

Python语言程序设计

成都信息工程大学区块链产业学院区块链工程专业

刘硕-信息楼411

电话13608005089

qq-645169109

邮箱-645169109@qq.com

参考教材

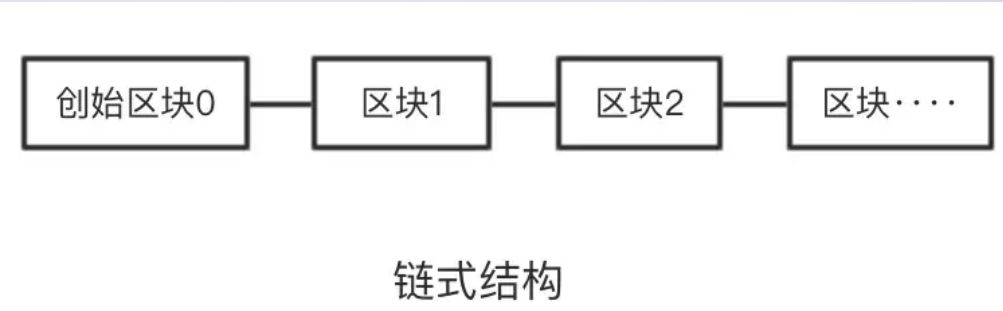
- 嵩天编著，《Python语言程序设计基础（第2版）》，高等教育出版社，2014.7出版，2021.11第21次印刷
- 董付国编著，《Python程序设计（第2版）》，清华大学出版社，2016.6出版，2019.12第16次印刷

- 什么是区块链（Blockchain）？
一个电子记账本，以比特币和其他密码加密货币进行的交易公开地、按照日期顺序记录其中。

总的来说，它是一个公开的数据库，新的数据存储在在一个称为区块的容器中，并且附加到一个“不可变”的链条（即区块链）上，链条上还有以前附加的数据。这里的“不可变”指的是以前的数据一旦附加到链条上，后面是不可更改的。所以说白了，区块链就是一个特殊的历史不可更改的链表数据结构而已。

在比特币和其他密码加密货币的情况下，这些数据指的是一组组交易数据。当然在其他情况下，这个数据可以是任何数据类型。

区块链技术引起了新的“全电子”货币，如比特币和莱特币的发展，这些货币不是由一个集中式的权威机构发行和管理的。同时，区块链也给分布式计算带来了革命，带来了以太坊技术，出现了像智能合约（smart contracts）这样的概念。



```
Block #1 has been added to the blockchain!  
Hash: 0692004087f7dba4d766214dd4272a06c87d81d76aa9ee0e66dee95a45a28fd0  
  
Block #2 has been added to the blockchain!  
Hash: 11b05ef6dee7b41c42be967c09782c78c3bb7057c640ebe8301f8e21a650956e  
  
Block #3 has been added to the blockchain!  
Hash: b44a8a0bd82a866a7c3fc4639af80aa5cd8b05e1c9567575b1d19c3956367f  
  
Block #4 has been added to the blockchain!  
Hash: 5510cb22ac97f6daf1118d450d200aaafdba3ca863ac6e8229eed9567c1bc73a  
  
Block #5 has been added to the blockchain!  
Hash: 56504c2ba6a3cec764854cf0346dc5d3e1ffc2441340604ee9c8fefcd597b1033  
  
Block #6 has been added to the blockchain!  
Hash: 8cc71314e147e88083a22a234b12525e49d7abcd07ca89a2c18b8674b8d6629d  
  
Block #7 has been added to the blockchain!  
Hash: c6c613186d54644c6ed05dc12babc8b1c18cfc65d62eca5b2401b51d3689dfcb  
  
Block #8 has been added to the blockchain!  
Hash: e3bde534b17d152f7cb6b10d454a05491f6fd00a0ca63dc639de16e24da87320  
  
Block #9 has been added to the blockchain!  
Hash: 26e7b59e551b644a5ccdcf1c67db2bd1a5153aad265e87ae6bdeec80efbcd74
```


课程目录

| | |
|------------|------------------------------|
| 第一章 | 程序设计基本方法及基础知识 |
| 第二章 | Python程序的基本元素和代码规范及示例 |
| 第三章 | Python的基本数据类型 |
| 第四章 | 程序的控制结构及选择与循环 |
| 第五章 | Python序列结构 |
| 第六章 | 函数和代码的复用 |
| 第七章 | 组合数据类型 |
| 第八章 | 面向对象的程序设计 |
| 第九章 | 科学计算和可视化 |
| 第十章 | 文件和数据格式化 |

课程目标

学时：32 (20理论+12实践)

学分：2.0

| 毕业要求 | 毕业要求指标点 | 对应课程目标 |
|-----------------|---|---|
| 毕业要求1：工程知识 | 熟悉Python程序设计语言，能熟练设计、调试和分析Python语言程序； | 课程目标1：能熟练表述Python语言的基本概念、语法规则及语言要素，能熟练使用Python语言集成开发环境，正确声明变量，能使用常量、变量及函数构造表达式。 |
| 毕业要求2：设计/开发解决方案 | 能使用Python语言开发数据计算、数据处理与数据分析系统； | 课程目标2：能根据解决问题的方案选择程序的控制结构，熟练设计顺序、分支和循环结构程序，具备设计算法进行数据的收集、计算、检索、分类、汇总能力。 |
| 毕业要求3：环境和可持续发展 | 具备一定的发现问题、描述问题、解决问题等计算思维能力，拥有在计算机数据处理及数据分析方面持续发展潜力； | 课程目标3：具备独立设计Python语言程序进行数据计算与数据分析、绘制常用统计分析图形的能力，能够熟练获取并安装Python第三方库，发布Python程序。为进一步学习计算机及信息类相关知识奠定基础。 |

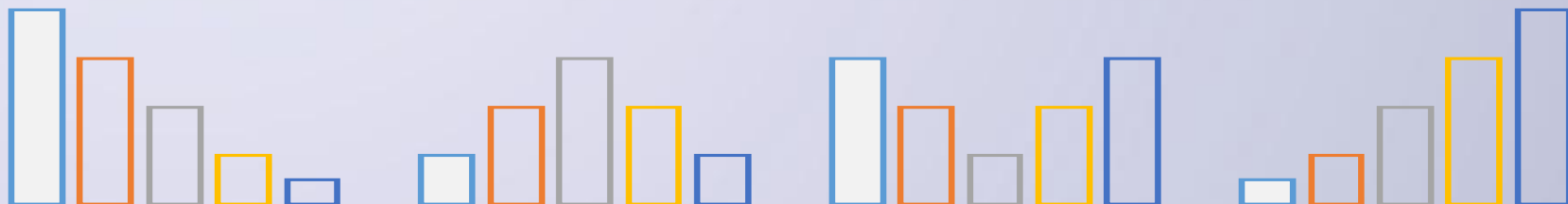
工程知识：能够将python相关的知识用于解决复杂工程问题。

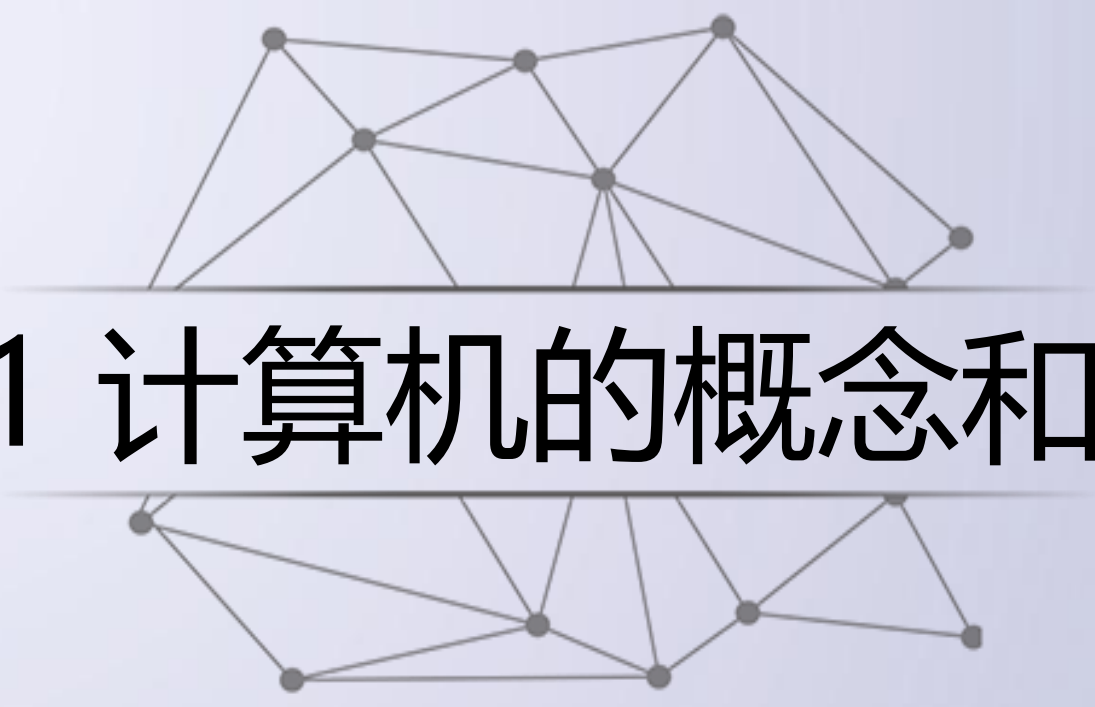
设计/开发解决方案：能够运用所学的python相关知识、技能和方法，对系统的解决方案进行合理的判断和选择，在客观约束条件下找到适当的折中方案，为一个或多个应用领域设计恰当的解决方案。能够基于区块链项目需求规格说明进行区块链技术系统的设计和实现。

目录

- 1.1 计算机的概念和发展**
- 1.2 程序设计语言**
- 1.3 Python语言概述**
- 1.4 Python开发环境配置**
- 1.5 程序的基本编写方法**
- 1.6 Python语言的版本更迭**

第1章 程序设计基本方法及基础知识





1.1 计算机的概念和发展



计算机的概念

- 计算机是根据指令操作数据的设备
- 具备功能性和可编程性两个基本特性

功能性指对数据的操作，表现为数据计算、输入输出处理和结果存储等。

可编程性指它可以根据一系列指令自动地、可预测地、准确地完成操作者的意图



计算机的概念


- 计算机硬件所依赖的集成电路规模按照摩尔定律以指数方式增长
- 计算机运行速度也接近几何级数快速增加
- 计算机所能高效支撑的功能不断丰富发展
- 程序设计语言经历了从机器语言到高级语言的发展过程，朝着更接近自然语言的方向发展。



摩尔定律 (Moore's Law)

■ 摩尔定律是计算机发展历史上最重要的预测法则

摩尔定律指出，单位面积集成电路上可容纳晶体管的数量约每两年翻一倍。由于计算机中几乎所有重要部件都由集成电路实现，因此揭示了半导体技术趋势。



1.2 程序设计语言

提纲

1. 2. 1 什么是程序设计语言

1. 2. 2 程序设计语言的演化

1. 2. 3 构建和运行程序

1. 2. 4 程序设计语言分类

1.2.1 什么是程序设计语言

一、语言的概念：

- 语言（Language）是人类最重要的交际工具，是人们进行沟通交流的主要**表达**方式。
- 语言就广义而言，是一套共同采用的 沟通符号、表达方式与处理规则。
- 语言的定义：由词汇和语法构成并能表达人类思想的符号系统。
语言：是用于表达和描述的工具。语言的基础是一组**字符（单词）**和一组**规则（语法）**。根据规则由单词构成的**字符串（句子）**的总体就是语言。

1.2.1 什么是程序设计语言

二、计算机语言的概念：

- **计算机语言 (Computer Language)** 指用于人与计算机之间通讯的语言。计算机语言是人与计算机之间传递信息的媒介。计算机系统最大特征是**指令**通过一种语言传达给机器。为了使电子计算机进行各种工作，就需要有一套用以编写**计算机程序**的数字、字符和语法规则，由这些字符和语法规则组成计算机各种指令（或各种语句）。这些就是计算机能接受的语言。

1.2.1 什么是程序设计语言

- 计算机系统由硬件和软件组成
 - 物理计算机和外围设备统称为硬件
 - 计算机执行的程序称为软件
- 软件一般分为系统软件和应用软件两大类
- 计算机程序指定计算机完成任务所需的一系列步骤
- 编程语言，又称为程序设计语言，是一组用来定义计算机程序的语法规则。每一种语言都有一套独特的关键字和程序指令语法

