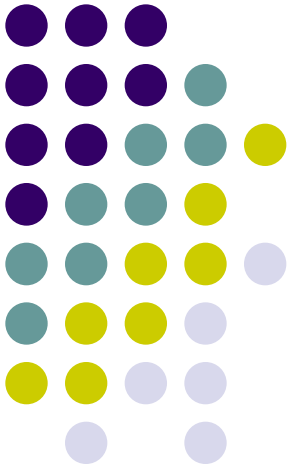
LẬP TRÌNH CĂN BẢN

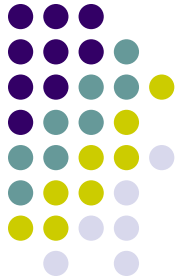
Phần 2 - Chương 3
CÁC CÂU LỆNH ĐƠN TRONG C



Nội dung chương này

- Câu lệnh
  - Khái niệm câu lệnh
  - Phân loại
- Các lệnh đơn
  - Lệnh gán
  - Lệnh nhập giá trị từ bàn phím cho biến
  - Lệnh xuất giá trị của biểu thức lên màn hình

Khái niệm câu lệnh

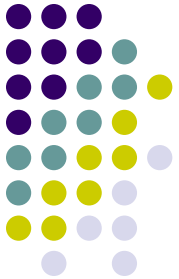


- “1 câu lệnh xác định 1 công việc mà chương trình phải thực hiện”
- Kết thúc bởi **;**

Phân loại

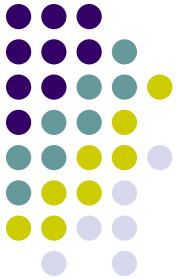
- **Có 2 loại**
  - **Lệnh đơn**
    - Không chứa 1 lệnh nào khác
    - Gồm: lệnh gán, nhập, xuất
  - **Lệnh có cấu trúc**
    - Chứa các lệnh khác
    - Gồm:
      - cấu trúc điều kiện rẽ nhánh
      - cấu trúc điều kiện lựa chọn
      - cấu trúc lặp
      - cấu trúc lệnh hợp thành

Các lệnh đơn



- Lệnh gán
- Lệnh nhập giá trị từ bàn phím cho biến
- Lệnh xuất giá trị của biểu thức lên màn hình

Lệnh gán (1)



- **Ví dụ:**

```
int main() {
      int x,y;
      x =10; //  Gán hằng số 10 cho biến x
      y = 2*x; //Gán giá trị của biểu thức 2*x (=20) cho biến y
      return 0;
}
```

- Cú pháp:

  <Tên biến> = <biểu thức>;

  - Ý nghĩa: Gán giá trị cho 1 biến

- Gán giá trị ngay tại lúc khai báo:

```
int x = 10, y=x;
```

Lệnh gán (2)

- Kiểu của biểu thức và của biến phải giống nhau

```
int main() {
        int x,y;
        x = 10; //  Gán hằng số 10 cho biến x
        y = "Xin chao";
                //y có kiểu int, còn "Xin chao" có kiểu char*
        return 0;
}
```

Error: "*Cannot convert 'char *' to 'int'*"

Lệnh gán (3)

- Thường thì có sự chuyển đổi kiểu tự động nếu có thể.

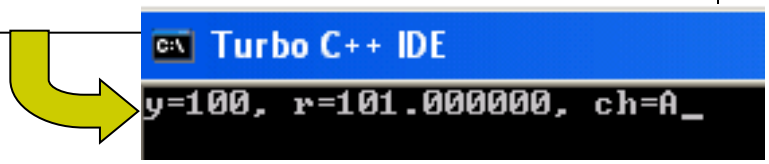
```
int main() {
    int x,y;
    float r;
    r = 9000;
    x = 10; // Gán hằng số 10 cho biến x
    y = 'd'; // y có kiểu int, còn 'd' có kiểu char
    r = 'e'; // r có kiểu float, 'e' có kiểu char
    char ch;
    ch = 65.7; // ch có kiểu char, còn 65.7 có kiểu float
    return 0;
}
```

Chuyển được

Lệnh gán (4)

