python 语言程序设计基础

Hengsheng Zhou

电信与智能制造学院

2025 年 3 月 26 日

郑州西亚斯学院
SIAS UNIVERSITY

Outline

1. 说课
   - 为什么学 python
   - 怎么学
   - 课程思政
2. 本节内容概述
3. 容器
   - 元组
   - 集合
   - 字典

1 说课
- 为什么学 python
- 怎么学
- 课程思政

2 本节内容概述

3 容器

1 说课
- 为什么学 python
- 怎么学
- 课程思政

2 本节内容概述

3 容器

1. Python 用途广泛，生态丰富，无论是初学者还是专业开发者，都能在不同领域找到合适的应用场景！

1. Python 用途广泛，生态丰富，无论是初学者还是专业开发者，都能在不同领域找到合适的应用场景！

应用场景 \ 示例	框架
数据分析	Matplotlib/Seaborn（数据可视化）
自动化	批量文件处理
数据采集	Scrapy

1. Python 用途广泛，生态丰富，无论是初学者还是专业开发者，都能在不同领域找到合适的应用场景！

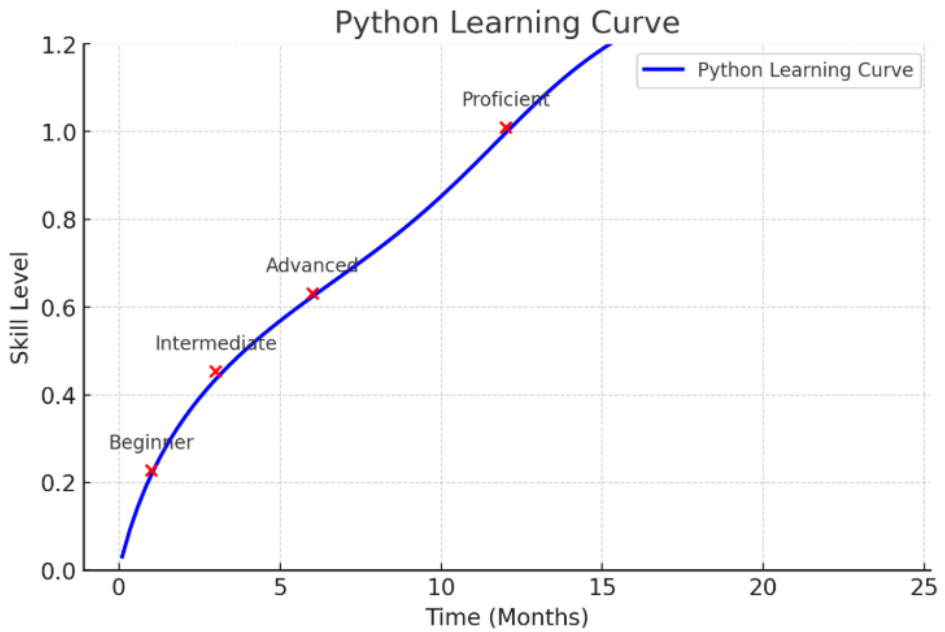
应用场景 \ 示例	框架
数据分析	Matplotlib/Seaborn（数据可视化）
自动化	批量文件处理
数据采集	Scrapy

