

Cấu trúc dữ liệu và giải thuật

TS. Phạm Tuấn Minh

Khoa Công nghệ Thông tin, Đại học Phenikaa

minh.phamtuan@phenikaa-uni.edu.vn

<https://sites.google.com/site/phamtuanminh/>

Chương 2: Mảng và danh sách liên kết

- ❑ Cấu trúc lưu trữ mảng
 - Mảng một chiều
 - Mảng nhiều chiều
- ❑ Danh sách liên kết
- ❑ Ngăn xếp
- ❑ Hàng đợi

Mảng một chiều

- ❑ **Khai báo mảng, khởi tạo giá trị, thao tác trên mảng**
- ❑ Con trỏ và mảng
- ❑ Mảng là tham số của hàm

1-3

- ❑ Tại sao học và sử dụng mảng?

1-4

Tại sao học và sử dụng mảng

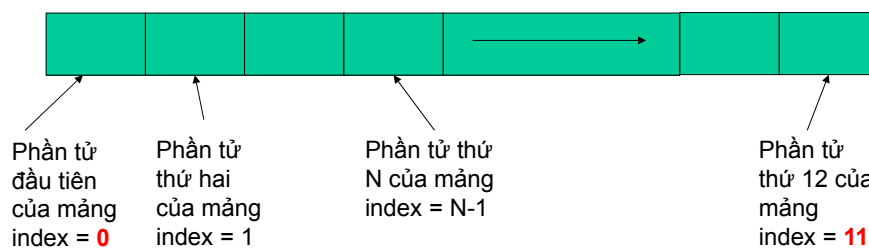
- ❑ Mảng là khái niệm cơ bản trong hầu hết các ngôn ngữ lập trình (ví dụ C, C++, Java, ...)
- ❑ Ý tưởng: Có biến dùng để tập hợp một nhóm các phần tử có cùng kiểu dữ liệu
- ❑ Ví dụ: Để mô tả sinh viên 50 sinh viên, nếu không sử dụng mảng, có thể cần định nghĩa 50 biến

1-5

Khái niệm mảng

- ❑ **Mảng** là một **danh sách** các phần tử có **cùng kiểu dữ liệu**. Giá trị của mỗi phần tử được lưu trữ tại vị trí được đánh số cụ thể trong mảng
- ❑ Mảng dùng một số nguyên làm chỉ số (**index**) để tham chiếu tới một phần tử trong mảng
- ❑ **Kích thước** của mảng là cố định mỗi khi tạo mảng
- ❑ Index bắt đầu với giá trị **0**

Mảng có kích thước bằng **12**



1-6

Khai báo mảng

- ❑ Khai báo mảng **không khởi tạo** giá trị của các phần tử

```
float sales[365]; /*mảng 365 phần tử float */
char name[12]; /*mảng 12 phần tử character*/
int states[50]; /*mảng 50 phần tử integer*/
int *pointers[5]; /* mảng 5 con trỏ tới phần tử integer */
```
- ❑ Khi khai báo mảng, chương trình dịch sẽ cấp phát một số vị trí **vùng nhớ liên tục** cho toàn bộ mảng (2 hoặc 4 byte cho kiểu integer tùy thuộc vào máy)

```
int h[4];
```

h[0]	h[1]	h[2]	h[3]
?	?	?	?

Phần tử:	1	2	3	4
Địa chỉ vùng nhớ:	2021	2023	2025	2027

- ❑ Kích thước mảng phải là **hằng số nguyên** hoặc **biểu thức hằng số nguyên**

```
char name[i]; // biến i ==> không hợp lệ
int states[i*6]; // biến i ==> không hợp lệ
```

1-7

Khởi tạo mảng

- ❑ Khởi tạo mảng khi khai báo mảng

```
#define MTHS 12 /* khai báo hằng số */
```

```
int days[MTHS]={31,28,31,30,31,30,31,31,30,31,30,31};
```

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
days	31	28	31	30	31	30	31	31	30	31	30	31

