第 0 讲　编程语言的基本结构与思路

这一讲的目的，一是帮助那些难以理解编程的同学理解编程到底在干什么，是怎么一回事。由于作者自己并不是科班出身，对编程的理解难免不够深入，但希望能减轻这些同学们的痛苦。二是从考试的角度提纲性地说明一下如何准备 python 的考试（尤其是理论考）

一　编程语言的基本结构

1. 编程是对数据的处理

· **计算机能够实现各种各样的功能，完成各种各样的任务**

原因在于，所有这些计算机的功能，背后都是在对海量数据进行计算处理（万物皆可是数据）

· **编程，即编写程序，也就是告诉计算机如何处理指定的数据**

计算机本身是机器，只能读懂电信号，于是大佬们开发了编程语言，让人们可以用人能看懂的字符来命令计算机做事。（简单来讲就是你用人类语言符号编写的程序会被大佬的程序转换成电信号给计算机）

· **实际上，计算机很笨，它只能做一些**非常基础的操作（如将两个数相加）

而神奇之处在于，**所有对数据的处理过程都能够分解成一连串基础操作**

→ 虽然“一连串”可能意味着非常多的操作，但计算机虽然笨却算得快，并且不会累。

· 上面这件事也意味着，**人们不能直接命令计算机去做一件复杂的事**，必须要先嚼碎了，把它**分解成基本操作**，再命令计算机去做。因此，我们学习编程主要有两个任务：

→ **知道计算机能做哪些基本操作**（也就是编程语言能提供哪些功能）

→ **如何将一件复杂的事情分解为基本操作**（也就是算法）

2. python 编程语言中的基本概念

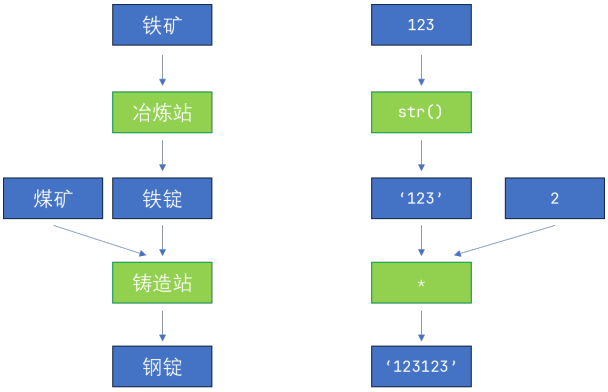
① python 将所有数据类型统一称为**对象**，好比工厂中的各个加工阶段的原料、中间产物、产品

而处理数据的操作有很多，包括**运算符**、**函数**、**方法**等，好比工厂里功能各不相同的加工机器

> 本讲义的 1、2 两讲将会涵盖本课程涉及的几种基本对象（数据类型），包括
> · 这些数据类型在编程语言中如何表示
> · 常见**运算符**会将它们处理成什么
> · 这些数据类型适用的**函数**和自带的**方法**

② 对象和运算符、函数、方法组成的代码称为**表达式**，其特点是**可以被运算，最终得到一个值**

> · 本讲义的第 4、5 讲会涉及函数以及输入输出相关的表达式



工厂生产和数据处理具有高度的相似性

③ 为了丰富数据处理流程，实现更多的功能，编程语言还提供了**语句**

语句的功能包括**数据的临时保存**（赋值语句）、**复杂流程**（分支、循环）、**异常处理**等

> · 本讲义的第 3 讲涵盖以上几种语句类型

3. 一般程序的基本结构

· 程序由多行代码组成，每一行代码都代表一个数据处理步骤

在不包含复杂结构的前提下，计算机将**从上到下依次执行**程序每一行代表的处理步骤

若包含分支、循环等结构，程序可能会跳段或者重新回到某处，但一定是从上到下执行到最后的

若包含函数，程序会跳转到其它代码从上到下执行，并在那段代码执行结束后回到一开始跳转出去的地方，继续向下执行。

因此总体上讲，程序大体上一定是**从上到下依次执行到末尾**

· 本课程涉及的简单程序主要结构为 **输入数据 → 处理数据 → 输出数据**

二 《python 程序设计》的学习方法

如果你是想真正学会如何用 python 写程序

· 立即放下这份讲义，到网上找教程自学，跟着写一些有实际意义的程序或项目
一定要多练习，编程是写出来的，不是看书看出来的

· 除了学习语言本身和算法以外，一定要学习代码风格与规范，写出清爽简洁的代码
（不要一开始就养成堆 shit 山的习惯）

· 以及还要学习如何调试（debug），积累找 bug 修 bug 的能力
没有人能一口气写出完全正确的代码，擅长解决 bug 的人才是真的强

· 语法的细枝末节不需要记，实战其实很难用到，真用到的时候也没人会阻止你查资料（问 GPT）

1. 实验考准备方法

· 据身边统计学，《python 程序设计实验考》挂科的主要原因包括：

看到题目，毫无思路，无从下手

明知有 bug 但怎么找也找不到

· 实际上，实验考的题目都很简单，不会超出平时作业的难度（除非不布置作业），因此：

多做题，多练习，在过程中总结常见的算法（比如进制转换、找质数等），知道怎么解决它们

培养好的代码格式规范（与分数无直接关联，但**简洁、清爽、有序的代码可以帮助你更高效地找到 bug**）

学会使用 IDE（如 pycharm），学会调试，积累找 bug 的经验

· 这份讲义目前还没有涉及到实验考，实在抱歉

2.**我想要拿下理论考**

· 据身边统计学，《python 程序设计理论考》低分的主要原因包括：

看到大段代码（通常格式混乱）无从下手，根本不知道它在干什么

考查非常细非常坑的知识点，自己恰好没背或粗心

· 因此，理论考需要掌握的技能与实验考截然不同：

熟读本讲义，在脑海中构建 python 语法知识的网络

学会分析代码：永远将代码看成数据处理链，搞清楚各个环节数据被处理成了什么类型、什么内容

有时也要将一块代码视为总体，看这一块代码对数据做了什么处理步骤

多做理论考试练习，整理错题，了解出题老师喜欢在什么地方挖坑（以及熟悉凌乱的代码风格）

不管你在大学前是否接触过计算机、是否接触过编程、高考是否选择了技术、是否参加过竞赛，不管你是理工农医还是人文社科，你都有能力学会编程。只要你认真对待 python 这门课，付出那么一点点时间，你必能拿下它。学校的课程可能不尽人意，但还是希望你能在编程中找到乐趣，小有所成