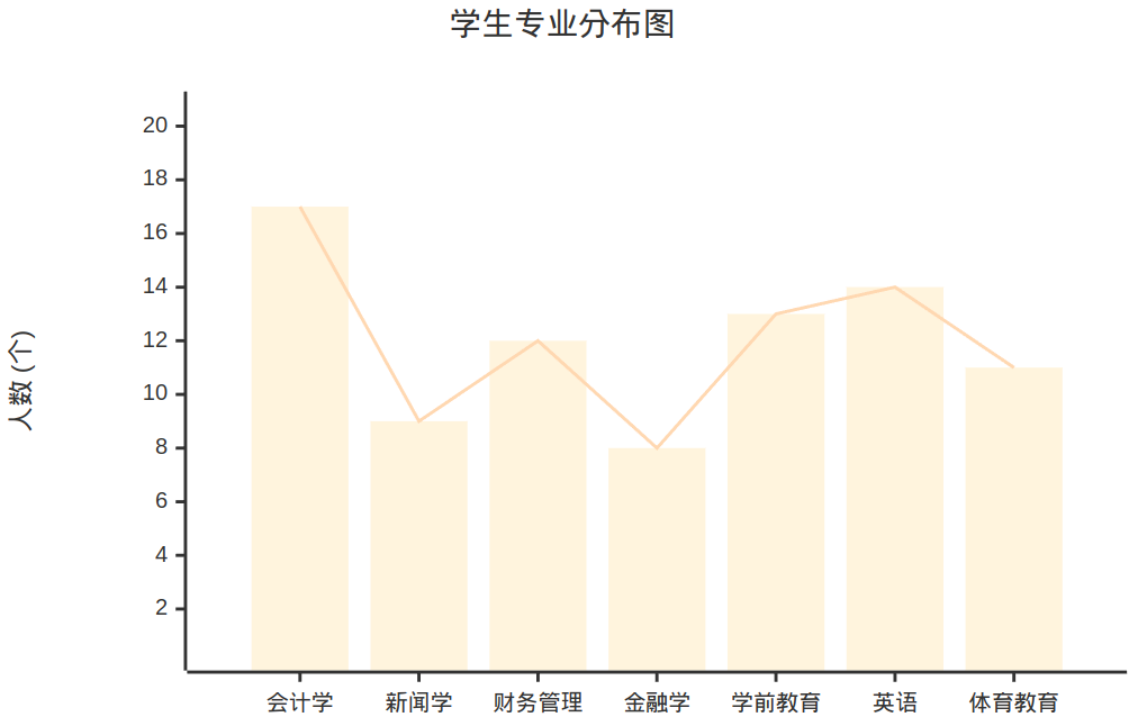
2. Python 语法简洁、易学易用，适合零基础入门.

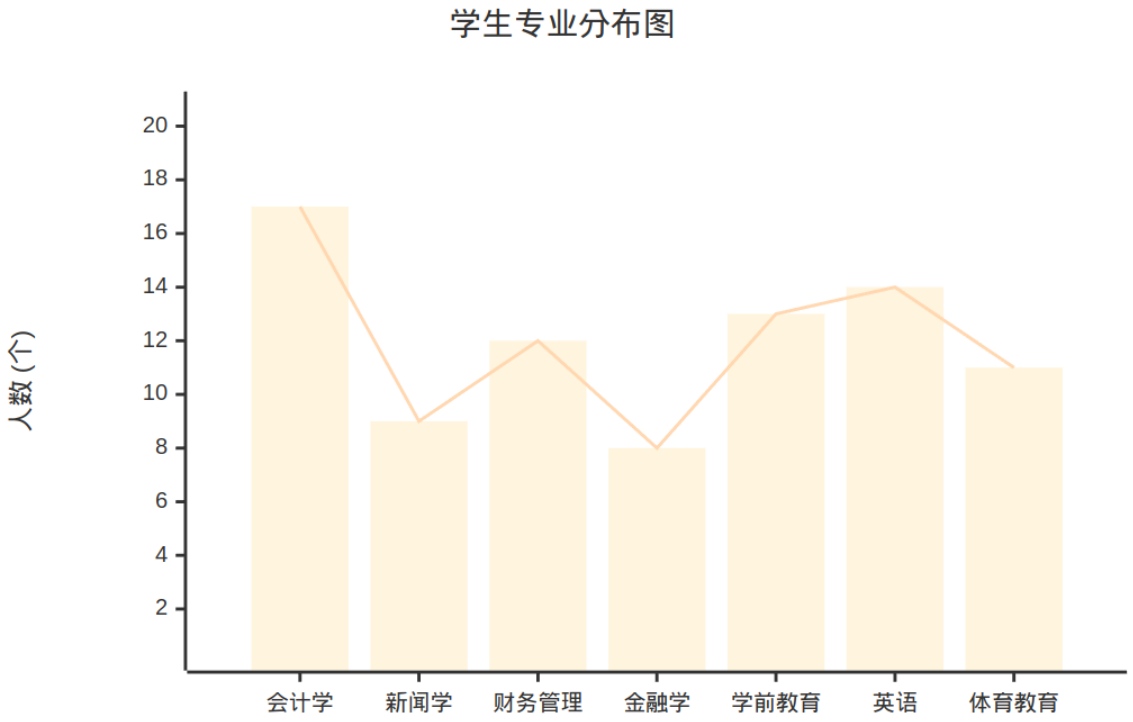
1. Python 用途广泛，生态丰富，无论是初学者还是专业开发者，都能在不同领域找到合适的应用场景！

