KỸ THUẬT LẬP TRÌNH C

Chương 2:
Các kiểu dữ liệu cơ sở

1. Các kiểu dữ liệu cơ sở
2. Biến
3. Hằng
4. Biểu thức
5. Các hàm thư viện C chuẩn

1. Các kiểu dữ liệu cơ sở

- Kiểu dữ liệu là sự kết hợp của 2 thành phần: Miền giá trị mà kiểu dữ liệu có thể lưu trữ và tập hợp các phép toán để thao tác dữ liệu.
- Có 4 kiểu dữ liệu cơ sở:

  Kiểu số nguyên, kiểu số thực, kiểu luận lý và kiểu ký tự.

1. Các kiểu dữ liệu cơ sở

Kiểu dữ liệu	Số bits (n)	Miền giá trị (từ -2^{n-1} đến $2^{n-1}-1$) / (từ 0 đến 2^n-1)
char	8	[-128, 127]
unsigned char	8	[0, 255]
int	16 / 32	[-32.768, 32.767] / [2.147.483.648, 2.147.483.647]
unsigned int	16 / 32	[0, 65.535] / [0, 4.294.967.295]
short	16	[-32.768, 32.767]
unsigned short	16	[0, 65.535]
long	32	[2.147.483.648, 2.147.483.647]
unsigned long	32	[0, 4.294.967.295]
float	32	[3.4E-38, 3.4E+38]
double	64	[1.7E-308, 1.7E+308]
long double	80	[3.4E-4932, 3.4E+4932]

a. Kiểu số nguyên:

Kiểu số nguyên thường được thực hiện với các phép toán: +, -, *, /, %, >, >=, <, <=, ==, …

- Các kiểu số nguyên có dấu

char	1	–128 … +127
short	2	–32768 … +32767
int	2	–32768 … +32767
long	4	–2147483648 … +2147483647

- Các kiểu số nguyên không dấu

unsigned char	1	0 … 255
unsigned short	2	0 … 65535
unsigned int	2	0 … 65535
unsigned long	4	0 … 4294967295

b. Kiểu số thực:

Kiểu số thực thường được thực hiện với các phép toán: +, -, *, /, >, >=, <, <=, ==, …

float	4	3.4*E–38 …	3.4*E+38
double	8	1.7*E–308 …	1.7*E+308

c. Kiểu luận lý

Kiểu luận lý thực chất cũng là một kiểu nguyên. Một số nguyên khác 0 được hiểu là đúng, một số nguyên bằng 0 được hiểu là sai.

Kiểu luận lý thường được thực hiện với các phép toán: && (And), || (Or), ! (Not), >, >=, <, <=, ==, …

Ví dụ:

0 (false), 1 (true), 2 (true)

1 > 2 (false), 1 < 2 (true)

