本地練習題

5 題 · 本地 clang　·　總分 動手寫

做法：讀題 → 自己在編輯器寫 → `./run.sh 你的檔.c` 編譯執行（已開 `-fsanitize=address,undefined`）→ 卡住或想對答案再看 `參考解答/` 裡對應的 `.c`。

參考解答是可直接編譯執行的程式檔（`參考解答/01_swap.c` … `05_dynamic_max.c`）。危險操作（懸空、use-after-free、leak）一定在本地用 sanitizer 跑，別只在瀏覽器試。

§ 1　練習題

練習 1：手寫 swap（傳位址）

寫一個 `swap` 函式與 `main`，使呼叫 `swap(&x, &y)` 後 `x`、`y` 互換並印出。並用一句話說明為何不能用傳值版 `void swap(int a, int b)`。

引導步驟

1. 參數收位址：`int *a, int *b`。
2. 用暫存變數，透過 `*a`、`*b` 讀寫指向的值。
3. `main` 裡 `swap(&x, &y)`，前後各印一次。

易錯

· 參數寫成傳值（交換無效）；呼叫時忘了 `&`。　**參考解答：`參考解答/01_swap.c`**

練習 2：動態配置一維陣列

寫 `int *make_squares(int n)`：動態配置 `n` 個 `int`，填入 `0,1,4,…,(n-1)²`，回傳指標；`main` 印出後 `free`。要含 `NULL` 檢查。

引導步驟

1. `malloc(n * sizeof(int))` → 檢查 `NULL`。
2. 迴圈 `a[i] = i * i`。
3. 回傳指標；呼叫端用完 `free` 並設 `NULL`。

易錯

· `malloc(n)` 漏乘 `sizeof(int)`；沒檢查 `NULL`；呼叫端忘了 `free`。 **參考解答：`參考解答/02_make_squares.c`**

練習 3：抓 bug 並修正（懸空指標）

下列函式為何危險？指出 bug 名稱，並用 `malloc` 改寫成安全版（呼叫端負責 `free`）。用 `./run.sh` 跑你的修正版，ASan 應乾淨。

```c
int *make_array(void) {
    int local[3] = {1, 2, 3};
    return local;            /* ? */
}
```

引導步驟

1. `local` 在 stack 還是 heap？函式回傳後還在嗎？
2. 回傳的位址叫什麼指標？為何不能用？
3. 改放 heap 讓它活過函式。

易錯

· 只說「會出錯」沒講出「懸空指標／stack 回收」；改 malloc 後忘了 `free`。 **參考解答：`參考解答/03_make_array.c`**

練習 4：用指標走訪陣列

給 `int a[6] = {3,1,4,1,5,9};`，**用指標（不用 `a[i]` 下標語法）**走訪，印出每個元素並求總和。

> **引導步驟**
>
> 1. `int *p = a;` 指向首元素。
> 2. 用 `*(p + i)` 取值。
> 3. 累加總和。

易錯

- 把 `*(p+i)` 寫成 `*p+i`（運算優先序錯）。 **參考解答：`參考解答/04_pointer_traverse.c`**

練習 5：動態讀入並求最大值

讀入一個整數 `n`，動態配置 `n` 個 `int`，再讀 `n` 個數，印出最大值，最後 `free`。要含 `NULL` 檢查。

> **引導步驟**
>
> 1. `scanf("%d", &n)` → `malloc(n * sizeof(int))` → 檢查 `NULL`。
> 2. 迴圈 `scanf("%d", &a[i])`（注意 `&`）。
> 3. 掃一遍求 max，印出後 `free`。

易錯

- `malloc` 漏 `sizeof(int)`、沒檢查 `NULL`、`scanf` 漏 `&`、忘了 `free`。 **參考解答：`參考解答/05_dynamic_max.c`**