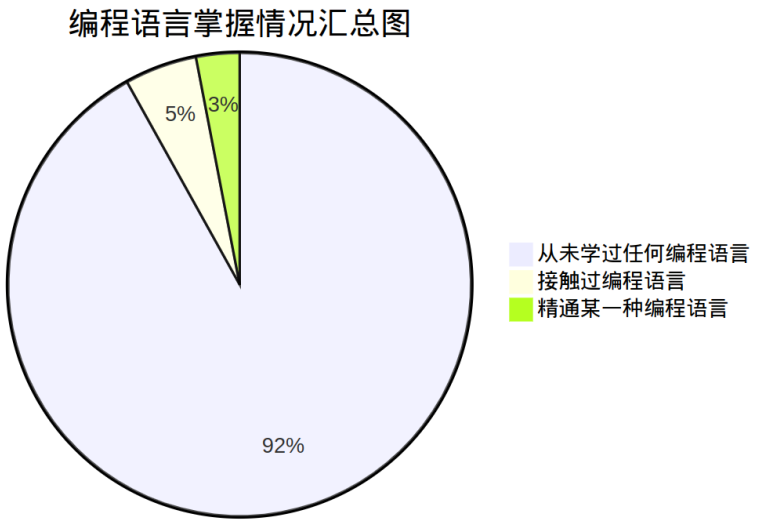
应用场景 \ 示例	框架
数据分析	Matplotlib/Seaborn（数据可视化）
自动化	批量文件处理
数据采集	Scrapy

2. Python 语法简洁、易学易用，适合零基础入门.




学生专业分布图
人数 (个)
20
18
16
14
12
10
8
6
4
2
会计学
新闻学
财务管理
金融学
学前教育
英语
体育教育


学生专业分布图
人数 (个)
20
18
16
14
12
10
8
6
4
2
会计学
新闻学
财务管理
金融学
学前教育
英语
体育教育


编程语言掌握情况汇总图
5%
3%
92%
从未学过任何编程语言
接触过编程语言
精通某一种编程语言

1 说课
- 为什么学 python
- 怎么学
- 课程思政

2 本节内容概述

3 容器

- 避免陷入“理论陷阱”

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

solution

遇到复杂的问题就先跳过，由浅入深

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

solution

遇到复杂的问题就先跳过，由浅入深

- 熟练使用 AI 辅助工具

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

solution

遇到复杂的问题就先跳过，由浅入深

- 熟练使用 AI 辅助工具

solution

学会使用 deepseek,chatGPT 等 AI 辅助工具编写代码

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

solution

遇到复杂的问题就先跳过，由浅入深

- 熟练使用 AI 辅助工具

solution

学会使用 deepseek,chatGPT 等 AI 辅助工具编写代码

- 找 python 开发社区交流经验

- 避免陷入“理论陷阱”

