NLP 南京大学 自然语言处理研究组 Natural Language Processing Group

南京大学 NANJING UNIVERSITY

初识程序

黄书剑

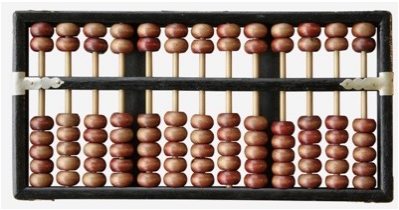
- 计算机和程序
- C和C++语言
- 第一个C/C++程序
- 程序的构成
- 词法规则
- 句法规则（初步）
- 编译器的语法检查

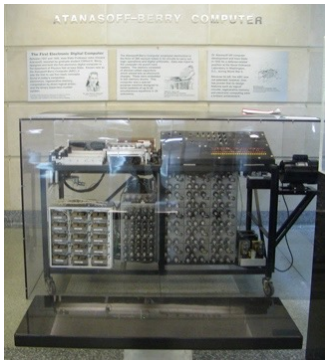
南京大学
NANJING UNIVERSITY

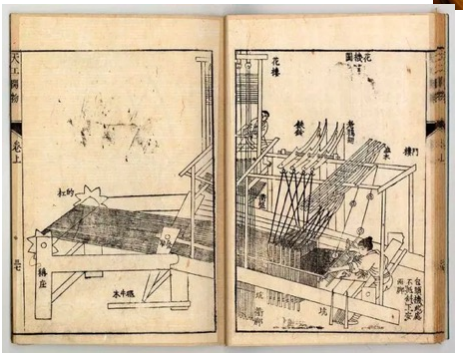
计算机和程序

计算设备

- **存储设备 -» 执行设备　　机械设备 -» 电子设备**







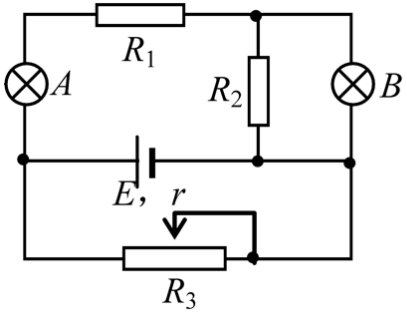
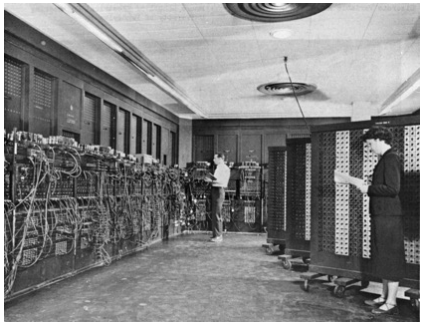


图2. 天工开物中记录的织布机 (17世纪) 图片来自知乎
图3. 巴贝奇Charles Babbage的差分机 (18世纪) 图片来自维基
图4-5. Atanasoff-Berry Computer 第一台电子计算机 及 其中的加减法单元

计算机的工作模型

- **1946年出现第一台通用电子计算机ENIAC** (Electronic Numerical Integrator And Computer)



