



TRƯỜNG ĐẠI HỌC THỦY LỢI
Khoa Công nghệ thông tin
Bộ môn Tin học và KTTT

NHẬP MÔN LẬP TRÌNH

INTRODUCTION TO COMPUTER PROGRAMMING

CSE102

Giảng viên: TS.Nguyễn Quỳnh Diệp

Email: diepnq@tlu.edu.vn



CHƯƠNG 2

TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH C



NỘI DUNG CHÍNH



1. Giới thiệu tổng quan Ngôn ngữ lập trình C
2. Làm việc với Ngôn ngữ C
3. Nhập, xuất dữ liệu



Giới thiệu tổng quan C

1. Giới thiệu Ngôn ngữ lập trình C
2. Các phần tử cơ bản trong C
3. Cấu trúc của 1 chương trình C
4. Biên dịch chương trình C

1. Giới thiệu Ngôn ngữ C

- C là ngôn ngữ lập trình cấp cao, được sử dụng phổ biến để lập trình hệ thống cùng với Assembler và phát triển các ứng dụng
- C ra đời do nhu cầu viết lại HĐH Unix cho các hệ máy tính khác nhau
 - Dùng ngôn ngữ Assembly
 - Công việc nặng nề, phức tạp
 - Khó chuyển đổi chương trình giữa các hệ máy tính khác nhau
 - Cần ngôn ngữ mới
 - Đơn giản việc lập trình
 - Tính khả chuyển cao
- C ra đời tại Bell Lab thuộc tập đoàn AT&T
 - Tác giả Brian W. Kernighan & Dennis Ritchie
 - Dựa trên ngôn ngữ BCPL & B
 - Phát triển năm 1970, hoàn thành 1972
- C là ngôn ngữ lập trình hệ thống mạnh và mềm dẻo, có thư viện nhiều hàm(function) đã được tạo sẵn, hỗ trợ nhiều phép toán.

■ Đặc điểm

- Ngôn ngữ lập trình hệ thống
- Tính khả chuyển, linh hoạt cao
- Có thể mạnh trong xử lý dữ liệu số, văn bản, cơ sở dữ liệu, ..

■ Phạm vi sử dụng

- Viết các chương trình hệ thống
- Các trình điều khiển thiết bị (device driver)
- Xử lý ảnh

2. Các phần tử cơ bản trong C

1. Tập ký tự
2. Từ khóa
3. Định danh
4. Các kiểu dữ liệu
5. Hằng
6. Biến
7. Hàm
8. Biểu thức
9. Câu lệnh
10. Chú thích

- Ký tự là các phần tử cơ bản tạo nên chương trình
- Chương trình: tập các câu lệnh nhằm giải quyết nhiệm vụ đặt ra
- Câu lệnh: các từ (từ vựng) liên kết với nhau theo cú pháp của ngôn ngữ lập trình
 - Ví dụ:

```
while(i < N ){  
    for(i = 0; i<n; i++)
```
- Các từ: Tổ hợp các ký tự theo nguyên tắc xây dựng từ vựng
 - Ví dụ: TenFile, BaiTap2...

Tập ký tự

- 26 chữ cái hoa: A B C ... X Y Z
- 26 chữ cái thường: a b c ... x y z.
- 10 chữ số: 0 1 2 3 4 5 6 7 8 9.
- Các kí hiệu toán học: + - * / = < >
- Các dấu ngăn cách: . ; , : space tab
- Các dấu ngoặc: () [] { }
- Các kí hiệu đặc biệt: _ ? \$ & # ^ \ ! ' " ~
...

Từ khóa (keyword)

- Được định nghĩa sẵn trong C
- Dành riêng cho các mục đích xác định
 - Đặt tên cho kiểu dữ liệu:
 - int, float, double...
 - Mô tả các lệnh, các cấu trúc lập trình
 - if, else, while, case, for...

- Các từ khóa thông dụng

break	case	char	const	continue	default
do	double	else	enum	float	for
goto	if	int	interrupt	long	return
short	signed	sizeof	static	struct	switch
typedef	union	unsigned	void	while	

- Lưu ý: Tất cả từ khóa trong C đều viết bằng **chữ cái thường**

Định danh (Identifier)

- Định danh (tên) là một dãy các kí tự dùng để gọi tên các đối tượng trong chương trình.
 - Các đối tượng trong chương trình
 - Biến
 - Hằng số
 - Hàm
 - Kiểu dữ liệu
- Định danh có thể được đặt bởi
 - Ngôn ngữ lập trình → các từ khóa
 - Người lập trình

- Định danh được bắt đầu bởi chữ cái hoặc dấu gạch dưới “_” (underscore)
- Các kí tự tiếp theo chỉ có thể là: chữ cái, chữ số hoặc dấu gạch dưới “_”
- Định danh do người lập trình đặt **không được trùng với các từ khóa của C**
- Độ dài định danh tùy thuộc phiên bản C, tối đa 32 kí tự
- Chú ý: C là ngôn ngữ **có phân biệt chữ hoa và chữ thường**

- Định danh hợp lệ:
 - i, x, y, a, b, _function,
 - _MY_CONSTANT, PI, gia_tri_1

- Định danh không hợp lệ
 - 1_a, 3d, 55x (bắt đầu bằng **chữ số**)
 - so luong, sin() (có **kí tự không hợp lệ**, dấu cách, dấu ngoặc..)
 - int, char (**trùng** với từ khóa của C)

- Định danh nên có tính gợi nhớ
- Nên sử dụng dấu gạch dưới để phân tách các định danh gồm nhiều từ
 - Có thể dùng cách viết hoa chữ cái đầu mỗi từ
 - Ví dụ: sinh_vien, sinhVien, SinhVien
- Quy ước thường được sử dụng:
 - Hằng số dùng chữ cái hoa
 - Ví dụ: PI, EPSILON,...
 - Các biến, hàm, cấu trúc dùng chữ cái thường
 - Biến điều khiển vòng lặp: i, j, k...
 - Hàm: NhapDuLieu, TimKiem,...
 - Cấu trúc: SinhVien, MatHang,...

- Một kiểu dữ liệu là một tập hợp các giá trị mà một dữ liệu thuộc kiểu đó có thể nhận được.

Ví dụ:

- Số nguyên: số nguyên có dấu, số nguyên không dấu
 - Kích thước: 2 hoặc 4 byte
 - Dải giá trị: 0->65.536 (hoặc -32.768-32.767)
hoặc 0->4294967296 (hoặc -2,147,483,648 tới 2,147,483,647)
- Số thực:
 - Float: 4 byte
 - Double: 8 byte
- Ký tự (char)
 - 1 byte
- Trên mỗi kiểu dữ liệu, xác định một số phép toán tương ứng.

- Một số phép toán trên kiểu dữ liệu số nguyên (int) của C

Tên phép toán	Ký hiệu	Ví dụ
Đảo dấu	-	
Cộng; Trừ; Nhân	+ - *	
Chia lấy nguyên	/	$17/3 \rightarrow 5$
Chia lấy phần dư	%	$17\%3 \rightarrow 2$
So sánh	> < >= <= == !=	

- Hằng là đại lượng có giá trị không đổi trong chương trình.
- Giá trị hằng do người lập trình xác định.
- Các loại hằng
 - Hằng số nguyên
 - Vd: 2567
 - Hằng số thực: biểu diễn bằng dấu phẩy động hoặc tĩnh
 - Vd: $123.456 = 12.456E+1 = 1.23456E+2 = 1234.56E-1$
 - Hằng ký tự: đặt trong cặp dấu nháy đơn ''
 - Vd: 'A', '&', '@'
 - Hằng chuỗi/xâu ký tự: đặt trong cặp dấu nháy kép ""
 - Vd: "Thuyloi", "xin chao"

Biến (variable)

- Biến là đối tượng dùng để lưu giữ dữ liệu và có thể thay đổi giá trị khi chương trình được thực thi.
- Biến được đặt trong các ô nhớ thuộc bộ nhớ chính của máy tính.
- Số ô nhớ dành cho 1 biến phụ thuộc kiểu dữ liệu của biến.
 - VD mỗi biến kiểu float chiếm 4byte trong bộ nhớ; biến kiểu int chiếm 2byte,...
- Tên biến phải được đặt theo quy tắc định danh.