solution

学一点就写代码，实践出真知，避免只看不练

- 先掌握最基础的知识

solution

遇到复杂的问题就先跳过，由浅入深

- 熟练使用 AI 辅助工具

solution

学会使用 deepseek,chatGPT 等 AI 辅助工具编写代码

- 找 python 开发社区交流经验

solution

学习在 GitHub 和 Stack Overflow 等开源社区寻找学习资源

1 说课
- 为什么学 python
- 怎么学
- 课程思政

2 本节内容概述

3 容器

在专业课程教学中融入思政教育，实现知识传授和价值引导的统一，是教学过程中不可或缺的重要环节，课程思政教育贯穿本课程教学全过程。

思政主题	Python 结合方式	目标
家国情怀	Python + 数据分析（环保、扶贫、疫情）	关注社会，培养责任感
工匠精神	Pythonic 代码风格 + 代码优化	追求卓越，提升质量意识
团队精神	Python 项目开发 + 开源贡献	强化合作，培养沟通能力
职业道德	Python + 网络安全（合法爬虫、数据加密）	规范行为，树立安全意识

在专业课程教学中融入思政教育，实现知识传授和价值引导的统一，是教学过程中不可或缺的重要环节，课程思政教育贯穿本课程教学全过程。

思政主题	Python 结合方式	目标
家国情怀	Python + 数据分析（环保、扶贫、疫情）	关注社会，培养责任感
工匠精神	Pythonic 代码风格 + 代码优化	追求卓越，提升质量意识
团队精神	Python 项目开发 + 开源贡献	强化合作，培养沟通能力
职业道德	Python + 网络安全（合法爬虫、数据加密）	规范行为，树立安全意识

不同思政目标所对应的实现方式

1 说课

2 本节内容概述

3 容器

1 说课

2 本节内容概述

3 容器
- 元组
- 集合
- 字典

1 说课

2 本节内容概述

3 容器
- 元组
- 集合
- 字典

元组

创建元组

definition

元组是有序的、不能更改的、可重复的容器。

元组

创建元组

definition

元组是有序的、不能更改的、可重复的容器。

example

创建元组 thistuple = ("apple", "banana", "cherry", "apple", "cherry") 使用构造器创建元组 thistuple = tuple(("apple", "banana", "cherry", "apple", "cherry"))

元组

创建元组

definition

元组是有序的、不能更改的、可重复的容器。

example

创建元组 thistuple = ("apple", "banana", "cherry", "apple", "cherry") 使用构造器创建元组 thistuple = tuple(("apple", "banana", "cherry", "apple", "cherry"))

定义单个元素的元组

创建单元素的元组必须在元素后添加逗号 thistuple = ("apple",) print(type(thistuple)) thistuple = ("apple") print(type(thistuple))

元组

访问元组

- 索引 (正向，反向，截取)
- 遍历
- 筛选

元组

添加元素/删除元素

因为 tuple 是不可更改的，如果需要更改 tuple 中的元素需要将其转化为 list 类型的变量

元组
解包

将元组中的元素一次赋给多个变量

example

```
fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")
(green, yellow, *red) = fruits
print(green) print(yellow) print(red)
```

元组

方法

- count()：输出某个元素在 tuple 中出的次数
- index()：输出某个元素在元组中第一次出现位置的索引值

1 说课

2 本节内容概述

3 容器
- 元组
- 集合
- 字典

集合 set

set 创建

definition

集合是无序、不可更改、不可重复、无索引的容器.(不可更改🄵的是集合元素的值无法更改，但不影响集合本身添加删除元素)

集合 set

set 创建

definition

集合是无序、不可更改、不可重复、无索引的容器.(不可更改🄵的是集合元素的值无法更改，但不影响集合本身添加删除元素)

example

thisset = {"apple", "banana", "cherry"} print(thisset) 通过构造器创建 set thisset = set(("apple", "banana", "cherry")) note the double round-brackets

集合 set

set 创建

definition

集合是无序、不可更改、不可重复、无索引的容器.(不可更改🄵的是集合元素的值无法更改，但不影响集合本身添加删除元素)

example

thisset = {"apple", "banana", "cherry"} print(thisset) 通过构造器创建 set thisset = set(("apple", "banana", "cherry")) note the double round-brackets

set 元素不允许重复

thisset = {"apple", "banana", "cherry", True, 1, 2}
1 和 true 在 set 中被认为是值相同的元素 print(thisset)