1.2.1 什么是程序设计语言

三、程序设计语言的概念

- **程序**：指令（操作）序列。
 - 按照工作步骤事先编排好的、具有特殊功能的指令序列。
- **程序设计**：程序设计是给出解决特定问题程序的过程，是程序构造活动中的重要组成部分。程序设计往往以某种程序设计语言为工具，给出这种语言下的程序。
- **程序设计语言**：用于书写计算机程序的语言，用于**表达和描述**要加工的数据以及求解问题的步骤和过程。是根据预先定义的规则（语法）、由一个有限字母表上的字符构成的字符串的总体。

举例：程序与程序设计语言

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      printf("Hello, world!");
6      return 0;
7  }
```

单词：include, main, return, printf

语句（句子）： return 0;

注：句子是以“;”结束的字符串。

语法（规则）：{ 语句[语句]}

举例：程序及程序设计语言

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

```
main()
{
```

```
    int number1; /*变量声明*/
```

```
    int number2;
```

```
    int result;
```

```
    scanf( “%d%d” ,&number1,&number2); /*输入 从键盘读取2个整数*/
```

```
    result = number1 * number2; /*乘积运算*/
```

```
    printf ( “the result is : %d\n ” , result) ; /*输出结果*/
```

```
    return 0;
```

```
}
```

这是一段用C语言写的程序，表达描述的是两个数的乘法计算过程

操作序列：先读入两个数->乘法运算->输出结果

根据预先定义的规则（语法），由一个有限字母表上的字符构成的字符串的总体

举例：程序及程序设计语言

| | | | | |
|----|----------|-----------|----------|-------------------|
| 1 | 00000000 | 00000100 | | 0000000000000000 |
| 2 | 01011110 | 00001100 | 11000010 | 0000000000000010 |
| 3 | | 11101111 | 00010110 | 0000000000000101 |
| 4 | | 11101111 | 10011110 | 0000000000001011 |
| 5 | 11111000 | 101011101 | 11011111 | 0000000000010010 |
| 6 | | 01100010 | 11011111 | 0000000000010101 |
| 7 | 11101111 | 00000010 | 11111011 | 0000000000010111 |
| 8 | 11110100 | 10101101 | 11011111 | 0000000000011110 |
| 9 | 00000011 | 10100010 | 11011111 | 00000000000100001 |
| 10 | 11101111 | 00000010 | 11111011 | 00000000000100100 |
| 11 | 01111110 | 11110100 | 10101101 | |
| 12 | 11111000 | 101011110 | 11000101 | 00000000000101011 |
| 13 | 00000110 | 10100010 | 11111011 | 00000000000110001 |
| 14 | 11101111 | 00000010 | 11111011 | 00000000000110100 |
| 15 | | 00000100 | 00000100 | 00000000000111101 |
| 16 | | 00000100 | 00000100 | 00000000000111101 |

程序：按照工作步骤事先编排好的、具有特殊功能的指令序列

提纲

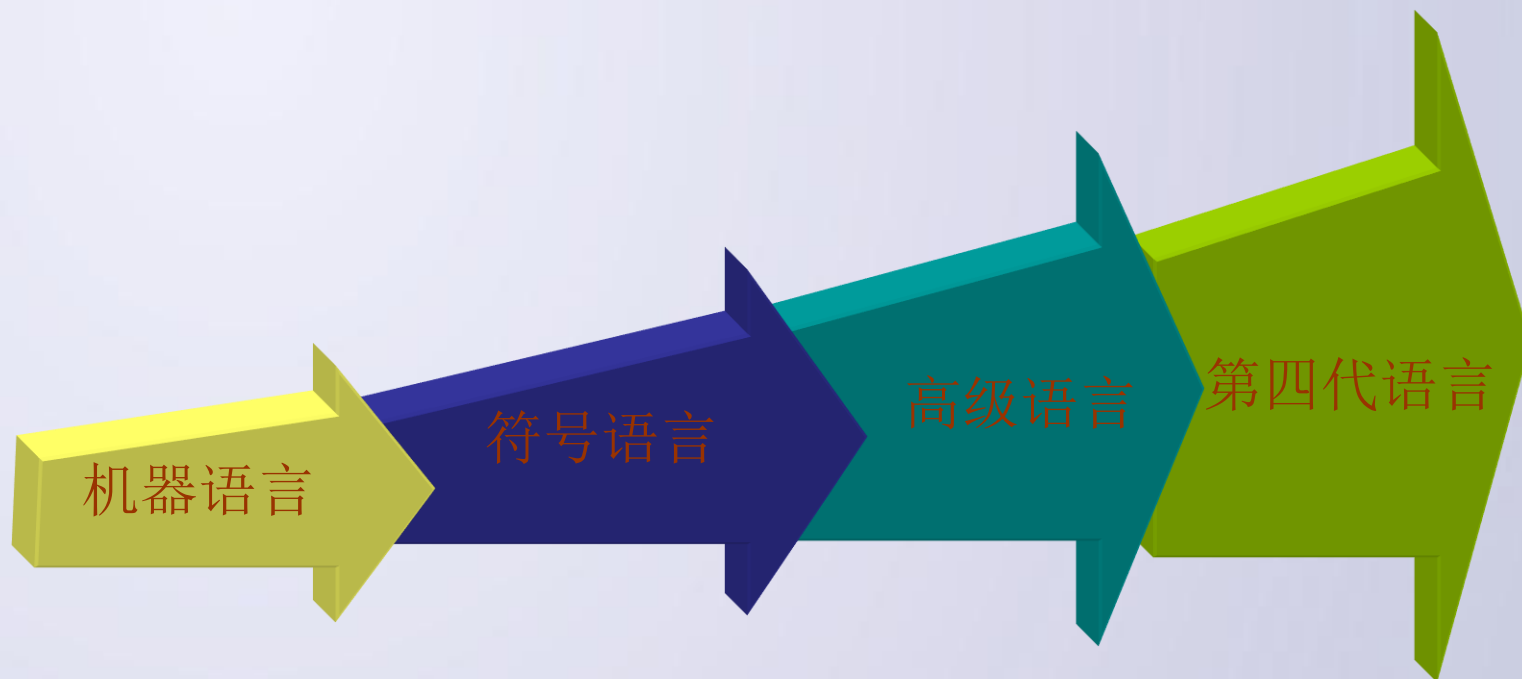
1. 2. 1 什么是程序设计语言

1. 2. 2 程序设计语言的演化

1. 2. 3 构建和运行程序

1. 2. 4 程序设计语言分类

1.2.2 程序设计语言的演化



编程语言

- 编程语言分为低级语言和高级语言两类
 - 低级语言与特定的机器有关
 - 高级语言独立于机器
- 机器语言和汇编语言属于低级语言
 - 机器语言是第一代程序设计语言，使用二进制代码编写程序，可读性差，但能够直接被计算机识别和执行
 - 汇编语言是第二代程序设计语言，使用简单的助记符来表示指令
- 高级语言是独立于计算机体系结构的语言，其最大特点是类似自然语言的形式描述对问题的处理过程
 - C、C++、C#、Java和Python

按范式对编程语言分类

- 面向过程的编程语言
 - FORTRAN、COBOL、Basic、Ada、Pascal、C
- 面向对象的编程语言
 - Java、C#、C++、Smalltalk、Visual Basic
- 函数式编程语言
 - Lisp、Scheme、Haskell、F#
- 逻辑式编程语言：Prolog
- Python程序设计语言属于多范式编程语言

计算思维和程序设计方法

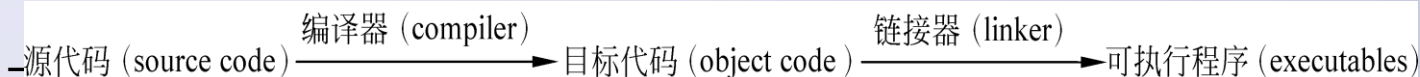
- 人类在认识世界和改造世界过程中形成了以下三种基本的思维
 - 逻辑思维：以推理和演绎为特征，以数学学科为代表
 - 实验思维：以实验和验证为特征，以物理学科为代表
 - 计算思维：以设计和构造为特征，以计算机学科为代表
- 计算思维的本质是抽象（**Abstraction**）和自动化（**Automation**）
- 程序设计方法属于计算思维的范畴，常见的程序设计方法主要包括两种
 - 结构化程序设计和面向对象的程序设计

程序设计方法

- 结构化程序设计通常采用自顶向下（**Top-Down**）、逐步求精（**Stepwise Refinement**）的程序设计方法
- 自顶向下程序设计的基本思想
 - 问题分解、算法实现、组合程序
- 采用自顶向下方法设计的程序，一般通过自底向上（**Bottom-Up**）的方法来实现
 - 先实现、运行和测试每一个基本函数，再测试由基本函数组成的整体函数，这样有助于定位错误

程序的编写和执行

- 用文本编辑器编写和编辑程序
 - Notepad、Vim、Emacs、Sublime等
 - 专用的IDE开发环境，例如IDLE、Spyder、Visual Studio Code等
- 使用文本编辑器编写一个程序后，将文件保存到磁盘上，包含程序代码的文件称之为源文件（**source file**）
- 将源文件转换成机器语言有以下两种转换方法
 - 编译：编译器（**Compiler**）将源代码翻译成目标语言



根据执行机制分类高级编程语言

- 静态语言和脚本语言两类
- 采用编译方式执行的语言属于静态语言，例如C、C++、C#、Java等
 - 优点在于：编译后的目标代码可以直接运行；编译所产生的目标代码执行速度通常更快
- 采用解释方式执行的语言属于脚本语言，例如JavaScript、PHP、Python等
 - 优点在于：源代码可以在任何操作系统上的解释器中运行，可移植性好；解释执行需要保留源代码，因此程序纠错和维护十分方便

1.2.2 程序设计语言的演化

一、机器语言

- ◆ 计算机发展早期使用的语言；面向计算机指令。
- ◆ 指令由“0”和“1”的二进制码组成，是指挥计算机工作的命令，是计算机唯一可以直接识别的语言；
- ◆ 与具体机器有关，不同的机器能识别的机器语言也不同；

1.2.2 程序设计语言的演化—机器语言

| | | | | |
|----|----------|-----------|----------|--------------------|
| 1 | 00000000 | 00000100 | | 0000000000000000 |
| 2 | 01011110 | 00001100 | 11000010 | 0000000000000010 |
| 3 | | 11101111 | 00010110 | 00000000000000101 |
| 4 | | 11101111 | 10011110 | 00000000000001011 |
| 5 | 11111000 | 101011101 | 11011111 | 00000000000010010 |
| 6 | | 01100010 | 11011111 | 00000000000010101 |
| 7 | 11101111 | 00000010 | 11111011 | 00000000000010111 |
| 8 | 11110100 | 10101101 | 11011111 | 00000000000011110 |
| 9 | 00000011 | 10100010 | 11011111 | 000000000000100001 |
| 10 | 11101111 | 00000010 | 11111011 | 000000000000100100 |
| 11 | 01111110 | 11110100 | 10101101 | |
| 12 | 11111000 | 101011110 | 11000101 | 000000000000101011 |
| 13 | 00000110 | 10100010 | 11111011 | 000000000000110001 |
| 14 | 11101111 | 00000010 | 11111011 | 000000000000110100 |
| 15 | | 00000100 | 00000100 | 000000000000111101 |
| 16 | | 00000100 | 00000100 | 000000000000111101 |

1.2.2 程序设计语言的演化—机器语言

机器语言的缺陷

- ◆ 非常晦涩难读；
- ◆ 书写工作量大，且容易出错，不易修改；
- ◆ 由于和具体的机器相关，要求开发人员对计算机的硬件和指令系统要有很正确深入的理解，并且有熟练的编程技巧，因此只有少数专家能达到此要求；
- ◆ 移植性不好(在一台机器上编写的机器语言程序在不同型号的另一台机器上可能不能运行)；

1.2.2 程序设计语言的演化—符号语言

二、符号语言

- ◆ 20世纪50年代早期，数学家Grace Hopper发明了符号语言，即用符号或助记符来表示不同的机器语言指令（包括操作码和和操作数地址）。
- ◆ 程序员可用ADD、SUB、MUL、DIV等符号来分别表示加法、减法、乘法、除法的操作码。
- ◆ 符号语言又称汇编语言。

求 $d = b^2 - 4ac$ 的汇编语言程序

操作码

程序

操作数地址

注释(运算结果)

| | | | | |
|----|-----|---|---|----------------------|
| 1. | MUL | B | B | ; b^2 送入B |
| 2. | MUL | A | E | ; 4a送入A |
| 3. | MUL | A | C | ; 4ac送入A |
| 4. | SUB | B | A | ; $b^2 - 4ac$ 送入B |
| 5. | MOV | D | B | ; $b^2 - 4ac$ 从B传送到D |
| 6. | HLT | | | ; 停机 |

用符号
或助记
符来表
示指令
中的操
作码和
操作
数地址

| | |
|---|---|
| A | a |
| B | b |
| C | c |
| D | d |
| E | 4 |

1~6为指令，MUL为乘法指令，SUB为减法指令，MOV为传送指令、HLT为停机指令；

A、B、C、D、E分别表示存储数a、b、c、d以及常数4的寄存器。

1.2.2 程序设计语言的演化—符号语言

- 汇编语言编写的程序需要翻译成机器语言（二进制代码）才能运行，这个翻译过程由汇编程序来实现。

1.2.2 程序设计语言的演化—符号语言

汇编语言的局限

- ◆ 汇编语言的语法、语义结构仍然和机器语言基本一样，而与人的传统解题方法相差甚远。
- ◆ 汇编语言的大部分指令是和机器指令一一对应的，因此代码量大。
- ◆ 和具体的机器相关，人们终究还是要对计算机的硬件和指令系统有很正确深入的理解，而且还是要记住机器语言的符号(助记符)。移植性不好。

1.2.2 程序设计语言的演化—高级语言

三、高级语言

- ◆ 由于汇编语言的局限性，后来出现了高级语言。
- ◆ 高级语言与自然语言（尤其是英语）很相似，因此高级语言程序易学、易懂、也易查错。

两数相乘的C语言程序

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

