



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# TIN HỌC ĐẠI CƯƠNG

## Phần 2. Lập trình

### Chương 1: Tổng quan về ngôn ngữ C

# Các khái niệm

- Ngôn ngữ lập trình (Programming Language)
  - Ngôn ngữ được thiết kế và chuẩn hóa để truyền các chỉ thị cho máy tính,
  - Được sử dụng để tạo ra các chương trình nhằm mục đích điều khiển hoặc mô tả các thuật toán
- Chương trình (Program)
  - Tập hợp các chỉ thị được biểu thị qua ngôn ngữ lập trình nhằm thực hiện các thao tác máy tính nào đó

# Các khái niệm

- Lập trình (Programming)
  - Quá trình con người tạo ra chương trình máy tính sử dụng ngôn ngữ lập trình
- Mã nguồn (Source Code)
  - Tập hợp các chỉ thị được biểu thị qua ngôn ngữ lập trình nhằm thực hiện các thao tác máy tính nào đó





# Các khái niệm

- Đặc điểm của ngôn ngữ lập trình: mỗi ngôn ngữ lập trình có thể được xem như là một tập hợp của các chi tiết kỹ thuật
  - Dữ liệu và cấu trúc dữ liệu
  - Câu lệnh và dòng điều khiển
  - Các tên và tham số
  - Các cơ chế tham khảo và tái sử dụng

# Các ngôn ngữ lập trình



# Các ngôn ngữ lập trình

Language Rank	Types	Spectrum Ranking
1. Python	 	100.0
2. C	  	99.7
3. Java	  	99.5
4. C++	  	97.1
5. C#	  	87.7
6. R		87.7
7. JavaScript	 	85.6
8. PHP		81.2
9. Go	 	75.1
10. Swift	 	73.7

# Nội dung



## 1.1. Lịch sử phát triển

1.2. Các phần tử cơ bản của ngôn ngữ C

1.3. Cấu trúc cơ bản của chương trình C

1.4. Biên dịch chương trình C

1.5. Các trình biên dịch

## 1.1. Lịch sử phát triển

- Ngôn ngữ C ra đời tại phòng thí nghiệm BELL của tập đoàn AT&T (Hoa Kỳ)
- Do Brian W. Kernighan và Dennis M. Ritchie phát triển vào đầu 1970, hoàn thành 1972
- C dựa trên nền các ngôn ngữ BCPL (*Basic Combined Programming Language*) và ngôn ngữ B.
- Tên là ngôn ngữ C như là sự tiếp nối ngôn ngữ B.





## 1.1. Lịch sử phát triển

- 1978: C được giới thiệu trong phiên bản đầu của cuốn sách "*The C programming language*"
- Sau đó, C được bổ sung thêm những tính năng và khả năng mới → Đồng thời tồn tại nhiều phiên bản nhưng không tương thích nhau.
- Năm 1989, Viện tiêu chuẩn quốc gia của Hoa Kỳ (*American National Standards Institute* - ANSI) đã công bố phiên bản chuẩn hóa của ngôn ngữ C: *ANSI C* hay *C chuẩn* hay C89

## 1.1. Lịch sử phát triển

- Các phiên bản ngôn ngữ C
  - ANSI C: C chuẩn (1989)
  - Các phiên bản khác thường bổ sung thêm thư viện của ANSI C
- Hiện nay cũng có nhiều phiên bản của ngôn ngữ C khác nhau, gắn liền với một bộ chương trình dịch cụ thể của ngôn ngữ C
  - Turbo C++ và Borland C++ của Borland Inc.
  - MSC và VC của Microsoft Corp.
  - GCC của GNU project...

## 1.1. Lịch sử phát triển

- Đặc điểm của ngôn ngữ lập trình C
  - 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
- C thường được sử dụng để viết các chương trình hệ thống
  - Hệ điều hành Unix có 90% mã C, 10% hợp ngữ
  - Các trình điều khiển thiết bị (device driver)
  - Xử lý ảnh ...