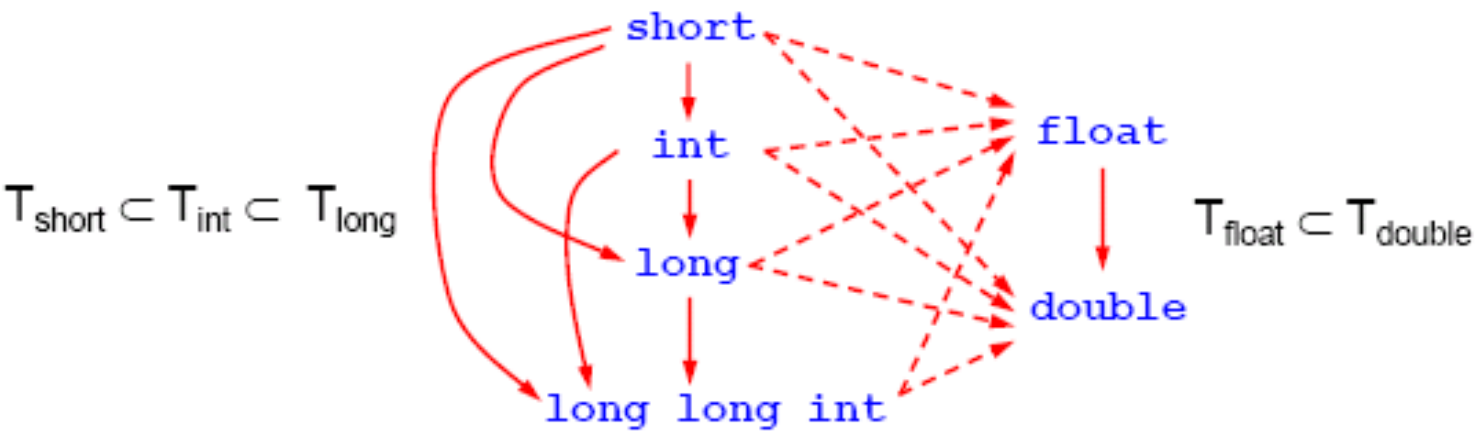
- Kết quả chương trình sau là gì?

```
#include <conio.h>
#include <stdio.h>
int main(){
        int x,y;
        float r;
        char ch;
        clrscr();
        r=9000;
        x=10;
        y='d';
        r='e';
        ch='A';
        printf("y=%d, r=%f, ch=%c",y,r,ch);
        getch();
        return 0;
}
```



Lệnh gán (5)

- Trong C, các chuyển đổi kiểu sau được làm tự động.



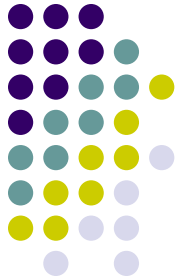
- Những chuyển đổi trên đảm bảo không làm mất đi sự chính xác (loss of precision).
- Việc chuyển đổi theo các hướng khác có thể làm mất sự chính xác
- **Ví dụ:**

```
double aDouble = 3.14;
int anInt;
anInt = aDouble;
aDouble = anInt;
```

„loss of precision“

⇒ aDouble = 3.000000

Lệnh gán (6)

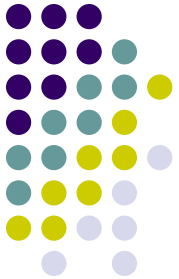


- **Ép kiểu (casting type)**

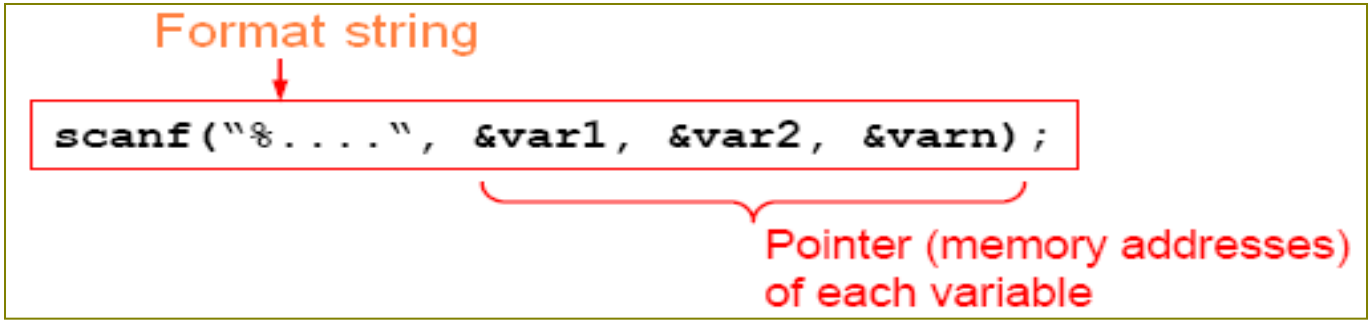
```
valueInNewType = (NewType)valueInOldType;
```

```
int anInt;
double aDouble = 3.14159;

aDouble = (int)aDouble;      // truncation of fraction
printf("%f\n", aDouble);     // --> 3

anInt = aDouble;             // truncation of fraction
aDouble = anInt/2;           // integer division 3/2 = 1
printf("%f\n", aDouble);     // --> 1

aDouble = (double)anInt/2; // floating-point division 3/2
printf("%f\n", aDouble);     // --> 1.50000

aDouble = anInt/2.0;         // floating-opint division 3/2
printf("%f\n", aDouble);     // --> 1.50000
```