```
main()
```

```
{
```

```
    int number1;
```

```
    int number2;
```

```
    int result;
```

```
    printf("please input the two numbers:\n");
```

```
    scanf("%d %d",&number1,&number2); //读取乘数和被乘数
```

```
    result = number1 * number2; //两数相乘
```

```
    printf("the result is : %d\n ", result) ;//输出结果
```

```
    return 0;
```

```
}
```

1.2.2 程序设计语言的演化—高级语言

高级语言的优势

- ◆ 高级语言程序易学、易懂、也易查错。
- ◆ 使程序员可以完全不用与计算机的硬件打交道、不必了解机器的指令系统。
- ◆ 高级语言与具体机器无关，在一种机器上运行的高级语言程序有可能可以不经改动地移植到另一种机器上运行，大大提高了程序的通用性。

1.2.2 程序设计语言的演化

四、第四代语言

第四代语言(Fourth-Generation Language, 以下简称4GL)是一种面向问题的程序设计语言, 实现了在更高级层次上的抽象, 可以极大地提高软件生产率, 缩短软件开发周期。4GL提供了功能强大的非过程化问题定义手段, 用户只需告知系统做什么, 而无需说明怎么做, 程序就能够自动生成算法, 自动进行处理。典型的4GL语言有ADA, MODULA-2, SMALLTALK-80等。

1.2.2 程序设计语言的演化

- 按照**4GL**的功能可以将其划分为查询语言和报表生成器、图形语言、应用生成器、形式规格说明语言等几类。
- **查询语言**:是数据库管理系统的主要工具, 它提供用户对数据库进行查询的功能。如**SQL(Structured Query Language,结构化查询语言)**
- **报表生成器**:是为用户提供的自动产生报表的重要工具, 它提供非过程化的描述手段让用户很方便地根据数据库中的信息来生成报表, 如**ADF**。
- **图形语言**:用图形的方式来进行软件开发。
- **应用生成器**: 是重要的一类综合的**4GL**工具, 它用来生成完整的应用系统。应用生成器让用户不必使用多个软件, 而只用这样一个综合工具来实现多种功能。
- **形式规格说明语言**:形式化地对软件应满足的功能、性能及其它重要方面的陈述, 可避免自然语言的二义性, 也是软件自动化的基础。

1.2.2 程序设计语言的演化

五、自然语言

理想情况下，计算机能够理解自然语言（如英语、汉语等）并立即执行请求。

大量关于自然语言的工作正在实验室中进行。

但迄今为止，自然语言的使用仍然是相当有限的。

总结

程序设计语言：

用于书写计算机程序的语言，用于**表达和描述**要加工的数据以及求解问题的步骤和过程。是根据预先定义的**规则（语法）、**由一个有限字母表上的字符构成的**字符串（单词和句子）**的总体。

提纲

1. 2. 1 什么是程序设计语言

1. 2. 2 程序设计语言的演化

1. 2. 3 构建和运行程序

1. 2. 4 程序设计语言分类

1.2.3 构建和运行程序

1、程序写到哪？存到哪？ --- 文件作为载体

- ◆高级语言程序（C程序）是字符串的集合，字符即文本，类似于我们经常使用的英文文本；
- ◆用文本编辑器（windows word、TXT编辑器等）都可以写并保存C语言写的程序；我们称之为“**源程序**”；
- ◆由于计算机只能执行二进制编码的指令，而C语言写的程序是不能够直接在计算机上执行，
- ◆我们需要将C语言写的程序“变换”到机器指令序列程序，我们称之为“**目标程序或可执行程序**”

1.2.3 构建和运行程序

2、翻译程序---编译程序和解释程序

- ◆ 高级语言程序翻译成机器语言程序需要借助于翻译程序。
- ◆ 翻译实际上是“变换”过程，也即：计算过程
- ◆ 翻译过程是根据程序语言的语法规则进行，所以，同时可以检查语法或词法错误；
- ◆ 翻译程序也是一个程序，由第三方提供，我们只是使用
- ◆ 翻译程序有编译程序和解释程序两种。

1.2.3 构建和运行程序

◆ **编译程序 (Compiler)**：将编写的源程序中全部语句**一次性**翻译成机器语言程序后，再运行机器语言程序。编译和运行是两个独立分开的阶段。若想多次运行同一个程序，只要源程序不变，则不需要重新编译；源程序若有修改，则需要重新编译。

1.2.3 构建和运行程序

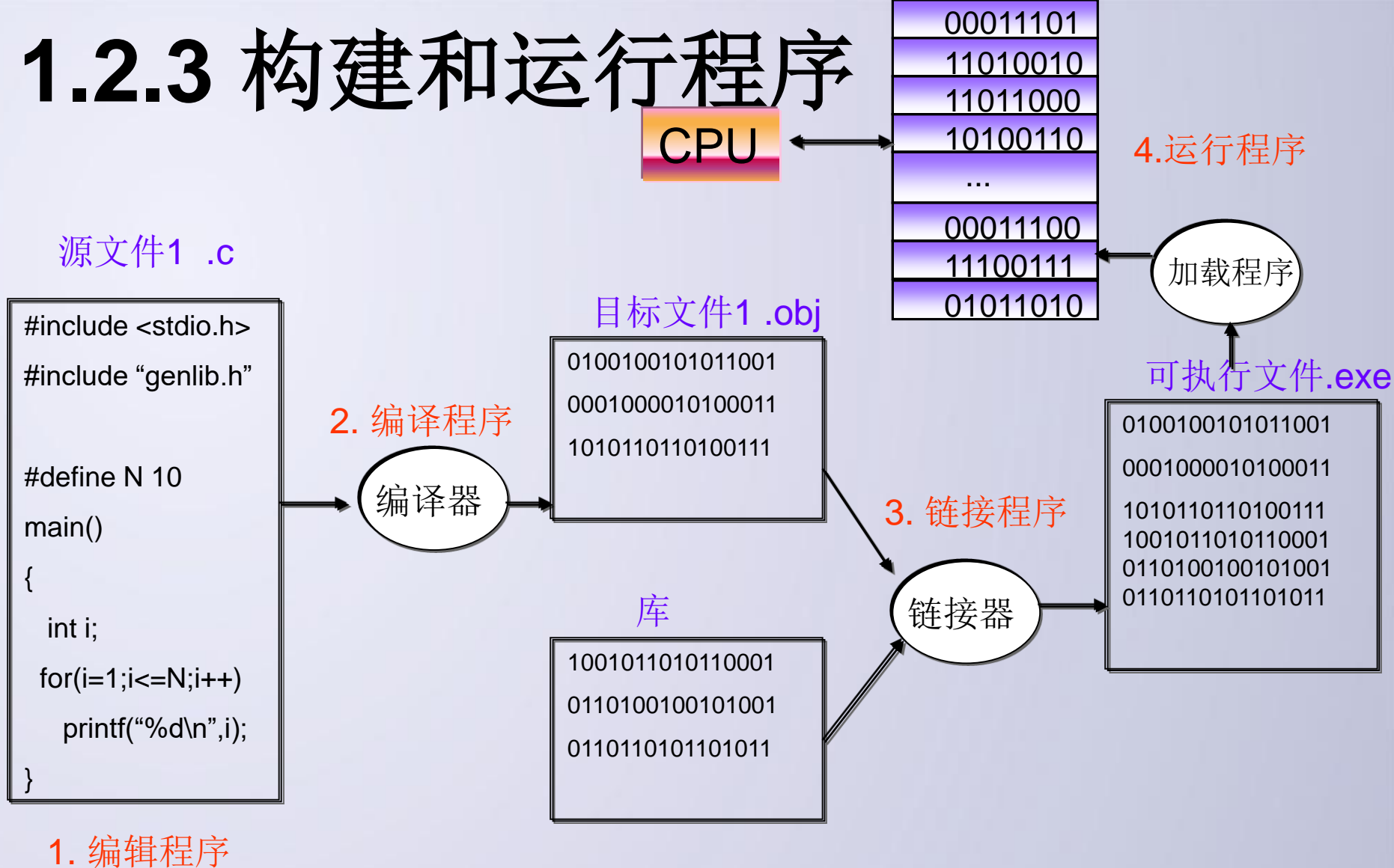
- ◆ **解释程序**：将源程序中的一条语句翻译成机器语言后立即执行它（并且不再保存刚执行完的机器语言程序），然后再翻译执行下一条语句。如此重复，直到程序结束。如果一条语句要重复执行，则每一次的重复执行都要重新翻译该语句，故效率很低。
 - ◆ 著名的解释程序有：**BASIC**语言解释程序、**LISP**语言解释程序、**UNIX**命令语言（**shell**）解释程序、数据库查询语言**SQL**解释程序等。
 - ◆ 当前大部分语言如**C**、**C++**、**FORTRAN**、**ALGOL**等是用编译程序进行翻译的。而**BASIC**、**PASCAL**、**LISP**等既有编译程序、又有解释程序。

1.2.3 构建和运行程序

3、链接程序

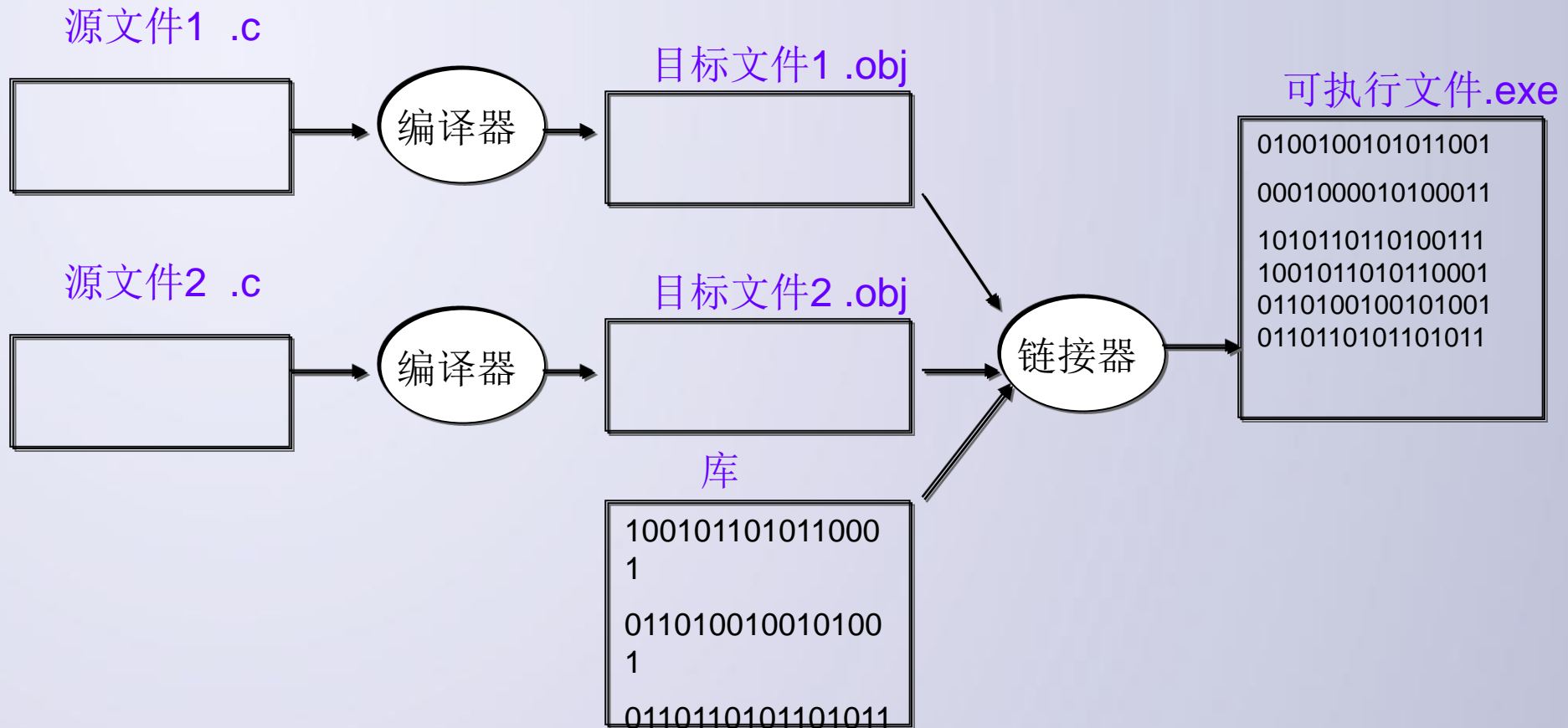
- 链接程序，编译器和汇编程序都经常依赖于链接程序，它将分别在不同的目标文件(.obj)中编译或汇编的代码收集到一个可直接执行的文件中（.exe）。
- **链接程序（linker）**在这种情况下，目标代码即还未被连接的机器代码，与可执行的机器代码之间就有了区别。连接程序还连接目标程序和用于标准库函数的代码，以及连接目标程序和由计算机的操作系统提供的资源（例如，存储分配程序及输入与输出设备）。连接过程对操作系统和处理器有极大的依赖性。

1.2.3 构建和运行程序



C标准库：有丰富的函数集，可供程序员直接使用。

1.2.3 构建和运行程序



当程序由多个源文件组成时

源文件 .c

