SSPU 高级程序语言设计笔记

2025 年 10 月 11 日-2026 年 1 月 7 日

Contents

1　Python 语言概述 (2025 年 10 月 15 日)

2　基本数据类型 (2025 年 10 月 22 日)

3　程序控制结构 (2025 年 10 月 29 日)

4　组合数据类型 (2025 年 11 月 5 日)

5 函数与模块 (2025 年 11 月 12 日)

5.1 函数的变量作用域

基本概念：
1. 局部变量：函数体内声明的
2. 全局变量：函数体外声明的
在函数体内定义全局，使用 global

5.2 模块

import 模块调用为模块. 函数名
from 模块 import 函数调用为直接函数名
导入所有函数为 from 模块 import * 调用为直接函数名

5.3 函数装饰器

输入为函数，输出也为函数

5.4 类

类相当于函数的集合，不过类是一群带有性质的函数的集合

5.4.1 数据成员 2025 年 11 月 26 日

首先可以简要理解类与成员的关系为工厂与产品的关系，类是工厂，成员是产品。
类可以通过某种形式来生成对象，就像工厂通过某种生产规则生产出产品一样。

写一段代码理解一下：

Listing 1: 类的数据成员示例

```
class Student:
  school = "SSPU"

  def __init__(self, name, age):
    self.name = name
    self.age = age
  def display_info(self):
    print(f"姓名:{self.name},年龄:{self.age},学校:{self.school}")
student1 = Student("张三",20)
student2 = Student("李四",22)
student1.display_info()  # 输出：姓名：张三，年龄：20，学校：SSPU
student2.display_info()  # 输出：姓名：李四，年龄：22，学校：SSPU
```

注意：类变量是所有实例共享的，而实例变量是每个对象独有的。

Listing 2: 私有变量示例

```
  class BankAccount:
    def __init__(self,balance):
      self._balance = balance
    def get_balance(self):
      return self._balance
    def deposit(self,amount):
      self._balance += amount
  account=BankAccount(1000)
  print(account.get_balance()) #正确方式，可以输出1000
  print(account._balance) #错误方式，私有变量无法访问
```

在这里，Student 就是一个类，school = ”SSPU” 是全体实例的属性 (也就是都是二工大的学生)

这类似于一个模版，然后具体输入张三和李四，就会生成二工大学生张三以及李四的这样的信息。

而这里的私有变量就相当于学生的隐私，比如张三或者李四的银行账户与存款等等

5.5 基类和派生类

Listing 3: 基类和派生类示例

```
class Student:    #创建学生基类
  def __init__(self,name,number):
    self.name = name
    self.number = number

  def do_homework(self):
    print("学生完成作业")

class G_student(Student):     #研究生派生类
  def __init__(self,name,number,advisor):
    super().__init__(name,number)
    self.advisor = advisor

  def do_homework(self):
    print("研究生作研究")

class U_student(Student):   #本科派生类
  def __init__(self,name,number,teacher):
    super().__init__(name,number)
    self.teacher = teacher

  def do_homework(self):
    print("本科生找工作")

def homework(student): #多态函数
  student.do_homework()
g_student=G_student("张三","2025001","王教授") #研究生张三，学号2025001，导师王教授
u_student=U_student("李四","2025002","李老师") #本科生李四，学号2025002，授课老师李老师

homework(g_student) #调用多态函数，输出：研究生学生完成作业 （做研究）
homework(u_student) #调用多态函数，输出：大学本科生完成作业 （找工作）
```

多态函数的好处在于老师不需要知道每个阶段的学生具体类型，只需要调用 *homework* 函数即可，这样的话不同阶段的学生会做它们自己的任务。

6 我在学习 Python 过程中出现的错误（2025 年 12 月 3 日）

2025-12-03 12:00 可见空格符问题：代码块每行前有多余空格，已在导言区添加配置保留缩进但不渲染可见空格。

2025-12-03 12:02 构造函数名写错：把 _init_ 写成了 __init__（少一个下划线）。

2025-12-03 12:05 继承语法错误：类继承写法不正确，应为 class G_student(Student):。

2025-12-03 12:06 中文引号问题：使用了中文引号，应使用标准 ASCII 双引号。

2025-12-03 12:08 代码块嵌套错误：类定义位置不正确。

2025-12-03 12:10 注释与输出不一致：建议同步注释与实际输出。