d. Kiểu ký tự

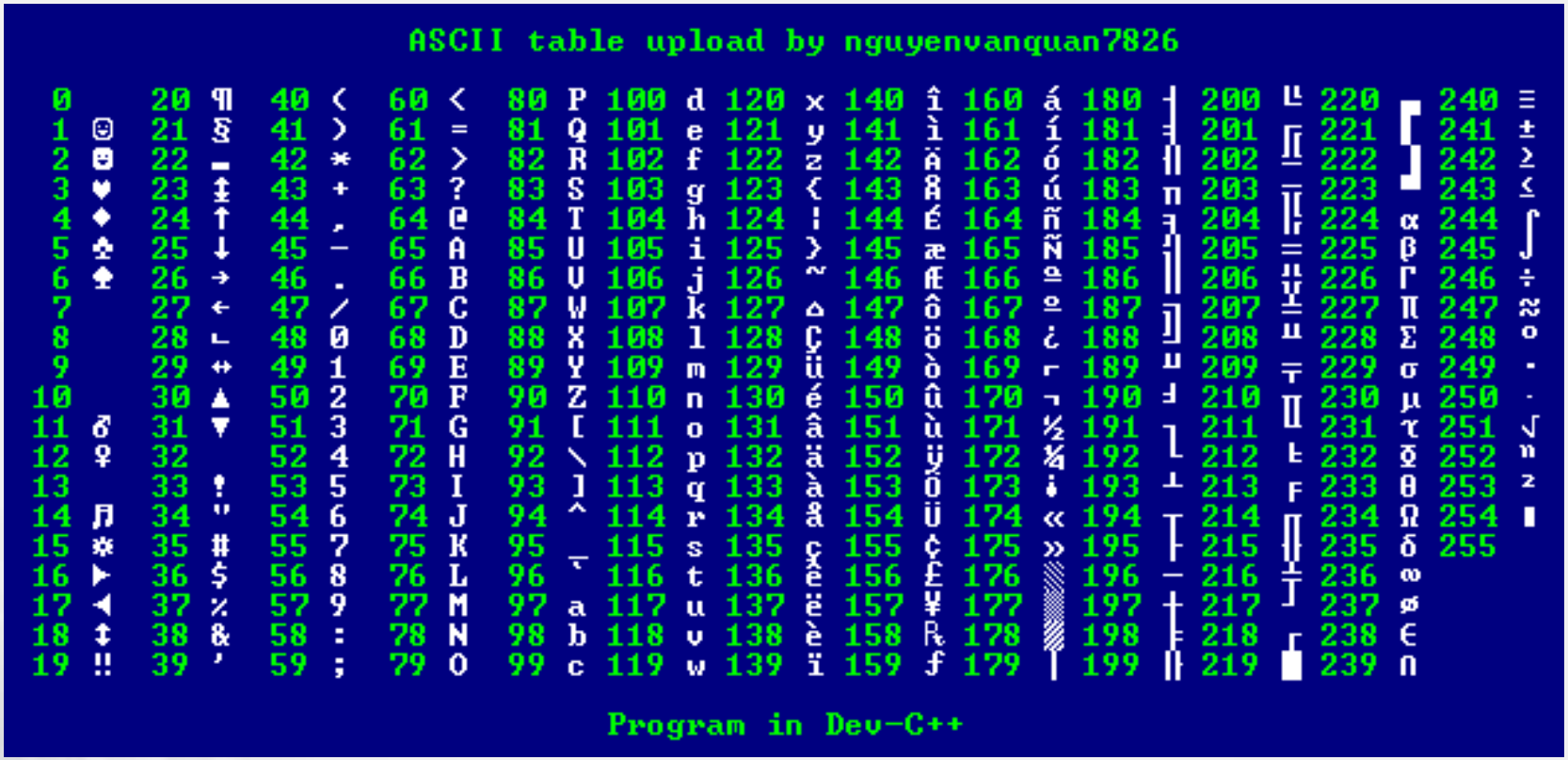
- Có tên kiểu là char, miền giá trị là 256 ký tự trong bảng mã ASCII.
- Đây cũng chính là kiểu số nguyên vì kiểu này không lưu trực tiếp ký tự mà chỉ lưu mã ASCII của ký tự đó.

Ví dụ:

Lưu số 65 tương đương việc lưu với ký tự ‘A’

Lưu số 97 tương đương việc lưu với ký tự ‘a’

d. Kiểu ký tự



Bảng mã Ascii

2. Biến

- Biến là một vùng nhớ được lưu tại một địa chỉ nào đó trong bộ nhớ máy tính.
- Biến được đặt tên thông qua khai báo biến
- Giá trị mà biến lưu trữ có thể bị thay đổi nhiều lần trong suốt quá trình chương trình thi hành.

Cú pháp:

<kiểu> <Tên biến>;

<kiểu> <Tên biến 1>, <Tên biến 2>;

Ví dụ:

```
int i;
int j, k;
unsigned char dem;
float ketqua, delta;
```

3. Hằng

- Hằng cũng là một vùng nhớ được lưu trữ tại một địa chỉ nào đó trong bộ nhớ máy tính .
- Hằng được đặt tên thông qua khai báo hằng.
- Giá trị mà hằng lưu trữ **không thay đổi** trong suốt quá trình chương trình thi hành.