```
#include <stdio.h>
#include "genlib.h"
#define N 10
main()
{
    int i;
    for(i=1;i<=N;i++)
        printf("%d\n",i);
}
```

stdlib.h

.....
.....

genlib.h

.....

目标文件 .obj

```
0100100101011001
0001000010100011
1010110110100111
```

编译器

预处理程序 + 翻译程序

预处理程序：通常是把其他文件包含到要编译的文件中、以及用程序文本替换专门的符号。C语言中预处理指令都以#开头。

翻译程序：将预处理后的程序翻译成二进制代码。

1.2.3 构建和运行程序

源文件 .c

```
#include <stdio.h>
#include "genlib.h"
#define N 10
main()
{
    int i;
    for(i=1;i<=N;i++)
        printf("%d\n",i);
}
```

预处理

文件stdio.h
和genlib.h的内容

```
main()
{
    int i;
    for(i=1;i<=10;i++)
        printf("%d\n",i);
}
```

1.2.3 构建和运行程序

源文件 .c

```
#include <stdio.h>
#include "genlib.h"
#define N 10
main()
{
    int i;
    for(i=1;i<=N;i++)
        printf("%d\n",i);
}
```

预处理

文件stdio.h
和genlib.h的内容

```
main()
{
    int i;
    for(i=1;i<=10;i++)
        printf("%d\n",i);
}
```

翻译

目标文件 .obj

```
0100100101011001
0001000010100011
1010110110100111
```

库

```
1001011010110001
011010010010100
1
0110110101101011
```

链接器

可执行文件.exe

```
010010010101100
1
00010000101000
11
101011011010011
1
100101101011000
1
011010010010100
1
011011010110101
1
```

1.2.3 构建和运行程序

**示范如何在DEV C++集成编程环境下构建和运行一个C程序
基本步骤：**

- 1、使用“文件”菜单功能创建并编写源程序，并存储为“.c”文件。
- 2、使用“运行”菜单中的编译功能，对所编写的源程序进行编译。编译中进行语法检查，如果有语法错误须改错。
- 3、编译结果无错误告警，方可运行程序，使用“运行”菜单中的运行功能。看程序运行的结果。

1.2.3 构建和运行程序

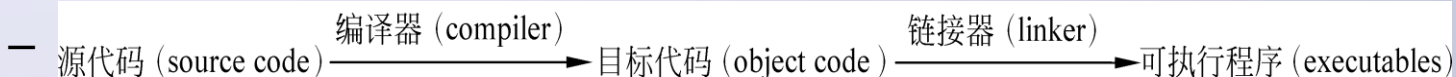
扩展思考：

程序的编译可以认为是“计算”过程，是根据程序语言的语法规则进行计算，那么语法规则该如何描述呢？

参考答案：人们将用一组**数学符号**和**规则**来描述事物的方式称为**形式描述**，而把所用的数学符号和规则称为**形式语言**。研究程序设计语言以及自然语言的形式化定义、分类、结构等有关理论，以及**识别**各类语言的形式化模型（自动机模型）及其相互关系。

程序的编写和执行

- 用文本编辑器编写和编辑程序
 - Notepad、Vim、Emacs、Sublime等
 - 专用的IDE开发环境，例如IDLE、Spyder、Visual Studio Code等
- 使用文本编辑器编写一个程序后，将文件保存到磁盘上，包含程序代码的文件称之为源文件（**source file**）
- 将源文件转换成机器语言有以下两种转换方法
 - 编译：编译器（**Compiler**）将源代码翻译成目标语言



提纲

1. 2. 1 什么是程序设计语言

1. 2. 2 程序设计语言的演化

1. 2. 3 构建和运行程序

1. 2. 4 *程序设计语言分类

1.2.4 程序设计语言的分类

- 根据人们对于求解问题的方法进行分类可以将程序设计语言进行分类。

过程化语言

函数型语言

面向对象语言

说明性语言

专用语言

1.2.4 程序设计语言的分类

1、过程化语言

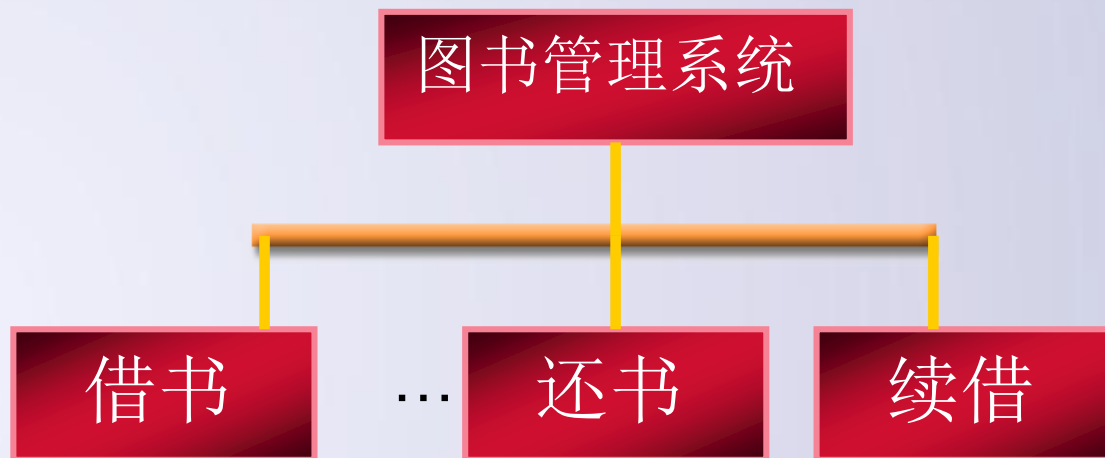
基于冯·诺依曼结构描述计算过程的语言。

面向过程的程序设计特点：

- ◆ 面向动作（活动）。求解问题（程序）是一个复杂的计算过程，该计算过程可被分解成若干个动作（活动）。即：一个计算过程可看做是一系列动作。
- ◆ 一个动作可以是一条语句（基本操作）或是一个可以继续分解的子过程（子程序）。

1.2.4 程序设计语言的分类

- ◆ 子过程（子程序）：每一个子程序实现一个明确、独立的功能。应用上述方法可继续分解为一系列动作。
- ◆ 循环往复直至将“动作”分解为“基本操作—语句”。
- ◆ 总之，面向过程的设计即是实现某一计算的操作过程和操作步骤，然后用过程化语言来描述这些操作过程和步骤。



图书管理系统被划分成借书、还书、续借等功能，每一个功能的实现对应于一个执行过程，有一系列的执行步骤。

一个简单的借书过程可能如下：

- 1.读取借阅者的借书证信息，验证借阅者身份合法性；
- 2.若身份合法，则验证借阅者是否有欠款未交；
- 3.若有欠款，则记录借阅者的付款信息；
- 4.验证借阅者当前是否已经借满书；
- 5.若可以借书，则将借阅者的借书信息记录到系统中；

.....

用过程化
语言描述
的操作过
程

1.2.4 程序设计语言的分类

| | |
|---------|---|
| FORTRAN | 第一个高级语言，由Jack Backus领导下的一批IBM工程师设计，1957年商用。适用于科学计算。 |
| COBOL | 面向商业数据处理的语言。 |
| PASCAL | 1971，Niklaus Wirth在瑞士的苏黎士发明。通过强调结构化编程方法来教初学者编程。适用于科研和教学，但从未在工业界获得广泛的流行。 |
| C | Dennis Ritchie在20世纪70年代初发明，最初是想用于编写操作系统和系统软件。具有高级数据结构和控制结构，亦可调用底层功能，指令短、高效。 |
| ADA | 为美国国防部开发，分布式系统控制语言。 |

1.2.4 程序设计语言的分类

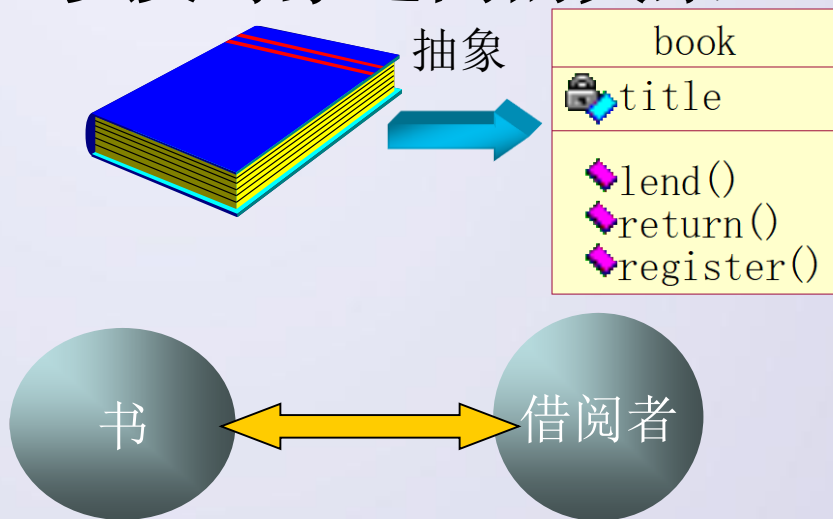
2、面向对象语言

面向对象基本思想：

- ◆ 从现实世界中客观存在的事物出发来构建软件系统，强调直接以问题域（现实世界）中的事物为中心来思考问题、认识问题，并根据这些事物的本质特征，把它们抽象地表示为系统中的对象，作为系统的基本构成单位。（对象）
- ◆ 对象包含属性和操作，每一个对象有明确的职责，完成一定的功能。（属性和操作）
- ◆ 对象之间不是孤立的，而是具有各种关系。
- ◆ 对象与对象之间通过消息进行通信，相互协作。（消息）
- ◆ 同类型的对象可以进一步抽象出共性，形成类。（类）

1.2.4 程序设计语言的分类—面向对象语言

- 面向对象语言可以用来描述参与问题解决的对象以及对象之间的关系。



用Java语言
描述