Hàm (function)

- Hàm là chương trình con thực hiện 1 chức năng nào đó
 - Nhận dữ liệu đầu vào (các tham số vào)
 - Thực hiện một công việc nào đó
 - Có thể trả về một kết quả gọi là giá trị của hàm
 - Ví dụ: hàm $\sin(x)$
 - $\sin(3.14/2) \rightarrow 1.000$
 - $\sin(3.14/6) \rightarrow 0.499770$
- Có 2 loại hàm:
 - Hàm có sẵn của các thư viện C cung cấp
 - Hàm do người lập trình tự định nghĩa
- Chi tiết về hàm sẽ được học ở phần sau

Hàm	Ý nghĩa	Ví dụ
<code>sqrt(x)</code>	Căn bậc 2 của x	<code>sqrt(16.0) → 4.0</code>
<code>pow(x,y)</code>	x mũ y (x^y)	<code>pow(2,3) → 8</code>
<code>fabs(x)</code>	Trị tuyệt đối của x ($ x $)	<code>fabs(-5.0) → 5.0</code>
<code>exp(x)</code>	E mũ x (e^x)	<code>exp(1.0) → 2.71828</code>
<code>log(x)</code>	Logarithm tự nhiên của x ($\ln x$)	<code>log(2.718) → 0.999</code>
<code>log10(x)</code>	Logarithm cơ số 10 của x ($\log x$)	<code>log10(100) → 2.00</code>
<code>sin(x)</code> <code>cos(x)/ tan(x)</code>	Các hàm lượng giác	
<code>ceil(x)</code>	Số nguyên nhỏ nhất không nhỏ hơn x ($\lceil x \rceil$)	<code>ceil(2.5)=3</code> <code>ceil(-2.5)=-2</code>
<code>floor(x)</code>	Số nguyên lớn nhất không lớn hơn x ($\lfloor x \rfloor$)	<code>floor(2.5)=2</code> <code>floor(-2.5)=-3</code>

- Biểu thức là sự kết hợp các toán hạng (operand) với các toán tử (operator) theo một quy tắc xác định.
 - Toán hạng có thể là biến, hằng, hàm, biểu thức...
 - Các toán tử rất đa dạng: cộng, trừ, nhân, chia...
- Khi biểu thức được tính toán sẽ trả về một kết quả.

Câu lệnh (statement)

- Câu lệnh diễn tả một hoặc một nhóm các thao tác trong giải thuật.
 - Chương trình được tạo thành từ dãy các câu lệnh.
- Các câu lệnh trong C được kết thúc bởi dấu chấm phẩy (;)
 - Chú ý: lệnh khối (sẽ học sau) **không cần** kết thúc bằng dấu ;
- Câu lệnh đơn là những câu lệnh không chứa câu lệnh khác.
 - Ví dụ: lệnh gán, gọi hàm, vào/ra dữ liệu
- Các câu lệnh phức: là những câu lệnh chứa câu lệnh khác.
 - Ví dụ: Lệnh khối/khối lệnh: là tập các lệnh đơn nhóm lại với nhau và đặt trong cặp ngoặc nhọn { }
 - Các lệnh điều khiển cấu trúc chương trình
 - Ví dụ: Lệnh rẽ nhánh, lệnh lặp.

- Lời mô tả, giải thích trong chương trình
 - Giúp việc đọc hiểu chương trình dễ dàng hơn
 - Chú thích không phải là câu lệnh \Rightarrow không ảnh hưởng tới chương trình. Khi gặp chú thích, trình biên dịch sẽ bỏ qua
- Cách viết chú thích
 - Chú thích một dòng: sử dụng //
 - Chú thích nhiều dòng: sử dụng /* và */

3. Cấu trúc của 1 chương trình đơn giản

Khai báo các tệp tiêu đề
`#include`

Khai báo các đối tượng toàn cục

- Khai báo biến, hằng

Định nghĩa hàm `main()`

```
{  
  
  
}
```

Khai báo tệp tiêu đề

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

Định nghĩa hằng, khai
báo biến toàn cục

```
#define PI 3.14159  
const float G = 9.8;  
float r, S;
```

Hàm chính

```
void main(){  
    printf("Cho ban kinh r = "); scanf("%f",&r);  
    printf("Dien tich hinh tron S = %f\n", PI*r*r);  
    printf("Gia toc trong truong G = %2.1f",G);  
}
```

- Liệt kê danh sách thư viện sẽ được sử dụng trong chương trình
 - Các hàm của C đều thuộc một thư viện nào đó
 - Nếu không khai báo thư viện, trình biên dịch sẽ không hiểu được hàm (có thể báo lỗi)
- Cú pháp khai báo:
 - `#include <ThuVien.h>`
 - Ví dụ: `#include <stdio.h>`

Khai báo các tệp tiêu đề
`#include`

Khai báo các đối tượng toàn cục
- Khai báo biến, hằng

Định nghĩa hàm `main()`
{
}

- Các đối tượng toàn cục có phạm vi sử dụng trong **toàn bộ** chương trình
 - Các hằng, biến
 - Các kiểu dữ liệu mới
- Tuân theo nguyên tắc khai báo đối tượng

Khai báo các tệp tiêu đề
#include

Khai báo các đối tượng
toàn cục
- Khai báo biến, hằng

Định nghĩa hàm main()
{

}

- **Bắt buộc phải có**
- Là hàm đặc biệt trong C, đánh dấu điểm bắt đầu của mọi chương trình C
- Khi thực hiện một chương trình C, hệ thống sẽ gọi tới hàm main() đầu tiên, sau đó sẽ thực hiện lần lượt các câu lệnh (bao gồm cả lời gọi tới các hàm khác) nằm trong hàm main()

Khai báo các tệp tiêu đề
#include

Khai báo các đối tượng
toàn cục
- Khai báo biến, hằng

Định nghĩa hàm main()
{
}

▪ Cách 1: Cú pháp

```
void main(){  
  
    ....  
  
}
```

▪ Cách 2: Cú pháp

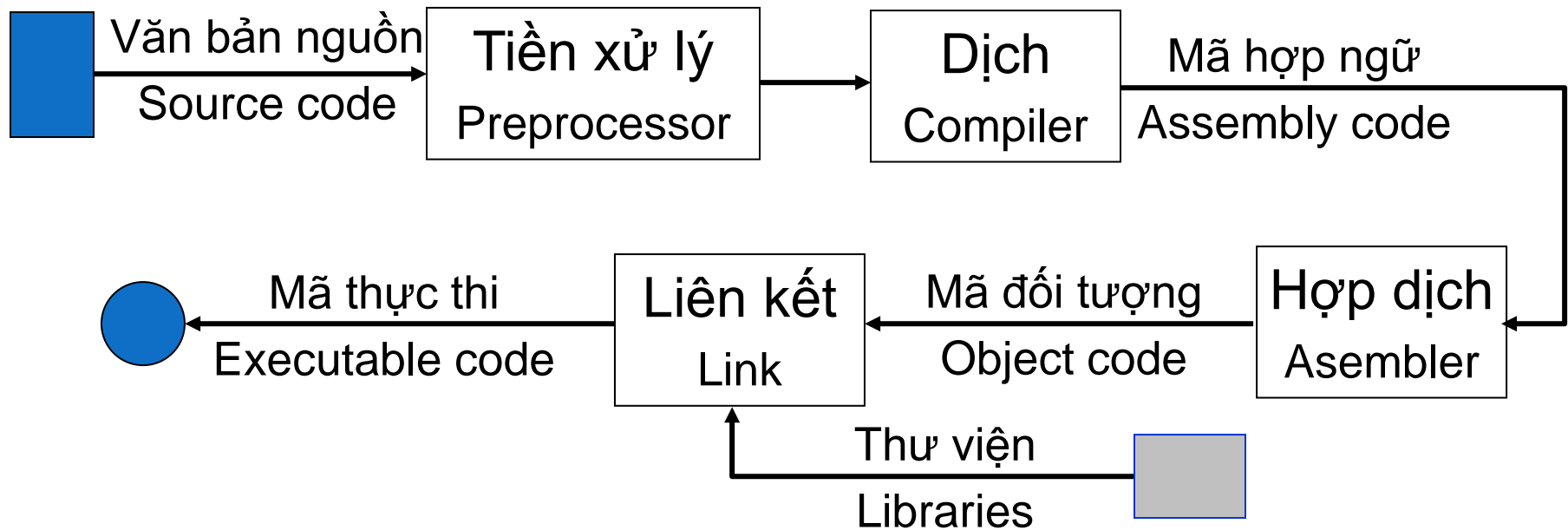
```
int main(){  
  
    ....;  
  
    return 0;  
  
}
```

```
1. #include <stdio.h>
2. int main() { //Không cần tham số
3.     printf("Hello world! \n");
4.     return 0; //Trả về giá trị 0
5. }
```

1. Khai báo thư viện **stdio.h**, đây là thư viện vào ra chuẩn (standard input output)
2. Điểm bắt đầu thực hiện của chương trình. Máy tính thực hiện các câu lệnh nằm trong cặp ngoặc {} của **main()**
3. Hàm **printf** in ra một xâu. Chú ý dấu xuống dòng (\n)
4. Hàm main trả về giá trị 0 (0 thường dùng để thể hiện chương trình hoạt động bình thường, không có lỗi).

4. Biên dịch Chương trình

- Chương trình được viết bằng ngôn ngữ bậc cao phải được dịch ra mã máy để thực thi
 - Công việc dịch được thực hiện bởi trình biên dịch (compiler)
- Các giai đoạn dịch chương trình



5. Công cụ lập trình

▪ Dev C++:

- Là một môi trường phát triển tích hợp tự do (IDE), hỗ trợ việc lập trình bằng ngôn ngữ C/C++. Dev C++ được phát triển bởi lập trình viên Colin Laplace.
- Dev C++ có giao diện sử dụng khá dễ dàng.