- **从ENIAC开始，计算机元器件的发展带来了计算能力的提升**
  - 电子管、晶体管、集成电路、超大规模集成电路

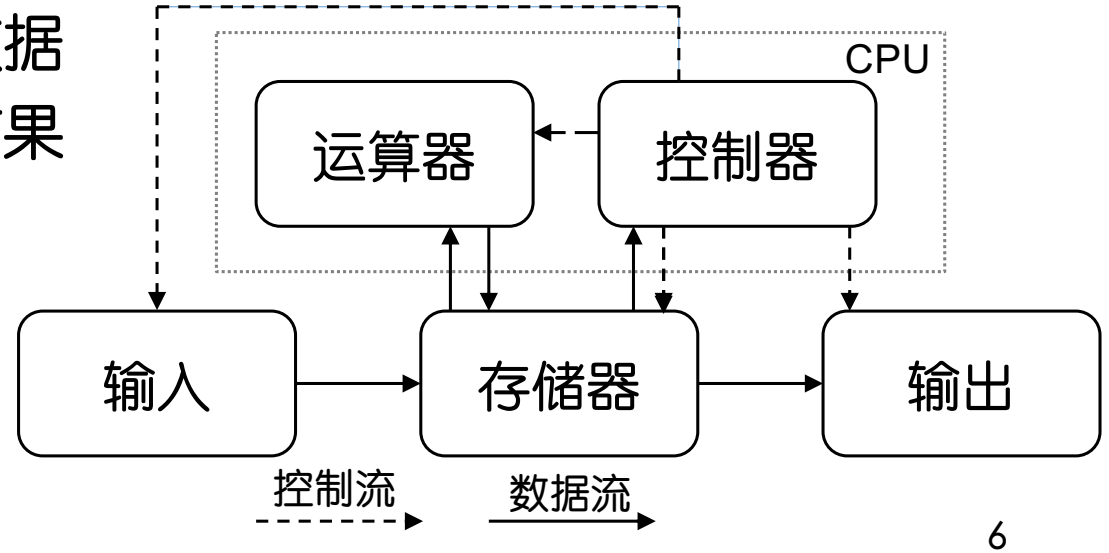
芯片

- **尽管计算机已经得到飞速发展，但目前大部分计算机仍然采用冯诺依曼体系结构。**

图1-2. ENIAC (1946)及其背后的部分电子管 图片来自维基

冯·诺依曼体系结构（Von Neumann）

- **冯•诺依曼提出计算机应由5个单元构成：**
  - 存储单元：存储程序（指令序列）和数据
  - 运算单元：进行算术/逻辑运算
  - 控制单元：控制程序的执行流程和根据指令向其它单元发出控制信号
  - 输入单元：从外界获得数据
  - 输出单元：向外界输出结果



程序作为特殊的数据存储在存储器中
从而将硬件和软件的分离开来

计算机软件和硬件

- **一台计算机由硬件和软件两方面构成：**
  - 硬件：计算机的物理构成，即构成计算机的元器件和设备。
    - 中央处理器（CPU，Central Processing Unit）
    - 内存（memory）
    - 外围设备
      - 外存（external storage，例如硬盘、U盘、光盘等）
      - 输入设备（例如键盘、鼠标等）
      - 输出设备（例如显示器、打印机等）

计算机软件和硬件

- **一台计算机由硬件和软件两方面构成：**
  - 硬件：计算机的物理构成，即构成计算机的元器件和设备。
  - 软件：计算机程序以及相关的文档。
    - 系统软件（例如Windows、Unix、Linux、Mac OS等）
    - 支撑软件（例如集成开发环境、软件测试工具等）
    - 应用软件（例如财务软件、自动控制软件等）

计算机软件和硬件

- **一台计算机由硬件和软件两方面构成：**
  - 硬件：计算机的物理构成，即构成计算机的元器件和设备。
  - 软件：计算机程序以及相关的文档。

- **一台计算机的性能主要由硬件决定，而它的功能则主要是由软件来提供。**

程序（program）

- **程序是一组连续的相互关联的计算机指令：**

  CPU能执行的指令包括：

  算术运算指令：实现加、减、乘、除等；

  比较指令：比较两个操作数的大小；

  数据传输指令：实现CPU的寄存器、内存以及外设之间的数据传输；

  流程控制指令：用于确定下一条指令的内存地址

  顺序、转移、循环、子程序调用/返回

- **程序是指示计算机处理某项计算任务的任务书，计算机根据该任务书，执行一系列操作，并产生有效的结果。**
  - **计算（compute）：并非单指数值计算，而是指根据已知数据（data）经过一定的步骤（算法）获得结果的过程**

程序 = 算法 + 数据（结构）

程序设计语言（programming language）

- **机器语言**
  - 适配硬件设备的计算机语言，只有0、1两种符号
- **自然语言**
  - 人类日常交流沟通使用的语言，一般不能用自然语言直接进行程序设计（programming）。
- **程序设计语言**
  - 人类设计的具有特定形式和语法规则的语言
- **利用程序设计语言设计、编写的程序（源程序），通过相应的翻译、优化工具（编译器），可以形成计算机能够理解的机器语言程序（目标程序），从而可以被计算机执行（可执行文件）**

It is no exaggeration to regard this as the most fundamental idea in programming:

> The evaluator, which determines the meaning of expressions in a programming language, is just another program.