# Nội dung

1.1. Lịch sử phát triển



1.2. Các phần tử cơ bản của ngôn ngữ C

1.3. Cấu trúc cơ bản của chương trình C

1.4. Biên dịch chương trình C

1.5. Các trình biên dịch

# Ví dụ chương trình C đơn giản

```
#include <stdio.h>
#include <conio.h>
void main() {
    printf("Hello World\n");
    getch();
}
```

# Ví dụ chương trình C đơn giản

`#include <stdio.h>` → Khai báo thư viện vào/ra chuẩn

`#include <conio.h>` → Khai báo thư viện vào/ra của MS-DOS

`void main()` { → Khai báo hàm main

`printf("Hello World\n");`

`getch();`

}

Câu lệnh in chuỗi ký tự ra màn hình

Câu lệnh chờ ký tự bất kỳ từ bàn phím

## 1.2.1. Tập ký tự

- Tập ký tự là tập các phần tử cơ bản tạo nên chương trình
  - Tổ hợp các ký tự → từ
  - Liên kết các từ theo cú pháp → câu lệnh
  - Tổ chức các câu lệnh → chương trình
- Ví dụ:
  - include, void, main...
  - printf(...), getch();

Câu lệnh printf

```
#include <stdio.h>
#include <conio.h>
void main(){
    printf("Hello World\n");
    getch();
}
```

## 1.2.1. Tập ký tự

- Tập ký tự trong C
  - 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: \_ ? \$ & # ^ \ ! ' " ~ .v.v.



## 1.2.2. Từ khóa

- Từ khóa (keyword)
  - Có sẵn trong mỗi ngôn ngữ lập trình
  - 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, while, case...
- Chú ý:
  - *C là ngôn ngữ phân biệt chữ hoa, chữ thường*
  - *Tất cả từ khóa trong C đều viết bằng chữ cái thường*

```
int x;  
for (x=1; x<20; x++)  
    if (x/2 == 1) break;
```

## 1.2.2. Từ khóa

- Một số từ khóa hay dùng trong Turbo C

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	

## 1.2.3. Định danh

- Định danh (*Identifier* – hoặc còn gọi là *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... (sẽ làm quen ở các mục sau)
- Định danh có thể được đặt bởi
  - Ngôn ngữ lập trình → các từ khóa
  - Người lập trình

## 1.2.3. Định danh

- Quy tắc đặt tên định danh trong C
  - Các kí tự được sử dụng: chữ cái, chữ số và dấu gạch dưới “\_” (*underscore*)
  - Bắt đầu của định danh phải là chữ cái hoặc dấu gạch dưới “\_”, không được bắt đầu định danh bằng chữ số.
  - Định danh do người lập trình đặt không được trùng với các từ khóa của C

## 1.2.3. Định danh

- Ví dụ
  - 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, ti le (có dấu cách - kí tự không hợp lệ)
    - int, char (trùng với từ khóa của ngôn ngữ C)

## 1.2.3. Định danh

- Một số quy ước (code convention)
  - Nên sử dụng dấu gạch dưới để phân tách các định danh gồm nhiều từ
  - Định danh nên có tính gợi nhớ
  - Quy ước thường được sử dụng:
    - Hằng số dùng chữ cái hoa
    - Các biến, hàm, cấu trúc dùng chữ cái thường
- Ví dụ

Định danh	Loại đối tượng
HANG_SO_1, _CONSTANT_2	Hằng số
a, b, i, j, count	Biến
nhap_du_lieu, tim_kiem, xu_li	Hàm
sinh_vien, mat_hang	Cấu trúc

