

01 - 初识计算机与程序设计

C++ 程序设计基础

SOJ 信息学竞赛教练组

2024 年 5 月 30 日

目录

1 计算机的组成

2 程序设计及编程语言的发展

3 编写第一个程序

4 输出语句

5 总结

你见过的计算机长什么样？



计算机由哪些部分组成？



计算机由哪些部分组成？

主机



计算机由哪些部分组成？

主机



显示器（输出设备）



计算机由哪些部分组成？

主机



显示器（输出设备）



音箱（输出设备）



计算机由哪些部分组成？

主机



显示器（输出设备）



音箱（输出设备）



鼠标（输入设备）

计算机由哪些部分组成？

主机



显示器（输出设备）



音箱（输出设备）



键盘（输入设备）



鼠标（输入设备）

主机里有什么？



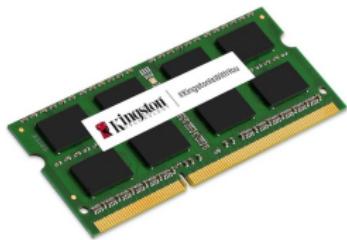
主机里有什么？



主机里有什么？



主机里有什么？



主机里有什么？



计算机的组成部件

- 输入设备：键盘、鼠标、触摸屏等

计算机的组成部件

- 输入设备：键盘、鼠标、触摸屏等
- 输出设备：显示器、打印机、音箱等

计算机的组成部件

- 输入设备：键盘、鼠标、触摸屏等
- 输出设备：显示器、打印机、音箱等
- 中央处理器（即 CPU），含有运算器和控制器

计算机的组成部件

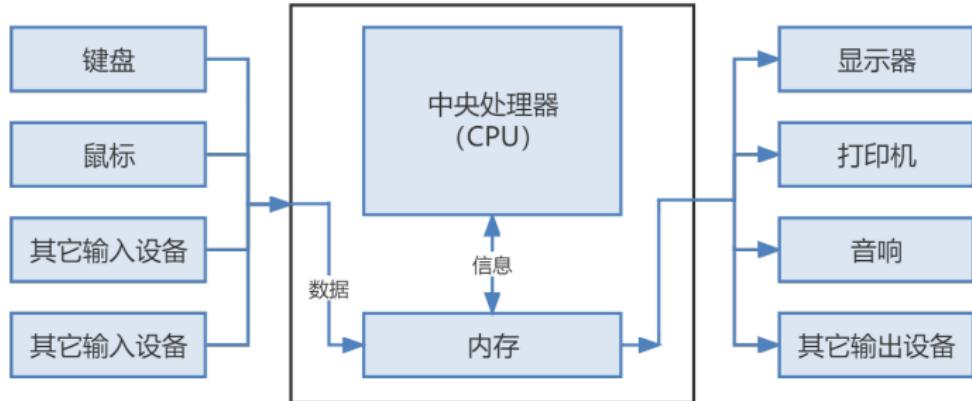
- 输入设备：键盘、鼠标、触摸屏等
- 输出设备：显示器、打印机、音箱等
- 中央处理器（即 CPU），含有运算器和控制器
- 存储器：内存、硬盘等

计算机的组成部件

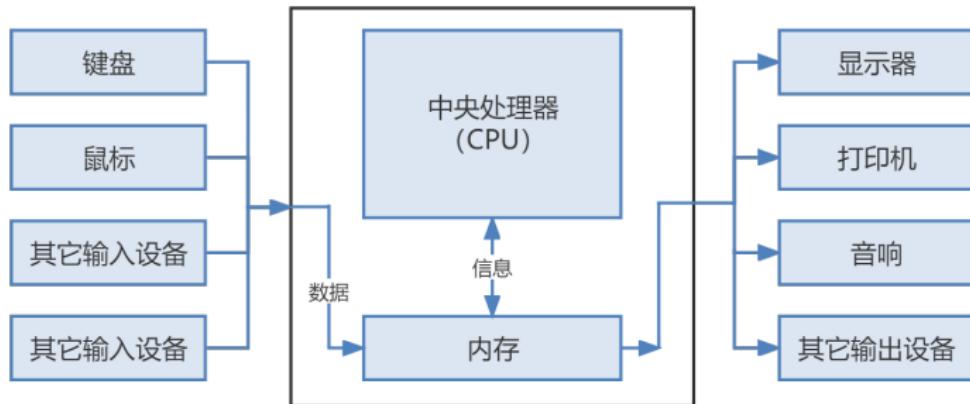
- 输入设备：键盘、鼠标、触摸屏等
- 输出设备：显示器、打印机、音箱等
- 中央处理器（即 CPU），含有运算器和控制器
- 存储器：内存、硬盘等
- 其他设备（可选）：独立显卡、网卡等

