教案纸

第一讲

第一章 Python 语言初体验

课程时长：2 学时

教学目标

1. 了解 Python 语言的历史和发展。
2. 掌握 Python 的安装过程。
3. 熟悉 Python 的开发环境，包括 IDLE 的使用和 pip 工具的安装与使用。
4. 能够编写并执行简单的 Python 程序。

教学内容

1.1 Python 语言概述（10 分钟）

1. Python 的历史和发展。
    - Guido van Rossum 创建 Python 的背景。
    - Python 2.x 到 Python 3.x 的演进。
2. Python 语言的特点。
    - 简洁易读的语法。
    - 面向对象编程。
    - 可移植性。
    - 解释型语言。
    - 开源。
3. Python 语言的应用领域。
    - Web 开发。
    - 数据科学。
    - 人工智能与机器学习。

教案纸

- 自动化运维。
- 游戏开发。
- 桌面应用与网络编程。

1.2、Python 的安装（15 分钟）

1. 访问 Python 官方网站下载 Python 安装包。
2. 选择适合操作系统的 Python 版本进行下载。
3. 安装 Python，注意勾选“Add Python to PATH”选项。
4. 安装完成后验证 Python 是否安装成功（通过命令行输入 `python --version`）。

1.3、Python 的开发环境（40 分钟）

1. IDLE 集成开发环境介绍。
   - 启动 IDLE。
   - 交互式编程与文件式编程的区别。
   - 在 IDLE 中创建、编辑和运行 Python 程序。
2. pip 工具的使用。
   - pip 工具简介。
   - 使用 pip 命令安装第三方库（如 requests 库）。
   - 使用 pip 命令查看已安装的库列表。
   - 使用 pip 命令升级和卸载库。
3. 第一个 Python 程序。
   - 在 IDLE 中编写并运行打印“Hello, World!”的程序。
   - 解释程序执行过程和结果。

课堂练习与讨论（15 分钟）

1. 学生自行安装 Python 并验证安装是否成功。
2. 学生使用 IDLE 编写并运行一个简单的 Python 程序（如计算两个数的和）。

教案纸

3. 讨论 Python 语言的特点和应用领域，分享个人对 Python 的第一印象。

课后作业

1. 复习 Python 语言的历史和特点。
2. 使用 pip 工具安装一个感兴趣的 Python 第三方库（如 NumPy 或 pandas），并简单了解其用途。
3. 编写一个 Python 程序，实现以下功能：
    - 从用户输入中获取两个整数。
    - 计算并输出这两个整数的和、差、积和商（除不尽时保留两位小数）。

教学评估

- 通过课堂练习和讨论评估学生对 Python 安装和开发环境的掌握情况。
- 通过课后作业评估学生对 Python 语言特点和第三方库安装的理解以及编程实践能力。

教案纸

第二讲

第二章 Python 基本语法概述

课程时长：2 学时

教学目标

1. 掌握 Python 语言的编程规范，包括缩进、注释等。
2. 理解常量、变量与对象在 Python 中的概念和应用。
3. 熟悉 Python 中常用的数据类型，包括数字、字符串、列表、元组、字典和集合。