```
Public class Book
{
    Private String title;
    Public Boolean lend(){
        .....
    }
    Public Boolean return(){
        ...
    }
}
```

图书管理系统中的2个
主要类：书、借阅者

1.2.4 程序设计语言的分类

主要的两种面向对象语言

| | |
|------|--|
| C++ | 扩展的C语言，具有了面向对象的特性。 |
| JAVA | 纯面向对象语言。Sun公司开发，在C和C++的基础上发展而来，但去除了C++中的一些语言特性，从而更加强壮。 |

1.2.4 程序设计语言的分类一

函数型说明语言

3、函数型说明语言

基于数学建模求解问题的思想。

程序的基本单元是函数。函数可以理解成一个黑盒，完成从一系列输入（X）到输出（Y）的映射。

函数型语言主要实现以下功能：

定义一系列可供任何程序员调用的基本函数；

允许程序员通过组合若干基本函数创建新的函数。

1.2.4 程序设计语言的分类—函数 数型说明语言

例如：定义一个称为**First**的基本函数。由它来完成从一个列表中抽取第一个元素的功能。再定义另一个函数**rest**，由它完成从一个列表中抽取出除第一个元素以外的所有元素。通过组合使用两个函数，可以在程序中定义一个函数来完成对第三个元素的抽取。

1.2.4 程序设计语言的分类—函数型说明语言

| | |
|--------|--|
| LISP | <p>20世纪60年代早期由麻省理工大学（MIT）科研小组设计开发。</p> <p>把所有的一切都看成是列表，把列表作为处理对象。</p> |
| Scheme | <p>定义了一系列基本函数。函数名和函数的输入列表写在括号内，结果是一个可用于其他函数输入的列表。</p> <p><code>(car 2 3 7 8 11 17 20) == => 2</code> <code>(cdr 2 3 7 8 11 17 20) == => 3 7 8 11 17 20</code></p> <p>现在可以组合这两个函数实现从列表中取出第三个元素7：<code>(car (cdr (cdr list)))</code></p> |

1.2.4 程序设计语言的分类—说明性 (逻辑) 语言

4、说明性（逻辑）语言

根据已知正确的一些论断（事实），运用逻辑推理的可靠准则推导出新的论断（事实）。

If (A is B) and (B is C), then (A is C)

将此准则应用于下面的事实：

事实1: Socrates is a human \rightarrow A is B

事实2: A human is mortal \rightarrow B is C

则可以推导出下面的事实：

事实3: Socrates is mortal \rightarrow A is C

1.24 程序设计语言的分类—说明性 (逻辑) 语言

- ◆ 程序员需要知道该领域内的所有已知的事实，还应该精通逻辑上如何严谨地定义准则，这样程序才能推导并产生新的事实。
- ◆ 由于要收集大量的事实信息，所以程序是针对特定领域的。说明性程序设计迄今为止只是局限于人工智能领域。

1.2.4 程序设计语言的分类—说明性 (逻辑) 语言

PROLOG: 适用于人工智能的专家系统的说明性语言

已知事实:

human (John)

mortal (human)

用户可以进行查询

?-mortal (John)

程序响应是(yes)。

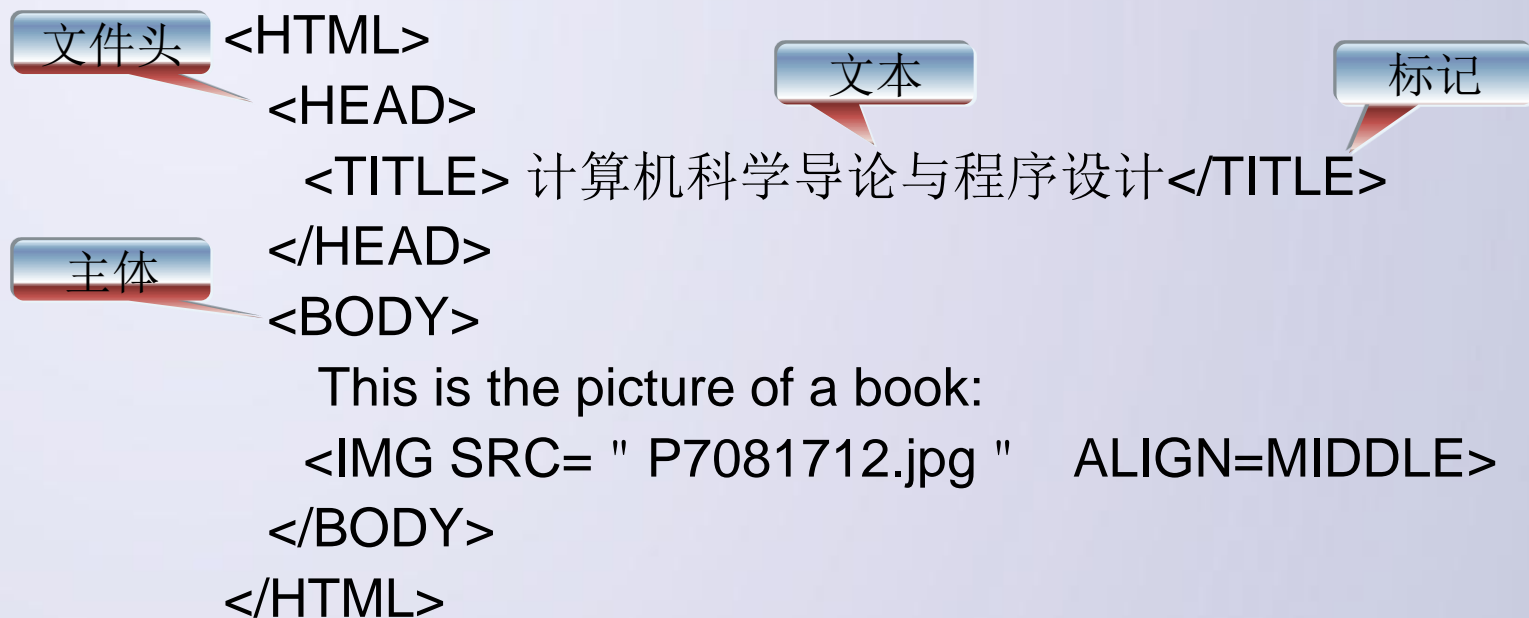
1.2.4 程序设计语言的分类

5、专用语言

- ◆ **HTML**：超文本标记语言（Hypertext Markup Language, HTML），是World Wide Web Consortium (W3C)组织发布的标准，专门为在互联网上发布信息而设计。
- ◆ **HTML文件**由文本和标签组成（类似<head>,<table>等）。标签是预先规范和定义好的，用户不能自己创建。
- ◆ 存放在服务器端，可以由浏览器下载到本地进行解析、显示。

1.2.4 程序设计语言的分类—HTML

HTML文件由文件头和主体组成。



文件头：包含页面标题和其他浏览器将要使用到的参数。

文本是页面中所包含的实际信息。标记则定义了文档呈现的格式。

1.2.4 程序设计语言的分类

- **PERL**：实用摘要和报表语言（**Practical Extraction and Report Language, PERL**）
- **SQL**：结构查询语言（**Structured Query Language, SQL**），是一种对数据库进行查询、操作的语言



程序设计语言概括

■ 程序设计语言包括编译执行和解释执行两种方式

程序设计语言是计算机能够理解和识别用户操作意图的一种交互体系，它按照特定规则组织计算机指令，使计算机能够自动进行各种运算处理。按照程序设计语言规则组织起来的一组计算机指令称为计算机程序。



程序设计语言概括

■ 机器语言

机器语言是一种二进制语言，它直接使用二进制代码表达指令，是计算机硬件可以直接识别和执行的程序设计语言。

例如：执行数字2和3的加法，16位计算机上的机器指令为：11010010 00111011，不同计算机结构的机器指令不同



程序设计语言概括

■ 汇编语言

使用助记符与机器语言中的指令进行一一对应，在

计算机发展早期帮助程序员提高编程效率

例如：执行数字2和3的加法，汇编语言指令为：

add 2, 3, result, 运算结果写入result。

- 机器语言和汇编语言都直接操作计算机硬件并基于此设计，所以它们统称为低级语言。



程序设计语言概括

■ 高级语言

高级语言区别于低级语言在于，高级语言是接近自然语言的一种计算机程序设计语言，更容易地描述计算问题并利用计算机解决计算问题。

例如：执行数字2和3加法的高级语言代码为：

```
result = 2 + 3
```



编译和解释

高级语言按照计算机执行方式的不同可分成两类

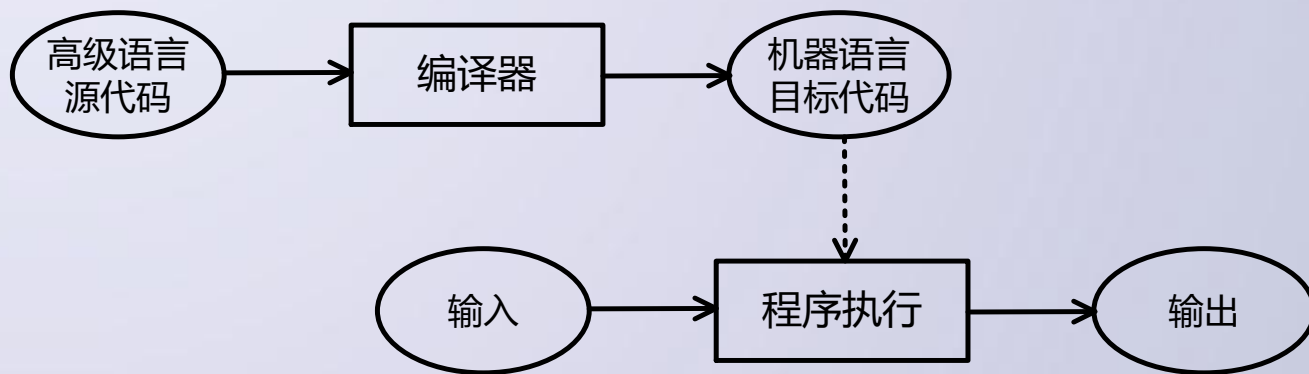
- 静态语言

- 脚本语言

这里所说的执行方式指计算机执行一个程序的过程，静态语言采用编译执行，脚本语言采用解释执行。

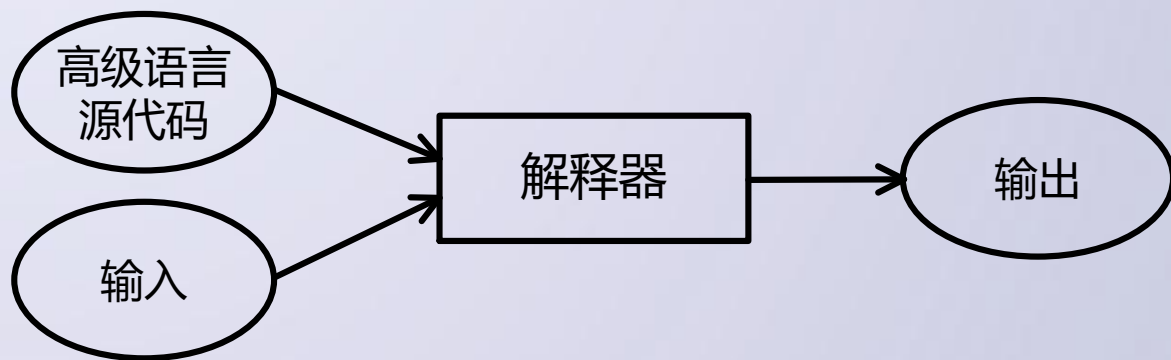
编译和解释

- 编译是将源代码转换成目标代码的过程，通常，源代码是高级语言代码，目标代码是机器语言代码，执行编译的计算机程序称为编译器



编译和解释

- 解释是将源代码逐条转换成目标代码同时逐条运行目标代码的过程。执行解释的计算机程序称为解释器。





编译和解释

编译是一次性地翻译，一旦程序被编译，不再需要编译程序或者源代码。

- 对于相同源代码，编译所产生的目标代码执行速度更快。
- 目标代码不需要编译器就可以运行，在同类型操作系统上使用灵活。



编译和解释

解释则在每次程序运行时都需要解释器和源代码。

■解释执行需要保留源代码，程序纠错和维护十分方便。


■只要存在解释器，源代码可以在任何操作系统上运行，可移植性好



计算机编程

——为什么要学习计算机编程？

——因为“编程是件很有趣的事儿”！



计算机编程

计算思维是区别于以数学为代表的逻辑思维 and 以物理为代表的实证思维的第三种思维模式。

编程是一个求解问题的过程

- 首先需要分析问题，抽象内容之间的交互关系
 - 设计利用计算机求解问题的确定性方法，
 - 进而通过编写和调试代码解决问题
- 这是从抽象问题到解决问题的完整过程。



1.3 Python语言概述

Python语言的诞生

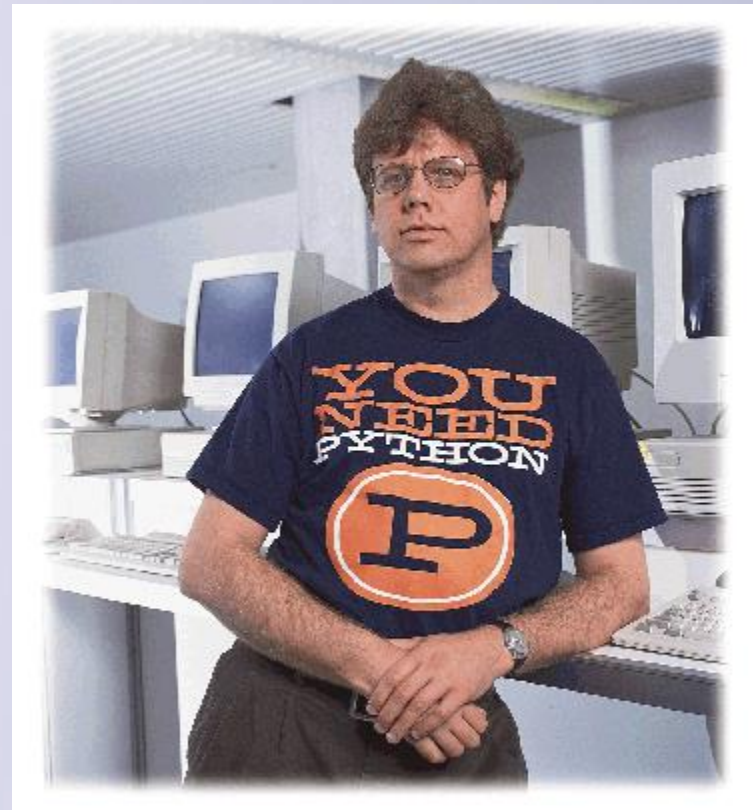
Guido van Rossum

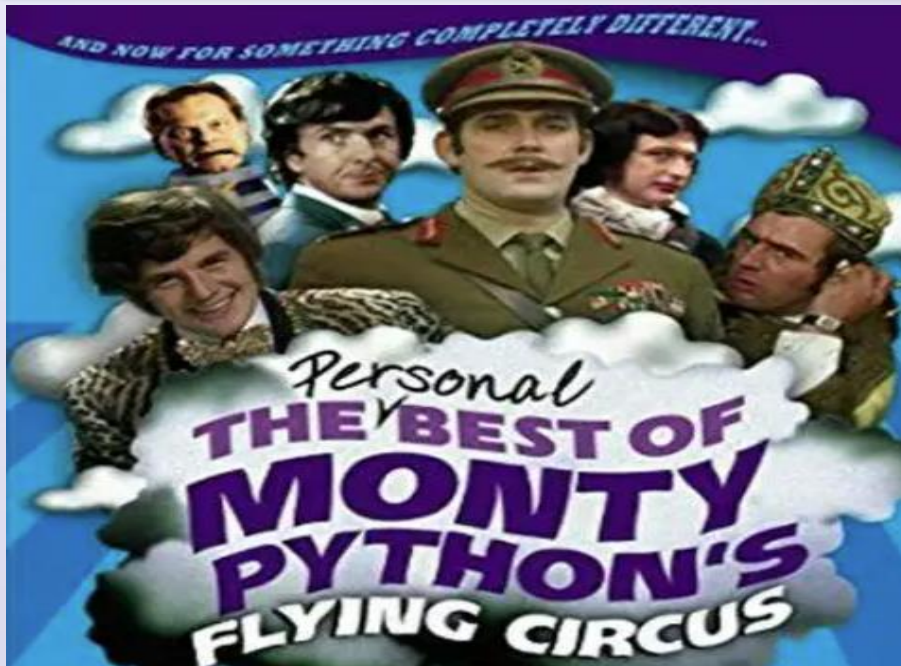
Benevolent Dictator For Life

Python(1990)语言创立者

2000年, Python 2.x

2008年, Python 3.x





Monty Python组合

Python是一种怎样的语言

- Python是一门跨平台、开源、免费的解释型高级动态编程语言。
- Python支持命令式编程(HOW)、函数式编程(WHAT)，完全支持面向对象程序设计，语法简洁清晰，拥有大量的几乎支持所有领域应用开发的成熟扩展库。
- 胶水语言：可以把多种不同语言编写的程序融合到一起实现无缝拼接，更好地发挥不同语言和工具的优势，满足不同应用领域的需求。

Python相关的资源

- 几个重要网址
 - <https://www.python.org/>
 - <https://www.python.org/doc/>
 - <http://bugs.python.org/>
 - <https://hackerone.com/python>
 - <http://stackoverflow.com/questions/tagged/python>

The screenshot shows the Python bug tracker website. At the top, there's a navigation bar with the Python logo and 'bug tracker' text. Below it, a search bar and a filter dropdown (set to 'open') are visible. A prominent red-bordered box contains a migration notice: 'This issue tracker has been migrated to GitHub, and is currently read-only. For more information, see the GitHub FAQs in the Python's Developer Guide.' Below this notice is a table of issues.

| ID | GH | 最近修改 | 标题 | 状态 | 创建人 | 分配给 | 类型 | Msgs |
|-------|-------|------|---|------|--------------|--------------|----------|------|
| 2771 | 47020 | 5月之前 | Test issue | open | gvanrossum | ezio.melotti | behavior | 145 |
| 24778 | 68966 | 5月之前 | [CVE-2015-20107] mailcap.findmatch: document shell command Injection danger in filename parameter | open | TheRegRunner | docs@python | security | 15 |
| 47261 | 91417 | 6月之前 | RFC: Clarify usage of macros for PySequence_Fast within the Limited C API | open | rgoswami | docs@python | | 2 |



编写Hello程序

使用Python语言编写的Hello程序只有一行代码