To appreciate this point is to change our images of ourselves as programmers. We come to see ourselves as designers of languages, rather than only users of languages designed by others.

—— SICP Ch 4 Metalinguistic Abstraction

- **(C/C++语言的）编译器是用什么语言实现的?**

南京大学
NANJING UNIVERSITY

C/C++语言

C/C++语言的来历

- ALGOL 60 (algorithmic language，国际委员会，1960)
  - 简洁、科学的定义
- CPL (combined programming language，剑桥、伦敦大学，1963)
  - 接近硬件、规模大
- BCPL (basic ~，剑桥大学Matin Richards，1967)
  - 简化
- B (贝尔实验室Ken Thompson，1970)
  - 精华
- C (贝尔实验室Dennis M. Ritchie，1972~1973)
  - 既保持了BCPL和B语言的优点(精练、高效、接近硬件等)
  - 又克服了它们的缺点(过于简单、数据无类型、功能有限等)
- C++ (贝尔实验室Bjarne Stroustrup，1979)
  - 为支持面向对象程序设计而设计(先是C with Class)

C 语言标准（specification）

- 1978年Dennis M. Ritchie与Brian W. Kernighan出版了《The C Programming Language》，此书是最初的C语言标准（K&R C）
- 随着C语言的应用和发展，形成了多种C语言的实现版本，各种版本在功能和函数库的设置内容上存在差别。
- 1983年，ANSI开始制定统一的C语言标准，直至1989 年底正式批准名为ANSI X3.159-1989的标准（C89），1990年，ISO采纳了C89并以ISO/IEC 9899:1990颁布
- 2000年初，ISO颁布了ISO/IEC 9899:1999（C99）
- 2011年底，ISO发布了ISO/IEC 9899:2011 （C11）
- C17, C23...

