:▲0,▲HOLE_DEAD_FOUR:▲1↵
15▲~▲19:▲White▲HOLE_DEAD_FOUR↵
[3:23▲user@ws▲p6]▲./a.out↵
.▲1▲1▲.▲1▲1▲0▲.▲.▲.▲.▲.▲.▲.▲.▲1▲1▲1▲0↵
HOLE_THREE:▲0,▲HOLE_DEAD_FOUR:▲1↵
2▲~▲6:▲Black▲HOLE_DEAD_FOUR↵
[3:23▲user@ws▲p6]▲./a.out↵
.▲1▲.▲1▲1▲.▲.▲0▲0▲.▲0▲.▲0▲.▲1▲.▲1▲1▲1↵
HOLE_THREE:▲2,▲HOLE_DEAD_FOUR:▲1↵
2▲~▲5:▲Black▲HOLE_THREE↵
8▲~▲11:▲White▲HOLE_THREE↵
15▲~▲19:▲Black▲HOLE_DEAD_FOUR↵
[3:23▲user@ws▲p6]▲./a.out↵
1▲1▲1▲1▲.▲1▲1▲.▲1▲1▲0▲.▲.▲.▲.▲.▲.▲.▲.↵
HOLE_THREE:▲0,▲HOLE_DEAD_FOUR:▲1↵
6▲~▲10:▲Black▲HOLE_DEAD_FOUR↵
[3:23▲user@ws▲p6]▲./a.out↵
.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲.▲1▲.▲.▲.▲.▲.↵
No▲HOLE_THREE▲or▲HOLE_DEAD_FOUR▲detected.↵
[3:23▲user@ws▲p6]
```

- 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）：
  - **chess.c**
  - **chess.h**