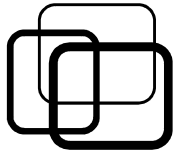
Mảng

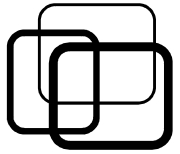
GV. Nguyễn Minh Huy

Nội dung



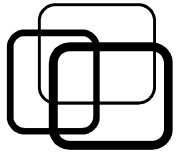
- Khái niệm mảng.
- Thao tác trên mảng.
- Chuỗi ký tự.

Nội dung



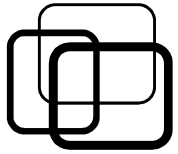
- **Khái niệm mảng.**
- Thao tác trên mảng.
- Chuỗi ký tự.

Khái niệm mảng



- Xét chương trình sau:
  - Nhập 5 số nguyên, sau đó xuất 5 số vừa nhập.
    - Khai báo 5 biến int a, b, c, d, e.
  - Nhập 50 số nguyên, sau đó xuất 50 số vừa nhập.
    - Khai báo 50 biến int!!
  - ➔ Làm sao khai báo nhiều biến cùng lúc?
  - ➔ Mảng.

Khái niệm mảng



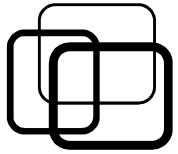
- Mảng trong ngôn ngữ C:
  - Mảng là một dãy biến liên tục có cùng kiểu.
  - Các biến trong dãy là phần tử mảng.
  - Khai báo:

    <Kiểu dữ liệu> **<Tên mảng>[** <Số phần tử> **]**;

    <Số phần tử>: phải là một hằng số.

```
int     m1[ 10 ];       // Dãy 10 số nguyên.
float   m2[ 50 ];       // Dãy 50 số thực.

int     N;
float   m3[ N ];        // Sai

const int K = 100;
float   m4[ K ];        // Đúng
```

Khái niệm mảng



- Mảng trong ngôn ngữ C:
  - Sau khi khai báo, phần tử mảng có giá trị bao nhiêu?

int **m[5]**; **m** | ? | ? | ? | ? | ? |

  - Khởi tạo giá trị mảng:

<Kiểu dữ liệu> **<Tên mảng>[**<Số phần tử>**]** = { **<Giá trị PT1>**, **<Giá trị PT2>**, … };

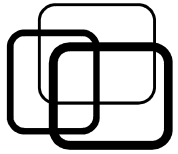
int **m1[5]** = { 1, 2, 3, 4, 5 }; **m1** | 1 | 2 | 3 | 4 | 5 | // Khởi tạo tất cả phần tử

int **m2[5]** = { 1, 2 }; **m2** | 1 | 2 | 0 | 0 | 0 | // Khởi tạo vài phần tử đầu // các phần tử sau tất cả = 0

int **m3[5]** = { 0 }; **m3** | 0 | 0 | 0 | 0 | 0 | // Khởi tạo tất cả = 0

int **m3[ ]** = { 1, 2, 3, 4, 5 }; **m3** | 1 | 2 | 3 | 4 | 5 | // Tự động biết số phần tử

Khái niệm mảng



- Mảng trong ngôn ngữ C:
  - Truy xuất phần tử mảng:

    <Tên mảng> **[<Chỉ số mảng>]**

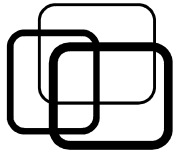
    <Chỉ số mảng>: một số nguyên từ **0** đến **<Số phần tử> - 1**.

int **a[10]** = { 0 };

	0	1	2	3	4	5	6	7	8	9
a	0	0	0	0	0	0	0	0	0	0

```
a[ 0 ] = 5;
a[ 1 ] = 6;
a[ 2 ] = a[ 0 ] + a[ 1 ];

a[ -1 ] = 7;          // Sai
a[ 10 ] = 8;          // Sai
```

Khái niệm mảng



- Mảng trong ngôn ngữ C:
  - Truyền tham số mảng:
    - Khai báo tham số mảng giống khai báo mảng.

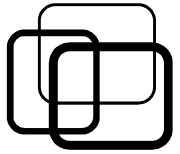
      void foo(**int a[100], int size**);
    - Khai báo tham số mảng có thể bỏ số phần tử.

      void foo(**int a[ ], int size**);
    - Phần tử mảng CÓ THỂ THAY ĐỔI sau khi ra khỏi hàm.

```
void foo( int a[ ], int size )
{
        a[ 2 ] = 9;
        a[ 5 ] = 8;
}
```

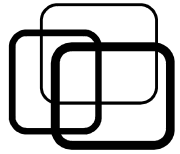
```
void main()
{
        int  a[ 100 ] = { 0 };

        foo( a, 100 );
        // a[2], a[5] bị thay đổi.
}
```

Nội dung



- Khái niệm mảng.
- **Thao tác trên mảng.**
- Chuỗi ký tự.