你用计算机做过什么事情？

计算机如何工作

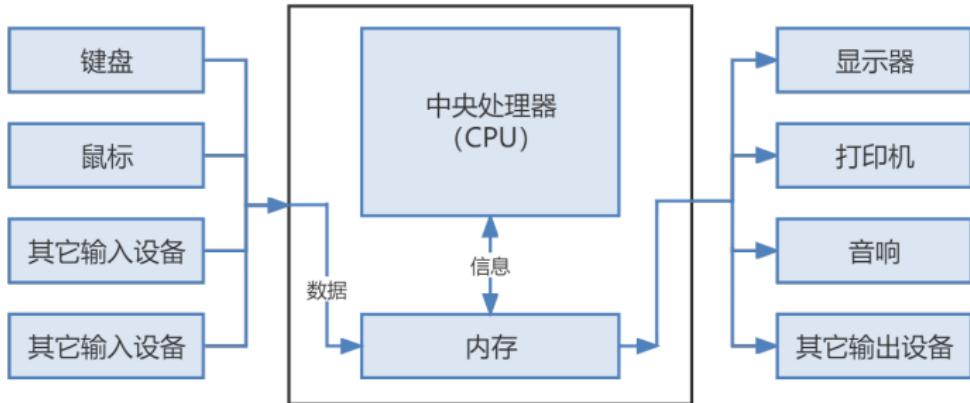


计算机如何工作



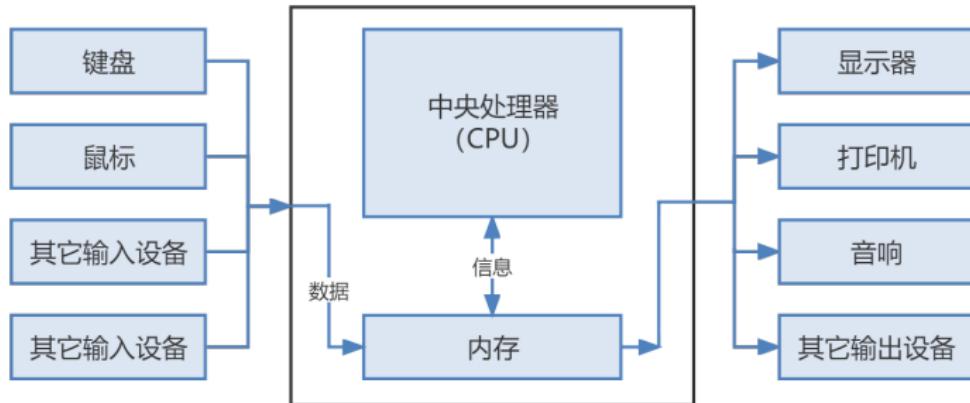
程序从输入设备中获得用户的输入信息，然后对信息进行处理，再把处理结果通过输出设备展示给用户。

计算机如何工作



处理过程是 CPU 和内存一起完成的：CPU 执行“处理动作”，而内存中既存放了用户输入数据，也存放了告诉 CPU 如何处理数据的“说明书”——程序。

计算机如何工作



计算机之所以可以完成不同的功能，正是因为我们给它编写了不同的程序。

目录

1 计算机的组成

2 程序设计及编程语言的发展

3 编写第一个程序

4 输出语句

5 总结

什么是程序

- 程序是给计算机的指令，我们通过程序告诉计算机做什么和怎么做

什么是程序

- 程序是给计算机的指令，我们通过程序告诉计算机做什么和怎么做
- 计算机不懂得人类的语言（自然语言），所以你需要用计算机语言（编程语言）来和计算机交流

什么是程序

- 程序是给计算机的指令，我们通过程序告诉计算机做什么和怎么做
- 计算机不懂得人类的语言（自然语言），所以你需要用计算机语言（编程语言）来和计算机交流
- 因此，程序需要使用编程语言来编写

程序设计语言的发展

- 机器语言 (Machine Language)
 - 机器语言是每台计算机的 CPU 指令集合

110110101001101

程序设计语言的发展

- 机器语言 (Machine Language) 110110101001101
 - 机器语言是每台计算机的 CPU 指令集合
- 汇编语言 (Assembly Language) ADD R1, R2, R3
 - 汇编语言用符号 (助记符) 来代替 0/1 码, 运行时通过汇编器转换为机器语言

程序设计语言的发展

- 机器语言 (Machine Language) 110110101001101
 - 机器语言是每台计算机的 CPU 指令集合
- 汇编语言 (Assembly Language) ADD R1, R2, R3
 - 汇编语言用符号 (助记符) 来代替 0/1 码, 运行时通过汇编器转换为机器语言
- 高级语言 (High-level Languages) C = A + B
 - 高级语言用自然语言符号来代替 0/1 码, 运行时通过编译器转换为机器语言
 - 高级语言更接近我们日常使用的自然语言

C++ 语言的特点

- C/C++ 的应用范围广
 - 许多设备的驱动程序和操作系统的都是用 C/C++ 写的
 - 编程开发方面的工作中 C/C++ 仍然是热门语言

C++ 语言的特点

- C/C++ 的应用范围广
 - 许多设备的驱动程序和操作系统的都是用 C/C++ 写的
 - 编程开发方面的工作中 C/C++ 仍然是热门语言
- C/C++ 的程序运行效率更快

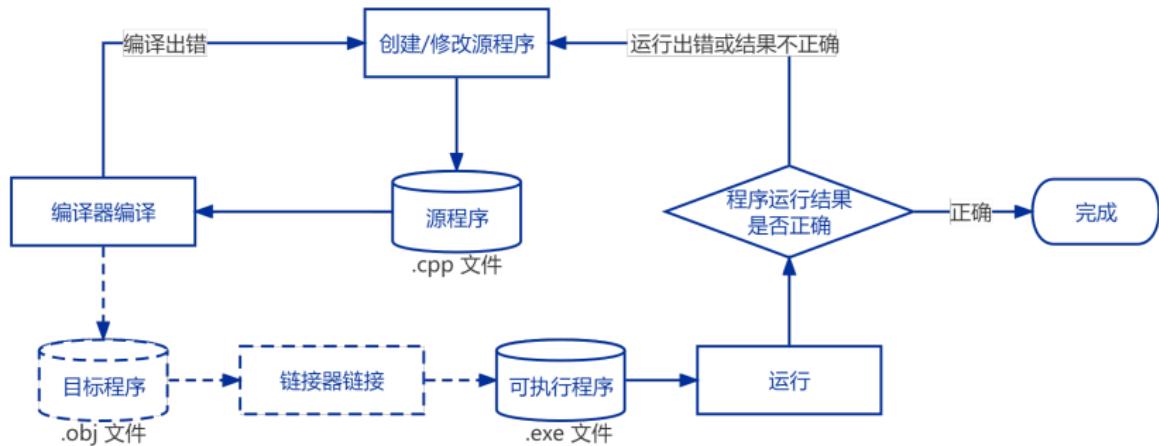
C++ 语言的特点

- C/C++ 的应用范围广
 - 许多设备的驱动程序和操作系统的都是用 C/C++ 写的
 - 编程开发方面的工作中 C/C++ 仍然是热门语言
- C/C++ 的程序运行效率更快
- C/C++ 是高级语言中最接近低级语言的，因而如果想更深入了解计算机是如何运作的，C/C++ 是首选