American National Standards Institute

International Organization for Standardization

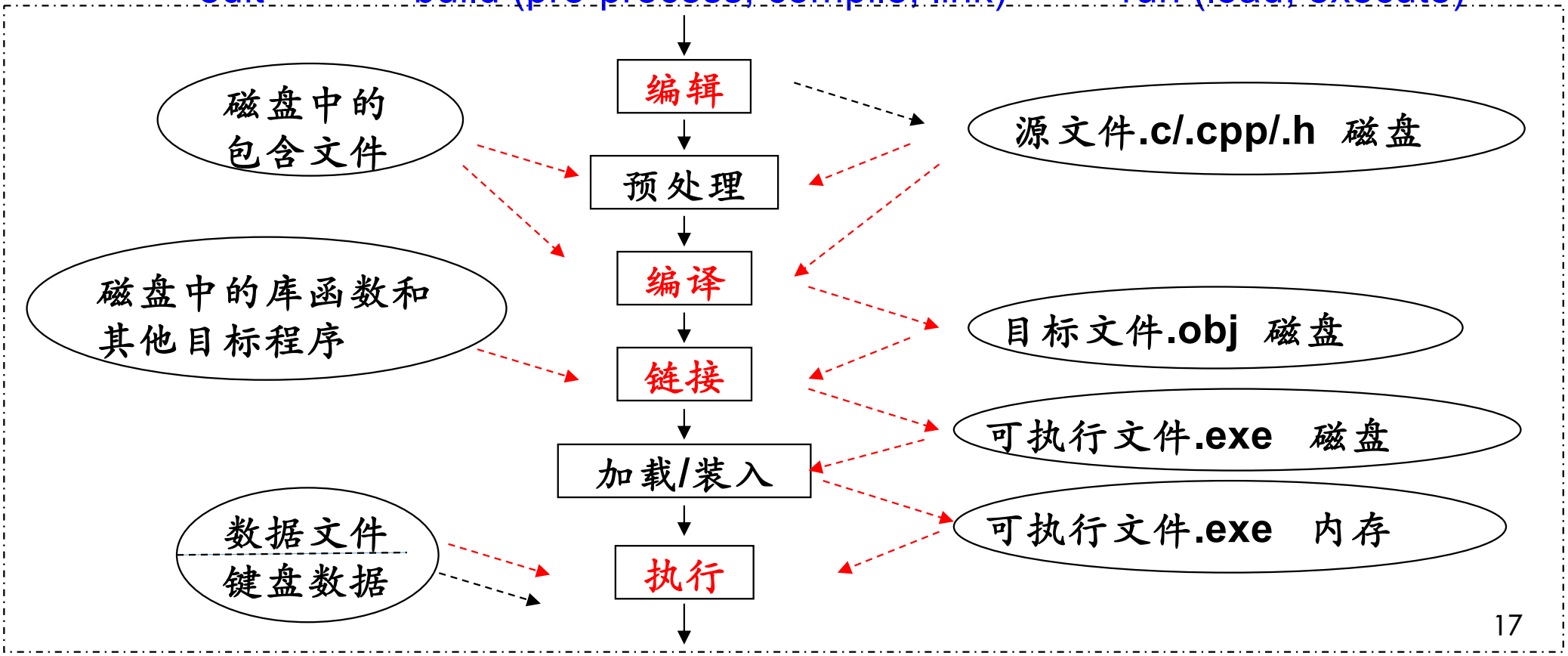
C++ 语言标准（specification）

- 1979年Bjarne Stroustrup在Simula影响下开始设计C with Classes
- 1983年C with Classes 更名为 C++
- 1985年 《C++ Programming Language》 著作面世
- 1990年 《Annotated C++ Reference Manual》 、Turbo C++
- 1998年 ISO发布了第一个C++标准 ISO/IEC 14882:1998 （C++98）
- 2003年发布C++03；
- 2005年计划TR1(C++0x)，并于2011年正式发布(C++11)
- C++14, C++17, C++20, C++23, C++26 ...

International Organization for Standardization

C/C++程序的运行步骤

- **编辑 → 生成（预处理、编译、链接） → 执行(加载、运行)**
  edit build (pre-process, compile, link) run (load, execute)



开发环境

- **上述C/C++程序的编辑、编译等步骤都需要特定软件的支撑**
  - VC++编译器、GCC编译器等
  - 单独执行某个步骤可以得到对应的结果
- **使用带有编辑程序、编译程序等支撑软件的集成开发环境（IDE）进行程序开发，可以提升开发效率**
  - Visual Studio C++、Dev-C++、Eclipse CDT、Code Block等
  - 在这些集成环境中，往往比较方便就能完成所有的步骤。
  - *程序开发的几个步骤在不同开发环境中的安排也不尽相同
    - Visual Studio C++中的Build菜单包含编译和链接步骤，执行步骤一般在Debug菜单中
    - Dev-C++中，编译、链接和执行步骤则安排在Execute菜单下

开发环境

- *不同的IDE所含的编译程序及其对标准的支持程度不尽相同
  - 初学者应该以掌握程序设计的基本方法为目标
  - 尽量避免被语言标准及各个开发环境实现的细节所纠缠
  - 有些问题和概念在计算机系统原理、编译技术、程序设计语言概论等后续课程中讨论更为合适。

程序的开发和调试

- **程序必须符合程序设计语言的规范**
  - 一般由编译器负责检查，编译器会对错误予以报告
- **部分程序可能在其他步骤发生错误，因而不能正常编译、链接或执行**
  - 相应负责的程序（链接程序等）一般也会进行错误报告
- **程序中还可能包含逻辑错误，造成程序运行结果与预期不一致**
  - 往往需要更复杂的方法才能发现
  - IDE通常含有辅助调试（debug）程序的软件
  - 修正逻辑错误时往往需要修改程序，整个过程可能会重复多次，直到得出正确的执行结果。

南京大学
NANJING UNIVERSITY

第一个C/C++程序

Hello World!