教学内容

2.1 Python 语言的编程规范

一、代码缩进

- 介绍 Python 中缩进的重要性，解释为什么缩进是强制性的。
- 演示如何正确使用 4 个空格进行缩进，并强调不要混用空格和制表符。

二、注释

- 解释注释的作用，提高代码的可读性和可维护性。
- 演示单行注释和多行注释的写法，如使用#和三个引号（'''或"""）。

2.2 常量、变量与对象

一、常量

- 介绍常量的概念，说明 Python 中并没有真正的常量类型，但可以使用约定俗成的方式表示常量（如全大写字母）。

二、变量

- 解释变量的定义和作用，说明 Python 中不需要事先声明变量类型。
- 演示如何给变量赋值，并改变变量的值。

三、对象

- 介绍 Python 中一切皆对象的概念，解释对象的标识、类型和值。

教案纸

- 通过示例演示变量如何引用对象，并解释可变对象和不可变对象的区别。

2.3 数据类型

一、数字类型

- 介绍整数（int）、浮点数（float）和复数（complex）的表示方法和运算规则。
- 演示数字类型的常见操作，如加、减、乘、除等。

二、字符串类型

- 解释字符串类型的定义和作用，介绍字符串的定界符（单引号、双引号、三引号）。
- 演示字符串的常见操作，如连接、切片、查找等。

三、列表类型

- 介绍列表的定义和作用，说明列表是有序的元素集合。
- 演示列表的创建、修改和常用操作（如追加、删除元素）。

四、元组类型

- 解释元组的定义和作用，说明元组是不可变的有序元素集合。
- 演示元组的创建和使用。

五、字典类型

- 介绍字典的定义和作用，说明字典是无序的键值对集合。
- 演示字典的创建、访问和修改。

六、集合类型

- 解释集合的定义和作用，说明集合是无序的唯一元素集合。
- 演示集合的创建和基本操作（如并集、交集）。

课堂练习

1. 编写代码，演示正确的缩进和注释使用。
2. 定义一个变量和一个常量，并输出它们的值。
3. 创建一个包含不同数据类型的列表、元组、字典和集合，并进行相关操作。

教案纸

作业布置

1. 复习 Python 的编程规范，确保在编写代码时严格遵守。
2. 编写一个程序，使用列表和字典存储学生信息，并允许用户通过姓名查询学生信息。
3. 尝试使用集合解决一个实际问题，如找出两个列表中的共同元素。

教案纸

第三讲

《Python 基本语法概述》教案 - 2.4 Python 的语句概述 & 2.5 输入输出函数

课程时长：2 学时

教学目标

1. 理解 Python 中表达式和语句的概念及其关系。
2. 掌握赋值语句的多种形式和用法。
3. 学会使用 import 语句导入模块和对象。
4. 理解控制流语句（特别是 if 和 for 语句）的基本概念。
5. 熟悉输入输出函数 input()和 print()的用法。

教学内容

2.4 Python 的语句概述

一、表达式

教案纸

- 解释表达式的定义和组成元素（数字、变量、运算符、括号、字面量）。
- 示例演示表达式与语句的区别。

二、赋值语句

- 解释赋值语句的基本形式和用法。
- 演示多重赋值、增强赋值、链式赋值和解构赋值的示例。

三、引入语句（import 语句）

- 介绍 import 语句的三种格式及其用法。
- 示例演示标准库 math 和 random 的导入和使用。

四、控制语句

- 简要介绍选择结构和循环结构。
- 示例演示 if 语句和 for 语句的用法。

2.5 输入输出函数

一、input()函数

- 解释 input()函数的作用和用法。
- 示例演示 input()函数接收用户输入，并处理为字符串类型。
- 引入 eval()函数与 input()函数的结合使用，以处理数值输入。

二、print()函数

- 解释 print()函数的作用和用法。
- 演示 print()函数输出单个值、多个值、指定分隔符和结束符的用法。
- 介绍 print()函数将输出重定向到文件的用法。

课堂练习

1. 编写代码，使用赋值语句给变量赋值，并进行简单的算术运算。
2. 编写代码，使用 if 语句判断用户输入的分数等级（如 A、B、C 等）。
3. 编写代码，使用 for 循环打印数字 1 到 10。

教案纸

4. 编写一个程序，使用 input()函数接收用户输入的两个数字，并使用 print()函数输出它们的和。

作业布置

1. 复习赋值语句和控制流语句的基本概念，编写一个简单的程序，根据用户输入的年龄输出对应的年龄段描述（如“儿童”、“青少年”、“成年”等）。
2. 编写一个程序，使用 input()函数接收用户输入的姓名和年龄，并使用 print()函数输出一条包含姓名和年龄的问候信息。

教学评估

通过课堂练习和作业完成情况评估学生对 Python 语句、表达式、赋值语句、引入语句、控制流语句以及输入输出函数的理解和掌握程度。

教案纸

教案纸

第四讲

《Python 基本语法概述》教案 - 2.6 turtle 库

课程时长：2 学时

教学目标

1. 了解 turtle 库的基本概念及其在图形编程中的应用。
2. 掌握 turtle 库中常用的函数和命令，用于绘制基本图形。
3. 熟悉 turtle 库的坐标系和角度控制。
4. 能够使用 turtle 库绘制简单的图形和图案。