C++ 语言的特点

- C/C++ 的应用范围广
 - 许多设备的驱动程序和操作系统的都是用 C/C++ 写的
 - 编程开发方面的工作中 C/C++ 仍然是热门语言
- C/C++ 的程序运行效率更快
- C/C++ 是高级语言中最接近低级语言的，因而如果想更深入了解计算机是如何运作的，C/C++ 是首选
- 熟练掌握 C/C++ 语言，在学习其他高级编程语言时更加轻松

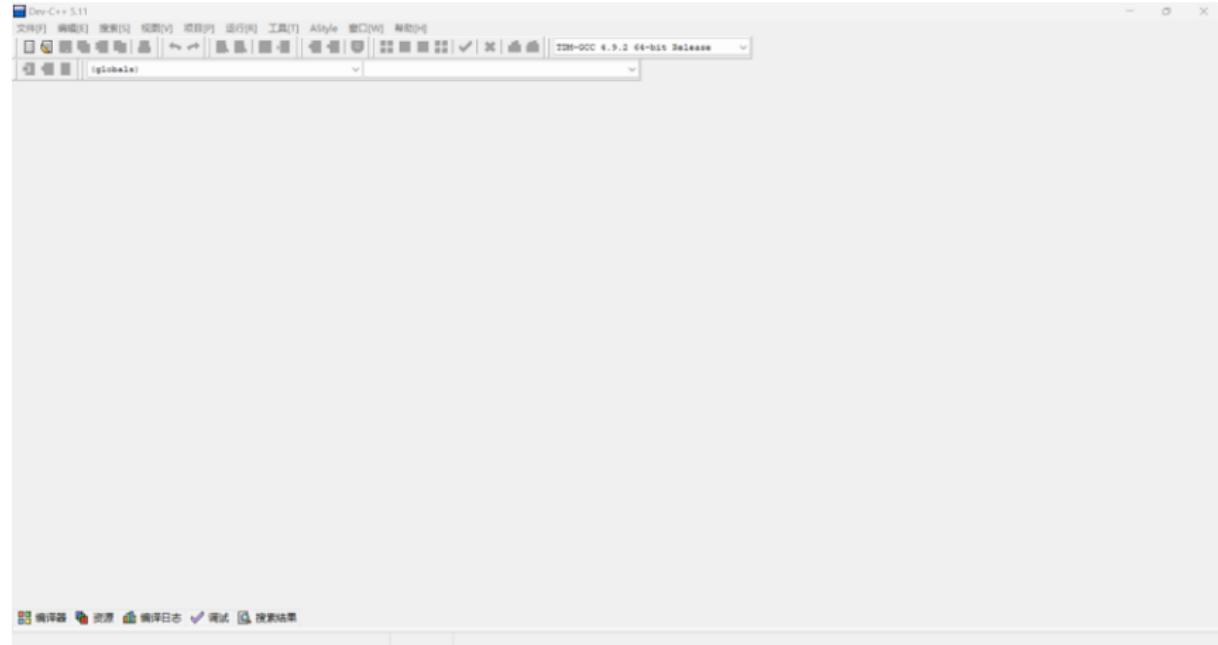
C++ 编译运行的流程



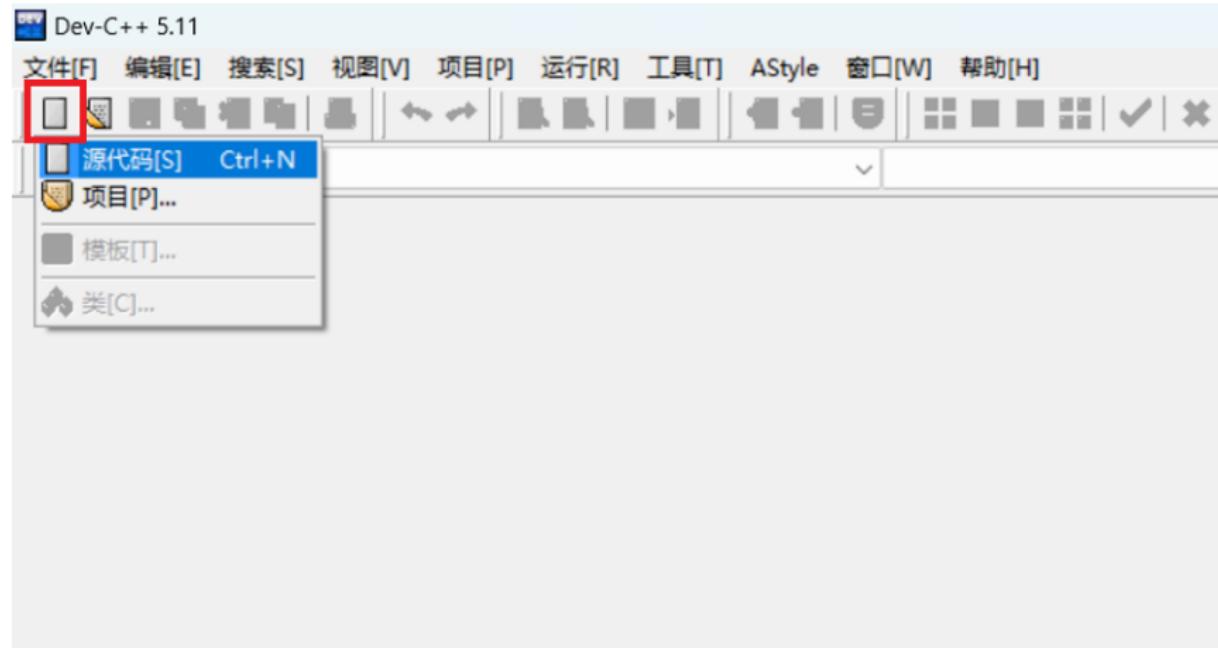
目录

- 1 计算机的组成**
- 2 程序设计及编程语言的发展**
- 3 编写第一个程序**
- 4 输出语句**
- 5 总结**