- 书写Hello World程序：

```c
#include <stdio.h>


int main()
{
   printf("Hello World! \n");
   return 0;
}
```

C 版本

```cpp
#include <iostream>
using namespace std;

int main()
{
   cout<<"Hello World! \n";
   return 0;
}
```

C++ 版本

main函数：

- 每一个程序都应该有的内容（灰色内容可以省略）：

```
int main(void)
{


   return 0;
}
```

main函数与程序执行

- **main函数是C/C++系统约定的程序入口**
  - 一个程序中必须定义一个名字为的main函数
  - 一个程序中只能有一个main函数

- **执行程序时，总是从main函数体中的第一条语句开始执行，当执行到main函数中的return语句时，整个程序结束**
  - 编译、运行上述程序，将依次执行main中的语句，并得到预期的运行结果。

- Tips:
  - 对于简单的程序，可以仅包含main函数。程序功能都在main中实现。
  - 对于复杂程序，可以使用多个函数协作完成，将在后续予以介绍。

理解main函数

- 每一个程序都应该有的内容（灰色内容可以省略）：

```
int main(void)
{


   return 0;
}
```

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

main函数的更多说明

- **main是一个函数，遵从函数的形式约定（后续将详细介绍）**
  - 返回值类型：int是约定的main函数返回值的类型，表示main函数执行完毕返回给执行环境的信号是一个整数，与return后面的0一致
  - 形参列表：函数名后面必须有一对圆括号，括号内为形参列表。void（空类型）表示该main函数不带参数，void可以省略
  - 函数体：接下来必须是一对花括号，花括号里是函数体
  - 返回值的返回：一般约定正常程序段的末尾写“return 0;”，异常处理程序段（实际应用中往往要编写专门的异常处理程序段）的末尾写“return -1;”。执行环境根据接收到的-1或0，决定要不要采取故障恢复等措施

“空” 的main函数

- 每一个程序都应该有的内容（灰色内容可以省略）：

```
int main(void)
{


   return 0;
}
```

仅有上述内容也是一个完整的程序。该程序的main函数中没有任何实质性的计算，因此是一个可执行的“空”程序。

南京大学
NANJING UNIVERSITY

程序的构成

一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                    //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);        //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

符号（变量、值）、语句、注释、输入输出

从读程序开始！

西班牙语是西班牙的官方语言，也是拉丁美洲大多数国家的官方语言，同时还是联合国的工作语言之一。下面展示的是 10 个西班牙语语句以及它们的英语译文，请据此按要求回答问题。

西班牙语	英语
Garcia y una pistola.	Garcia and a pistol.
Carlos Garcia tiene tres asociados.	Carlos Garcia has three associates.
Sus asociados no son fuertes.	His associates are not strong.
Garcia tambien tiene empresas en Europa.	Garcia has companies in Europe too.
Sus clientes están enfadados.	His clients are angry.
Los asociados tambien están enfadados.	The associates are also angry.
Los clients y los asociados son enemigos.	The clients and the associates are enemies.
Sus empresa no venden pistola.	His company do not sell pistol.
Los grupos pequeños no son modernos.	The small groups are not modern.
Los asociados están sorprendidos.	The associates are surprised.

问题一 请将下面（1）中的西班牙语句子翻译成英语，并按要求回答（2）中的问题。

（1）Los asociados enfadados y Garcia no son enemigos.

（2）简单而言，“语序”指的就是不同成分在语句中的排列顺序。请回答，从前面这句话中可看出西班牙语和英语的语序有哪两点不同？

问题二 请将下面的英语句子翻译成西班牙语：

（3）Three small clients sell pistols in Europe too.

问题三 请把下面（4）、（5）两个英语句子翻译成西班牙语（已知 hungry 和 poor 在翻译出来的西班牙语句子中都是 hambrientos），并按要求回答（6）中的问题。

（4）The associates are hungry.

