

第一步 下載 Zuvio 校園 APP

App Store 或 Google Play 中
下載「Zuvio 校園」



第二步 註冊

在 Zuvio 校園 APP 註冊並
登入帳號



第三步 加選課程

在「學習」功能中點擊+，
輸入課程通行碼：

113713813



Environment

- Don't use Dev C++ !!!!

- If you use windows, please install [mingw](#)
 - You can use “gcc” to compile
 - Then, use “./a.out” to execute
- For the editor, just use anyone you like
 - I would like to use VScode

```
C/week 02$ gcc week02_inversion.c
C/week 02$ ./a.out
```

Week 1 assignment

```
1 #include <stdio.h>
2 #define S 10
3
4 int main(void)
5 {
6     int layers, i, j, symmetric = 1, matrix[S][S];
7     printf("Num of layers: ");
8     scanf("%d", &layers);
9
10    for (i = 0; i < layers; i++)
11    {
12        for (j = 0; j < layers; j++)
13        {
14            scanf("%d", &matrix[i][j]);
15        }
16    }
17
18    for (i = 0; i < layers; i++)
19    {
20        for (j = 0; j < layers; j++)
21        {
22            if (i < j)
23            {
24                if (matrix[i][j] != matrix[j][i])
25                {
26                    symmetric = 0;
27                    break;
28                }
29            }
30        }
31    }
```

Topic 1: Pointer concepts

Pointer & reference

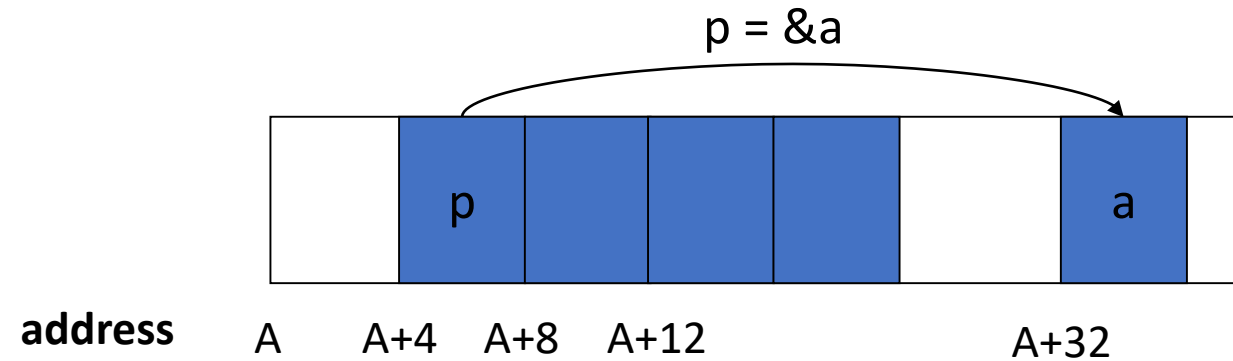
- Pointer concept
 - Pointer is **still a variable** but with a special usage
 - Declaration of a pointer → `int *p`
 - p is an address
 - A shortcut to a space
 - *p means the **content** that p points to
- Reference concept
 - Reference is the address space of a variable

Pointer & reference

```
#include <stdio.h>
int main (void)
{
    int *p;
    int a;

    a = 10;
    p = &a;

    return 0;
}
```



*p = 10	a = 10
p = A+32	&a = A+32
&p = A+4	
&>(*p) = A+32	

```
4  int main(void)
5  {
6      int a = 4;
7      int *p;
8
9      p=&a;
10     printf("%p %p %p %p\n", &a, &p, &(*p), p);
11
12 }
```

```
ryanpan@RyanPanPC /Volumes/MyWorks/D_Data/teaching/110/C/week 03$ ./a.out
0x7ffee94da85c 0x7ffee94da850 0x7ffee94da85c 0x7ffee94da85c
```