Cú pháp:

#define <Tên hằng> <Giá trị>

const <kiểu> <Tên hằng> = <Giá trị>;

Ví dụ:

```
#define MAX 100
#define PI 3.14
Hoặc
const int MAX = 100;
const float PI = 3.14;
```

4. Biểu thức

- BT Là một sự kết hợp giữa các toán tử (+, –, *, /, %….) và các toán hạng (hằng, biến, lời gọi hàm, …).
- Mỗi BT đều có một giá trị.

Ví dụ:

```
double p = (a + b + c) / 2 ; // Nửa chu vi
double s = sqrt(p*(p-a)*(p-b)*(p-c)); //Diện tích tam giác
```

a. Toán tử gán

Dùng để gán giá trị cho biến.

Cú pháp:

<biến> = <biểu thức>;

Ví dụ:

```
int a, b, c, d, e;
a = 12;
b = a;
c = a + b;
e = d = c; //Phép gán liên tiếp
```

b. Toán tử số học

- Toán tử 1 ngôi: Là toán tử chỉ tác động lên một toán hạng

  ++ (tăng 1 đơn vị) -- (giảm 1 đơn vị)

Chú ý:

Khi toán tử đặt trước toán hạng thì **toán hạng được tăng/giảm trước khi được dùng.**

còn khi toán tử đặt sau toán hạng thì **toán hạng được dùng trước khi tăng/giảm**.

Ví dụ:

```
int x, y;
x = 10;
y = x++ + 5; // y = 15 và x = 11
x = 10;
y = ++x + 5; // x = 11 và y = 16
```

- Toán tử 2 ngôi: Là toán tử tác động lên hai toán hạng.

  \+ – * / % (chia lấy phần dư) += -= *= /= %=

  Ví dụ 1:

```
int a = 9, b = 2,  x, y;
float c = 9.0, z;
x = a/b; // x = 4 (chia nguyên)
y = a%b; // y = 1
z = c/b; // z = 4.5 (chia thực)
```

Ví dụ 2:

```
int n;
n = 10;
n += 10;    // n = n + 10 → n = 20
n -= 5;     // n = n – 5 → n = 15
n *= 2;     // n = n * 2 → n = 30
n /= 4;     //n = n / 4 → n = 7
n %= 2;     //n = n % 2 → n = 1
```

c. Toán tử quan hệ

Toán tử quan hệ cho ra giá trị đúng hoặc sai.

== (bằng) > < >= < <= != (khác)

Ví dụ:

```
bool b1 = (1 == 2);      //b1 → false
bool b2 = (1 != 2);      //b2 → true
bool b3 = (1 > 2);       //b3 → false
bool b4 = (1 >= 2);      //b4 → false
bool b5 = (1 < 2);       //b5 → true
bool b6 = (1 <= 2);      //b6 → true
```

d. Toán tử luận lý

Toán tử luận lý cho ra giá trị đúng/sai. && (and) || (or) ! (not)

a	b	a && b	a || b	!a
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Ví dụ:

```
bool b1 = (1 > 2) && (3 > 4);  //b1 → false
bool b2 = (1 < 2) || (3 > 4);  //b2 → true
bool b3 = !b2;                 //b3 → false
```

e. Toán tử biểu thức điều kiện

Đây là toán tử 3 ngôi (gồm có 3 toán hạng), ký hiệu ?:

Cú pháp:

<Bt1> ? <Bt 2> : <Bt 3>

biểu thức này cho kết quả là <Bt2> nếu <Bt1> đúng, ngược lại cho kết quả là <Bt3>

Ví dụ:

```
int a = 10, b = 20, max, min;
max = (a > b) ? a : b;
min = (a < b) ? a : b;
```

f. Toán tử chuyển kiểu

Toán tử chuyển kiểu dùng để chuyển một kiểu dữ liệu bất kỳ sang một kiểu dữ liệu mong muốn.