## 1.2.4. Các kiểu dữ liệu

- Định nghĩa:
  - Một kiểu dữ liệu là một tập hợp các giá trị mà một dữ liệu thuộc kiểu dữ liệu đó có thể nhận được.
  - Trên một kiểu dữ liệu ta xác định một số phép toán đối với các dữ liệu thuộc kiểu dữ liệu đó.
- Ví dụ:
  - Kiểu dữ liệu int (số nguyên) trong C
  - Một dữ liệu thuộc kiểu dữ liệu int
    - Là một số nguyên
    - Nhận giá trị từ  $-32,768$  ( $-2^{15}$ ) đến  $32,767$  ( $2^{15} - 1$ )

## 1.2.4. Các kiểu dữ liệu

- Ví dụ (tiếp)
  - Một số phép toán được định nghĩa trên kiểu dữ liệu int của C

Tên phép toán	Ký hiệu
Đảo dấu	-
Cộng	+
Trừ	-
Nhân	*
Chia lấy phần nguyên	/
Chia lấy phần dư	%
So sánh	>, <, >=, <=, ==, !=



## 1.2.5. Hằng số

- Định nghĩa:
  - Hằng số (*constant*) là đại lượng có giá trị không đổi trong chương trình.
- Biểu diễn hằng số trong ngôn ngữ C:
  - Hằng số nguyên
  - Hằng số thực
  - Hằng ký tự
  - Hằng chuỗi ký tự

## 1.2.5. Hằng số

- Biểu diễn hằng số nguyên: trong C, một hằng số nguyên có thể biểu diễn dưới 3 dạng
  - Dạng thập phân
  - Dạng thập lục phân
  - Dạng bát phân

Giá trị thập phân	Giá trị thập lục phân	Giá trị bát phân
2007	<b>0x</b> 7D7	<b>0</b> 3727
396	<b>0x</b> 18C	<b>0</b> 614

## 1.2.5. Hằng số

- Biểu diễn hằng số thực: trong C, một hằng số thực có thể biểu diễn dưới 2 dạng
  - Dạng số thực dấu phẩy tĩnh
  - Dạng số thực dấu phẩy động
- Ví dụ

Số thực dấu phẩy tĩnh	Số thực dấu phẩy động
3.14159	31.4159 E-1
123.456	12.3456 E+1 hoặc 1.23456 E+2

## 1.2.5. Hằng số

- Biểu diễn hằng ký tự: trong C, một hằng ký tự có thể biểu diễn theo hai cách
  - Bằng ký hiệu của ký tự đặt giữa hai dấu nháy đơn
  - Bằng số thứ tự của ký tự đó trong bảng mã ASCII (số nguyên -> tuân thủ quy tắc biểu diễn hằng số nguyên)
- Ví dụ

Ký tự cần biểu diễn	Cách 1	Cách 2
Chữ cái A	'A'	65, 0x41, 0101
Số 1	'1'	49, 0x31, 061
Dấu nháy đơn	'\''	39, 0x27, 047
Ký tự tab	'\t'	9, 0x09, 011

## 1.2.5. Hằng số

- Biểu diễn hằng xâu (chuỗi) ký tự:
  - Hằng xâu kí tự được biểu diễn bởi dãy các kí tự thành phần có trong xâu đó và được đặt trong cặp dấu nháy kép.
- Ví dụ:
  - “ngon ngu lap trinh C”
  - “Tin hoc dai cuong”
  - “Dai hoc Bach Khoa Ha Noi”

## 1.2.6. Biến

- Định nghĩa:
  - Biến (*variable*) là đại lượng mà giá trị có thể thay đổi trong chương trình.
- Biến phải thuộc một kiểu dữ liệu nào đó.
- Biến được sử dụng để lưu trữ dữ liệu, phục vụ cho xử lý tính toán trong chương trình.
- Biến phải được đặt tên theo qui tắc đặt tên.
- Chú ý:
  - Biến được cấp phát vùng nhớ trong bộ nhớ chính, còn hằng thì không.