Pointer & array

- The interaction between pointer and array
 - `int ex_arr [5]`
 - `int *ptr;`
 - `ptr = ex_arr;`
 - \rightarrow `ex_arr[i]` equals to `*(ptr+i)`
`ptr = ptr + i` will move to the next `i` element of `ex_arr`
 - Don't use \rightarrow `ptr = &ex_arr;`
Because that array can be considered as a form of pointer

```
warning: incompatible pointer types assigning to 'int *' from 'int (*)[3]' [-Wincompatible-pointer-types]
```


Pointer & array (Quiz)

```
#include <stdio.h>

#define N 4
int main (void)
{
    int array[N] = {0};
    int *p;
    int i;

    p = array;

    array[0] = 1;
    p[0] = 2;
    *(p+1) = 3;

    p++;

    *(p+2) = 4;
    *(p+3) = 5;
```

Legal ! But abnormal
May not error !

```
p = array;

    for (i=0; i < N; i++)
    {
        printf("array[%d]: %d, *(p+%d): %d\n",
               i, array[i], i, *(p+i));
    }
}
```

array[0]: 2, *(p+0): 2
array[1]: 3, *(p+1): 3
array[2]: 0, *(p+2): 0
array[3]: 4, *(p+3): 4

Pointer & array (Don't do this!)

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

```
#define N 4
```

```
int main (void)
```

```
{
```

```
    int array[N] = {0};
```

```
    int arrayy[N] = {0};
```

```
    int *p;
```

```
    int i;
```

```
    printf("Array_ref: %p, Arrayy_ref: %p, \n",  
           p_ref: %p, i_ref: %p\n",  
           array, arrayy, &p, &i);
```

```
    p = array;
```

```
    array[0] = 1;
```

```
    p[0] = 2;
```

```
    *(p+1) = 3;
```

```
    p++;
```

```
    *(p+2) = 4;
```

```
Array_ref: 0xbfbfebe0,  
Arrayy_ref: 0xbfbfebd0,  
p_ref: 0xbfbfebcc,  
i_ref: 0xbfbfebcb
```

```
p = array;
```

```
for (i=0; i < N; i++)
```

```
{
```

```
    printf("array[%d]: %d, *(p+%d): %d\n",  
           i, array[i], i, *(p+i));
```

```
}
```

```
p = arrayy;
```

```
*(p+4) = 300;
```

Overwrite array[N]
(Memory pollution)

```
for (i=0; i < N; i++)
```

```
{
```

```
    printf("array[%d]: %d\n", i, array[i]);
```

```
}
```

```
}
```

```
array[0]: 2, *(p+0): 2  
array[1]: 3, *(p+1): 3  
array[2]: 0, *(p+2): 0  
array[3]: 4, *(p+3): 4
```

```
array[0]: 300  
array[1]: 3  
array[2]: 0  
array[3]: 4
```

Array allocation

```
#include <stdio.h>
#include <stdlib.h>
#define N 4
int main (void)
{
    int *array;
    int *p;
    int i;

    array = (int *) malloc (sizeof(int)*N);
    p = array;

    array[0] = 1;
    p[0] = 2;
    *(p+1) = 3;

    p++;

    *(p+2) = 4;
    *(p+3) = 5;
```

```
    p = array;

    for (i=0; i < N; i++)
    {
        printf("array[%d]: %d, *(p+%d): %d\n",
               i, array[i], i, *(p+i));
    }
}
```

Week 2 assignment

- Input $1+x^2$ numbers
(The first number is total number, $x>3$)
- After alignment, change first row and third row, and then first column and third column

```
$ ./a.exe 16 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

9 10 11 12
5 6 7 8
1 2 3 4
13 14 15 16

11 10 9 12
7 6 5 8
3 2 1 4
15 14 13 16
```

Requirements

- Let you know how to get rid of “array accessing”
- Only one set of brackets, i.e., [], is allowed in your program
 - That is command line argument → `int main(int argc, char *argv[])`
- Don't use [] to access (read or write) array elements

Something you may need

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

→ You can check out from the Internet