教学内容

一、引入 turtle 库（5 分钟）

- 介绍 turtle 库的概念及其在图形编程中的作用。
- 演示如何使用 `import turtle` 语句引入 turtle 库。

二、turtle 库的坐标系（10 分钟）

- 讲解 turtle 库中的空间坐标系和角度坐标系。
- 演示如何理解和使用这两种坐标系来控制海龟的移动和旋转。
- 通过示例代码展示海龟在画布上的初始位置和初始方向。

三、turtle 的画布函数（10 分钟）

- 介绍 `screensize()`和 `setup()`函数的作用和用法。
- 演示如何设置画布的大小、位置和背景颜色。

教案纸

- 通过示例代码展示不同画布设置的效果。

四、turtle 的画笔函数（20 分钟）

1. 画笔状态函数
    - 讲解 `pendown()`, `penup()`, `pensize()`, `pencolor()`, `color()`, `begin_fill()`, `end_fill()`, `clear()`, `reset()`, `hideturtle()`, `showturtle()`, `isvisible()`, `write()` 等函数的作用和用法。
    - 演示如何使用这些函数来控制画笔的状态和颜色填充。
    - 通过示例代码展示不同画笔设置的效果。
2. 画笔运动函数
    - 讲解 `forward()`, `backward()`, `right()`, `left()`, `goto()`, `setheading()`, `circle()`, `home()`, `undo()`, `speed()`等函数的作用和用法。
    - 演示如何使用这些函数来控制海龟的移动和旋转。
    - 通过示例代码展示不同画笔运动的效果。

五、turtle 库综合实践（25 分钟）

1. 绘制红边正方形（5 分钟）
    - 讲解并演示使用 turtle 库绘制红边正方形的代码。
    - 学生动手实践，绘制自己的红边正方形。
2. 绘制实心正方形（5 分钟）
    - 讲解并演示使用 turtle 库绘制实心正方形的代码，包括填充颜色的设置。
    - 学生动手实践，绘制自己的实心正方形。
3. 绘制电脑印花图案（10 分钟）
    - 讲解并演示使用 turtle 库绘制电脑印花图案的代码，包括循环和角度控制的应用。
    - 学生动手实践，尝试绘制自己的印花图案。

教案纸

4. 绘制劳动勋章（5 分钟）
    - 讲解并演示使用 turtle 库绘制劳动勋章的代码，包括两个海龟对象的创建和协同工作。
    - 学生动手实践，尝试绘制自己的劳动勋章或其他复杂图案。

课堂小结（5 分钟）

- 总结 turtle 库在图形编程中的应用和重要性。
- 强调代码规范性和可读性的重要性。
- 鼓励学生在课后继续探索 turtle 库的其他功能和用法。

作业布置

- 要求学生使用 turtle 库绘制一个自己喜欢的图形或图案，并附上代码和解释。
- 鼓励学生尝试使用 turtle 库完成一些创意性的图形设计任务。

教案纸

第五讲

第三章 Python 的基本数据类型-3.1 数字类型&3.2 数字类型的运算

课程时长：2 学时

学习目标

1. 熟练掌握数字类型（整数、浮点数、复数）的概念及使用。
2. 熟练掌握数字类型的运算。

教学内容

一、引言（5 分钟）

- 简述 Python 中数据的重要性及数字类型在编程中的基础地位。

二、数字类型（25 分钟）

3.1 数字类型

1. 整数（int）

- 定义整数类型及其特点。
- 演示整数在不同进制下的表示方法（十进制、二进制、八进制、十六进制）。
- 解释整数对象的不可变性及 Python 中的整数缓存机制。