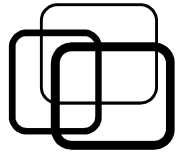
Thao tác trên mảng



- Cách thức chung:
  - B1: Duyệt mảng.
    - Dùng **vòng lặp** + **biến đếm**.
    - Mỗi vòng lặp → thao tác một phần tử.
  - B2: Thao tác trên từng phần tử.
    - Dùng biến đếm truy xuất phần tử.

```
// Duyệt mảng M có kích thước N.
for ( int i = 0; i < N; i++ )
{
        <Lệnh truy xuất phần tử M[ i ]>;
}
```

Thao tác trên mảng

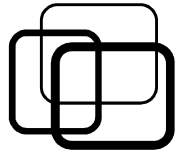


- Nhập mảng:

```
// Nhập mảng số nguyên a, kích thước n
void nhapMang( int a[ ], int &n )
{
        printf("Nhap kich thuoc = ");
        scanf("%d", &n);

        for (int i = 0; i < n; i++)
        {
            printf("Nhap phan tu %d = ", i);
            scanf("%d", &a[ i ]);
        }
}
```

```
#define MAX   100

void main()
{
        int  a[ MAX ], size1;
        int  b[ MAX ], size2;

        nhapMang(a, size1);
        nhapMang(b, size2);
}
```

Thao tác trên mảng

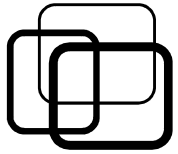


- Xuất mảng:

```
// Xuất mảng số nguyên a, kích thước n
void xuatMang( int a[ ], int n )
{
        for ( int i = 0; i < n; i++ )
              printf("%d ", a[ i ]);
}
```

```
#define MAX    100

void main()
{
        int  a[ MAX ], size1;
        int  b[ MAX ], size2;

        nhapMang(a, size1);
        nhapMang(b, size2);

        xuatMang(a, size1);
        xuatMang(b, size2);
}
```

Thao tác trên mảng

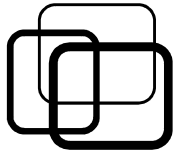


- Tính tổng phần tử mảng:

```
// Tính tổng mảng a, kích thước n
long tinhTong( int a[ ], int n )
{
        long  tong = 0;

        for ( int i = 0; i < n; i++ )
              tong += a[ i ];

        return tong;
}
```

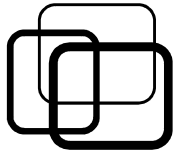
```
#define MAX    100

void main()
{
        int  a[ MAX ], size1;
        int  b[ MAX ], size2;

        nhapMang(a, size1);
        nhapMang(b, size2);

        long  tong1 = tinhTong(a, size1);
        long  tong2 = tinhTong(b, size2);
}
```

Nội dung



- Khái niệm mảng.
- Thao tác trên mảng.
- **Chuỗi ký tự.**

Chuỗi ký tự



- Chuỗi ký tự trong C:
  - Chuỗi ký tự: mảng ký tự, **phần tử cuối = '\0'**.

    ➔ Chiều dài chuỗi = số phần tử mảng – 1;
  - Khai báo chuỗi:

    char **<Tên chuỗi> [** <Chiều dài chuỗi> **+ 1]**;

char **s1[5]**;

	0	1	2	3	4
s1	?	?	?	?	\0

  - Khởi tạo chuỗi:

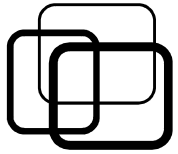
    char <Tên chuỗi> **[ ] = "<Chuỗi khởi tạo>"**;

char **s2[ ]** = "Hello";

	0	1	2	3	4	5
s2	H	e	l	l	o	\0

// Khởi tạo chuỗi

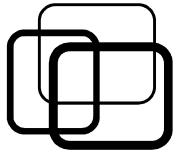
Chuỗi ký tự



- Thao tác trên chuỗi ký tự:
  - Nhập chuỗi:
    - scanf("%s", &chuoi).
      ➔ Chỉ nhập từ đầu tiên.
    - gets(chuoi);
      ➔ Nhập nguyên chuỗi.
  - Xuất chuỗi:
    - printf("%s", chuoi).
    - puts(chuoi).

```
#define MAX   100

void main()
{
        char  s1[ MAX ];
        char  s2[ MAX ];

        printf("Nhap chuoi s1 = ");
        scanf("%s", &s1);
        printf("Nhap chuoi s2 = ");
        gets(s2);

        printf("Chuoi s1 = %s", s1);
        puts(s2);
}
```