Dev-Cpp Ver 4.9.9.2 hoặc Dev-Cpp 5.1.1

▪ Visual Studio Code:

- VSCode là một công cụ lập trình (Code Editor) do Microsoft phát triển, VSCode có thể cài đặt và sử dụng trên cả Windows, MacOS và Linux.



Làm việc với C

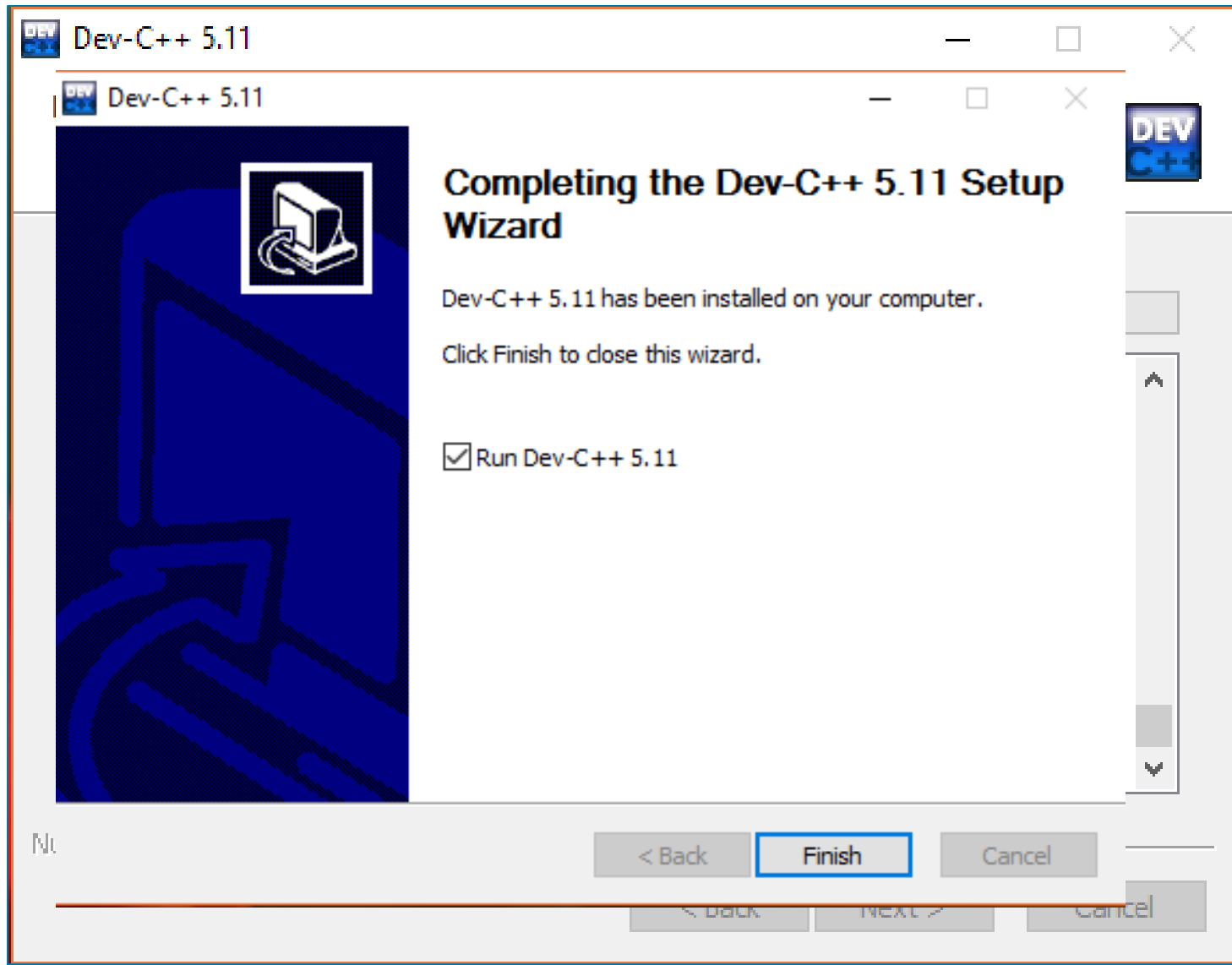
1. Cài đặt Dev-Cpp

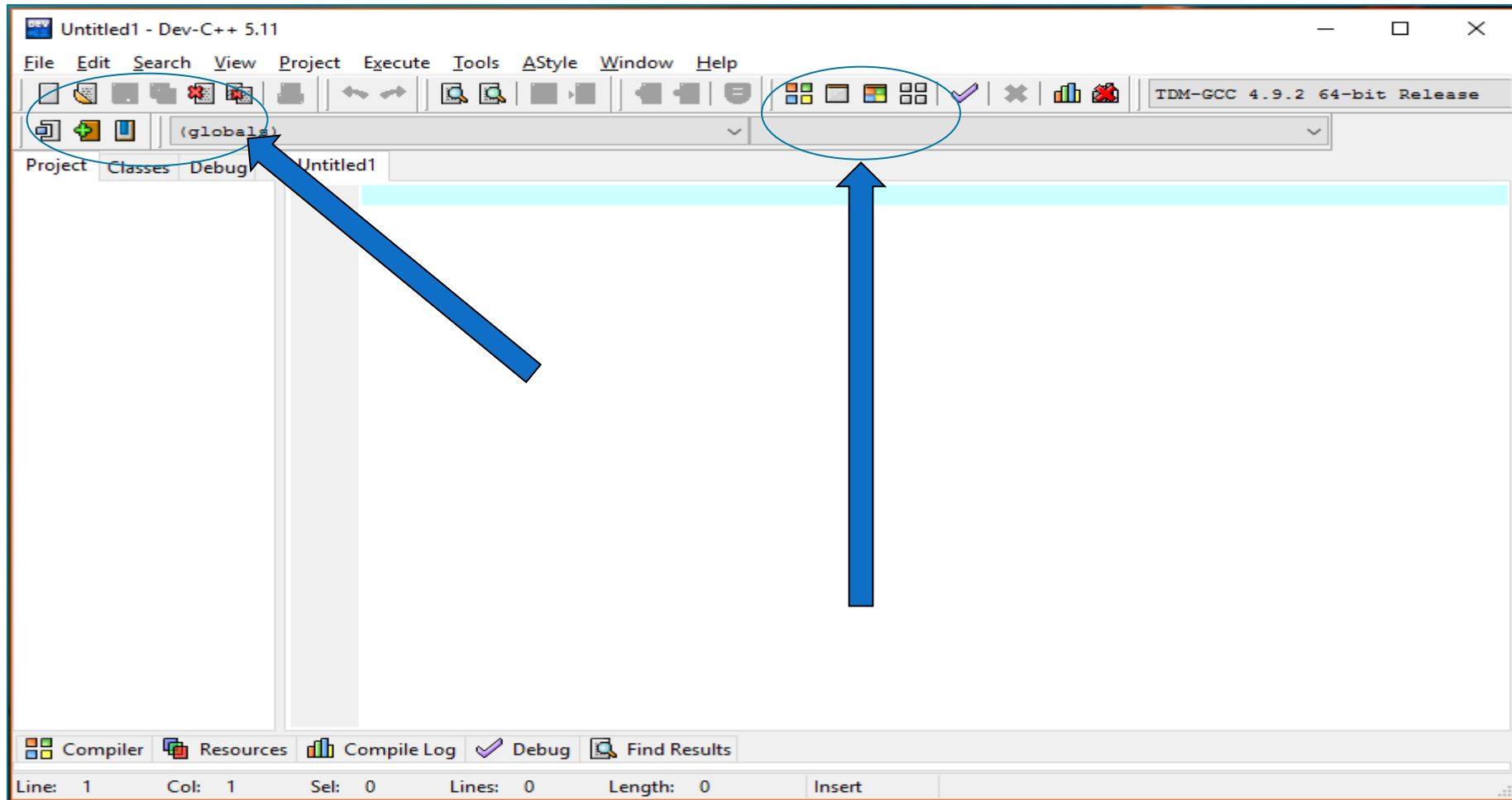
■ Tải Dev-C++

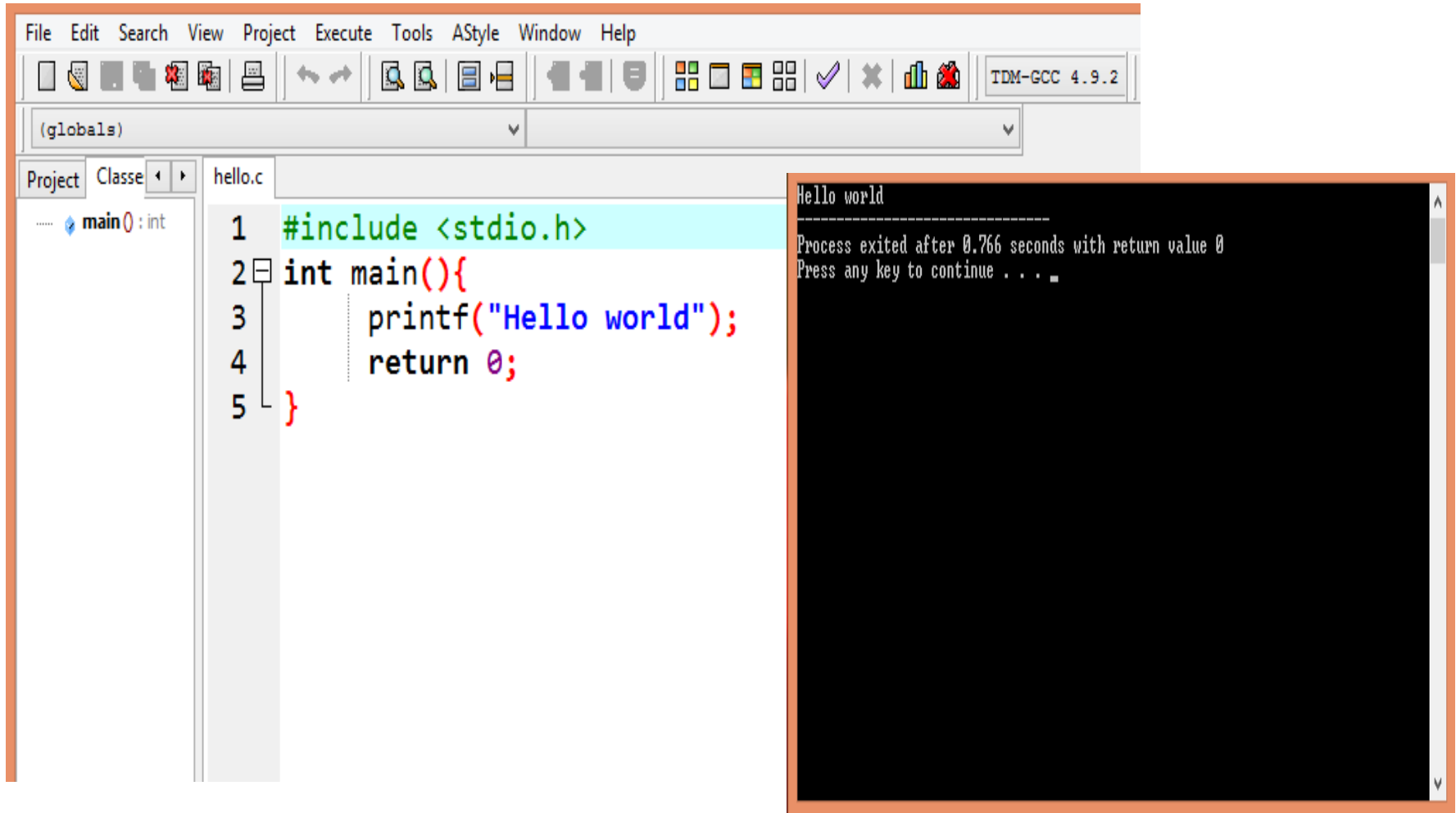
- Google \Rightarrow Dev-C++
- Tìm đến phiên bản thích hợp, thực hiện tải về
 - Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe
 - Devcpp_v4_9_9_2_setup.exe



■ Thực thi file tải về, làm theo hướng dẫn







The screenshot displays the DEV-C++ integrated development environment. The top menu bar includes File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, and Help. Below the menu is a toolbar with various icons for file operations, editing, and execution. The status bar at the bottom right indicates the compiler version as TDM-GCC 4.9.2.

The main workspace is divided into two panes. The left pane shows the project structure with a tree view containing 'main0 : int'. The right pane displays the source code for 'hello.c' with line numbers 1 through 5. The code is as follows:

```
1 #include <stdio.h>
2 int main(){
3     printf("Hello world");
4     return 0;
5 }
```

The right pane also shows the output of the program execution. The output is displayed in a black window with white text, showing 'Hello world' followed by a separator line, then 'Process exited after 0.766 seconds with return value 0', and finally 'Press any key to continue . . . '.

- Google Play/App store
- Gõ **Mobile C** (chữ **C** màu đỏ)
- Tải về điện thoại
- New Folder (+ thư mục), new file (+ file)
- Nhập tên thư mục/ tên file
- Soạn thảo chương trình (đuôi **.c**), sử dụng bàn phím ảo của chương trình. Bàn phím có thể phóng to (nhiều chức năng)/thu nhỏ (chỉ có phím chữ)
- Chạy CT bằng nút **Run** trên màn hình

2. Các kiểu dữ liệu

■ Kiểu đơn

Tên kiểu	Ý nghĩa	Kích thước	Miền dữ liệu
char	Kí tự; Số nguyên có dấu	1 byte	-128 ÷ 127
short int	Số nguyên có dấu	2 byte	-32.768 ÷ 32.767
int	Số nguyên có dấu	2 hoặc 4 byte	
long long int	Số nguyên có dấu	4 byte	-2,147,483,648 ÷ 2,147,483,647
float	Số thực dấu phẩy động	4 byte	$\pm 3.4E-38 \div \pm 3.4E+38$