2. 浮点数（float）

- 定义浮点数类型及其特点。
- 演示浮点数的表示方法（普通表示法和科学计数法）。

教案纸

- 讲解浮点数比较中可能出现的问题，并介绍解决方法（如使用 `abs()`函数或 `round()`函数）。

3. 复数（complex）

- 定义复数类型及其表示方法（实部和虚部）。
- 演示复数的创建及访问实部和虚部的方法。

三、数字类型的运算（35 分钟）

3.2 数字类型的运算

1. 算术运算操作符

- 列出并解释 Python 中的算术运算操作符（+、-、*、/、//、%、**）。
- 演示操作符与赋值符号结合形成的增强赋值操作符。
- 强调数值运算可能改变数据类型的情况。

2. 数值运算函数

- 介绍与数值运算相关的内置函数（`abs()`, `divmod()`, `pow()`, `round()`, `max()`, `min()`）。
- 通过示例代码展示这些函数的用法。

课堂练习（25 分钟）

1. 编写代码，使用不同进制表示整数并进行转换。
2. 编写代码，演示浮点数比较问题的解决方法。
3. 编写代码，创建复数并进行基本运算。
4. 编写代码，使用数值运算函数（如 `abs()`, `divmod()`, `pow()`, `round()`）进行计算。
5. 编写代码，计算用户输入的两个数的算术运算结果（加、减、乘、除、整除、取余、幂运算）。

课堂小结（5 分钟）

- 总结数字类型的特点及用途。
- 强调数字类型运算在编程中的重要性。

教案纸

- 鼓励学生继续探索 Python 中其他数据类型和运算。

作业布置

1. 编写一个程序，接收用户输入的任意两个数（可以是整数、浮点数或复数），并计算它们的和、差、积、商（如果适用的话）、整除结果、余数及幂运算结果。
2. 研究 Python 中数值运算的其他高级话题，如精度控制、大数运算等。

教案纸

第六讲

第三章 Python 的基本数据类型-3.3 字符串类型&3.4 字符串类型的操作

课程时长：2 学时

学习目标

1. 掌握字符串的表示方法，包括定界符和转义字符。
2. 理解字符串的编码方式，特别是 Unicode 编码。
3. 掌握字符串的索引和切片操作。
4. 熟悉字符串的基本操作符和常用的操作函数。
5. 学会使用字符串的常用方法，如查找、替换、格式化等。

教学内容

一、引言（5 分钟）

- 简述字符串在编程中的重要性。
- 引出本节课的主题：字符串类型。

二、字符串的表示（10 分钟）

1. 字符串定界符
    - 介绍单引号、双引号、三单引号和三双引号作为字符串定界符的用法。
    - 演示不同定界符之间的嵌套使用。
2. 转义字符
    - 解释转义字符的概念和用途。

教案纸

- 列出常见的转义字符及其作用（如换行符`\n`、制表符`\t`、反斜杠`\\`等）。
- 演示转义字符在字符串中的使用。
- 介绍原始字符串（raw string）的概念和使用方法。

三、字符串的编码（5 分钟）

- 介绍 Python 3 中字符串默认采用 Unicode 编码的特点。
- 演示如何使用 `chr()`和 `ord()`函数进行 Unicode 编码值与其对应字符的转换。
- 解释字符串运算与 Unicode 编码之间的关系。

四、字符串索引和切片（10 分钟）

1. 字符串索引
   - 介绍字符串索引的概念和正向递增序号、反向递减序号的用法。
   - 演示通过索引访问字符串中特定字符的示例代码。
2. 字符串切片
   - 介绍字符串切片的概念和用法。
   - 演示切片操作的格式和参数（起始索引、结束索引、步长）。
   - 通过示例代码展示切片操作的不同效果。

五、字符串类型的操作（40 分钟）

1. 字符串操作符
   - 介绍字符串操作符（连接、重复、成员测试）的用法。
   - 演示示例代码。
2. 字符串操作函数
   - 介绍常用的字符串操作函数（如 `len()`, `str()`, `chr()`, `ord()`, `int()`, `float()`）。
   - 演示这些函数的用法和示例代码。