```
print("Hello World")
```

```
>>>print("Hello World")  
Hello World
```

第一行的“>>>”是Python语言运行环境的提示符

第二行是Python语句的执行结果



C语言的Hello程序

```
#include <stdio.h>
int main(void)
{
printf("Hello World\n");
return 0;
}
```

一般来说，同样功能的程序，Python语言实现的代码行数仅相当于C语言的1/5至1/10，简洁程度取决于程序的复杂度和规模。



Python语言的优势

脚本语言 + 语句执行

例1:

```
print("Hello World! 大家好!")
```

例2:

```
sum = 99999 * 99999  
print(sum)
```



Python语言的优势

例3:

```
months="JanFebMarAprMayJunJulAugSepOctNovDec"
```

```
n = 4
```

```
monthAbbrev = months[(n-1)*3:(n-1)*3+3]
```

```
print(monthAbbrev)
```



Python语言的优势

简洁 + 强制可读性

例4:

```
def mean(numbers):  
    s = 0.0  
    for num in numbers:  
        s = s + num  
    return s / len(numbers)  
  
nums = [0,1,2,3,4,5,6,7,8,9]  
print(mean(nums))
```

0.0



Python语言的优势

简洁 + 强制可读性

例4:

```
def mean(numbers) :  
    s = 0.0  
    for num in numbers:  
        s = s + num  
    return s / len(numbers)  
nums = [0,1,2,3,4,5,6,7,8,9]  
print(mean(nums))
```

4.5

Python语言的优势

跨平台 + 开源

<http://pypi.python.org/>

目前有许多开源库，覆盖各类计算问题

(requests django flask

twisted beautifulsoup4 numpy

pandas matplotlib tensorflow

tkinter keras pytorch)





Python语言的优势

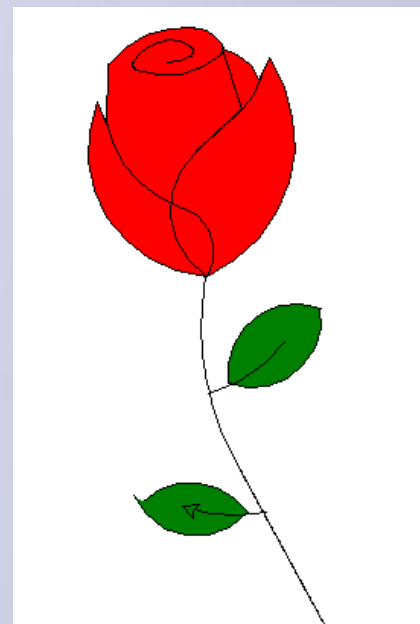
Python语言的优势：面向过程 + 面向对象

灵活的介绍程序设计理念

Python语言的优势：图形界面

Windows窗口

玫瑰花





Python语言特点

- Python语言是通用语言
- Python语言是脚本语言
- Python语言是开源语言
- Python语言是跨平台语言
- Python语言是多模型语言

Python的两种编程方式

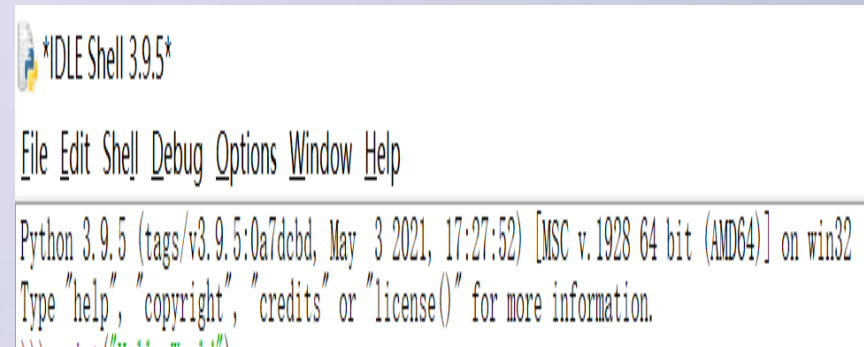
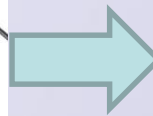
交互式和文件式

- **交互式：对每个输入语句即时运行结果，适合语法练习**
- **文件式：批量执行一组语句并运行结果，编程的主要方式**

Python的两种编程方式

- IDLE提前登场

IDLE是一个Python Shell，shell的意思就是“外壳”，就是一个通过键入文本与程序交互的途径！



实例1: 圆面积的计算

根据半径计算圆面积

*可以表示乘法运算符，字符串接*可以按照后边的数字输出多次。

```
>>> r = 25
>>> area = 3.1415 * r * r
>>> print(area) 1963.4375000000002
>>> print("{:.2f}".format(area))
1963.44
```

字符串方法

交互式

```
>>> print (good *8)
Traceback (most recent call last):
  File "<pyshell#18>", line 1, in <module>
    print (good *8)
NameError: name 'good' is not defined
>>> print ("good study"*2)
good studygood study
>>> print ("good study\n"*2)
good study
good study
```

实例1: 圆面积的计算

根据半径计算圆面积

```
r = 25
area = 3.1415 * r * r
print(area)
print("{:.2f}".format(area))
```

输出结果如下:

```
1963.43750000000002
1963.44
```

保存为CalCircle.py文件并运行

文件式

- `format()`-行走的翻译官 模具



位置参
数...

....

.....

{0}

{1}

{2}

关键字
参数

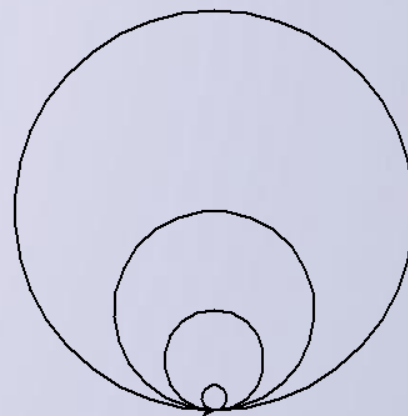
兼职工作的除法运算符%-数字进制转换

实例2: 同切圆绘制

绘制多个同切圆

```
import turtle
turtle.pensize(2)
turtle.circle(10)
turtle.circle(40)
turtle.circle(80)
turtle.circle(160)
```

模块



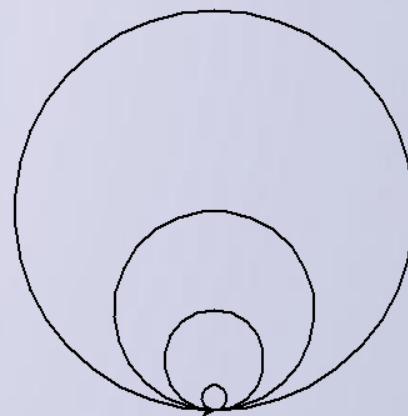
保存为TangentCirclesDraw.py文件并运行

文件式

实例2: 同切圆绘制

绘制多个同切圆

```
>>> import turtle  
>>> turtle.pensize(2)  
>>> turtle.circle(10)  
>>> turtle.circle(40)  
>>> turtle.circle(80)  
>>> turtle.circle(160)
```



交互式

实例3: 输出日期和时间

```
IDLE Shell 3.10.7
File Edit Shell Debug Options Window Help
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> from datetime import datetime
>>> now=datetime.now()
>>> print(now)
2023-05-04 10:00:13.769983
>>> now.strftime("%x")
'05/04/23'
>>> now.strftime("%X")
'10:00:13'
>>> help("datetime")
Help on module datetime:

NAME
datetime - Fast implementation of the datetime type.

MODULE REFERENCE
https://docs.python.org/3.10/library/datetime.html

The following documentation is automatically generated from the Python
source files. It may be incomplete, incorrect or include features that
are considered implementation detail and may vary between Python
implementations. When in doubt, consult the module reference at the
location listed above.

