Python 程序开发课程笔记

一、数据存储与处理基础

数据可长期存储在磁盘（如 SSD、HDD）或磁带中。在进行数据呈现、计算加工或编解码时，需借助通电的 CPU 和内存，以进程为单位在操作系统中处理。例如，Microsoft Word 启动后是一个进程，可将磁盘中的文档加载到内存进行查看和编辑，最后再保存回磁盘。Python 解释器启动后也是一个进程，可执行 Python 代码，并调用操作系统或其他软件完成相应任务。

二、Python 基本概念

1. **变量 (variable)**：用于存储数据的标识符，可以随时修改其值。

```
python
```

运行

```
x = 10
name = "Alice"
```

1. **函数 (function)**：一段可重复使用的代码块，用于执行特定的任务。

```
python
```

运行

```
def add(a, b):
    return a + b

result = add(3, 5)print(result)
```

1. **对象 (object)**：Python 中一切皆对象，每个对象都有其类型和属性。

```
python
```

运行

```
s = "hello"print(type(s))  # <class'str'>
```

1. **类型 (type)**：用于表示对象的种类，如字符串 (str)、整数 (int)、列表 (list)、字典 (dict) 等。
2. **属性 (attribute)**：对象所具有的特性。

```python
运行

my_list = [1, 2, 3]print(len(my_list))  # 调用列表的 len 属性
```

1. **方法 (method)**：对象的函数，可对对象进行操作。

```python
运行

my_list = [1, 2, 3]

my_list.append(4)  # 调用列表的 append 方法 print(my_list)
```

1. **调用 (call)**：执行函数或方法。

```python
运行

def greet(name):

    print(f"Hello, {name}!")


greet("Bob")  # 调用 greet 函数
```

1. **形参 (parameter)**：函数定义时的参数。

```python
运行

def multiply(a, b):

    return a * b
```

1. **实参 (argument)**：函数调用时传递的参数。

python

运行

```
result = multiply(2, 3)  # 2 和 3 是实参
```

1. **返回值 (return value)**：函数执行后返回的结果。

python

运行

```
def divide(a, b):
    return a / b

div_result = divide(10, 2)print(div_result)
```

三、任务实践

1. **创建 Conda 环境**：在 VS Code 中打开项目目录，新建 `environment.yml` 文件，指定安装 Python 3.12，然后运行 `conda env create` 命令创建 Conda 环境。
2. **创建联系人文件**：新建 `contacts.txt` 文件，按要求格式写入联系人信息。
3. **编写 main.py 代码**：读取 `contacts.txt` 文件内容，处理后输出到 `emails.txt` 文件，按邮箱域名和用户名排序。

python

运行

```
def read_contacts(file_path):
    contacts = []
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            name, gender, email = line.strip().split()
            contacts.append((name, gender, email))
    return contacts
```

```
def generate_emails(contacts):
    emails = []
    for name, gender, email in contacts:
        salutation = f"尊敬的 {name}{'女士' if gender == '女' else '先生'}"
        email_content = f"to: <{email}>\n{salutation}，您的会员资格即将到期，请及时续费。\n---\n"
        emails.append(email_content)
    return emails
def sort_emails(emails):
    def key_func(email):
        parts = email.split('@')
        domain = parts[1].split('<')[0]
        username = parts[0].split('to: <')[1]
        return domain, username

    return sorted(emails, key=key_func)
if __name__ == "__main__":
    contacts = read_contacts('contacts.txt')
    generated_emails = generate_emails(contacts)
    sorted_emails = sort_emails(generated_emails)

    with open('emails.txt', 'w', encoding='utf-8') as file:
        file.writelines(sorted_emails)
```

四、调试工具使用

1. **安装扩展**：在 VS Code 扩展商店里安装 Python 扩展和 Ruff 扩展。
2. **调试命令**：运行 `python -m pdb main.py` 进入调试模式，使用 `l` 显示代码、`n` 执行当前行、`p` 打印表达式、`s` 步入调用、`pp` 美观打印、`c` 继续执行等命令进行调试。
3. **检查对象**：安装 `wat-inspector` 第三方软件包，在调试过程中检查各种对象。