Lệnh nhập giá trị từ bàn phím cho biến (1)



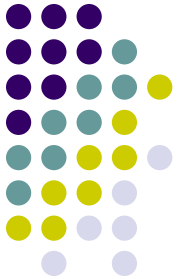
- **scanf** đọc dữ liệu từ bàn phím và gán vào biến



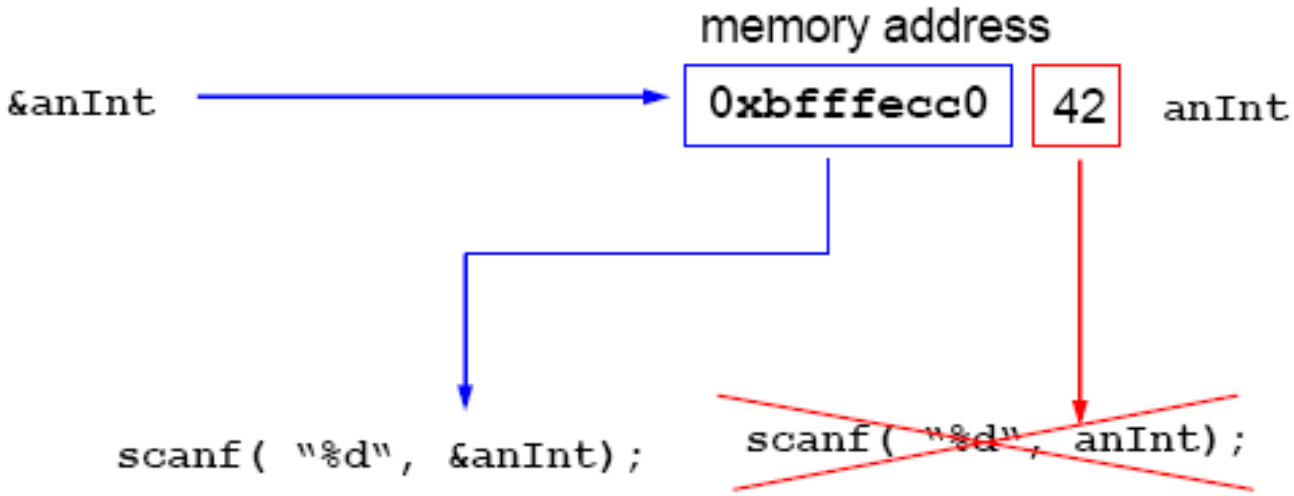
- **Chuỗi định dạng (format string):** để qui định kiểu dữ liệu, cách biểu diễn, độ rộng, số chữ số thập phân, …

Định dạng	Ý nghĩa
%[số ký số]d	Nhập số nguyên có tối đa <số ký số>
%[số ký số] f	Nhập số thực có tối đa <số ký số> tính cả dấu chấm
%c	Nhập một ký tự

Lệnh nhập giá trị từ bàn phím cho biến (2)



- **scanf** phải lưu giá trị vào 1 biến
  - **scanf("%d",anInt):** không đúng, vì ***anInt*** xác định giá trị hiện hành của 1 biến.
  - **scanf("%d",&anInt):** đúng, vì địa chỉ của ***anInt*** đã được xác định.



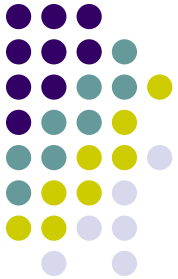
Ví dụ - Dùng Standard Input

```
#include <stdio.h>
main()
{
    int age;
    float size;

    printf("your age: ");
    scanf("%d", &age);
    printf("your size in meter: ");
    scanf("%f", &size);

    printf("You are %d years old and %f m.\n",
            age, size);
}
```

Output:

```
your age:                    ⇐ 42
your size in meter:          ⇐ 1.76
You are 42 years old and 1.760000 m.
```

Lệnh xuất giá trị của biểu thức lên màn hình (1)



```
printf("Hello World!");
```

- Cần ít nhất 1 đối số.
- Đối số đầu tiên là 1 chuỗi
- Chuỗi có thể chứa:

\n	new line
\t	horizontal tab stop
\"	double quote "
\\	back slash \

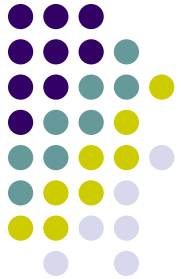
- **Ví dụ:**

```
printf("Hello World\n\t\"This is a quoted string.\"");
```

Output

```
Hello World
        "This is a quoted string."
```

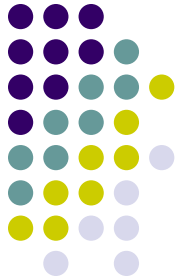
Lệnh xuất giá trị của biểu thức lên màn hình (2)



- Nếu muốn in ra các biến và biểu thức, ta truyền nó vào printf như các đối số.
- Các định dạng (format) khác nhau cho các kiểu giá trị khác nhau (dùng %).

```
printf("String %....", arg1, arg2, ... argn)
```

Lệnh xuất giá trị của biểu thức lên màn hình (3)



- Các định dạng:

Định dạng	Ý nghĩa
%d	Xuất số nguyên
%[.số chữ số thập phân] f	Xuất số thực có <số chữ số thập phân> theo quy tắc làm tròn số.
%o	Xuất số nguyên hệ bát phân
%x	Xuất số nguyên hệ thập lục phân
%c	Xuất một ký tự
%s	Xuất chuỗi ký tự
%e hoặc %E hoặc %g hoặc %G	Xuất số nguyên dạng khoa học (nhân 10 mũ x)

Ví dụ - Output từ C

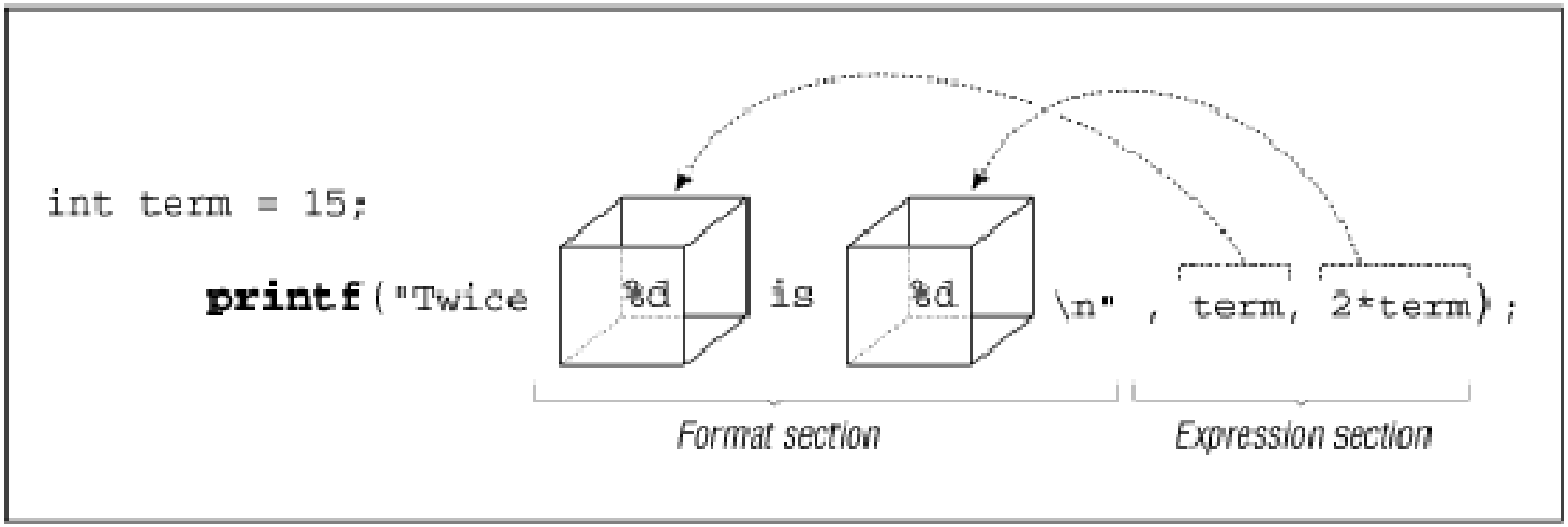
Ví dụ:

```
#include <stdio.h>
main()
{
    const double pi = 3.141592654;
    int anInt = 42;
    double big = 1.24e23;

    printf("pi    = %f\n", pi);
    printf("anInt = %d\n", anInt);
    printf("big   = %e\n", big);
    printf("anInt is at address: %p\n, &anInt);
}
```

Output:

```
pi    = 3.141592
anInt = 42
big   = 1.240000e+23
anInt is at address: 0xffbefaf4
```

Giải thích thêm về printf




Hết chương