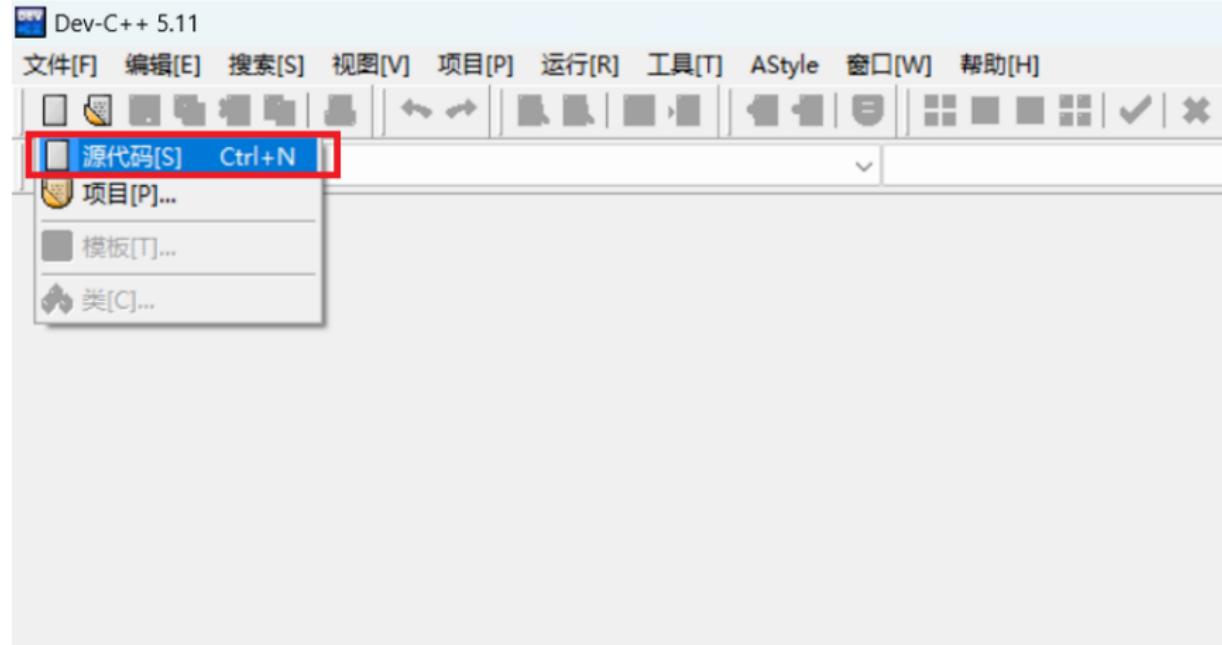
IDE 的使用



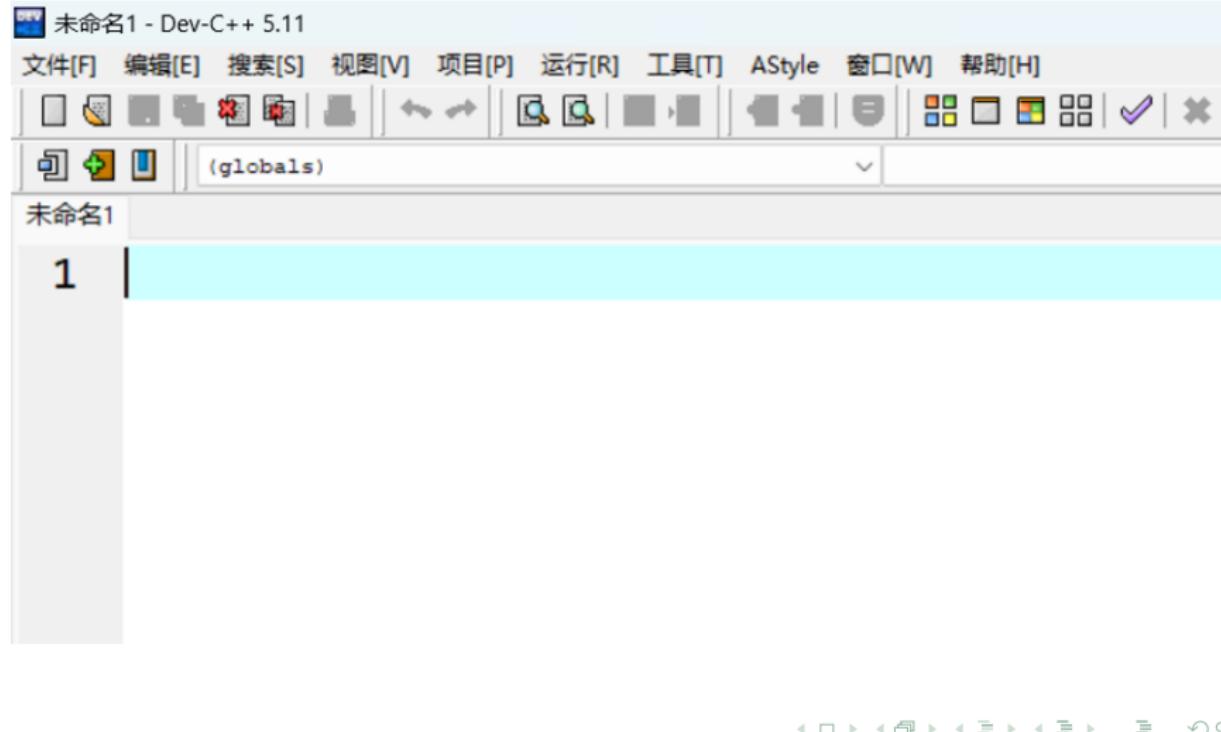
IDE 的使用



IDE 的使用



IDE 的使用



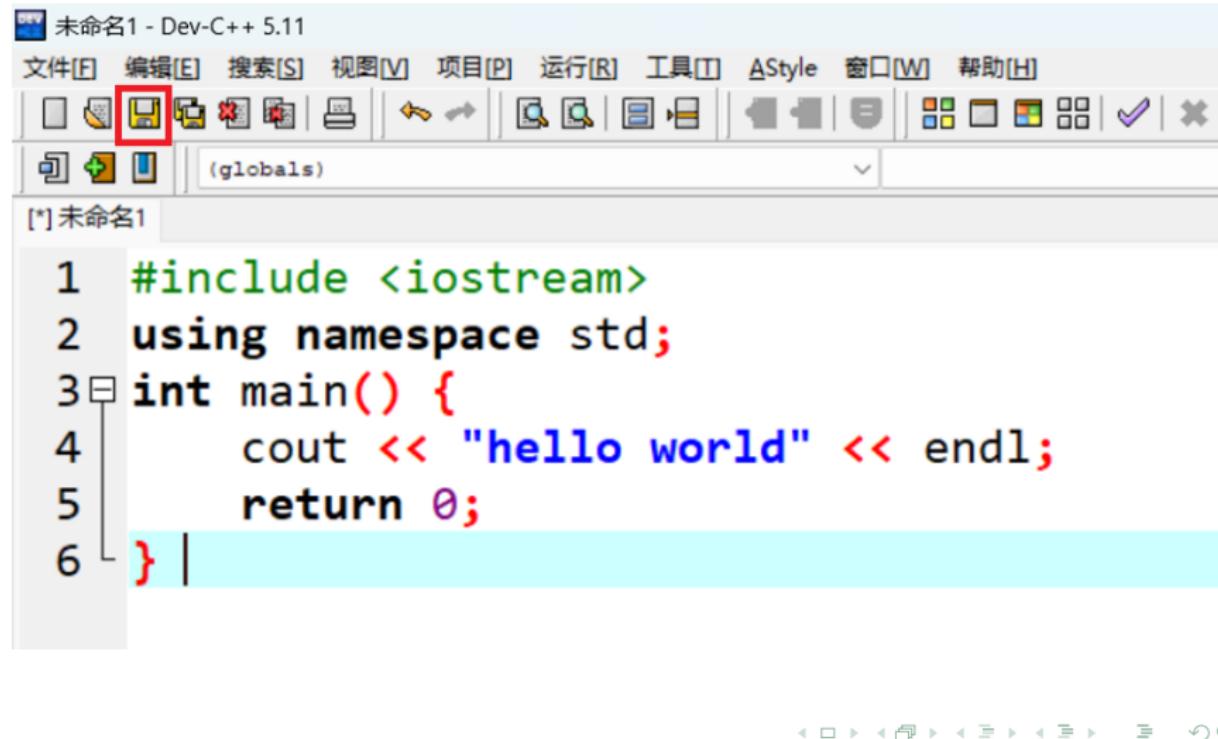
IDE 的使用

The screenshot shows the Dev-C++ 5.11 IDE interface. The menu bar includes: 文件 [F] | 编辑 [E] | 搜索 [S] | 视图 [V] | 项目 [P] | 运行 [R] | 工具 [T] | AStyle | 窗口 [W] | 帮助 [H]. The toolbar