程序的组成

- **逻辑上，一个程序由一些程序实体的定义构成，这些程序实体主要包括：**
  - 常量：不变的数据
  - 变量：可变的数据
  - 数据类型：用于对数据的特征进行描述
  - 表达式：对数据的加工过程
  - 函数：对数据的加工过程（子程序）

程序 = 算法 + 数据结构

程序的语法和语义

- **一个语言包括语法、语义和语用**
  - 语法：是指构作结构正确的语言成分所需遵循的规则集合。
  - 语义：是指语言各个成分的含义。
  - 语用：是指语言成分的使用场合及所产生的实际效果。（初学者最不容易掌握）
- **语法用于描述程序的语义（算法+数据）**
  - 特定的语法规则描述了这个语言的表现形式
  - 严格限制的语法规则是程序语言区别于自然语言的重要特点
- **语法一般包括词法和句法：**
  - 词法是指语言的构词规则
  - 句法是指由词构成句子（程序）的规则

南京大学
NANJING UNIVERSITY

词法规则

词法规则

- **词法描述语言的基本单元的组成规律，包括：**
  - 基本符号（字符集）
  - 单词的构成规则
- **字符集：构成语言的基本符号**
- **单词：由字符集中的字符按照一定规则构成的具有一定意义的最小语法单位**

C/C++的词法规则-字符集

- **C/C++的字符集由下列符号构成：**
  - 大小写英文字母：a～z, A～Z
  - 数字：0～9
  - 特殊字符：

    ! # % ^ & * _ - + = ～ < > / \ | . , : ; ?

    ‘ “ ( ) [ ] { } 空格 横向制表 纵向制表 换页 换行

×	√	π	`	@	$	“ ”	,	‘ ’	;

键盘上没有的符号

键盘上有的符号

汉字库里的符号

C/C++的词法规则-单词构成

- **C/C++的单词有：**
  - 关键词
  - 标识符
  - 字面常量（直接量）
  - 操作符（运算符）
  - 标点符号

关键字（keyword）

- **系统保留的单词，有固定的作用和含义**
  - 通常由小写字母组成，在程序中不能用作其他目的
  - 表示语句：
    - break、continue、do、else、for、goto、if、return、switch、while…
  - 表示数据类型：
    - auto、char、const、double、enum、float、int、long、register、short、signed、struct、union、unsigned、void…
  - 表示标号：
    - case、default…
  - 其他关键字：
    - extern、sizeof、static、typedef…

教材表1-3中给出了C++中关键词的详细列表

回顾：一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                      //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

红色标识的为C语言中的关键字

标识符（identifier）

- **用于命名程序中的各种实体**
  - 变量、函数、形式参数、类型、符号常量、以及语句标号等
  - 包括系统预定义标识符和用户自定义标识符两部分
- **由字符集中的大小写英文字母、阿拉伯数字和下划线组成，且首字符是字母或下划线**

i、price_car、myFun、Node_3、Loop4、PI、pi

5x、stu.score、number 1、y-average	×

回顾：一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                      //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

红色标识的为标识符（包括预定义和自定义两种）

标识符的使用说明

- **关键词不能作为用户自定义的标识符！**
- **使用预定义标识符可能会引发潜在的问题!**

- **此外，还需要注意标识符的命名风格（Tips）**
  - 尽量不与其他标识符同名。
  - 尽量不在同一个程序中定义拼写相同、大小写不同的标识符。
    - 标识符中的大写字母与小写字母是有区别的，比如PI和pi是两个不同的标识符。
  - 尽量使用不太长的有意义的单词
    - 标识符的有效长度由具体编译器决定。

字面常量和操作符

- **字面常量（literal）**
  - 字面常量 在程序中直接书写的常量值，又称直接量（literal）。如：128、3.14、'A'、"abcd"等 。
- **操作符（operator，运算符）**
  - 操作符 由系统规定，用于表示基本运算。如：+，-，*，/，=， >，<，==，!=，>=，<=，||，&&等。

一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                      //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

红色标识的为字面常量，绿色标识的为操作符

标点符号（punctuation）

- **标点符号 起到某些语法、语义上的作用。**
  - 如：逗号、分号、冒号、括号等。
- **续行符**
  - 由一个反斜杠（\）后面紧跟一个换行符（↙）构成
- **空白符**
  - C++程序是由单词序列构成，没严格的格式要求。
    - 单词之间一般需要用空白符分开，以保证形式上独立。
      - 如，double␣r;　（“␣”代表空白符）
    - 单词和操作符、标点符号之间可以不加空白符
      - 适当加上空格可以提升代码可读性
  - 空白符包括：空格符、制表符、回车符、注释符