double	Số thực dấu phẩy động	8 byte	$\pm 1.7E-308 \div \pm 1.7E+308$

- Kiểu kết hợp: Với số nguyên, thêm từ khóa **unsigned** để chỉ ra số **không dấu**

Kiểu dữ liệu	Ý nghĩa	Kích thước	Miền dữ liệu
unsigned char	Số nguyên không dấu	1 byte	$0 \div 255$
unsigned short	Số nguyên không dấu	2 byte	$0 \div 65.535$
unsigned int	Số nguyên không dấu	2 hoặc 4 byte	
unsigned long unsigned long int	Số nguyên không dấu	4 byte	$0 \div 4,294,967,295$
long double	Số thực dấu phẩy động,	10 byte	$\pm 3.4E-4932 \div \pm 1.1E+4932$

- Một biến **phải được khai báo trước khi sử dụng**
- Cú pháp khai báo:

```
<KieuDuKieu> TenBien;  
<KieuDuLieu> TenBien_1, ..., TenBien_N;
```

Ví dụ:

```
//Khai báo biến x là một số nguyên 4 byte có dấu  
    int x;  
//Khai báo các biến y, z là các số thực 4 byte  
    double y, z;  
//Sau khi khai báo, có thể sử dụng  
    x = 3; y = x + 1;
```

- Sau khi khai báo, biến chưa có giá trị xác định.
 - Biến cần được gán giá trị trước khi sử dụng
- C cho phép kết hợp khai báo và khởi tạo biến

```
<KieuDuLieu> TenBien = Giatribandau;  
<KieuDuLieu> TenBien_1 = Giatri1, ...,  
TenBien_N = GiatriN;
```

- Ví dụ:

```
//Khai báo biến nguyên a và khởi tạo giá trị  
bằng 3
```

```
int a = 3;
```

```
//Khai báo biến thực x,y và khởi tạo giá trị  
bằng 5.0 và 7.6
```

```
double x = 5.0, y = 7.6;
```

```
int n; m = 2 * n;  $\Rightarrow$  m=?
```

Khai báo kiểu
nguyên nhưng
khởi tạo là số
thực

```
#include <stdio.h>
#include <math.h> //de su dung hang M_PI

void main(){
    int r = 3.5;
    printf("Dien tich hinh tron voi ban kinh r = %d la S = %f.\n", r, M_PI*r*r);
}
```

```
Dien tich hinh tron voi ban kinh r = 3 la S = 28.274334.
```

```
-----
```

```
Process exited after 0.05746 seconds with return value 57
Press any key to continue . . .
```

```
1 //giai phuong trinh bac 1
2 #include <stdio.h>
3 float a, b, x;
4 void main(){
5     printf("Cho he so cua phuong trinh ax + b = 0.\n");
6     printf("a = "); scanf("%f",&a);
7     printf("b = "); scanf("%f",&b);
8     if(a == 0)
9         if( b == 0)
10             printf("PT vo so nghiem!");
11         else
12             printf("PT khong co nghiem!");
13     else
14         printf("PT co nghiem x = %5.2f", -b/a);
15 }
```

- Chú ý: Giá trị của các hằng phải được xác định ngay khi khai báo.