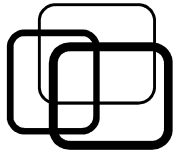
Chuỗi ký tự



- Thao tác trên chuỗi ký tự:
  - Thư viện <string.h>: **#include** <string.h>
    - strlen(chuoi): đếm chiều dài chuỗi.

```
char  s1[ ] = “Hello World”;                  // Tự thêm ‘\0’ ở cuối.
char  s2[ ] = “Hello World\n”;                // Tự thêm ‘\0’ ở cuối.
char  s3[ ] = { ‘H’, ‘e’, ‘l’, ‘l’, ‘o’, ‘ ‘, ‘W’, ‘o’, ‘r’, ‘l’, ‘d’, ‘\n’, ‘\0’ };

int  chieuDai1 = strlen(s1);    // chieuDai1 = 11
int  chieuDai2 = strlen(s2);    // chieuDai2 = 12
int  chieuDai3 = strlen(s3);    // chieuDai3 = 12
```

Chuỗi ký tự

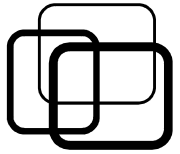


- Thao tác trên chuỗi ký tự:
  - Viết hoa chữ cái đầu mỗi từ:

```
#include <string.h>

void vietHoa( char  s[ ] )
{
        for (int i = 0; i < strlen(s); i++)
              if ( s[ i ] >= 'a' && s[ i ] <= 'z' &&
                    ( i == 0 || s[ i - 1 ] == ' ' ) )
                    s[ i ] = s[ i ] - 32;
}
```

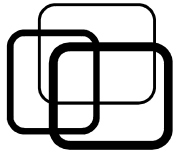
```
#define MAX  100

void main()
{
        char  s[MAX];

        printf("Nhap chuoi s = ");
        gets(s);

        vietHoa(s);

        printf("%s\n", s);
}
```

Tóm tắt



- Khái niệm mảng:
  - Dãy biến cùng kiểu.
  - Các biến trong dãy là phần tử mảng.
- Thao tác trên mảng:
  - Thao tác chung: duyệt + thao tác từng phần tử.
- Chuỗi ký tự:
  - Mảng ký tự kết thúc bằng ‘\0’.
  - Nhập xuất: printf, scanf, gets, puts.
  - Đếm chiều dài: strlen (thư viện <string.h>).

Bài tập



Bài tập 7.1:

Viết chương trình C (tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào mảng N số nguyên.
- Hãy cho biết:
  - a) Có bao nhiêu số âm trong mảng.
  - b) Có bao nhiêu số nguyên tố trong mảng.

Định dạng nhập:

Nhap N = 3

Phan tu 0 = 2

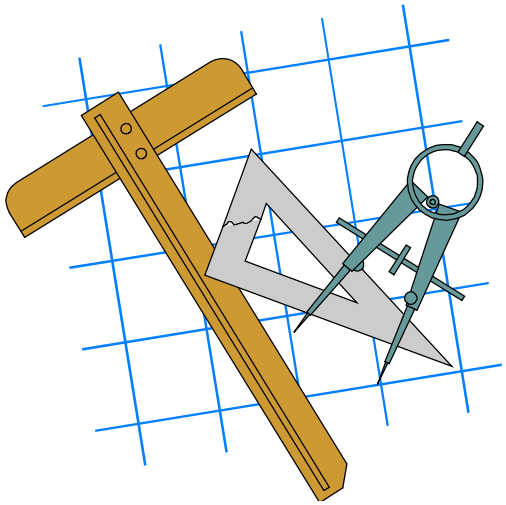
Phan tu 1 = 3

Phan tu 2 = -6

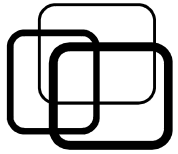
Định dạng xuất:

Co 1 so am.

Co 2 so nguyen to.