## Ví dụ 1. Minh họa sử dụng biến

```
#include <stdio.h>
#include <conio.h>
#define PI 3.14
void main() {
    float r, s; //Hai biến r, s thuộc kiểu số thực
    printf("Nhap ban kinh hình tron:");
    scanf("%f", &r);
    s = PI * r * r;
    printf("Dien tích hình tron %f\n", s);
    getch();
}
```

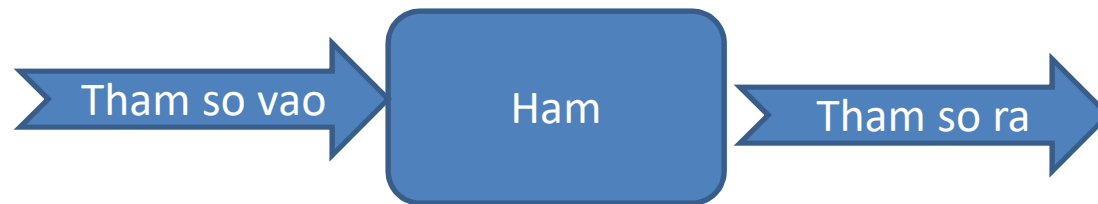
## Ví dụ 2

```
#include <stdio.h>
#include <conio.h>
void main() {
    float a, b, x;
    printf("Nhap he so a khac 0"); scanf("%f", &a);
    printf("Nhap he so b, b="); scanf("%f", &b);
    x = -b/a;
    printf("Nghiem cua phuong trinh %f", x);
    getch();
}
```



## 1.2.7. Hàm

- Mô tả:
  - Hàm (function) là một chương trình con có chức năng nhận dữ liệu đầu vào (các tham số đầu vào), thực hiện một chức năng nào đó và đưa ra các kết quả.



- 2 loại hàm:
  - Hàm có sẵn trong thư viện
  - Hàm do người lập trình định nghĩa (viết ra)

## 1.2.7. Hàm

- Ví dụ hàm có sẵn trong thư viện:
  - Thư viện `stdio.h` hàm: `scanf`, `printf`, ...
  - Thư viện `math.h` hàm: `pow`, `sin`, `cos`, `sqrt`, ...

Hàm	Ý nghĩa	Ký hiệu toán học	Ví dụ
<code>pow(x,y)</code>	x mũ y	$x^y$	<code>pow(2,3)=8</code>
<code>sin(x)</code>	Sin của x	$\sin(x)$	<code>sin(0)=0</code>
<code>cos(x)</code>	Cos của x	$\cos(x)$	<code>cos(0)=1</code>

## 1.2.8. Biểu thức

- Định nghĩa:
  - Biểu thức là sự ghép nối các toán tử (*operator*) và các toán hạng (*operand*) theo một quy tắc xác định.
  - Các toán hạng có thể là biến, hằng
  - Các toán tử rất đa dạng: cộng, trừ, nhân, chia..
- Ví dụ: Biểu thức tính biệt thức delta
$$\mathbf{b*b - 4*a*c}$$
  - a, b, c, 4 là các toán hạng
  - Các toán tử (phép toán): \*, -

## 1.2.9. Câu lệnh

- Câu lệnh (*statement*) diễn tả một hoặc một nhóm các thao tác trong giải thuật.
- Chương trình được tạo thành từ dãy các câu lệnh.
- **Cuối mỗi câu lệnh bắt buộc có dấu chấm phẩy ';' để đánh dấu kết thúc câu lệnh**

## 1.2.9. Câu lệnh

- Có 2 loại nhóm câu lệnh
  - Nhóm các câu lệnh đơn: những câu lệnh không chứa câu lệnh khác.
  - Ví dụ:
    - Lệnh gán: `delta = 100;`
    - Lệnh xuất ra màn hình: `printf("Hello World");`
  - Nhóm các câu lệnh phức: những câu lệnh chứa câu lệnh khác.
  - Ví dụ
    - Lệnh khối đặt trong cặp ngoặc nhọn `{ }`