contains various icons for file operations like Open, Save, and Build. Below the toolbar is a toolbar for global search and navigation. The code editor window displays the following C++ code:

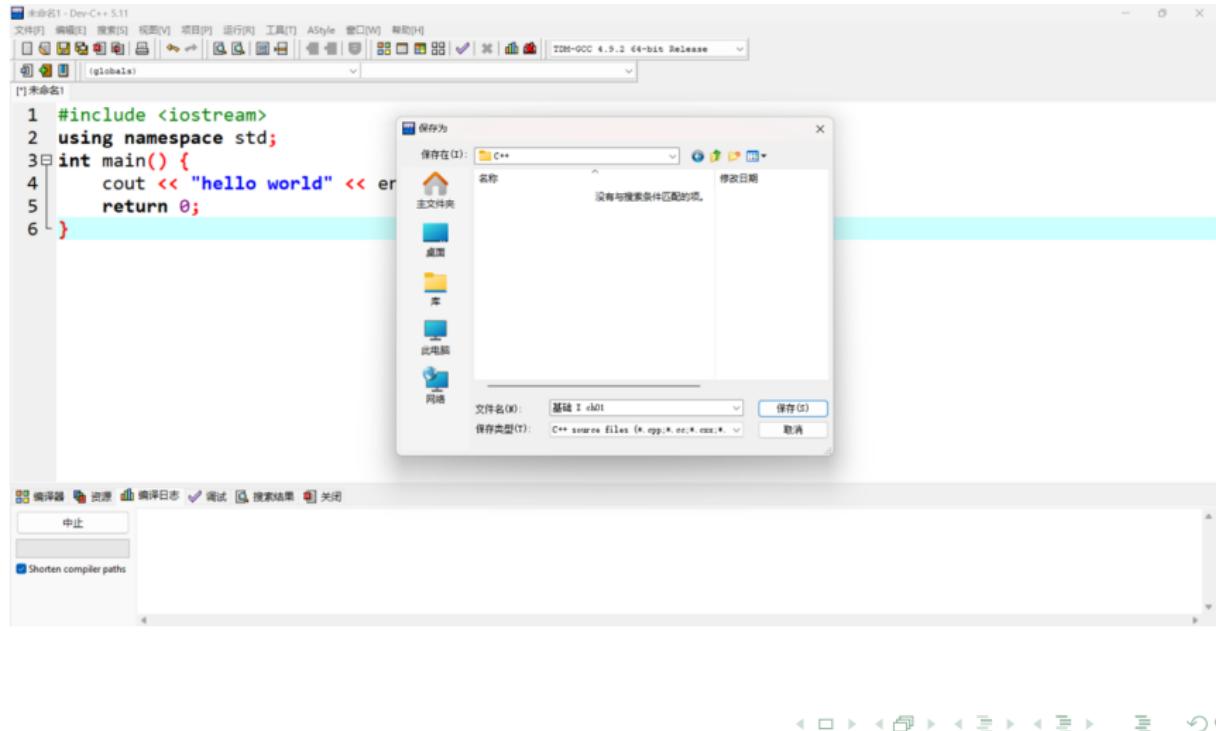
```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     cout << "hello world" << endl;
5     return 0;
6 }
```