CLASSES
builtins.object
    date
        datetime
    time
    timedelta
    tzinfo
        timezone

class date(builtins.object)
    date(year, month, day) --> date object

    Methods defined here:
```

模块

交互式

定义函数时如何采用交互式方式

例4:

```
def mean(numbers):
```

```
    s = 0.0
```

```
    for num in numbers:
```

```
        s = s + num
```

```
    return s / len(numbers)
```

```
nums = [0,1,2,3,4,5,6,7,8,9]
```

```
print(mean(nums))
```

```
>>> def mean(numbers):
    s = 0
    #numbers = [0,1,2,3,4,5,6,7,8,9]
    for num in numbers:
        s = s + (num)
    return s/len(numbers)
```

```
>>> nums = [0,1,2,3,4,5,6,7,8,9]
```

```
>>> print(mean(nums))
```

```
4.5
```

```
>>>
```

```
>>> def my_abs(x):
...     四个缩进if x >=0:
...         八个缩进return x
...     else:
...         return -x
```

```
>>> my_abs(-100)
```

```
100
```

```
>>> my_abs(99)
```

```
99
```



1.4Python开发环境配置



- 到Python主页下载并安装Python基本开发和运行环境，网址：www.python.org/downloads/
- 根据操作系统不同选择不同版本
- 下载相应的Python 3.0系列版本程序



Donate



Search

GO

Social

About

Downloads

Documentation

Community

Success Stories

News

Events

Download the latest version for Windows

Download Python 3.11.3

Looking for Python with a different OS? Python for [Windows](#),
[Linux/UNIX](#), [macOS](#), [Other](#)

Want to help test development versions of Python? [Prereleases](#),
[Docker images](#)



Active Python Releases

For more information visit the [Python Developer's Guide](#).

| Python version | Maintenance status | First released | End of support | Release schedule |
|----------------|--------------------|----------------|----------------|-------------------------|
| 3.11 | bugfix | 2022-10-24 | 2027-10 | PEP 664 |
| 3.10 | bugfix | 2021-10-04 | 2026-10 | PEP 619 |
| 3.9 | security | 2020-10-05 | 2025-10 | PEP 596 |
| 3.8 | security | 2019-10-14 | 2024-10 | PEP 569 |
| 3.7 | security | 2018-06-27 | 2023-06-27 | PEP 537 |

安装



```
C:\Users\PC>path
PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;D:\Program Files\Graphviz\bin;D:\Users\PC\AppData\Local\Programs\Python\Python310\Scripts\;D:\Users\PC\AppData\Local\Programs\Python\Python310\;C:\Users\PC\AppData\Local\Microsoft\WindowsApps;

C:\Users\PC>python
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> exit()

C:\Users\PC>.
```

```
C:\Users\PC>path
PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;D:\Program Files\Graphviz\bin;D:\Users\PC\AppData\Local\Programs\Python\Python310\Scripts\;D:\Users\PC\AppData\Local\Programs\Python\Python310\;C:\Users\PC\AppData\Local\Microsoft\WindowsApps;
```



启动

- 方法1：启动Windows命令行工具，输入python

```
C:\Users\PC>python
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> ■
```



启动

■ 方法2：调用IDLE来启动Python图形化运行环境

```
IDLE Shell 3.10.7
File Edit Shell Debug Options Window Help
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello,World")
Hello,World
>>>
```

启动

- 在IDLE环境下，除了撤销（Ctrl+Z）、全选（Ctrl+A）、复制（Ctrl+C）、粘贴（Ctrl+V）、剪切（Ctrl+X）等常规快捷键之外，其他比较常用的快捷键如下表所示。

| 快捷键 | 功能说明 |
|-----------|--|
| Alt+p | 浏览历史命令（上一条） |
| Alt+n | 浏览历史命令（下一条） |
| Ctrl+F6 | 重启Shell，之前定义的对象和导入的模块全部失效 |
| F1 | 打开Python帮助文档 |
| Alt+/ | 自动补全前面曾经出现过的单词，如果之前有多个单词具有相同前缀，则在多个单词中循环选择 |
| Ctrl+] | 缩进代码块 |
| Ctrl+[| 取消代码块缩进 |
| Alt+3 | 注释代码块 |
| Alt+4 | 取消代码块注释。 |
| Tab | 补全单词 |

启动

- 在IDLE中，如果使用交互式编程模式，那么直接在提示符“>>>”后面输入相应的命令并回车执行即可，如果执行顺利的话，马上就可以看到执行结果，否则会抛出异常。

```
>>> 3+4
```

```
7
```

```
>>> import math
```

```
>>> math.sqrt(16)
```

```
4.0
```

```
>>> 3*(2+6)
```

```
24
```

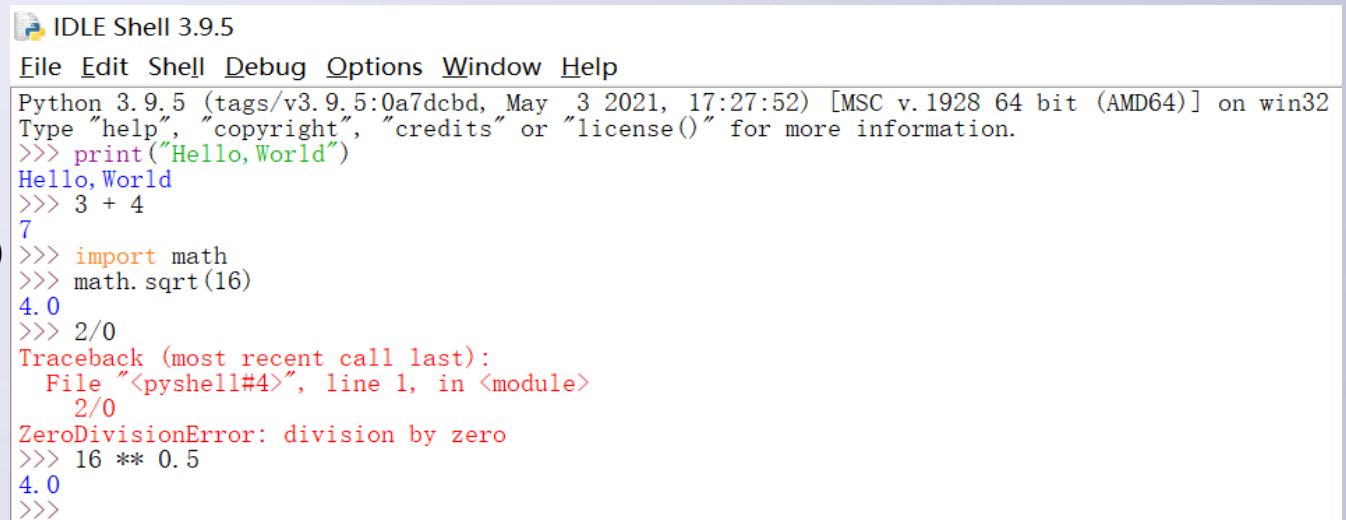
```
>>> 2/0
```

```
Traceback (most recent call last):
```

```
File "<pyshell#18>", line 1, in <module>
```

```
2/0
```

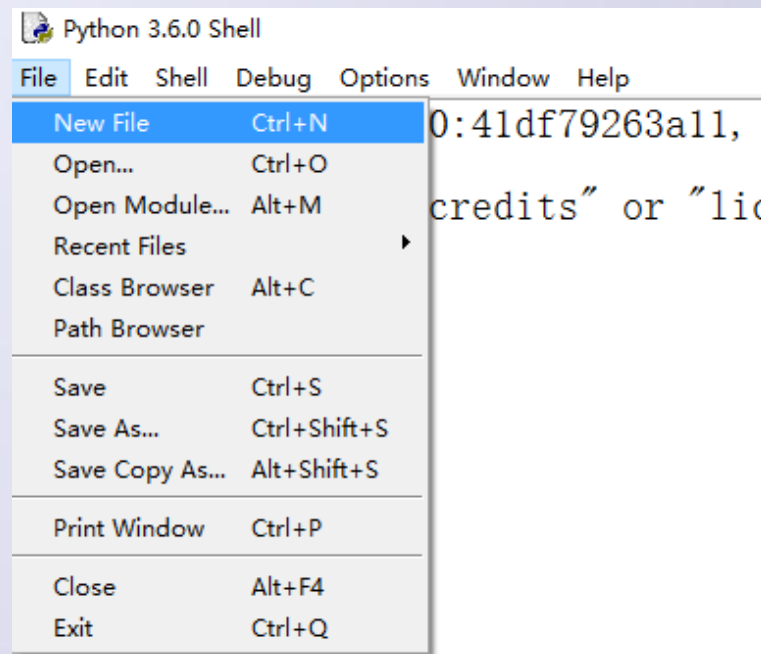
```
ZeroDivisionError: integer division or modulo by zero
```



```
IDLE Shell 3.9.5
File Edit Shell Debug Options Window Help
Python 3.9.5 (tags/v3.9.5:0a7dcbbd, May 3 2021, 17:27:52) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello,World")
Hello,World
>>> 3 + 4
7
>>> import math
4.0
>>> math.sqrt(16)
4.0
>>> 2/0
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    2/0
ZeroDivisionError: division by zero
>>> 16 ** 0.5
4.0
>>>
```

启动

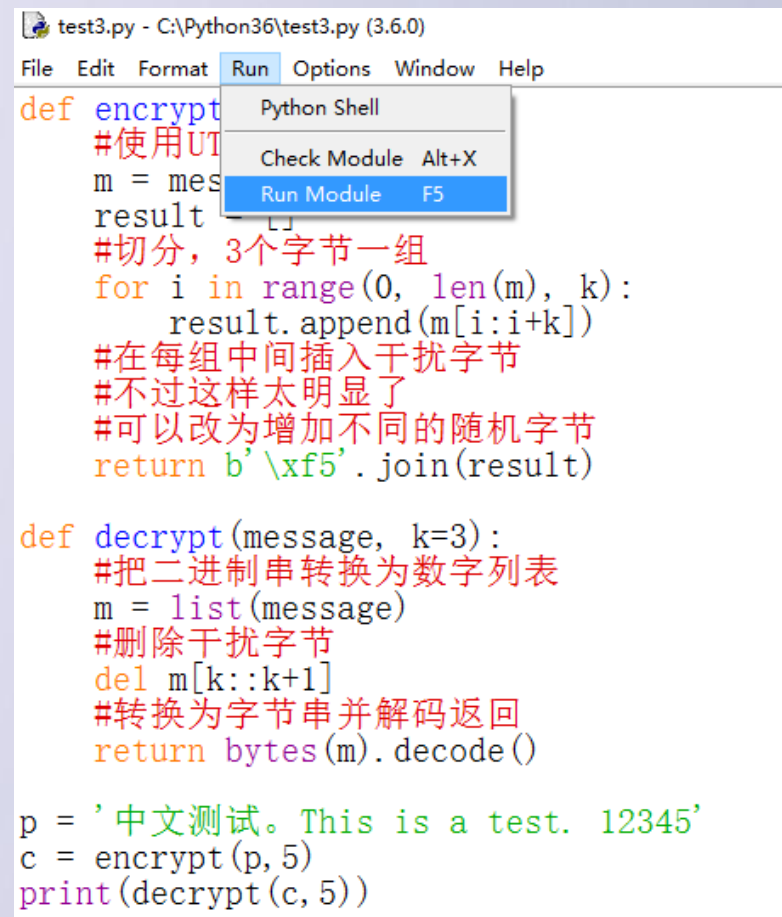
- 在IDLE界面中使用菜单“File” ==> “New File” 创建一个程序文件，输入代码并保存为.py文件。



启动

- 使用菜单

“Run” ==> “Check Module” 来检查程序中是否存在语法错误，或者使用菜单 “Run” ==> “Run Module” 运行程序，程序运行结果将直接显示在IDLE交互界面上。



```
test3.py - C:\Python36\test3.py (3.6.0)
File Edit Format Run Options Window Help
def encrypt(m):
    #使用UTF-8编码
    m = mes
    result = []
    #切分，3个字节一组
    for i in range(0, len(m), k):
        result.append(m[i:i+k])
    #在每组中间插入干扰字节
    #不过这样太明显了
    #可以改为增加不同的随机字节
    return b'\xf5'.join(result)

def decrypt(message, k=3):
    #把二进制串转换为数字列表
    m = list(message)
    #删除干扰字节
    del m[k::k+1]
    #转换为字符串并解码返回
    return bytes(m).decode()

p = '中文测试。This is a test. 12345'
c = encrypt(p, 5)
print(decrypt(c, 5))
```



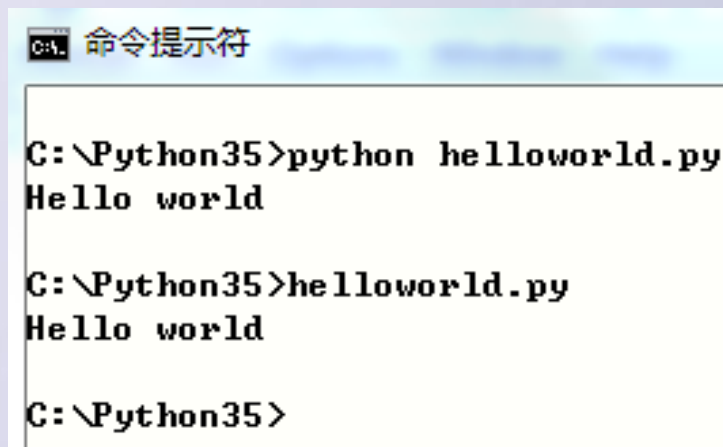

- 方法3：按照语法格式编写代码，编写可以用任何文本编辑器，保存为文件。