## 1.2.10. Chú thích

- Chú thích (comment):
  - Lời mô tả, giải thích vắn tắt cho một câu lệnh, một đoạn chương trình hoặc cả chương trình
  - Giúp việc đọc và hiểu chương trình dễ dàng hơn
  - Chú thích không phải là câu lệnh -> không ảnh hưởng tới chương trình
- Cách viết chú thích: trong C có hai cách
  - Chú thích một dòng: sử dụng `//`
  - Chú thích nhiều dòng: sử dụng `/*` và `*/`

# Nội dung

1.1. Lịch sử phát triển

1.2. Các phần tử cơ bản của ngôn ngữ C

⇒ 1.3. Cấu trúc cơ bản của chương trình C

1.4. Biên dịch chương trình C

1.5. Các trình biên dịch

## Ví dụ

```
#include <stdio.h>
void main() {
    // Khai bao cac bien
    int a, b;
    int tong, hieu, tich;
    // Nhap vao tu ban phim 2 so nguyen
    printf("\n Nhap vao so nguyen thu
nhat: ");
    scanf("%d", &a);
    printf("\n Nhap vao so nguyen thu
hai: ");
    scanf("%d", &b);
```



## Ví dụ

```
// Tính tong, hieu, tich cua 2 so do
tong = a+b; hieu = a - b;tich = a*b;
// Hien thi cac gia tri ra man hinh
printf("\n Tong cua 2 so vua nhap
la %d", tong);
printf("\n Hieu cua 2 so vua nhap
la %d", hieu);
printf("\n Tich cua 2 so vua nhap
la %d", tich);
// Doi nguoi dung an phim bat ki
getch();
}
```

## 1.3. Cấu trúc cơ bản của chương trình C

- **Gồm 6 phần có thứ tự như sau:**

Phần 1: Khai báo tệp tiêu đề: `#include`

Phần 2: Định nghĩa kiểu dữ liệu mới: `typedef ...`

Phần 3: Khai báo các hàm nguyên mẫu

Phần 4: Khai báo các biến toàn cục

Phần 5: Hàm **`main()`**

Phần 6: Nội dung các hàm đã khai báo

## 1.3. Cấu trúc cơ bản của chương trình C

- **Phần 1:** Khai báo tệp tiêu đề:
  - Thông báo cho chương trình dịch biết là chương trình có sử dụng những thư viện nào.
  - VD: `#include <stdio.h> // thao tác vào ra`  
`#include <conio.h> // hàm của DOS`
- **Phần 2:** Định nghĩa các kiểu dữ liệu mới
  - Định nghĩa các kiểu dữ liệu mới (nếu cần) dùng cho cả chương trình.

## 1.3. Cấu trúc cơ bản của chương trình C

- **Phần 3:** Khai báo các hàm nguyên mẫu:
  - Giúp cho chương trình dịch biết được những thông tin cơ bản của các hàm sử dụng trong chương trình.
- **Phần 4:** Khai báo các biến toàn cục
  - Ví dụ:  
`int a, b;`  
`int tong, hieu, tich;`

## 1.3. Cấu trúc cơ bản của chương trình C

- Phần 5: Hàm **main( )**
  - Khi thực hiện, chương trình sẽ bắt đầu bằng việc thực hiện các lệnh trong hàm **main( )**.
  - Trong hàm **main( )** có thể có lệnh gọi tới các hàm khác.
- Phần 6: Nội dung của các hàm đã khai báo
  - Cài đặt (viết mã) cho các hàm đã khai báo nguyên mẫu ở phần 3.