- ❑ Khởi tạo mảng một phần: Ví dụ khởi tạo 7 phần tử đầu tiên

```
#define MTHS 12 /* khai báo hằng số */
```

```
int days[MTHS]={31,28,31,30,31,30,31}; /* các phần tử còn lại sẽ được khởi tạo bằng 0 */
```

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
days	31	28	31	30	31	30	31	0	0	0	0	0

1-8

Khởi tạo mảng

- Bỏ qua kích thước mảng khi Khởi tạo mảng

```
int days[]={31,28,31,30,31,30,31}; /* mảng 7 phần tử */
```

	[0]	[1]	[2]	[3]	[4]	[5]	[6]
days	31	28	31	30	31	30	31

1-9

Thao tác trên mảng

- **Truy cập** phần tử mảng

```
sales[0] = 122.5;
```

```
if (sales[0] == 50.0) ... // sử dụng chỉ số mảng
```

- Chỉ số trong khoảng từ **0** tới **n-1** trong đó n là kích thước của mảng khi khai báo

```
char name[12];
```

```
name[12] = 'c'; // index out of range
```

- Làm việc với các giá trị

```
(1) days[1] = 29; OK ?
```

```
(2) days[2] = days[2] + 4; OK ?
```

```
(3) days[3] = days[2] + days[3]; OK ?
```


```
(4) days[MTHS] = {2,3,4,5,6}; OK ?
```

1-10

Duyệt mảng - Dùng chỉ số mảng

- Ví dụ: Duyệt mảng `days[]` để hiện giá trị của mỗi phần tử

days	31	28	31	30	31	30	31	31	30	31	30	31
chỉ số	0	1	2	3	4	5	6	7	8	9	10	11




1-11

Duyệt mảng: In giá trị các phần tử

```
#include <stdio.h>
#define MTHS 12 /* define a constant */
int main( )
{
    int i;
    int days[MTHS] = {31,28,31,30,31,30,31,31,30,31,30,31};
    /* print the number of days in each month */
    for (i = 0 ; i < MTHS ; i++)
        printf("Month %d has %d days\n", i+1, days[i]);
    return 0;
}
```

Output
Month 1 has 31 days
Month 2 has 28 days
...
Month 12 has 31 days

days	31	28	31	30	31	30	31	31	30	31	30	31
chỉ số	0	1	2	3	4	5	6	7	8	9	10	11



1-12

Duyệt mảng: Tìm kiếm giá trị

```
#include <stdio.h>
#define SIZE 5 /* define a constant */
int main ( )
{
    char myChar[SIZE] = {'b', 'a', 'c', 'k', 's'};
    int i;
    char searchChar;
    // Reading in user's input to search
    printf("Enter a char to search: ");
    scanf("%c", &searchChar);
    // Traverse myChar array and output character if found
    for (i = 0; i < SIZE; i++) {
        if (myChar[i] == searchChar){
            printf("Found %c at index %d", myChar[i], i);
            break;    //break out of the loop
        }
    }
    return 0;
```

Output

Enter a char to
search: a
Found a at index 1

1-13

Duyệt mảng: Tìm giá trị lớn nhất

```
#include <stdio.h>
int main( )
{
    int index, max, numArray[10];
    max = -1; printf("Enter 10 numbers: \n");
    for (index = 0; index < 10; index++)
        scanf("%d", &numArray[index]);
    // Find maximum from array data
    for (index = 0; index < 10; index++) {
        if (numArray[index] > max)
            max = numArray[index];
    }
    printf("The max value is %d.\n", max);
    return 0;
}
```

Output

Enter 10 numbers:
4 3 8 9 15 25 3 6 7 9
The max value is 25

1-14

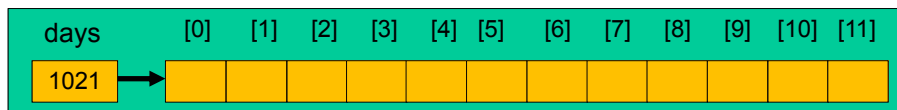
Mảng một chiều

- ❑ Khai báo mảng, khởi tạo giá trị, thao tác trên mảng
- ❑ **Con trỏ và mảng**
- ❑ Mảng là tham số của hàm

