

# SSPU 高级程序设计笔记

2025 年 10 月 11 日-2026 年 1 月 7 日

## Contents

<b>1 Python 语言概述 (2025 年 10 月 15 日)</b>	<b>2</b>
<b>2 基本数据类型 (2025 年 10 月 22 日)</b>	<b>3</b>
<b>3 程序控制结构 (2025 年 10 月 29 日)</b>	<b>4</b>
<b>4 组合数据类型 (2025 年 11 月 5 日)</b>	<b>5</b>
<b>5 函数与模块 (2025 年 11 月 12 日)</b>	<b>6</b>
5.1 函数的变量作用域 . . . . .	6
5.2 模块 . . . . .	6
5.3 函数装饰器 . . . . .	6
5.4 类 . . . . .	6
5.4.1 数据成员 2025 年 11 月 26 日 . . . . .	6

# 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

```
1 class Student:
2     school = "SSPU"
3
4     def __init__(self, name, age):
5         self.name = name
6         self.age = age
7     def display_info(self):
8         print(f"姓名:{self.name}, 年龄:{self.age}, 学校:{self.school}")
9 student1 = Student("张三",20)
10 student2 = Student("李四",22)
11 student1.display_info() # 输出: 姓名: 张三, 年龄: 20, 学校: SSPU
12 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

```
1 class BankAccount:
2     def __init__(self,balance):
3         self._balance = balance
4     def get_balance(self):
5         return self._balance
6     def deposit(self,amount):
7         self._balance += amount
8 account=BankAccount(1000)
9 print(account.get_balance()) #正确方式，可以输出1000
10 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