# Nội dung

1.1. Lịch sử phát triển

1.2. Các phần tử cơ bản của ngôn ngữ C

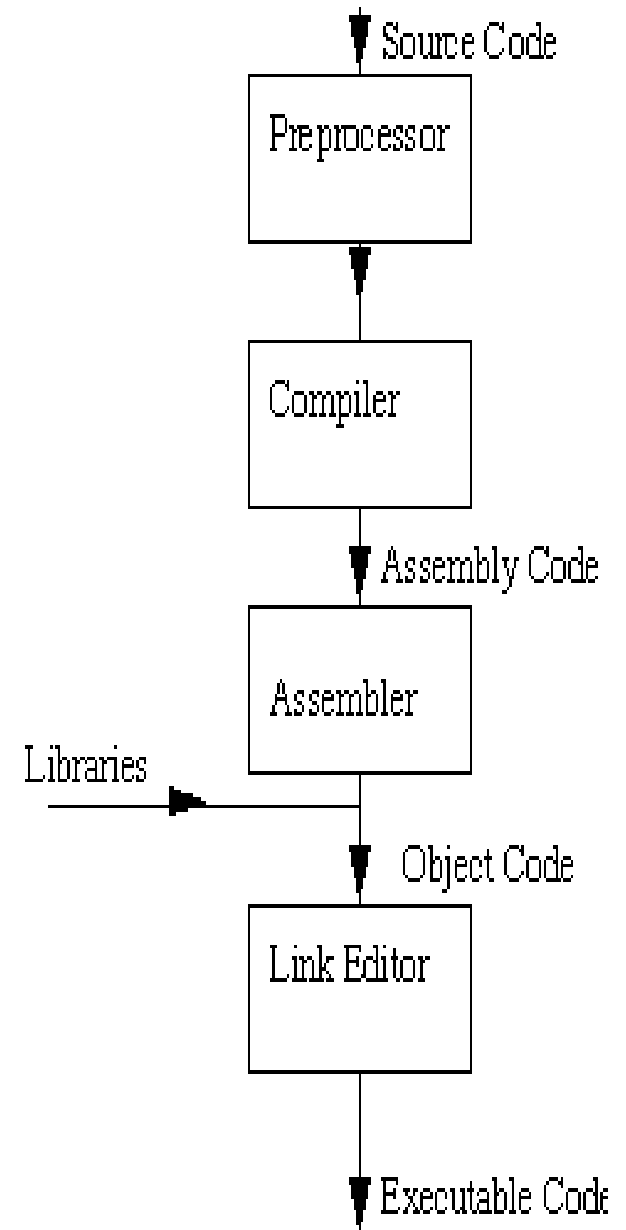
1.3. Cấu trúc cơ bản của chương trình C

⇒ 1.4. Biên dịch chương trình C

1.5. Các trình biên dịch

## 1.4. Biên dịch chương trình C

- **Preprocessor**
  - Loại bỏ các chú thích
  - Dịch các chỉ thị tiền xử lý bắt đầu là #
- **C Compiler**
  - Biên dịch mã nguồn thành mã assembly.
- **Assembler**
  - Tạo ra mã object.
    - Trên UNIX → file .o
    - Trên MS-DOS → file.OBJ
- **Link Editor**
  - Nếu tệp nguồn tham chiếu đến các hàm thư viện/hàm được định nghĩa thì *Link editor* kết hợp các hàm này với hàm `main()` để tạo ra tệp có thể thực thi được
    - Trong MS-DOS là file .exe