教案纸

3. 字符串处理方法
    - 介绍字符串的常用方法（如 `upper()`, `lower()`, `find()`, `strip()`, `split()`, `replace()`, `count()`, `center()`, `join()`）。
    - 演示这些方法的用法和示例代码。
    - 通过一个综合示例（首都单词处理）来加深理解。
4. `format()`方法
    - 介绍 `format()`方法的基本使用和格式控制。
    - 演示不同格式控制标记（填充、对齐、宽度、精度、类型）的用法和示例代码。

课堂练习（20 分钟）

- 练习字符串的基本操作，包括索引、切片、连接、替换等。
- 使用 `format()`方法格式化字符串输出。

课堂小结（5 分钟）

- 总结字符串类型的特点和重要性。
- 强调掌握字符串操作对于编程的重要性。
- 鼓励学生继续探索 Python 中字符串的其他高级用法。

作业布置

1. 编写一个程序，接收用户输入的字符串，输出该字符串的长度、第一个字符、最后一个字符、以及反转后的字符串。
2. 使用 `format()`方法格式化输出一个包含姓名、年龄、地址信息的字符串模板。

教案纸

第七讲

第三章 Python 的基本数据类型-3.5 精选案例

课程时长：2 学时

学习目标

1. 掌握通过不同方法处理三位整数，提取其各位数字并反向输出的技巧。
2. 学会将秒数转换为小时、分钟和秒格式的编程方法。
3. 理解如何从一串数字字符串中提取最大值和最小值。
4. 深化对 Python 基本数据类型及其操作的理解。

教学内容

一、引言（5 分钟）

- 简要回顾 Python 的基本数据类型（整数、浮点数、字符串等）。
- 强调数据类型操作在编程中的重要性。

二、应用举例 - 提取三位整数的各位数字并反向输出（30 分钟）

1. 方法一：数学方式求解
    - 演示代码，解释整除（//）和取余（%）运算符的应用。
    - 强调数据类型转换的必要性（从字符串到整数）。
2. 方法二：字符串方式求解
    - 演示代码，介绍字符串索引和切片操作。
    - 讨论输入有效性检查的重要性。
3. 方法三：一行代码求解
    - 演示代码，使用字符串切片[::-1]进行反转。
    - 讨论代码简洁性和可读性的平衡。
4. 课堂练习（10 分钟）
    - 学生尝试编写代码，实现以上三种方法。

教案纸

- 教师巡视指导，解答疑问。

三、应用举例 - 秒数转换为小时、分钟和秒格式（25 分钟）

1. 问题描述与分析
   - 阐述问题的实际需求和处理流程。
   - 强调整除和取模运算在时间转换中的应用。
2. 编写代码
   - 演示完整的代码实现，包括输入、处理和输出部分。
   - 介绍 format()函数的用法和格式化字符串的编写。
3. 课堂练习（10 分钟）
   - 学生编写代码，实现秒数到时间格式的转换。
   - 教师点评学生代码，强调编程规范和错误处理。

四、应用举例 - 提取数字字符串中的最大值和最小值（25 分钟）

1. 问题描述与分析
   - 阐述问题的实际需求和处理流程。
   - 讨论数据类型转换（从字符串到浮点数）的必要性。
2. 编写代码
   - 演示完整的代码实现，包括输入、字符串处理、类型转换和结果输出。
   - 介绍 split()函数和 max()、min()函数的应用。
3. 课堂练习（10 分钟）
   - 学生编写代码，实现数字字符串中最大值和最小值的提取。
   - 教师点评学生代码，强调数据处理的逻辑性和准确性。

课堂小结（5 分钟）

- 总结本节课学习的三种应用实例及其编程方法。
- 强调数据类型操作在解决实际问题中的重要性。

教案纸

- 鼓励学生多实践、多思考，加深对 Python 基础数据类型的理解。

作业布置

1. 编写一个程序，接收用户输入的一串数字（可能包含小数），并输出这些数字中的最大值和最小值。
2. 编写一个程序，将用户输入的秒数转换为“天:小时:分钟:秒”的格式输出。
3. 思考并尝试编写一个程序，实现将用户输入的字符串反转输出（不限制字符串长度）。