Bài tập



Bài tập 7.2:

Viết chương trình C kiểm tra mảng như sau:

(tổ chức theo dạng hàm và chia làm nhiều file):

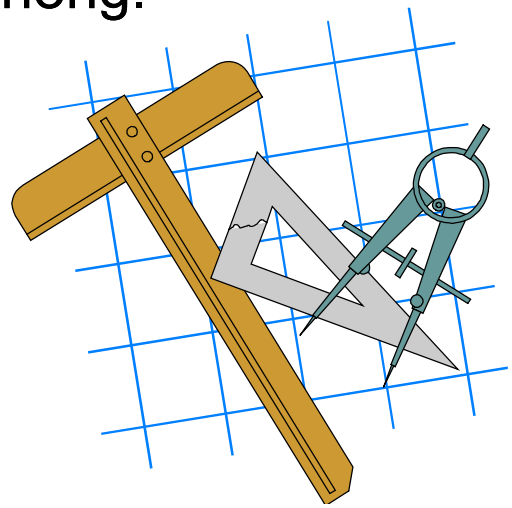
- Nhập vào mảng N số nguyên.
- Hãy cho biết:
  - a) Mảng có tăng dần không.
  - b) Mảng có đối xứng không.
  - c) Mảng có lập thành một cấp số cộng không.

Định dạng xuất:

Mang <tang/khong tang> dan.

Mang <doi xung/khong doi xung>.

Mang <lap thanh/khong lap thanh> cap so cong.



Bài tập

Bài tập 7.3:

Viết chương trình C thao tác chuỗi như sau:

(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào 2 chuỗi ký tự S1 và S2.
- Hãy chèn S2 vào ngay chính giữa S1 và xuất kết quả.

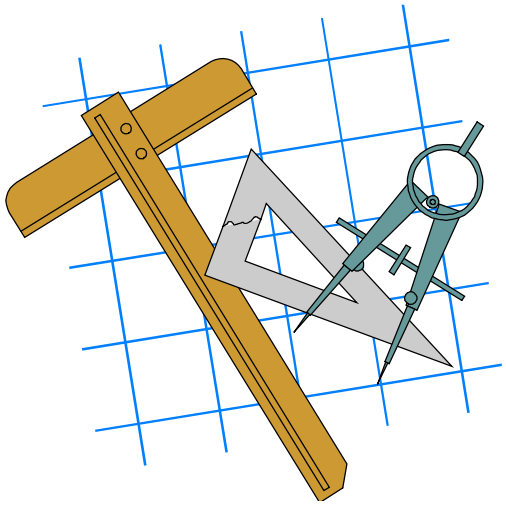
Định dạng nhập:

Nhap chuoi 1 = hello world

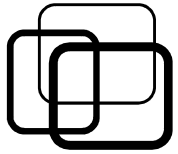
Nhap chuoi 2 = it is a good day

Định dạng xuất:

Ket qua = it is a hello worldgood day



Bài tập



Bài tập 7.4:

Viết chương trình C thao tác chuỗi như sau:

(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào chuỗi ký tự S.
- Đếm và in những ký tự xuất hiện nhiều nhất trong S.

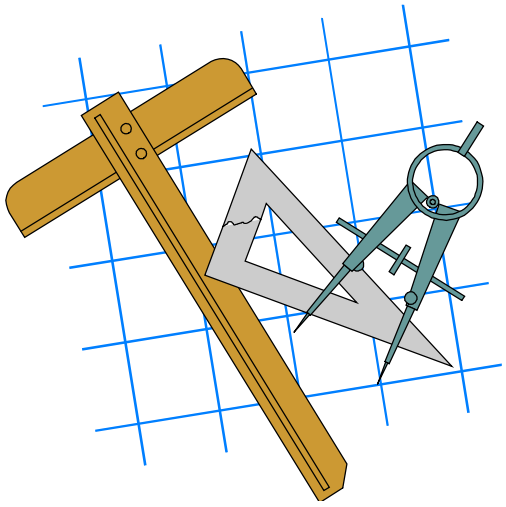
Định dạng nhập:

Nhap chuoi = tick tak tok

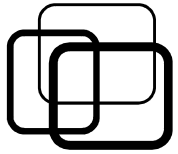
Định dạng xuất:

t: 3

k: 3



Bài tập



Bài tập 7.5:

Viết chương trình C thao tác chuỗi như sau:

(tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào chuỗi ký tự S.
- Đếm xem chuỗi S có bao nhiêu từ (cách nhau khoảng trắng hoặc kết thúc bằng dấu câu , . ? !).
- Chuẩn hóa chuỗi S:
  - + Bỏ khoảng trắng đầu và cuối chuỗi.
  - + Bỏ khoảng trắng dư thừa giữa các từ.
  - + Viết hoa các ký tự đầu mỗi từ.