# Nội dung

1.1. Lịch sử phát triển

1.2. Các phần tử cơ bản của ngôn ngữ C

1.3. Cấu trúc cơ bản của chương trình C

1.4. Biên dịch chương trình C



1.5. Các trình biên dịch

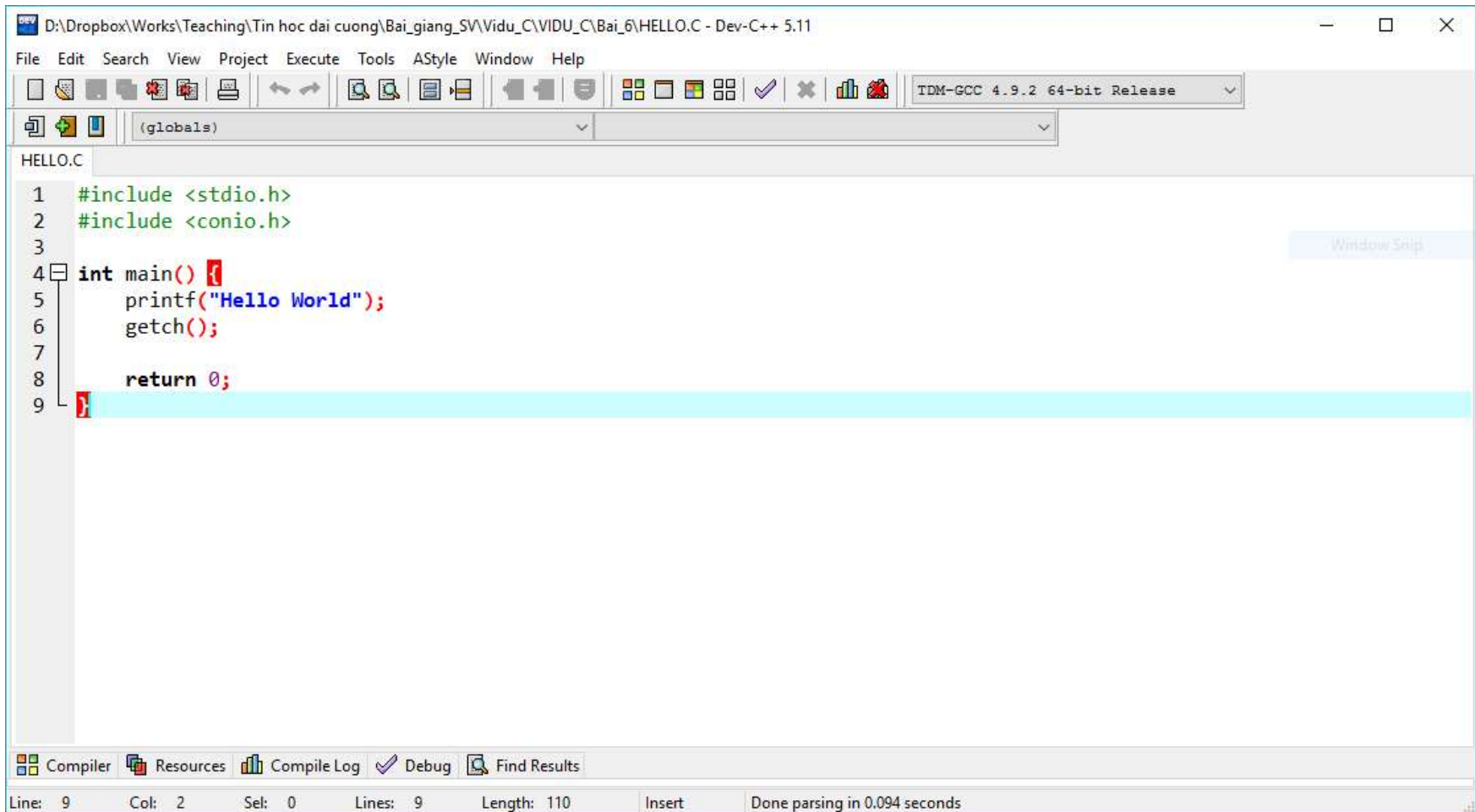


## 1.5.1. Giới thiệu

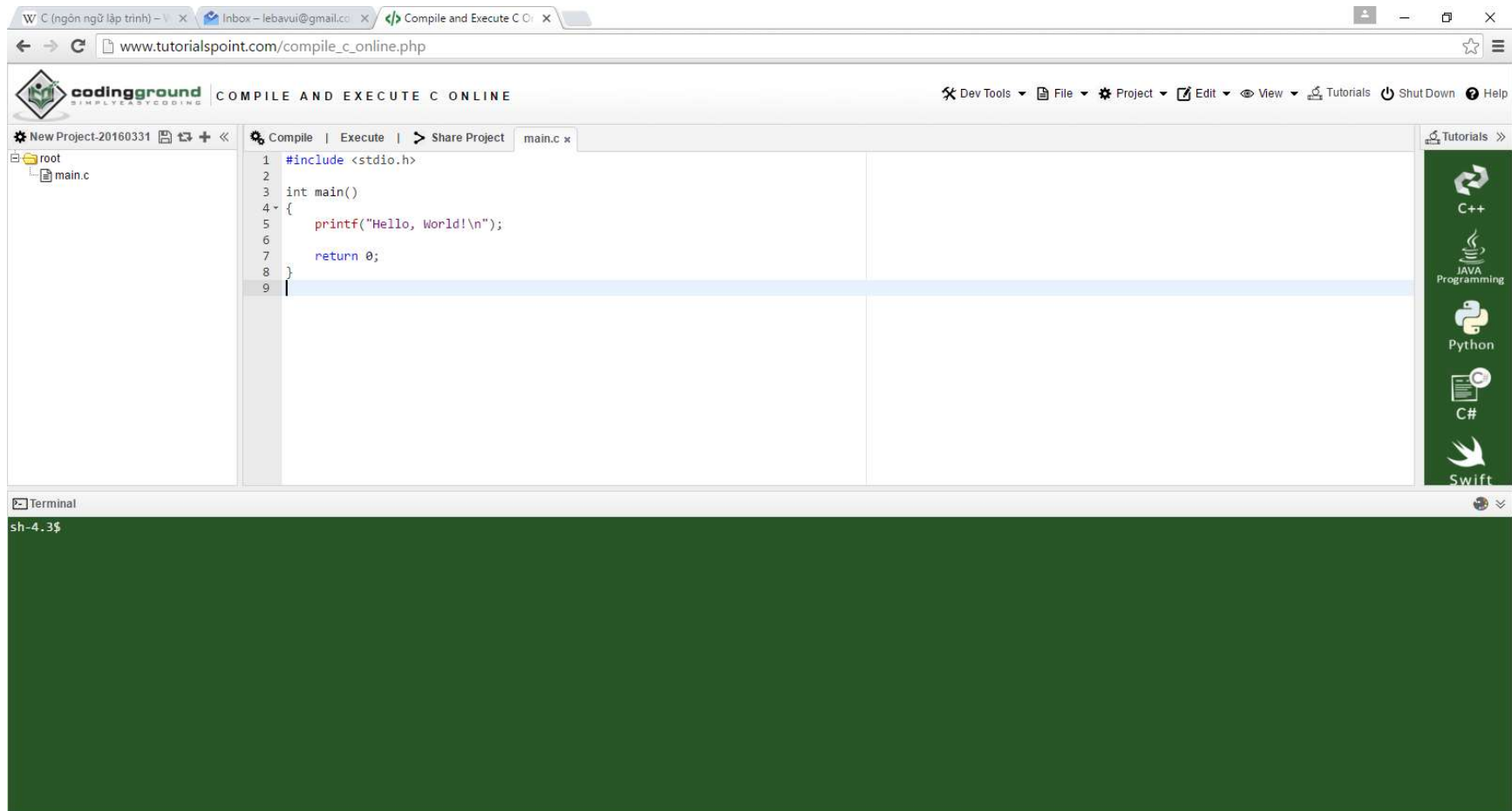
- Trình biên dịch (compiler): dịch mã nguồn (source code) thành file thực thi
- Các trình biên dịch C phổ biến
  - Turbo C++ của hãng Borland
  - VisualStudio của Microsoft
  - GCC của GNU
  - **Dev-C++ của Bloodshed Software**

## 1.5.2. Cài đặt và sử dụng Dev-C++

- Giao diện sử dụng của chương trình



# Chương trình soạn thảo và dịch C online



[http://www.tutorialspoint.com/compile\\_c\\_online.php](http://www.tutorialspoint.com/compile_c_online.php)