集合

访问 set

- 索引

集合

访问 set

- 索引

example

list(1,2,3)[2]

集合

访问 set

- 索引

example

list(1,2,3)[2]

- 遍历

集合

访问 set

- 索引

example

list(1,2,3)[2]

- 遍历

example

for item in 1,2,3:

集合

访问 set

- 索引

example

list(1,2,3)[2]

- 遍历

example

for item in 1,2,3:

- 筛选

集合

访问 set

- 索引

example

list(1,2,3)[2]

- 遍历

example

for item in 1,2,3:

- 筛选

example

[item for item in 1,2,3 if item $>$ 2]

集合

向集合中添加元素

- add()
- update() 更新原集合、union()=| 返回新集合

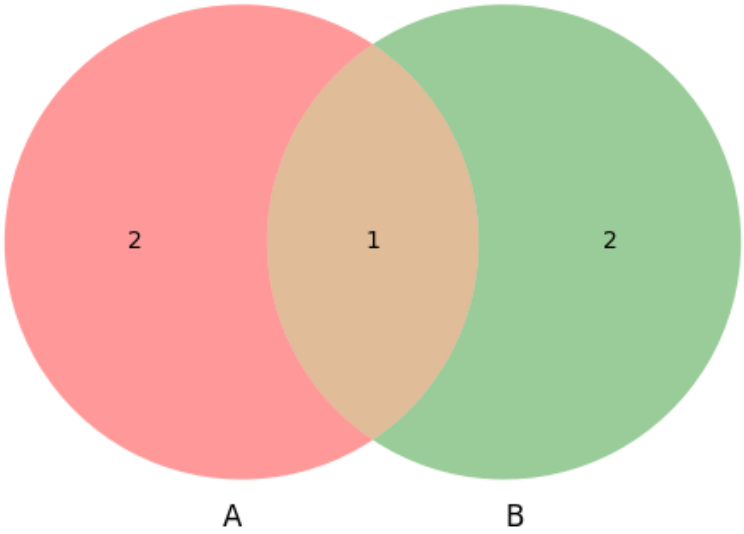
集合

删除集合中的元素

- remove(): 删除指定值的元素，若值不存在报错
- discard(): 删除指定值的元素，不报错
- pop(): 随机删除元素
- clear(): 清空元素

集合

对集合的操作



- intersection() 将两集合中所有重复的元素返回到新集合，取得 1
- difference() 将在另一个集合中出现过的元素筛掉形成新集合，取得 A2 或 B2
- symmetric_difference() 两个集合中所有差异的元素全部同步到新集合，取得 A2 和 B2

1 说课

2 本节内容概述

3 容器
- 元组
- 集合
- 字典

字典

创建字典

definition

字典是有序、可改值、不允许出现重复的键

字典

创建字典

definition

字典是有序、可改值、不允许出现重复的键

example

thisdict = { "brand": "Ford", "model": "Mustang", "year": 1964 }
thisdict = dict(name = "John", age = 36, country = "Norway") : 使用构造器创建集合
dict.fromkeys(Iterable, default_values): 快速构造默认值相同的字典

字典

字典元素的访问

- get(): 通过 key 值访问某个元素的 value
- keys(): 返回所有的 keys
- values(): 返回所有的 values
- items(): 返回所有的 items
- 遍历字典
  - for x in dictionary 和 keys(): 遍历 keys
  - for x in dictionary: dictionary[x] 和 values() 便利 values
  - for x,y in dictionary.items: 遍历 (key,value)

Attention

x =dictionary.keys(): 在获取字典的 keys 之后任何对字典的修改都会同步到 keys 列表,valuse 和 items 也类似

字典

向字典中添加元素

- dictionary[keys]=values
- update(): 函数的值可以是任何 iterable 类型的变量

字典

删除字典中的元素

- pop(key): 删除指定 key 的元素
- popitem(): 删除最后一个元素
- clear(): 清空字典