The code editor has syntax highlighting: green for comments, black for keywords, red for strings, and blue for operators. Line 6, which contains the closing brace '}', is highlighted with a light blue background. The status bar at the bottom shows navigation icons.

IDE 的使用



IDE 的使用



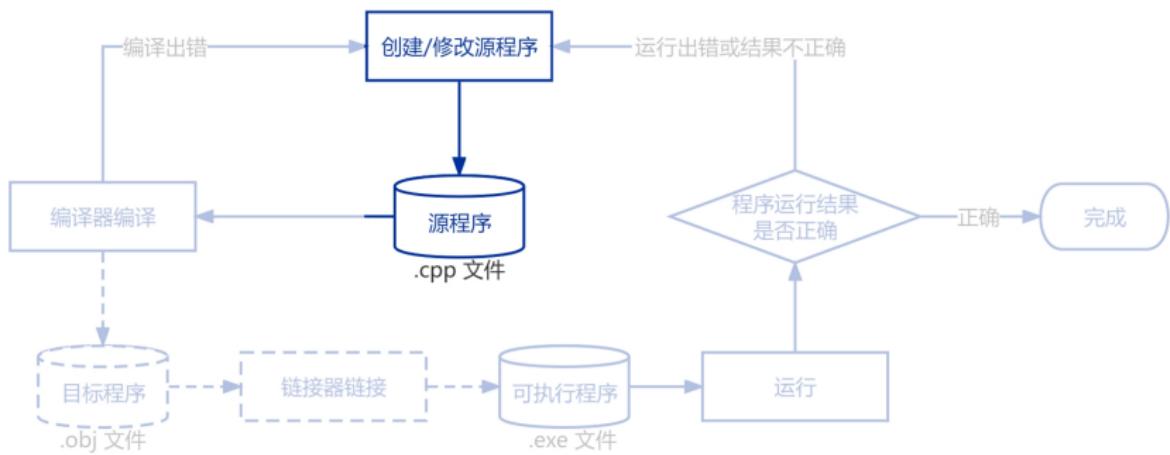
IDE 的使用

The screenshot shows the Dev-C++ IDE interface. The title bar indicates the file path: C:\Users\X\\Desktop\C++\基础\ch01.cpp - Dev-C++ 5.11. The menu bar includes: 文件(F) 编辑(E) 搜索(S) 视图(V) 项目(P) 运行(R) 工具(T) AStyle 窗口(W) 帮助(H). The toolbar contains various icons for file operations like Open, Save, and Build. The status bar at the bottom right shows the compiler: TDM-GCC 4.9.2 64-bit Release. The code editor displays the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     cout << "hello world" << endl;
5     return 0;
6 }
```

The code editor has a light gray background with syntax highlighting. The first line '#include <iostream>' is green. The keyword 'using' is red. The 'int main()' block and its contents are blue. The brace at line 6 is red. The status bar at the bottom right shows navigation icons for back, forward, search, and other functions.

IDE 的使用



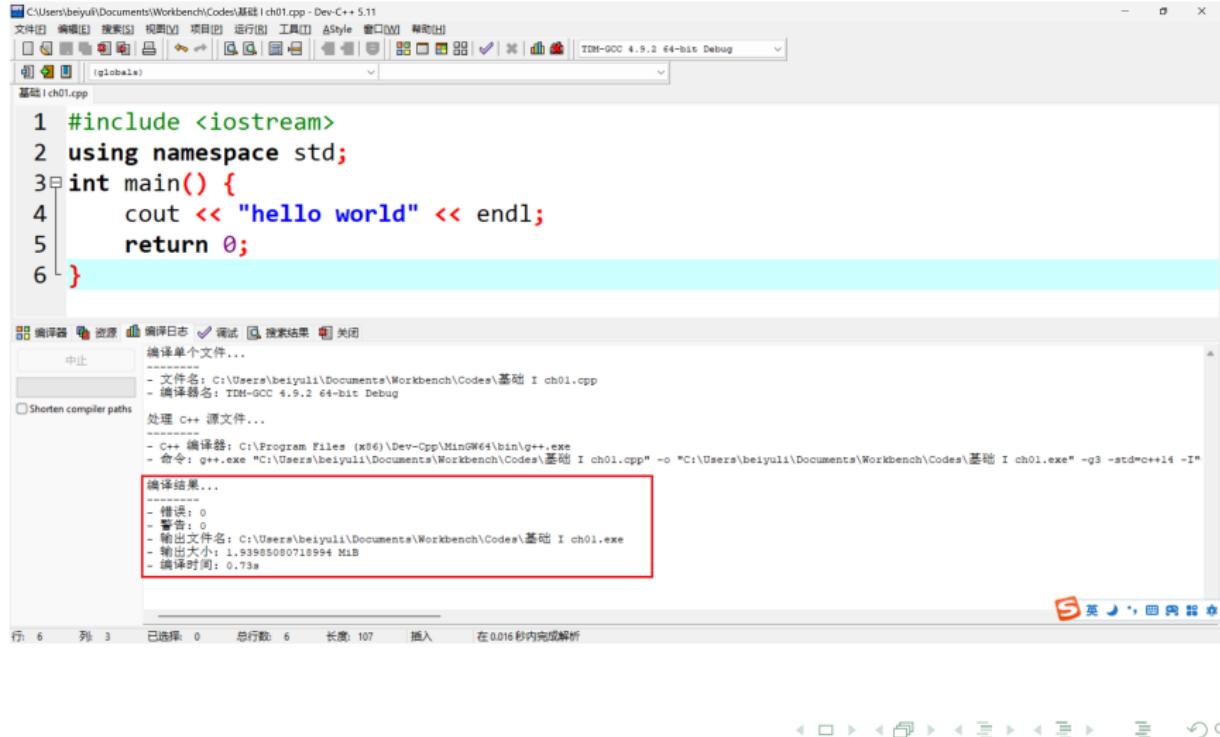
IDE 的使用

The screenshot shows the Dev-C++ IDE interface. The title bar displays the path: C:\Users\X\\Desktop\C++\基础\ch01.cpp - Dev-C++ 5.11. The menu bar includes: 文件(F), 编辑(E), 搜索(S), 视图(V), 项目(P), 运行(R), 工具(T), AStyle, 窗口(W), 帮助(H). The toolbar has various icons, with the fourth one from the left highlighted by a red box. The status bar at the bottom right shows: TDM-GCC 4.9.2 64-bit Release. The code editor window contains the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     cout << "hello world" << endl;
5     return 0;
6 }
```