一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                    //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);        //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

符号（变量、值）、语句、注释、输入输出

√

南京大學
NANJING UNIVERSITY

句法规则（初步）

回顾：程序的语法和语义

- **一个语言包括语法、语义和语用**
  - 语法：是指构作结构正确的语言成分所需遵循的规则集合。
  - 语义：是指语言各个成分的含义。
  - 语用：是指语言成分的使用场合及所产生的实际效果。（初学者最不容易掌握）
- **语法用于描述程序的语义（算法+数据）**
  - 特定的语法规则描述了这个语言的表现形式
  - 严格限制的语法规则是程序语言区别于自然语言的重要特点
- **语法一般包括词法和句法：**
  - 词法是指语言的构词规则
  - 句法是指由词构成句子（程序）的规则

操作符与表达式

- **操作符用于描述程序中的操作**
  - 字面常量、符号常量、变量与函数可以作为操作符的基本操作对象，又叫操作数（operand）

- **表达式(expression)：用操作符将操作数连接起来的式子**
  - 最简单的表达式是单个操作数，如：字面常量1，变量d
  - 表达式可以是单个操作符连接操作数，如： `d + 1`
  - 表达式也可以作为操作数参加运算，如：`d = d + 1`

recursion 递归

语句（statement）

- **表达式末尾加一个分号可以构成语句**
  - 如：　　`d = d + 1;`
  - 最简单的语句是一个分号，即空语句，不执行任何操作
  - 一行可以包含多个语句，语句跨行需要使用续行符
- **可以用一对花括号将多个语句括起来，形成复合语句，一个复合语句可以看作一个语句块**

```
{
 sum = sum + PI * d;
 d = d + 1;
}
```

- **一些关键字，如while、return等，与表达式、分号或（子）语句配合也可以构成语句（后续详细介绍）**

注释（comment）

- **注释 是为了方便对程序的理解而加在源程序中的说明性文字信息。**
- **注释不属于可执行程序的一部分，但对程序维护非常重要！**
- **注释分为：**
  - 多行注释：以符号“/*”开始到符号“*/”结束。
  - 单行注释（C++扩充）：从符号“//”开始到本行结束。

一个略微复杂的例子

- 例：计算一个正方形的周长（计量单位为米）。

```
#include <stdio.h>

int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                      //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

符号（变量、值）、语句、注释、输入输出

√ √ √

语法的形式描述

- 有时需要对一个语言的语法进行精确的描述，这往往需要采用另一个相对简单、没有歧义的语言（称为元语言）来完成。
- 较常用的用于描述语言语法的元语言是一种称为BNF（Backus-Naur Form）的形式语言。
  - 例如，C/C++标识符的构成规则可用BNF描述成：

```
<标识符> ::= <非数字字符>|<标识符><非数字字符>|<标识符><数字字符>
<非数字字符> ::=
        _|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|
        a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
<数字字符> ::= 0|1|2|3|4|5|6|7|8|9
```

由字符集中的大小写英文字母、阿拉伯数字和下划线组成，且首字符是字母或下划线

- **在BNF中，“<”、“>”、“|”和“::=”称为元语言符号，它们不属于被描述的语言。**
  - “::=”表示“定义为”；“|”表示“或者”；“<标识符>”、“<非数字>”以及“<数字>”称为元语言变量，它们代表被描述语言中的语法实体。
  - 另外，BNF也存在一些扩充形式，例如，方括号“[]”用于表示其中的内容可有可无；花括号“{}”用于表示其中的内容可以重复出现多次，等等。
  - 当元语言与被描述的语言有相同的符号时，应采用某种方式把它们区分开来。
- **本课程采用简化了的BNF形式加上自然语言来对部分语法进行描述**

南京大學
NANJING UNIVERSITY

编译器的语法检查

回顾：

It is no exaggeration to regard this as the most fundamental idea in programming:

> The evaluator, which determines the meaning of expressions in a programming language, is just another program.

To appreciate this point is to change our images of ourselves as programmers. We come to see ourselves as designers of languages, rather than only users of languages designed by others.

—— SICP Ch 4 Metalinguistic Abstraction

- 编译器也是程序！
- 按照语法解析程序代码，并转换为机器可以执行的形式

如何能做到呢?

程序 = 算法 + 数据

编译错误

- **编译器尝试按照语法规则解析代码，在无法解析时返回错误**