1-15

Hàng số con trỏ

- ❑ Tên mảng là **hàng số con trỏ**
- ❑ Giả sử một số integer biểu diễn bởi 4 byte (hoặc 2 byte tùy máy tính) và mảng **days** bắt đầu tại vị trí nhớ 1021
- ❑ `int days[12];` // days - hàng số con trỏ



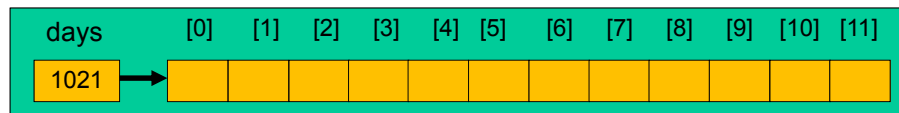
Địa chỉ nhớ: 1021 1023 1025 1027 1029 102B 102D 102F 1031 1033 1035 1037

- ❑ Địa chỉ của một phần tử mảng: ví dụ `int h[5];`
- ❑ **&h[0]** là **địa chỉ** của phần tử **đầu tiên**
- ❑ **&h[i]** là **địa chỉ** của phần tử thứ **(i+1)**

1-16

Hàng số con trỏ

- Tên mảng, **days**, là **địa chỉ (hoặc con trỏ)** của **phần tử đầu tiên** của mảng



Địa chỉ nhớ: 1021 1023 1025 1027 1029 102B 102D 102F 1031 1033 1035 1037



```
int days[12];
```

```
days == &days[0]
```

```
*days == days[0]
```

```
days + 1 == &days[1]
```

```
*(days+1) == days[1]
```

- Không thể thay đổi con trỏ cơ sở (base pointer) của mảng:

```
days += 5; // i.e., days = days+5; không hợp lệ
```

1-17

Biến con trỏ

- Biến con trỏ** có thể có **địa chỉ khác nhau**

```
int days[MTHS]; // days - hàng số con trỏ không thể thay đổi
```

```
/* pointer arithmetic */
#define MTHS 12
#include <stdio.h>
int main()
{
    int days[MTHS]= {31,28,31,30,31,30,31,31,30,31,30,31};
    int *day_ptr; // biến con trỏ
    day_ptr = days;
    printf("First element = %d\n", *day_ptr);
    day_ptr = &days[3]; /* points to the fourth element */
    printf("Fourth element = %d\n", *day_ptr);
    day_ptr += 3; /* points to the seventh element */
    printf("Seventh element = %d\n", *day_ptr);
    day_ptr--; /* points to the sixth element */
    printf("Sixth element = %d\n", *day_ptr);
    return 0;
}
```

Output

First element = 31
Fourth element = 30
Seventh element = 31
Sixth element = 30

1-18

Thao tác với con trỏ

Thao tác	day_ptr	days	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	...	[11]
			1021	1023	1025	1027	1029	102B	102D	102F	...	1037
int days[MTH] = {...};	?	1021	31	28	31	30	31	30	31	31	...	31
day_ptr=days;	1021	1021	31	28	31	30	31	30	31	31	...	31
day_ptr=&days[3];	1027	1021	31	28	31	30	31	30	31	31	...	31
day_ptr+=3;	102D	1021	31	28	31	30	31	30	31	31	...	31
day_ptr--;	102B	1021	31	28	31	30	31	30	31	31	...	31

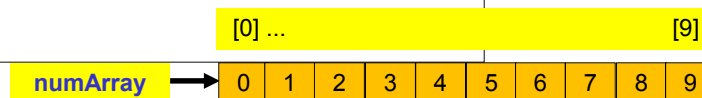
Dùng hằng số con trỏ - Tìm giá trị lớn nhất

```
#include <stdio.h>
int main( )
{
    int index, max, numArray[10];
    printf("Enter 10 numbers: \n");
    for (index = 0; index < 10; index++)
        scanf("%d", numArray + index);
    // Find maximum from array data
    max = *numArray;
    for (index = 1; index < 10; index++) {
        if (*(numArray + index) > max)
            max = *(numArray + index);
    }
    printf("The max value is %d.\n", max);
    return 0;
}
```