The code editor has a light gray background with syntax highlighting. The status bar at the bottom right shows navigation icons.

IDE 的使用



IDE 的使用

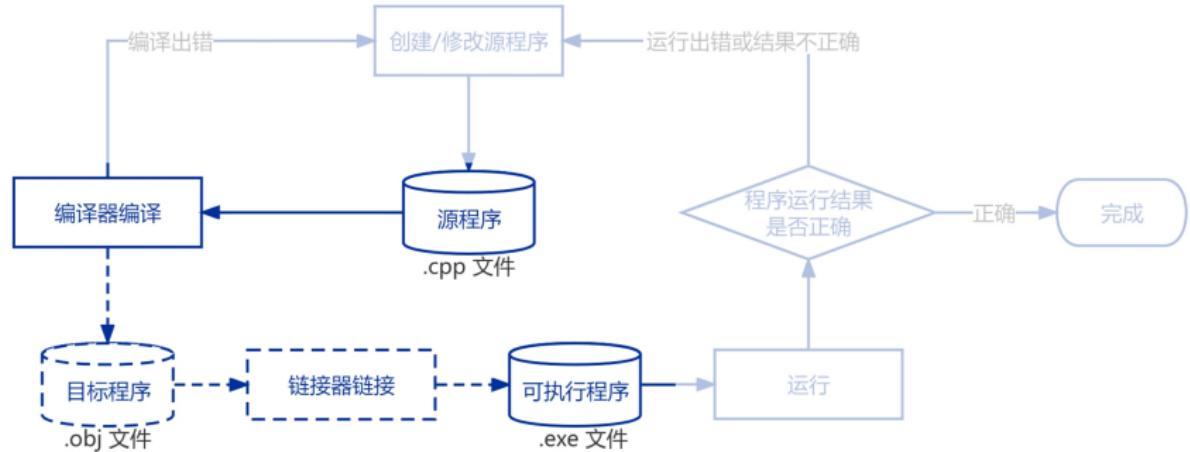


基础 | ch01.cpp C++ 代码文件



基础 | ch01.exe 可执行文件

IDE 的使用



IDE 的使用

The screenshot shows the Dev-C++ IDE interface. The title bar displays the path: C:\Users\X\\Desktop\C++\基础\ch01.cpp - Dev-C++ 5.11. The menu bar includes: 文件(F), 编辑(E), 搜索(S), 视图(V), 项目(P), 运行(R), 工具(T), AStyle, 窗口(W), 帮助(H). The toolbar has various icons, with the build icon (a square with a circle) highlighted by a red box. The status bar at the bottom right shows: TDM-GCC 4.9.2 64-bit Release. The code editor window contains the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     cout << "hello world" << endl;
5     return 0;
6 }
```

The code editor has a light gray background with syntax highlighting. The status bar at the bottom shows navigation icons.

IDE 的使用

The screenshot shows the Dev-C++ IDE interface. On the left, the code editor displays a file named ch01.cpp containing a simple "Hello World" program:

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     cout << "hello world" << endl;
5     return 0;
6 }
```