```
#include <stdio.h>
int mian( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                    //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

```
/usr/bin/g++ -g /Users/helloworld/test.cpp -o /Users/helloworld/test
Undefined symbols for architecture x86_64:
  "_main", referenced from:
     implicit entry/start for main executable
ld: symbol(s) not found for architecture x86_64
clang: error: linker command failed with exit code 1 (use -v to see invocation)
```

编译错误

- **编译器尝试按照语法规则解析代码，在无法解析时返回错误**

```
#include <stdio.h>
int main( ){
    intdi = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                      //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                   //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                //输出计量单位
    return 0;
}
```

```
/usr/bin/g++ -g /Users/helloworld/test.cpp -o /Users/helloworld/test
/Users/helloworld/test.cpp:5:5: error: use of undeclared identifier 'intdi'
    intdi = 1;                  //d是一个整型变量，表示直径，初始值是1
     ^
/Users/helloworld/test.cpp:8:37: error: use of undeclared identifier 'di'
    printf("The perimeter is: %d", 4 * di);      //输出计算结果
                                        ^
2 errors generated.
```

编译错误

- **编译器尝试按照语法规则解析代码，在无法解析时返回错误**

```
#include <stdio.h>
int main( ){
    int di = 1                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                      //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                   //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                //输出计量单位
    return 0;
}
```

```
/usr/bin/g++ -g /Users/helloworld/test.cpp -o /Users/helloworld/test
/Users/helloworld/test.cpp:5:14: error: expected ';' at end of declaration
      int di = 1                    //di是一个整型变量，表示直径，初始值是1
                ^
                ;
1 error generated.
```

编译错误

- **编译器尝试按照语法规则解析代码，在无法解析时返回错误**

```
#include <stdio.h>
int main( ){
    int di = 1;                          //di是一个整型变量，表示直径，初始值是1
    char mu = 'm';                       //mu是一个字符型变量，表示单位的英文符号
    scanf("%d", &di);                    //从控制台输入整数di的值，%d表示整型
    printf("The perimeter is: %d", 4 * di);          //输出计算结果
    printf("%c \n", mu);                 //输出计量单位
    return 0;
}
```

```
/usr/bin/g++ -g /Users/helloworld/test.cpp -o /Users/helloworld/test
/Users/helloworld/test.cpp:5:14: error: non-ASCII characters are not allowed
outside of literals and identifiers
      int di = 1;                  //d是一个整型变量，表示直径，初始值是1
                ^~
/Users/helloworld/test.cpp:5:14: error: expected ';' at end of declaration
      int di = 1;                  //d是一个整型变量，表示直径，初始值是1
2 errors generated.
```

编译错误

- **编译器尝试按照语法规则解析代码，在无法解析时返回错误**
  - 编译器可能会协助定位到具体错误位置
  - 一个语法错误可能引发多处编译错误，造成定位困难

- **养成仔细阅读编译器提示的习惯，根据提示推测可能错误原因！**
- **Tips：**
  - 如果需要执行的代码较多（二十行以上），可以考虑在部分代码完成后即开始进行编译调试，以便更早发现和定位错误。
  - 亦可利用函数/模块将程序分解为较小的单元（后续介绍）

小结

- **教学要求：**
  - 初步认识程序的语法和语义
  - 了解C++程序的编译执行过程

- **实践：**
  - 可以参照示例编写和运行具有简单功能的小程序
  - 能够阅读编译器给出的错误提示并修正代码中的语法错误

- **阅读：教材第一章相关内容**