Lesson1:程序的基本构造

我们为什么需要写一个程序

- 我们厌倦了一些重复性的工作，比如统计班级数据、重复用账号密码登陆某些网站，我们希望用一个脚本来解决问题
- 我们希望连接不同的工具甚至设备，让用户可以不必了解工具链上的每一环而以相对方便的方式甚至一键解决问题

一个程序需要什么

- 一些获取数据的方式
- 一些存储数据的结构
- 一些处理数据的方法
- 一条数据流动的管线
- 一个启动程序的入口
- 其他与外界交互的组件

一个微型项目的一个组件的样子

```
newsSpider/

spider/ 爬虫文件夹

government.py 中国政府网爬虫代码

tencent.py 腾讯新闻的爬虫代码

utils/ 工具文件夹

location/ 地理编码工具

nlp/ 自然语言处理工具

SimhashUtils.py Simhash编码工具

logUtils.py 日志模块

sqlUtils.py 数据库模块
```

```
to_redis.py 生成url

timeUtils.py 时间结构化模块

timer.py 定时任务

Structure/ 结构化代码

gov_data_filter.py 中国政府网的结构化代码

tencent_data_filter 腾讯新闻的结构化代码

main.py 主函数,决定运行哪个爬虫
```

- 抽象的过程:现实情景->抽象逻辑->抽象函数 + 抽象数据结构
- 实现的过程:组织函数并实现函数 + 组织数据并存储数据

封装

- 我们不希望每做一件事情都需要关注与这件事相关的所有细节，所以我们会把不同抽象层级的操作层层封装。
- 以开车为例，操作者只需要关注方向盘、档位、油门、刹车，而不需要关注变速箱、发动机、轮轴等具体是怎么工作的。而换一个更高的抽象层级，如考虑一次行程时，我们不需要关注具体怎样开车，只需要关注起点、终点、交通工具等信息，有关如何开车的信息在这一层级将被隐藏。
- 程序设计也是同理，我们会把一个大任务拆分出不同层级，将每个层级的内容封装进一个模块，外部只需要关注如何调用这些模块即可，这让程序变得简洁与有序

我们的主线

数据:

- 储存数据: 数据类型、变量及其作用域、文件
- 组织数据: 复合数据类型、数据结构、类
- 数据流动: IO，参数传递，所有权

函数:

- 定义函数
- 调用函数
- 组织函数

- 有关函数的函数