```
E:\>python hello.py  
Hello World
```

启动

- 在有些情况下可能需要在命令提示符环境中运行Python程序文件。在“开始”菜单的“附件”中单击“命令提示符”，然后执行Python程序。假设有程序HelloWorld.py内容如下。

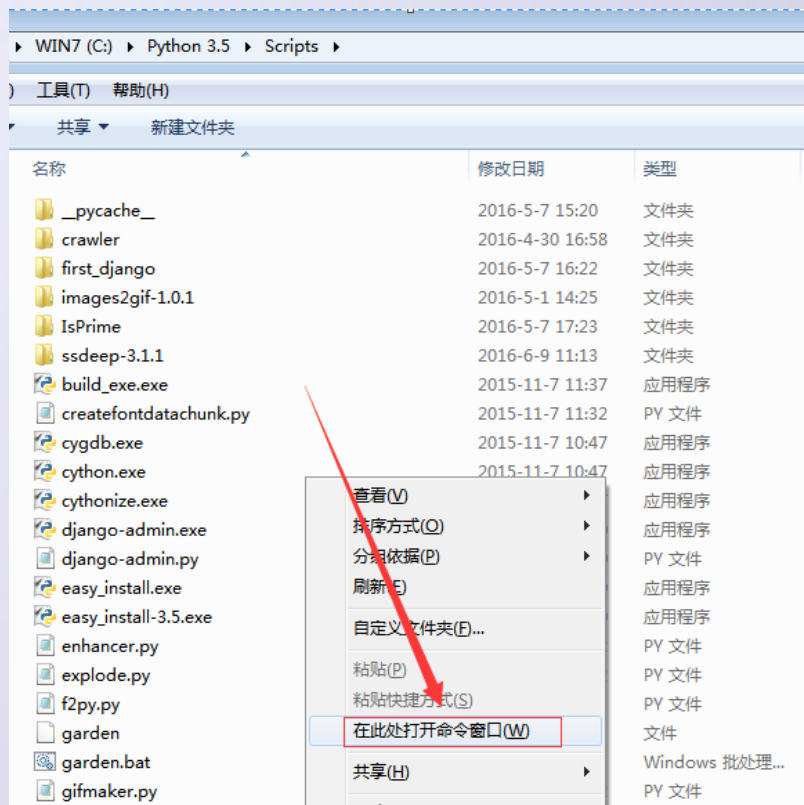
```
def main():  
    print('Hello world')  
main()
```



```
C:\Python35>python helloworld.py  
Hello world  
  
C:\Python35>helloworld.py  
Hello world  
  
C:\Python35>
```

启动

- 可以在资源管理器中切换至相应的文件夹并直接进入命令提示符环境。





- 方法4：打开IDLE，点击Ctrl+N打开一个新窗口，
输入语句并保存，使用快捷键F5即可运行该程序



- 方法5：将Python集成到Eclipse、PyCharm等面向较大规模项目开发的集成开发环境中

使用pip管理第三方包

```
Usage:
  pip <command> [options]

Commands:
  install           Install packages.
  download          Download packages.
  uninstall         Uninstall packages.
  freeze           Output installed packages in requirements format.
  list             List installed packages.
  show             Show information about installed packages.
  check            Verify installed packages have compatible dependencies.
  config           Manage local and global configuration.
  search           Search PyPI for packages.
  cache           Inspect and manage pip's wheel cache.
  wheel           Build wheels from your requirements.
  hash            Compute hashes of package archives.
  completion       A helper command used for command completion.
  debug           Show information useful for debugging.
  help            Show help for commands.

General Options:
  -h, --help           Show help.
  --isolated           Run pip in an isolated mode, ignoring environment variables and user configuration.
  -v, --verbose        Give more output. Option is additive, and can be used up to 3 times.
  -V, --version        Show version and exit.
  -q, --quiet          Give less output. Option is additive, and can be used up to 3 times (corresponding to
                        WARNING, ERROR, and CRITICAL logging levels).
  --log <path>        Path to a verbose appending log.
  --no-input           Disable prompting for input.
  --proxy <proxy>      Specify a proxy in the form [user:passwd@]proxy.server:port.
  --retries <retries> Maximum number of retries each connection should attempt (default 5 times).
  --timeout <sec>      Set the socket timeout (default 15 seconds).
  --exists-action <action> Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup,
                        (a)bort.
  --trusted-host <hostname> Mark this host or host:port pair as trusted, even though it does not have valid or any
                        HTTPS.
  --cert <path>        Path to PEM-encoded CA certificate bundle. If provided, overrides the default. See 'SSL
                        Certificate Verification' in pip documentation for more information.
```

```
C:\Users\liushuo>pip --version
pip 21.1.1 from d:\python39\lib\site-packages\pip (python 3.9)
```

```
C:\Users\liushuo>pip -V
pip 21.1.1 from d:\python39\lib\site-packages\pip (python 3.9)
```

```
C:\Users\liushuo>py -m ensurepip --upgrade
Looking in links: c:\Users\liushuo\AppData\Local\Temp\tmp2tnxf_75
Requirement already satisfied: setuptools in d:\python39\lib\site-packages (56.0.0)
Requirement already satisfied: pip in d:\python39\lib\site-packages (21.1.1)
```

使用pip管理第三方包

■ pip工具常用命令

| pip命令示例 | 说明 |
|---|--------------------------------|
| <code>pip download SomePackage[==version]</code> | 下载扩展库的指定版本，不安装 |
| <code>pip freeze [> requirements.txt]</code> | 以requirements的格式列出已安装模块 |
| <code>pip list</code> | 列出当前已安装的所有模块 |
| <code>pip install SomePackage[==version]</code> | 在线安装SomePackage模块的指定版本 |
| <code>pip install SomePackage.whl</code> | 通过whl文件离线安装扩展库 |
| <code>pip install package1 package2 ...</code> | 依次（在线）安装package1、package2等扩展模块 |
| <code>pip install -r requirements.txt</code> | 安装requirements.txt文件中指定的扩展库 |
| <code>pip install --upgrade SomePackage</code> | 升级SomePackage模块 |
| <code>pip uninstall SomePackage[==version]</code> | 卸载SomePackage模块的指定版本 |

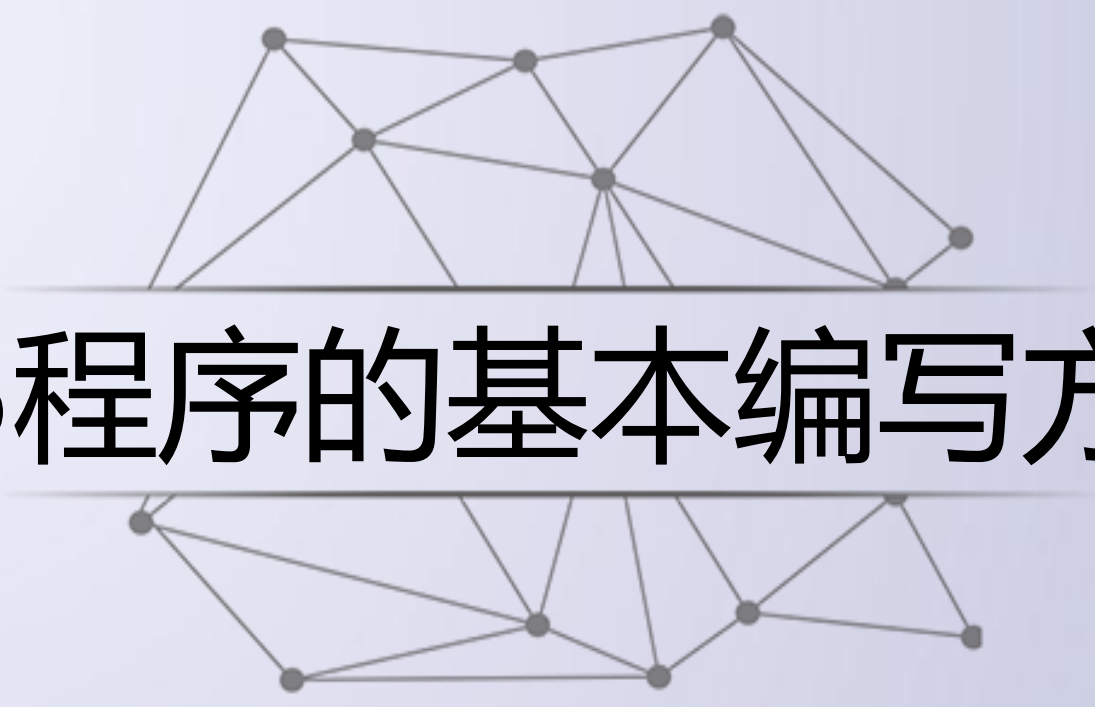
```
C:\Users\PC>pip list
```

| Package | Version | Editable project location |
|---------------------|-----------|---------------------------|
| abs1-py | 1.4.0 | |
| altgraph | 0.17.3 | |
| apriori | 1.0.0 | |
| apyori | 1.1.2 | |
| astunparse | 1.6.3 | |
| attrs | 22.1.0 | |
| automat | 20.2.0 | |
| av | 9.2.0 | |
| beautifulsoup4 | 4.11.1 | |
| biopython | 1.79 | |
| bs4 | 0.0.1 | |
| cachetools | 5.3.0 | |
| certifi | 2022.9.24 | |
| charset-normalizer | 2.1.1 | |
| click | 8.1.3 | |
| click-default-group | 1.2.2 | |
| cloup | 0.13.1 | |
| colorama | 0.4.5 | |
| colour | 0.1.5 | |
| commonmark | 0.9.1 | |
| constantly | 15.1.0 | |
| contourpy | 1.0.5 | |
| cycler | 0.11.0 | |

pip install -i <https://pypi.tuna.tsinghua.edu.cn/simple> tensorflow

Read timedout error

pip --default-timeout=100 install -i <https://pypi.tuna.tsinghua.edu.cn/simple> tensorflow



1.5程序的基本编写方法



IPO程序编写方法

- 输入数据-Input
- 处理数据-Process
- 输出数据-Output



IPO程序编写方法

■ 输入数据

输入（Input）是一个程序的开始。程序要处理的数据有多种来源，形成了多种输入方式，包括：文件输入、网络输入、控制台输入、交互界面输出、随机数据输入、内部参数输入等。



IPO程序编写方法

■ 处理数据

处理（Process）是程序对输入数据进行计算产生输出结果的过程。计算问题的处理方法统称为“算法”，它是程序最重要的组成部分。可以说，算法是一个程序的灵魂。



IPO程序编写方法

■ 输出数据

输出（Output）是程序展示运算成果的方式。程序的输出方式包括：控制台输出、图形输出、文件输出、网络输出、操作系统内部变量输出等。



IPO程序编写方法

■ 微实例1.1圆面积的计算

输入：圆半径radius


处理：计算圆面积 $area = \pi * radius * radius$ （此处， π 取3.1415）

输出：圆面积area



使用计算机解决问题

- 分析问题-分析问题的计算部分
- 划分边界-划分问题的功能边界
- 设计算法-设计问题的求解算法
- 编写程序-编写问题的计算程序
- 调试测试-调试和测试程序
- 升级维护-适应问题的升级维护



1.6Python语言版本更迭



Python语言的版本更迭

- 更高级别的3.0系列不兼容早期2.0系列
- 2008年至今，版本更迭带来大量库函数的升级替换，Python语言的版本更迭痛苦且漫长
- 到今天，Python 3.x系列已经成为主流

- 小练习

猜一猜用以下语句会得到什么结果？

```
myboss = '吉多'
```

```
yourboss = myboss
```

```
yourboss = '吉姆'
```

```
print(myboss)
```



- 小练习2

猜一猜用以下语句会得到什么结果？

```
myboss = '吉多'
```

```
yourboss = myboss
```

```
myboss = '吉姆'
```

```
print(yourboss)
```



- 小练习3

猜一猜用以下语句会得到什么结果？

```
mynumber = 666
```

```
yourword = '666'
```

```
mynumber = yourword
```

```
print(mynumber)
```



- 下面不是IPO模式的一部分的是（）
A.Input B.Program C.Process D.Output



本章小结

本章具体讲解了计算机的基本定义、计算机的功能性和可编程性、程序设计语言分类、编译和解释、Python语言的历史和发展、配置Python开发环境等内容，最后给出了Python版本的主要区别供参考。