*Cách 1: Dùng **#define**

- Cú pháp:

`# define Tên_hằng Giá_trị`

Không có dấu ;

- Ví dụ:

```
#define MAX_SINH_VIEN 50
```

```
#define CNTT “Cong nghe thong tin”
```

```
#define DIEM_CHUAN 23.5
```

* Cách 2: Dùng từ khóa **const**

■ Cú pháp:

const Kiểu Tên_hằng = giá_trị;

■ Ví dụ:

```
const int MAX_SINH_VIEN = 50;  
const char CNTT[20] = "Cong nghe thong tin";  
const float DIEM_CHUAN = 23.5;
```



```
#include <stdio.h>
#include <math.h> //de su dung hang M_PI

#define PI 3.14159
float r1, r2, s1, s2;

void main(){
    printf("Cho ban kinh r1 = "); scanf("%f",&r1);
    s1 = PI*r1*r1;
    printf("Dien tich hinh tron 1 = %f.\n", s1);
    printf("Cho ban kinh r2 = "); scanf("%f",&r2);
    s2 = M_PI*r2*r2;
    printf("Dien tich hinh tron 2 = %f.",s2);
}
```

```
Cho ban kinh r1 = 1
Dien tich hinh tron 1 = 3.141590.
Cho ban kinh r2 = 1
Dien tich hinh tron 2 = 3.141593.
-----
Process exited after 2.668 seconds with return value 33
Press any key to continue . . .
```

2. Các toán tử trong C

1. Toán tử số học
2. Toán tử so sánh
3. Toán tử logic
4. Toán tử logic bit
5. Toán tử gán

Toán tử	Ý nghĩa	Kiểu dữ liệu của toán hạng	Ví dụ (<i>int a = 12; float x=3.0</i>)
-	Đảo dấu	float, double, int, long,.. (<i>Số nguyên hoặc thực</i>)	-12, -12.34, - a, - x
+	Cộng	float, double, int, long,..	12 + x
-	Trừ	float, double, int, long,..	12.0 - x
*	Nhân	float, double, int, long,.. (<i>Số nguyên hoặc thực</i>)	12 * 3.0 12 * 3
/	Chia	Nếu có ít nhất 1 toán hạng là số thực	17.0/3.0 → 5.666667 17/3.0 → 5.666667 17.0/3 → 5.666667
/	Chia lấy thương	Số nguyên int, long,..	17/3 → 5
%	Chia lấy số dư	Số nguyên: int, long,..	17%3 → 2

Các toán tử so sánh

$<, >, <=, >=, ==, !=$

- Dùng cho phép so sánh giá trị 2 toán hạng
- Kết quả phép so sánh là một số nguyên:
 - 1 nếu quan hệ có kết quả là đúng
 - 0 nếu quan hệ có kết quả sai
- Ví dụ:
 - $6 > 4 \rightarrow$ Trả về giá trị 1
 - $6 < 4 \rightarrow$ Trả về giá trị 0
 - `int b = (x != y);`

Nếu x và y khác nhau, biểu thức đúng và b mang giá trị 1. Ngược lại biểu thức sai và b mang giá trị 0

- Sử dụng để xây dựng các biểu thức logic
- Biểu thức logic có kết quả logic đúng
 - Trả về giá trị **khác 0**
- Biểu thức logic có kết quả logic sai
 - Trả về giá trị **bằng 0**

▪ Và (**&&**) :

- Cho kết quả đúng (trả về giá trị 1) khi cả 2 toán hạng đều đúng (khác 0)
- Ví dụ: $3 < 5 \ \&\& \ 4 < 6 \rightarrow 1$; $3 < 5 \ \&\& \ 5 > 6 \rightarrow 0$

▪ Hoặc (**||**):

- Cho kết quả sai (trả về giá trị 0) chỉ khi cả 2 toán hạng đều sai (bằng 0)
- Ví dụ: $4 || 5 < 3 \rightarrow 1$; $5 < 5 || 2 > 6 \rightarrow 0$

▪ Phủ định (**!**):

- Cho kết quả đúng (1) hoặc sai (0) khi toán hạng là sai (0) hoặc đúng (khác 0)
- Ví dụ: $!3 \rightarrow 0$; $!(2 > 3) \rightarrow 1$;

- Cú pháp: biến = biểu thức;
Hoặc biến = giá trị;
- Là toán tử được sử dụng thường xuyên
 - Biểu thức bên phải dấu bằng được tính toán
 - Giá trị của biểu_thức được gán cho biến ở vế trái
- Ví dụ:
 - `int a, b, c;`
 - `a = 3;`
 - `b = a + 5;`
 - `c = a * b;`

- Biểu thức gán là biểu thức nên cũng có giá trị.
 - Giá trị của biểu thức gán bằng giá trị của biểu_thức bên phải toán tử
 - Có thể gán giá trị của biểu thức gán cho một biến khác
 - Có thể sử dụng như một biểu thức bình thường
- Ví dụ:
 - `int a, b, c;`
 - `a = b = 2007;`
 - `c = (a = 20) * (b = 30);` kết quả `c=600`

- Toán tử số học:

$+=$

$-=$

$*=$

$/=$

$\%=$

Ví dụ:

$a *= b$ tương đương với $a = a * b$

3. Biểu thức trong C

- Xuất hiện trong vế phải của lệnh gán.
- Làm toán hạng trong các biểu thức khác.
- Làm biểu thức kiểm tra trong các cấu trúc điều khiển
 - Cấu trúc lặp: for, while, do while.
 - Cấu trúc rẽ nhánh: if, switch.
- Làm tham số thực sự trong lời gọi hàm.



Các loại biểu thức



- Biểu thức số học
- Biểu thức quan hệ
- Biểu thức logic

Biểu thức số học

- Là biểu thức mà giá trị của nó là các đại lượng số học (số nguyên, số thực).
 - Sử dụng các toán tử là các phép toán số học (cộng, trừ, nhân, chia...),
 - Các toán hạng là các đại lượng số học (hằng số, biến, biểu thức khác).
- Ví dụ: a, b, c là các biến thuộc kiểu số thực.
 - $3 * 3.7$
 - $8 + 6/3$
 - $a + b - c$
- Chú ý: phép chia **số nguyên/số nguyên** → **số nguyên**

```
1 //thao tac tren so nguyen
2 #include <stdio.h>
3
4 int main(){
5     int a =40, b = 9;
6     printf("a + b = %d", a+b);
7     printf("\na - b = %d", a-b);
8     printf("\na * b = %d", a*b);
9     printf("\na / b = %d", a/b);
10    return 0;
11 }
```

```
a + b = 49
a - b = 31
a * b = 360
a / b = 4
-----
```

```
1 //thao tac tren so thuc
2 #include <stdio.h>
3
4 int main(){
5     float a =40, b = 9;
6     printf("a + b = %f", a+b);
7     printf("\na - b = %f", a-b);
8     printf("\na * b = %f", a*b);
9     printf("\na / b = %f", a/b);
10    return 0;
11 }
```

```
a + b = 49.000000
a - b = 31.000000
a * b = 360.000000
a / b = 4.444444
-----
```

Biểu thức so sánh

- Là những biểu thức có sử dụng các toán tử quan hệ như lớn hơn, nhỏ hơn, khác nhau...
- Chỉ có thể trả về một trong 2 giá trị logic Đúng (TRUE) hoặc Sai (FALSE)
- Ví dụ

• <code>5 > 7</code>	• <code>// có giá trị logic là sai, FALSE</code>
• <code>9 != 10</code>	• <code>// có giá trị logic là đúng, TRUE</code>
• <code>2 >= 2</code>	• <code>// có giá trị logic là đúng, TRUE</code>
• <code>a > b</code>	• <code>// giả sử a, b là 2 biến kiểu int</code>

- **Chú ý:** Ngôn ngữ C coi các giá trị nguyên khác 0 là giá trị logic đúng (TRUE), giá trị 0 là giá trị logic sai (FALSE)

- Là biểu thức trả về các giá trị logic Đúng/Sai (True/False)
 - Các phép toán logic gồm có
 - AND VÀ logic, sử dụng toán tử &&
 - OR HOẶC logic, sử dụng toán tử ||
 - NOT PHỦ ĐỊNH, sử dụng toán tử !

- `(5 > 7) && (9 != 10)` • `//` có giá trị logic là sai, FALSE
- `0 || 1` • `//` có giá trị logic là đúng, TRUE
- `(5 > 7) || (9 != 10)` • `//` có giá trị logic là đúng, TRUE
- `0` • `//` có giá trị logic là sai, FALSE
- `!0` • `//` phủ định của 0, có giá trị logic là đúng, TRUE
- `3` • `//` có giá trị logic là đúng, TRUE
- `!3` • `//` phủ định của 3, có giá trị logic là sai, FALSE
- `(a > b) && (a < b)` • `//` Có giá trị sai, FALSE. Giả sử a, b là 2 biến kiểu int


```
1
2
3
4 int main(){
5     int a = 12, b = 23;
6     printf("%d", 5>7);
7     printf("\n%d", 9!=10);
8     printf("\n%d", 2>=2);
9     printf("\n%d", a>b);
10    return 0;
11 }
```

```
0
1
1
0
-----
Process exited after 0.1547 seconds
Press any key to continue . . .
```

4. Một số toán tử khác

- Tăng/giảm tự động 1 đơn vị
- Lấy địa chỉ
- Biểu thức điều kiện
- Toán tử phủ

Tăng giảm tự động một đơn vị

++	Tăng tự động	++x, x++
--	Giảm tự động	--x, x--

- Tiền tố (hậu tố): biến được tăng(++)/giảm(--), trước (sau) khi sử dụng để tính toán biểu thức
- Ví dụ:

```
int    a = 5, b, c, d, e;  
b = a++;           // b = 5 sau đó a = 6  
c = ++a;           // a = 7 rồi tới c = 7  
d = a--;           // d = 7 rồi tới a = 6  
e = --a;           // a = 5 sau đó e = 5  
a++ = ++a = tăng 1 lên 1 đơn vị
```

`&Tên_biến`

- Biến thực chất là một vùng nhớ của máy tính được đặt tên → tên của biến
- Mọi ô nhớ trên bộ nhớ máy tính đều được đánh địa chỉ.
 - Mọi biến đều có địa chỉ
- Ví dụ:
`int a = 2006;`
`&a` là địa chỉ của ô nhớ chứa giá trị biến a

$\text{exp1} \ ? \ \text{exp2} \ : \ \text{exp3}$

- exp1 là biểu thức logic
- Nếu $\text{exp1} \neq 0$ (giá trị đúng), biểu thức điều kiện trả về giá trị của exp2
- Nếu $\text{exp1} = 0$ (giá trị sai) biểu thức điều kiện trả về giá trị của exp3
- Ví dụ:
 float x= 5.2, y = 3.8, z;
 $z = (x < y) \ ? \ x \ : \ y;$
 $\rightarrow z = 3.8 \quad // \ z \min\{x, y\}$
 $\Leftrightarrow \text{if } (x < y) \ z = x; \text{ else } z = y;$

Toán tử phẩy (, comma)

biểu_thức_1, biểu_thức_2, ..