教案纸

第八讲

第四章 Python 程序的控制结构-4.1 程序流程图&4.2 顺序结构&4.3 条件表达式

学习目标

1. 理解程序流程图的基本元素及其作用。
2. 掌握顺序结构的概念及其在程序中的应用。
3. 熟悉条件表达式及其组成元素（关系运算符、逻辑运算符等）。

教学内容

一、引言（5 分钟）

- 简要介绍程序控制结构的重要性及其在编程中的作用。
- 引出本章将要学习的内容：程序流程图、顺序结构、条件表达式。

二、4.1 程序流程图（15 分钟）

1. 程序流程图基本概念
    - 解释程序流程图的作用：描述程序的基本操作和控制流程。
    - 展示流程图的基本元素（起止框、判断框、处理框、输入输出框、注释框、流向线、连接点），并解释其含义。
2. 顺序结构流程图示例
    - 展示顺序结构流程图（如图 4.2 所示），解释其含义。
    - 讨论顺序结构在程序中的作用和重要性。

三、4.2 顺序结构（15 分钟）

教案纸

1. 顺序结构概念
    - 解释顺序结构的概念：程序按照书写顺序从前到后依次执行语句。
    - 讨论顺序结构在程序中的基础性和重要性。
2. 顺序结构示例
    - 编写一个简单的 Python 程序示例，展示顺序结构的应用。
    - 分析示例程序中的赋值语句、输入语句和输出语句，解释它们在顺序结构中的作用。

四、4.3 条件表达式（35 分钟）

1. 关系运算符
    - 介绍关系运算符（<、<=、>、>=、==、!=），解释其含义和用法。
    - 示例演示比较不同类型数据（数值、字符串）时的大小关系。
2. 逻辑运算符
    - 介绍逻辑运算符（not、and、or），解释其含义和用法。
    - 示例演示逻辑运算符的组合使用，特别是惰性求值的特点。
3. 条件表达式的特殊用法
    - 解释条件表达式中非 False 即为 True 的原则。
    - 示例演示包含函数调用等复杂表达式的条件判断。
4. 条件表达式练习
    - 提供几个练习题，让学生编写条件表达式，检查其对条件表达式的掌握情况。

五、课堂练习（20 分钟）

- 提供一些包含顺序结构和条件表达式的编程题目，让学生练习编写程序。
- 教师巡视指导，解答学生疑问。

六、课堂小结（5 分钟）

- 总结本章学习的重点和难点。

教案纸

- 强调程序控制结构在编写高效、健壮程序中的重要性。

作业布置

1. 编写一个 Python 程序，要求用户输入两个整数，判断并输出它们的大小关系。
2. 编写一个 Python 程序，根据用户输入的分数判断其等级（如 A、B、C 等），并输出结果。

教案纸

第九讲

第四章 Python 程序的控制结构-4.4 选择结构

课程时长：2 学时

学习目标

1. 理解选择结构（分支结构）的概念及其重要性。
2. 掌握单分支结构（if 语句）的语法和使用方法。
3. 掌握双分支结构（if-else 语句）的语法和使用方法。

教案纸

1. 理解多分支结构（if-elif-else 语句）的语法和使用方法。
2. 能够根据实际问题设计合适的选择结构。

教学内容

一、引言（5 分钟）

- 介绍选择结构的概念及其在编程中的重要作用。
- 简述本章将要学习的内容：单分支结构、双分支结构和多分支结构。

二、单分支选择结构（if 语句）（20 分钟）

1. 语法介绍
    - 展示 if 语句的基本语法格式。
    - 强调冒号“:”和缩进的重要性。
2. 流程图解析
    - 展示图 4.3 单分支结构流程图，解释流程。
3. 示例分析（例 4-1）
    - 分析回文字符串判断示例代码。
    - 讨论当条件不满足时程序的行为。
4. 练习与讨论
    - 提供练习题，让学生编写简单的单分支结构程序。
    - 讨论学生代码，强调 if 语句的正确使用。

三、双分支选择结构（if-else 语句）（25 分钟）

1. 语法介绍
    - 展示 if-else 语句的基本语法格式。
    - 解释条件成立与不成立时程序的执行流程。
2. 流程图解析

教案纸

- 展示图 4.4 双分支结构流程图，解释流程。

-