Output

Enter 10 numbers:
0 1 2 3 4 5 6 7 8 9

The max value is 9

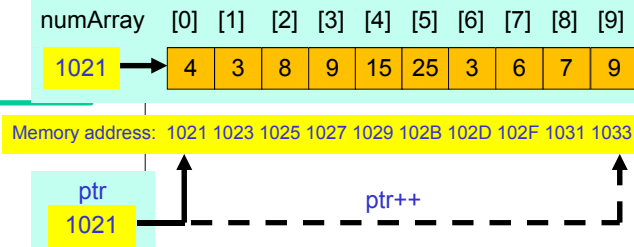


Dùng biến con trỏ - Tìm giá trị lớn nhất

```
#include <stdio.h>
int main(){
    int index, max, numArray[10];
    int *ptr;
    ptr = numArray;
    printf("Enter 10 numbers: \n");
    for (index = 0; index < 10; index++)
        scanf("%d", ptr++);
    // Find maximum from array data
    ptr = numArray;
    max = *ptr;
    for (index = 0; index < 10; index++) {
        if (*ptr > max)
            max = *ptr;
        ptr++;
    }
    printf("max is %d.\n", max);
    return 0;
}
```

Output

Enter 10 numbers:
4 3 8 9 15 25 3 6 7 9
max is 25.



Mảng một chiều

- ☐ Khai báo mảng, khởi tạo giá trị, thao tác trên mảng
- ☐ Con trỏ và mảng
- ☐ **Mảng là tham số của hàm**

Mảng là tham số của hàm

- ❑ Mảng với số chiều bất kì có thể truyền như tham số của hàm
fn(table); /* gọi hàm */
fn là hàm và **table** là mảng 1 chiều
- ❑ Mảng **table** được truyền vào hàm bằng **tham chiếu** (reference): **Địa chỉ** của **phần tử đầu tiên** của mảng được truyền vào hàm

1-23

Mảng là tham số của hàm

```
void fn(int table[], int n)  
{  
    ...  
} // n: kích thước của mảng
```

```
void fn(int table[ TABLESIZE])  
{  
    ...  
}
```

```
void fn(int *table, int n)  
{  
    ...  
}
```

- ❑ Prototype của hàm:
void fn(int **table**[], **int n**)
void fn(int **table**[TABLESIZE])
void fn(int ***table**, **int n**)

1-24

Truyền mảng như là tham số của hàm

```
#include <stdio.h>
int maximum(int table[ ], int n);
int main( )
{
    int max, index, n;
    int numArray[10];
    printf("Enter the number of values: ");
    scanf("%d", &n);
    printf("Enter %d values: ", n);
    for (index = 0; index < n; index++)
        scanf("%d", &numArray[index]);

    // find maximum
    max = maximum(numArray, n);
    printf("The maximum value is %d\n", max);

    return 0 ;
}
```

Output

Enter the number of values: 5
Enter 5 values: 1 2 3 4 5
The maximum value is 5

1-25

Truyền mảng như là tham số của hàm

```
int maximum(int table[ ], int n)
{
    int i, max;
    max = table[0];
    for (i = 1; i < n; i++)
        if (table[i] > max)
            max = table[i];
    return max;
}
```

(1) Dùng chỉ số

```
int maximum(int table[ ], int n)
{
    int i, max;
    max = *table;
    for (i = 1; i < n; i++)
        if (*(table+i) > max)
            max = *(table+i);
    return max;
}
```

(2) Dùng con trỏ

1-26

Chương 2: Mảng và danh sách liên kết

- ❑ Cấu trúc lưu trữ mảng
 - Mảng một chiều
 - Mảng nhiều chiều
- ❑ Danh sách liên kết
- ❑ Ngăn xếp
- ❑ Hàng đợi

1-27