- Toán tử phẩy (,) cho phép sử dụng nhiều biểu thức tại nơi chỉ cho phép viết một biểu thức
- Các biểu thức được tính toán từ trái qua phải
- Giá trị và kiểu của biểu thức là giá trị và kiểu của biểu thức cuối cùng, bên phải

■ Ví dụ:

if (i = 0, a != b)...

for(i = 0, j = 0; i < 100; i++, j++)....

(Kiểu) biểu thức

- Dùng để ép 1 biểu thức từ kiểu này sang kiểu khác
- Được sử dụng khi muốn chuyển kiểu dữ liệu
- Cách viết: (kiểu cần ép) <biểu thức>
- Ví dụ: (float) a/b

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

void main(){
    int a, b;
    printf("Cho so nguyen a = "); scanf("%d",&a);
    printf("Cho so nguyen b = "); scanf("%d",&b);
    printf("a%%b = %d\n", a%b);
    printf("a/b = %d\n",a/b);
    printf("a/b = %4.2f", (float) a/b); |
}
```

```
Cho so nguyen a = 14
Cho so nguyen b = 5
a%b = 4
a/b = 2
a/b = 2.80
-----
Process exited after 7.081 seconds with return value 10
Press any key to continue . . .
```

Ép kiểu

Thứ tự ưu tiên các toán tử

Mức	Toán tử	Chức năng
1	-> . [] () ++ _{hậu tố} -- _{hậu tố}	Lựa chọn, chỉ số...
2	++ -- ~ ! + - * & () sizeof	Toán tử 1 ngôi, ép kiểu,...
3	* / %	Toán tử số học lớp nhân
4	+ -	Toán tử số học lớp cộng
5	< <= > >=	Toán tử quan hệ
6	== !=	Bằng, khác
7	&&	AND logic
8		OR logic
9	? :	Toán tử phỏng điều kiện
10	= *= += <<= &= ...	Toán tử gán

Thứ tự ưu tiên các toán tử

■ Nguyên tắc

- Biểu thức con trong ngoặc được tính toán trước
- Phép toán một ngôi đứng bên trái toán hạng được kết hợp với toán hạng đi liền nó.
- Toán hạng đứng cạnh hai toán tử
 - Nếu hai toán tử có độ ưu tiên khác nhau thì toán tử nào có độ ưu tiên cao hơn sẽ kết hợp với toán hạng
 - Nếu hai toán tử cùng độ ưu tiên thì dựa vào trật tự kết hợp của các toán tử để xác định toán tử được kết hợp với toán hạng.

■ Ví dụ

$$a < 10 \ \&\& \ 2 * b < c \equiv (a < 10) \ \&\& \ ((2 * b) < c)$$

Chú ý: `int x = 5, a = 5 * x++;` kết quả: `a = 25, x = 6`

```
const int N=10;  
float S = 0.0;  
int b;  
S = N/3 +1;  
b=(S>4);
```

S= ? b = ?

```
int a=3, b=4, c;  
c = a++ * ++b;
```

a= ? b= ? c= ?

```
int k, num=30;  
k = num>5 ? (num <=10 ? 100 : 200) : 500;  
k=?
```

```
const int N=10;  
float S= 0.0;  
int b;  
S = N/3 +1;  
b=(S>4);
```

S= 4 b = 0

```
int a= 3, b=4, c;  
c = a++ * ++b;
```

a=4 b=5 c=15

```
int k, num=30;  
k = num>5 ? (num <=10 ? 100 : 200) : 500;  
k=200
```



Nhập, xuất dữ liệu



Nhập, xuất dữ liệu



- Xuất dữ liệu với `printf()`
- Nhập dữ liệu với `scanf()`

- Xuất dữ liệu:
 - printf()
- Nhập dữ liệu
 - scanf()
- Cần khai báo tệp tiêu đề
 - `#include <stdio.h>`

- Tác dụng:
 - Hiển thị ra màn hình các loại dữ liệu cơ bản
 - Số nguyên, số thực, kí tự, xâu kí tự
 - Tạo một số hiệu ứng hiển thị đặc biệt
 - Xuống dòng, sang trang,...
- Cú pháp: **printf(xau_dinh_dang [, DS_tham_so]);**
- **Xau_dinh_dang**: Là một xâu qui định cách thức hiển thị dữ liệu ra màn hình máy tính.
 - Bao gồm các nhóm kí tự định dạng
 - Nhóm kí tự định dạng thứ k xác định quy cách hiển thị tham số thứ k trong DS_tham_so
 - Số lượng tham số trong DS_tham_so bằng số lượng nhóm các kí tự định dạng trong xau_dinh_dang.
- **DS_tham_so**: Danh sách các biến/biểu thức sẽ được hiển thị giá trị lên màn hình theo cách thức được qui định trong xau_dinh_dang.


```
#include <stdio.h>
void main() {
    int a = 5;
    float x = 1.234;
    printf("Hien thi mot bieu thuc nguyen
%d va mot so thuc %f",2*a,x);
}
```

- Kết quả: Hien thi mot bieu thuc nguyen 10 va mot so thuc 1.234000

Xâu định dạng

- Các kí tự thông thường:
 - Được hiển thị ra màn hình.
- Các kí tự điều khiển:
 - Dùng để tạo các hiệu ứng hiển thị đặc biệt như xuống dòng ('\n').
- Các nhóm kí tự định dạng:
 - Xác định quy cách hiển thị các tham số trong phần danh_sach_tham_so.

- Mỗi nhóm ký tự định dạng chỉ dùng cho một kiểu dữ liệu

Ví dụ: %d dùng cho kiểu nguyên, %f dùng cho kiểu thực

- **DS_tham_so** phải phù hợp với các nhóm ký tự định dạng trong **xau_dinh_dang** về:

- Số lượng;
- Thứ tự;
- Kiểu dữ liệu;

Nếu không phù hợp sẽ hiển thị ra kết quả không như ý

```
printf(" %d ", 3.14); → -31457
```

Ký tự	Kiểu dữ liệu	Kết quả
%d	int	Số nguyên
%f	float	Số thực dấu phẩy tĩnh
%lf	double	Số thực dấu phẩy tĩnh
%g	float/double	Hiển thị dạng rút gọn, bỏ các số 0 vô nghĩa
%c	char	Kí tự
%s	char []	Hiển thị chuỗi kí tự kết thúc bởi '\0'
%e, %E	float/double	Số thực dấu phẩy động

Ký tự	Kiểu dữ liệu	Kết quả
%ld	long int	Số nguyên lớn
%lf	long double	Số thực lớn
%lo	long	Số hệ 8 (không có 0 đằng trước)
%lx, %LX	long	Số hệ hexa (chữ thường/chữ hoa)
%lu	unsigned long	Số thập phân

Ký tự	Kiểu dữ liệu	Kết quả
%%		Hiển thị kí tự %
%o	int, char	Số hệ 8 (không có 0 đằng trước)
%x %X	int, char	Số hệ hexa (chữ thường/chữ hoa)
%u	unsigned int/char	Số nguyên có dấu

- Có dạng “%**m**”,
 - m là một giá trị nguyên, không âm.
 - m cho biết số chỗ trống dành cho hiển thị biểu thức tương ứng

- Ví dụ:

```
int a = 1234;
```

```
printf(“%5d”,a) → 1234
```

```
printf(“%5d”,34) → 34
```

 ký hiệu cho dấu trắng (space)

- `printf("\n%3d %15s %3c", 1, "Nguyen Van A", 'g');`
- `printf("\n%3d %15s %3c", 2, "Tran Van B", 'k');`

□ □ 1 □ □ □ □ Nguyen □ Van □ A □ □ □ g

□ □ 2 □ □ □ □ □ □ Tran □ Van □ B □ □ □ k

- Có dạng “%m.n”,
 - m, n là 2 giá trị nguyên, không âm.
 - m cho biết **kích thước để hiển thị số thực**
 - n cho biết kích thước dành cho phần thập phân, nếu không đủ C sẽ làm tròn khi hiển thị
- Ví dụ:

```
1  #include <stdio.h>
2  void main() {
3      printf("%f",17.346);
4      printf("\n%.2f",17.346);
5      printf("\n%.2f",17.345);
6      printf("\n%8.2f",17.346);
7      printf("\n%8.2f",17.344);
8  }
```

```
17.346000
17.35
17.34
    17.35
    17.34
-----
Process exited after 0.09906 s
Press any key to continue . .
```

- Nếu số chỗ cần để hiển thị dữ liệu lớn hơn được cung cấp trong định dạng \Rightarrow Tự động cung cấp thêm chỗ mới để hiển thị đầy đủ, không cắt bớt nội dung của dữ liệu.
- Ví dụ:
`printf("%2d", 1234);` \rightarrow 1234
`printf("%6.3f", 123.456);` \rightarrow 123.456
`printf("%12.6e", 123.456);` \rightarrow 1.234560e+02
`printf("%12.3e", 123.456);` \rightarrow 1.235e+02

- Khi hiển thị dữ liệu có sử dụng tham số độ rộng, để căn lề trái cần thêm dấu trừ - vào ngay sau dấu %:

- Ngầm định, căn lề phải

%-

- Ví dụ:

```
printf("%-3d%-10s%-5.2f%-3c", 5, "Hello", 7.5, 'g')
```

→ 5□□Hello□□□□□7.50□g□□

- Tác dụng: Dùng để nhập dữ liệu từ bàn phím
- Các loại dữ liệu nhập:
 - Ký tự đơn lẻ
 - Chuỗi ký tự
 - Số nguyên: hệ 10, 8, 16
 - Số thực: Dấu phẩy tĩnh; Dấu phẩy động
- Cú pháp

`scanf(xau_dinh_dang [,DS_dia_chi]);`

```
scanf(Xau_dinh_dang [, DS_dia_chi]);
```

- **Xau_dinh_dang**: Gồm các ký tự được qui định cho từng loại dữ liệu được nhập vào.
 - Ví dụ: dữ liệu định nhập kiểu nguyên thì xâu định dạng là : %d
- **DS_dia_chi**: bao gồm địa chỉ của các biến (toán tử &), phân tách nhau bởi dấu phẩy (,)
- Phải phù hợp với các kí tự định dạng trong xau_dinh_dang về số lượng, kiểu, thứ tự

- Đọc các ký tự được gõ vào từ bàn phím
- Căn cứ vào cấu trúc định dạng, chuyển thông tin đã nhập sang kiểu dữ liệu phù hợp
- Gán những giá trị vừa nhập vào các biến tương ứng trong **DS_dia_chi**
- Ví dụ:

```
int a;
```

```
scanf("%d",&a); → 1234_ → a = 1234
```

- Thông tin được gõ vào từ bàn phím, được lưu ở vùng đệm trước khi được xử lý bởi hàm scanf() → Hàm scanf() đọc từ vùng đệm

```
#include <stdio.h>
int main(){
    int a, b;
    scanf("%d",&a);
    scanf("%d",&b);
    printf("%d %d", a, b);
    return 0;
}
```



Kí tự	Khuôn dạng dữ liệu nhập
%c	Đọc kí tự đơn lẻ
%d	Đọc số nguyên
%o	Đọc số hệ 8
%x	Đọc số hệ hexa
%u	Đọc số thập phân không dấu

Kí tự	Chú thích
%s	Đọc chuỗi kí tự tới khi gặp dấu phân cách
%f	Đọc số thực dấu phẩy tĩnh (float)
%ld	Đọc số nguyên kiểu long
%lf	Đọc số thực dấu phẩy tĩnh (double)
%e	Đọc số thực dấu phẩy động
%%	Đọc ký tự %

```
Nhap so nguyen int n = 123
Ket qua cua so nguyen n theo %d: 123
Ket qua cua so nguyen n theo %5d: 123
Ket qua cua so nguyen n theo %i: 123
Ket qua cua so nguyen n theo %o: 173
Ket qua cua so nguyen n theo %x: 7b
Ket qua cua so nguyen n theo %X: 7B
Ket qua cua so nguyen n theo %u: 123
(Loi) Ket qua cua so nguyen n theo %f: 0.000000
-----
Process exited after 1.946 seconds with return value 0
Press any key to continue . . .
```

```
1 #include<stdio.h>
2 #include<conio.h>
3 int main(){
4     int n;
5     printf("Nhap so nguyen int n = "); scanf("%d", &n);
6     printf("Ket qua cua so nguyen n theo %%d: %d",n);
7     printf("\nKet qua cua so nguyen n theo %%5d: %5d",n);
8     printf("\nKet qua cua so nguyen n theo %%i: %i",n);
9     printf("\nKet qua cua so nguyen n theo %%o: %o",n);
10    printf("\nKet qua cua so nguyen n theo %%x: %x",n);
11    printf("\nKet qua cua so nguyen n theo %%X: %X",n);
12    printf("\nKet qua cua so nguyen n theo %%u: %u",n);
13    printf("\n(Loi) Ket qua cua so nguyen n theo %%f: %f",n);
14    return 0;
15 }
```

```
Nhap so thuc x = 123.25
Ket qua cua so thuc x theo %10f: 123.250000
Ket qua cua so thuc x theo %10.4f: 123.2500
Ket qua cua so thuc x theo %.7f: 123.2500000
Ket qua cua so thuc x theo %e: 1.232500e+002
Ket qua cua so thuc x theo %g: 123.25
(Loi!) Ket qua cua so thuc x theo %d: 0
```

```
-----
Process exited after 4.363 seconds with return value 0
Press any key to continue . . .
```

```
1  #include<stdio.h>
2  #include<conio.h>
3  int main(){
4      float x;
5      printf("Nhap so thuc x = "); scanf("%f", &x);
6      printf("Ket qua cua so thuc x theo %10f: %10f",x);
7      printf("\nKet qua cua so thuc x theo %10.4f: %10.4f",x);
8      printf("\nKet qua cua so thuc x theo %%.7f: %%.7f",x);
9      printf("\nKet qua cua so thuc x theo %%e: %%e",x);
10     printf("\nKet qua cua so thuc x theo %%g: %%g",x);
11     printf("\n(Loi!) Ket qua cua so thuc x theo %d: %d",x);
12     return 0;
13 }
```

Minh họa hiển thị số thực (2)

```
1  #include<stdio.h>
2  #include<conio.h>
3  int main(){
4      double x;
5      printf("Nhap so thuc (long double) x = "); scanf("%lf",&x);
6      printf("Ket qua cua so thuc x theo %%lf: %lf",x);
7      printf("\nKet qua cua so thuc x theo %%e: %e",x);
8      printf("\nKet qua cua so thuc x theo %%g: %g",x);
9      printf("\nKet qua cua so thuc x theo %%10f: %10f",x);
10     printf("\nKet qua cua so thuc x theo %%10.4f: %10.4f",x);
11     printf("\nKet qua cua so thuc x theo %%0.7f: %0.7f",x);
12     printf("\n(Loi!) Ket qua cua so thuc x theo %%d: %d",x);
13     return 0;
14 }
```

```
Nhap so thuc (long double) x = 123.25
Ket qua cua so thuc x theo %lf: 123.250000
Ket qua cua so thuc x theo %e: 1.232500e+002
Ket qua cua so thuc x theo %g: 123.25
Ket qua cua so thuc x theo %10f: 123.250000
Ket qua cua so thuc x theo %10.4f: 123.2500
Ket qua cua so thuc x theo %.7f: 123.2500000
(Loi!) Ket qua cua so thuc x theo %d: 0
-----
Process exited after 6.749 seconds with return value 0
Press any key to continue . . .
```

Minh họa hiển thị kí tự, xâu kí tự

```

1  #include<stdio.h>
2  #include<conio.h>
3  int main(){
4      char c; char s[30];
5      printf("Nhap ki tu c = ");
6      fflush(stdin); scanf("%c", &c);
7      printf("Ket qua cua ki tu x theo %%c: %c",c);
8      printf("\nKet qua cua ki tu x theo %%5c: %5c",c);
9      printf("\nKet qua cua ki tu x theo %%-5c: %-5c",c);
10     printf("\n\nNhap xau ki tu (scanf()) s = ");
11     fflush(stdin); scanf("%s", &s);
12     printf("Ket qua cua xau ki tu s theo %%s: %s",s);
13     printf("\nKet qua cua xau ki tu s theo %%10s: %10s",s);
14     printf("\nKet qua cua xau ki tu s theo %%-10s: %-10s",s);
15     printf("\n\nNhap xau ki tu (gets()) s = ");
16     fflush(stdin); gets(s);
17     printf("Ket qua cua xau ki tu s theo %%s: %s",s);
18     printf("\nKet qua cua xau ki tu s theo %%10s: %10s",s);
19     printf("\nKet qua cua xau ki tu s theo %%-10s: %-10s",s);
20     return 0;
21 }

```