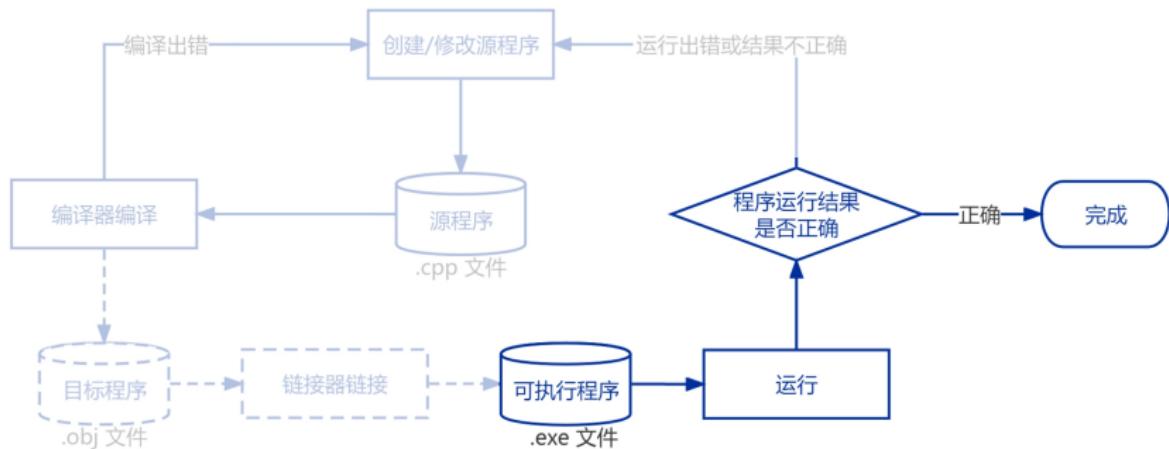
On the right, the terminal window shows the execution results:

```
C:\Users\��\Desktop\C++\基础 I ch01.exe
hello world
-----
Process exited after 0.4849 seconds with
return value 0
请按任意键继续. . .
```

At the bottom left, the status bar shows:

行 6 列 3 已选中 0 总行数 6 长度: 106 插入 在 0.406 秒内完成解析

IDE 的使用



IDE 的使用

- 快捷键
 - Ctrl + A 全选
 - Ctrl + C 复制
 - Ctrl + V 粘贴

程序的基本结构

```
1 #include <iostream>      // 头文件
2
3 using namespace std;    // 命名空间
4
5 // 程序的主体
6 int main() {           // 程序的入口：主函数
7     // 程序的内容
8
9     return 0;            // 程序的出口
10 }
```

程序的基本结构

```
1 #include <iostream>      // 头文件
2
3 using namespace std;    // 命名空间
4
5 // 程序的主体
6 int main() {           // 程序的入口：主函数
7     cout << "Hello, world!" << endl;
8
9     return 0;            // 程序的出口
10 }
```

目录

1 计算机的组成

2 程序设计及编程语言的发展

3 编写第一个程序

4 输出语句

5 总结

输出语句

- 使用 cout 语句，输出特定内容

输出语句

- 使用 cout 语句，输出特定内容
- 输出文本
 - 文本需要置于双引号 "" 中
 - `cout << "Hello"; // 输出文本信息 Hello`

输出语句

- 使用 cout 语句，输出特定内容
- 输出文本
 - 文本需要置于双引号 "" 中
 - `cout << "Hello"; // 输出文本信息 Hello`
- 输出多组文本
 - `cout << "Hello" << "C++"; // 输出 HelloC++`
 - `cout << "Hello" << " " << "C++"; // 输出 Hello C++`
 - 注意，空格也是文本

输出数值

- 输出数值

- `cout << 100; // 输出数值 100, 不需要双引号`

输出数值

- 输出数值

- `cout << 100; // 输出数值 100, 不需要双引号`

- 输出多个数值

- `cout << 100 << 200; // 输出 100200`
 - `cout << 100 << " " << 200; // 输出 100 200`

如何实现输出两行的效果？

- 下面的代码可以分开两行输出 Hello World 吗？
- ```
cout << "Hello";
cout << "World";
```

# 换行符

- C++ 的换行符是 endl

# 换行符

- C++ 的换行符是 endl
- cout << endl; 会输出一个换行符，使后面的输出从下一行开始

# 换行符

- C++ 的换行符是 endl
- cout << endl; 会输出一个换行符，使后面的输出从下一行开始
- cout << "Hello";  
cout << "World";

# 换行符

- C++ 的换行符是 endl
- cout << endl; 会输出一个换行符，使后面的输出从下一行开始
- ~~cout << "Hello";  
cout << "World";~~
- cout << "Hello" << endl;  
cout << "World" << endl;

# 格式控制

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6 cout << 3.14159 << endl;
7
8 return 0;
9 }
```

# 格式控制

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6 cout << 3.14159 << endl;
7
8 return 0;
9 }
```

- 输出结果中有几位小数?
- 假如想要保留 2 位小数呢?

# 保留小数

- 添加头文件 #include <iomanip>

# 保留小数

- 添加头文件 `#include <iomanip>`
- `cout << fixed << setprecision(2) << X.XXX << endl;`

# 保留小数

- 添加头文件 `#include <iomanip>`
- `cout << fixed << setprecision(2) << X.XXX << endl;`
- 四舍五入保留 2 位小数

# 格式控制

```
1 #include <iostream>
2 #include <iomanip>
3
4 using namespace std;
5
6 int main() {
7 cout << fixed << setprecision(2) << 3.14159 << endl;
8
9 return 0;
10 }
```

# 格式控制

```
1 #include <iostream>
2 #include <iomanip>
3
4 using namespace std;
5
6 int main() {
7 cout << fixed << setprecision(2) << 3.14159 << endl;
8
9 return 0;
10 }
```

- 输出结果：3.14

# 格式控制

```
1 #include <iostream>
2 #include <iomanip>
3
4 using namespace std;
5
6 int main() {
7 cout << fixed << setprecision(4) << 3.14159 << endl;
8
9 return 0;
10 }
```

# 格式控制

```
1 #include <iostream>
2 #include <iomanip>
3
4 using namespace std;
5
6 int main() {
7 cout << fixed << setprecision(4) << 3.14159 << endl;
8
9 return 0;
10 }
```

- 输出结果：3.1416

# 目录

- 1 计算机的组成**
- 2 程序设计及编程语言的发展**
- 3 编写第一个程序**
- 4 输出语句**
- 5 总结**

# 总结

- 计算机的工作原理
- 程序语言的发展
- C++ 的编译运行
- 输出语句

# Thank you!