Cú pháp: (<Kiểu>) <Biến>

Ví dụ:

```
int a = 9, b = 2;
int c = a / b; //4
float d = (float)a / b; //4.5
```

5. Các hàm thư viện C chuẩn

- Hàm thư viện là những hàm đã được định nghĩa sẵn trong một thư viện nào đó của C.
- Muốn sử dụng các hàm thư viện thì phải khai báo thư viện trước khi sử dụng bằng chỉ thị **#include <tên thư viện.h>** đặt ở đầu tập tin chương trình.

a. Hàm toán học

Các hàm toán học được định nghĩa sẵn trong thư viện **math.h**.

Tên hàm	Trả về
ceil(x)	Làm tròn lên của x
floor(x)	Làm tròn xuống của x
abs(x)	Trị tuyệt đối của x nguyên
fabs(x)	Trị tuyệt đối của x thực
sqrt(x)	Căn bậc hai của x
pow(x, y)	x lũy thừa y
exp(x)	exponential của x
log(x)	Logarithm tự nhiên của x

Ví dụ:

```
float x = 3.7;
int y = ceil(x); //y = 4
int z = floor(x); //z = 3
int a = -10; int b = abs(a); //b = 10;
float u = -20.5; float v = fabs(u); //v = 20.5
float t = pow(2.0, 10.0);  // t = 2^10 = 1024
float p = exp(log(100.0)/3) ; //y = e^(1/3)log(100) = 100^1/3
float q = exp(10.0*log(3.0)) ;     //q = e^10log(3) = 3^10
```

b. Hàm xuất dữ liệu

Hàm xuất dữ liệu được định nghĩa sẵn trong thư viện stdio.h.

Cú pháp:

printf(<chuỗi định dạng>[, <ts1>, <ts2>, …]);

- <chuỗi định dạng> là cách trình bày thông tin xuất và được đặt trong cặp nháy kép “ ”, gồm 3 loại:
  - Văn bản thường (literal text): Được xuất y hệt như lúc gõ trong chuỗi định dạng.
  - Ký tự điều khiển (escape sequence): Gồm dấu \ và một trong các ký tự như trong bảng sau:

Ký tự điều khiển	Ý nghĩa
\a	Tiếng chuông
\n	Xuống dòng
\t	Dấu Tab
\\	In dấu \
\”	In dấu “

- Đặc tả (conversion specifier): Gồm dấu % và một vài ký tự, nó xác định kiểu của biến/giá trị muốn xuất.

Đặc tả	**Ý nghĩa**
%c	Ký tự (char)
%d, %ld	Số nguyên có dấu (char, int, short, long)
%f, %lf	Số thực (float, double)
%s	Chuỗi ký tự (char [], char*)
%u	Số nguyên không dấu (unsigned int/short/long)

- Các tham số <ts1>, <ts2>, . . . chính là các biến/giá trị muốn xuất, được liệt kê theo thứ tự cách nhau dấu phẩy

Ví dụ:

```
int a = 10, b = 20;
printf("%d\n", a); //10
printf("%d\n", b); //20
printf("%d %d\n", a, b); //10 20
float x = 15.06;
printf("%f\n", x); //15.060000
printf("%0.2f\n", x); //15.06
printf("%f\n", 1.0/3); //0.333333
printf("%0.1f\n", 1.0/3); //0.3
```

c. Hàm nhập dữ liệu

Cú pháp:

scanf(<chuỗi định dạng>[, <đs1>, <đs1>, …]);

- <chuỗi định dạng> giống định dạng xuất nhưng chỉ có các đặc tả.
- Các tham số <ts1>, <ts2>, . . . là tên các biến sẽ chứa giá trị nhập và được đặt trước bởi toán tử &

Ví dụ:

```
int a, b;
scanf(“%d”, &a); // Nhập giá trị cho a
scanf(“%d”, &b); // Nhập giá trị cho b
scanf(“%d%d”, &a, &b); // Nhập giá trị cho a và b
float c, d;
scanf(“%f”, &c); // Nhập giá trị cho c
scanf(“%f”, &d); // Nhập giá trị cho d
scanf(“%f%f”, &c, &d); // Nhập giá trị cho c và d
char e, f;
scanf(“%c”, &e); // Nhập giá trị cho e
scanf(“%c”, &f); // Nhập giá trị cho f
scanf(“%c%c”, &e, &f); // Nhập giá trị e và f
```

Hết