```

Nhap ki tu c = A
Ket qua cua ki tu x theo %c: A
Ket qua cua ki tu x theo %5c:   A
Ket qua cua ki tu x theo %-5c: A

Nhap xau ki tu (scanf()) s = Bach Khoa
Ket qua cua xau ki tu s theo %s: Bach
Ket qua cua xau ki tu s theo %10s:      Bach
Ket qua cua xau ki tu s theo %-10s: Bach

Nhap xau ki tu (gets()) s = Bach Khoa
Ket qua cua xau ki tu s theo %s: Bach Khoa
Ket qua cua xau ki tu s theo %10s:  Bach Khoa
Ket qua cua xau ki tu s theo %-10s: Bach Khoa
-----

```

```
2  #include <stdio.h>
3  void main(){
4      // khai bao bien
5      int a; float x;
6      char ch;
7      char str[30];
8      // Nhap du lieu
9      printf("Nhap vao mot so nguyen: "); scanf("%d",&a);
10     printf("\nNhap vao mot so thuc: "); scanf("%f",&x);
11     fflush(stdin); //xoa bo nho dem
12     printf("\nNhap vao mot ki tu: "); scanf("%c",&ch);
13     printf("\nNhap vao mot xau ki tu: "); scanf("%s",str);
14     // Hien thi du lieu vua nhap vao
15     printf("\nNhưng du lieu vua nhap vao");
16     printf("\nSo nguyen: %d", a);
17     printf("\nSo thuc: %5.2f", x);
18     printf("\nKy tu: %c", ch);
19     printf("\nXau ky tu: %s", str);
20 }
```

Ví dụ → Kết quả thực hiện

```
Nhap vao mot so nguyen: 12
Nhap vao mot so thuc: 23.4456
Nhap vao mot ki tu: a
Nhap vao mot xau ki tu: Dai hoc Bach Khoa
Nhưng dữ liệu vừa nhập vào
Số nguyên: 12
Số thực: 23.45
Ký tự: a
Xâu ký tự: Dai
-----
Process exited after 0.00 seconds
Press any key to continue
```

Không hiển thị hết
xâu kí tự vừa nhập
"Dai hoc Bach Khoa"

■ Khi đọc số

- Hàm scanf() quan niệm rằng mọi kí tự số, dấu chấm ('.') đều là kí tự hợp lệ.
- Khi gặp các dấu phân cách như **tab**, **xuống dòng (enter)** hay **dấu cách (space bar)**, scanf() sẽ hiểu là kết thúc nhập dữ liệu cho một số.

■ Khi đọc kí tự

- Hàm scanf() cho rằng mọi kí tự có trong bộ đệm đều là hợp lệ, kể cả các kí tự tab, xuống dòng hay dấu cách

■ Khi đọc xâu kí tự:

- Hàm scanf() nếu gặp các kí tự dấu trắng, dấu tab hay dấu xuống dòng thì nó sẽ hiểu là kết thúc nhập dữ liệu cho một xâu kí tự.


```
2  #include <stdio.h>
3  void main(){
4      // khai bao bien
5      int a; float x;
6      char ch;
7      char str[30];
8      printf("Nhap vao mot so nguyen: "); scanf("%d",&a);
9      printf("\nNhap vao mot so thuc: "); scanf("%f",&x);
10     fflush(stdin); //xoa bo nho dem
11     printf("\nNhap vao mot ki tu: "); scanf("%c",&ch);
12     fflush(stdin);
13     printf("\nNhap vao mot xau ki tu: ");
14     gets(str); //Nhap xau ki tu co chua dau cach
15     printf("\nHung du lieu vua nhap vao");
16     printf("\nSo nguyen: %d", a);
17     printf("\nSo thuc: %5.2f", x);
18     printf("\nKy tu: %c", ch);
19     printf("\nXau ky tu: %s", str);
20 }
```

Nhập 2 số nguyên, đưa ra tổng, hiệu, tích...

```
#include <stdio.h>

int main(){
    int A, B;
    printf("Nhap vao 2 so nguyen : "); scanf("%d %d",&A,&B);
    printf("\n");
    printf("Tong %d + %d = %d \n", A, B, A + B);
    printf("Hieu %d - %d = %d\n", A, B, A - B);
    printf("Tich %d x %d = %d\n", A, B, A * B);
    printf("Thuong %d / %d = %.3f\n", A, B, (float)A / B);
    printf("Chia nguyen %d / %d = %d\n", A, B, A / B);
    printf("Chia du %d %% %d = %d\n", A, B, A % B);
    printf("\n");
    return 0;
}
```

```
Nhap vao 2 so nguyen : 17 5  
Tong 17 + 5 = 22  
Hieu 17 - 5 = 12  
Tich 17 x 5 = 85  
Thuong 17 / 5 = 3.400  
Chia nguyen 17 / 5 = 3  
Chia du 17 % 5 = 2
```

- Nhập tọa độ 3 điểm A,B,C và in ra diện tích $\triangle ABC$

```
#include <stdio.h>
#include <math.h>
int main(){
    float Ax,Ay, Bx, By, Cx, Cy, AB, BC, CA,p;
    printf("Nhap vao toa do diem A  : "); scanf("%f %f",&Ax,&Ay);
    printf("Nhap vao toa do diem B  : "); scanf("%f %f",&Bx,&By);
    printf("Nhap vao toa do diem C  : "); scanf("%f %f",&Cx,&Cy);

    //Tinh do dai cac canh cua tam giac
    AB = sqrt((Ax-Bx)*(Ax-Bx)+(Ay-By)*(Ay-By));
    BC = sqrt((Bx-Cx)*(Bx-Cx)+(By-Cy)*(By-Cy));
    CA = sqrt((Cx-Ax)*(Cx-Ax)+(Cy-Ay)*(Cy-Ay));
    p = (AB + BC + CA)/2;
    printf("Dien tich tam giac ABC la: %f",sqrt(p*(p-AB)*(p-BC)*(p-
CA))));
    printf("\n");
    return 0;
}
```

```
Nhap vao toa do diem A : 0 0
Nhap vao toa do diem B : 6 0
Nhap vao toa do diem C : 0 8
Dien tich tam giac ABC la: 24.000000
```

```
Nhap vao toa do diem A : -1 -2
Nhap vao toa do diem B : 5 -2
Nhap vao toa do diem C : 6 6
Dien tich tam giac ABC la: 24.000000
```

```
Nhap vao toa do diem A : 1 1
Nhap vao toa do diem B : 2 2
Nhap vao toa do diem C : 4 4
Dien tich tam giac ABC la: 0.000000
```

1. Viết chương trình nhập vào từ bàn phím bán kính một đường tròn và đưa ra màn hình diện tích và chu vi đường tròn
 2. Viết chương trình nhập vào từ bàn phím một số thực. Hãy đưa ra diện tích của hình tròn có bán kính là số thực đã nhập, diện tích hình vuông có cạnh là số thực đó, chu vi tam giác đều có cạnh là số thực vừa nhập.
- Lưu ý: có thể sử dụng hằng π theo 2 cách sau:
 - Khai báo hằng $PI = 3.1416$ trong chương trình.
 - Sử dụng hằng M_PI có sẵn trong tệp tiêu đề **math.h**

```
1  #include <stdio.h>
2  #include <math.h>
3  #define PI 3.1416
4  int main(){
5      float r;
6      printf("Cho ban kinh r = "); scanf("%f",&r);
7      printf("Chu vi hinh tron P = %f\n", 2*PI*r);
8      printf("Dien tich hinh tron S = %f\n", PI*r*r);
9      return 0;
10 }
```

```
Cho ban kinh r = 1
Chu vi hinh tron P = 6.283200
Dien tich hinh tron S = 3.141600
```

```
-----
Process exited after 1.585 seconds with return value 33
Press any key to continue . . .
```

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main(){
5      float r;
6      printf("Cho ban kinh r = "); scanf("%f",&r);
7      printf("Chu vi hình tron P = %f\n", 2*M_PI*r);
8      printf("Dien tích hình tron S = %f\n",M_PI*r*r);
9      return 0;
10 }
```

```
Cho ban kinh r = 1
Chu vi hình tron P = 6.283185
Dien tích hình tron S = 3.141593

-----
Process exited after 1.785 seconds with return value 0
Press any key to continue . . .
```


3. Viết chương trình nhập vào từ bàn phím chiều dài 3 cạnh của một tam giác, rồi đưa ra diện tích và các đường cao của tam giác đó.

4. Cho hàm số: $f(x) = x^7 + 5\sqrt[3]{x^5 + 3x^3 + 2} + 12$

Viết chương trình nhập vào 3 số thực a,b,c và đưa ra trung bình cộng của f(a), f(b), f(c).

5. Nhập vào x từ bàn phím và tính giá trị của biểu thức:

$$A = \frac{\cos 3a + \sqrt[5]{2x^3 + x + 1}}{\log_7(3^{x^2} + 2.14b)} \text{ trong đó } a = \sqrt{2^x + \pi} \text{ và } b = \ln(e^{x+1.23} + 1)$$