一、文件操作

1. 文件的打开与关闭

1. 编写代码，以只读模式打开名为 `test.txt` 的文件，然后关闭该文件（假设文件存在）。
2. 尝试以写入模式打开不存在的 `data.txt` 文件，然后查看是否生成了该文件，再关闭文件。

2. 文件读取操作

1. 有一个 `info.txt` 文件，内容如下：使用 `read()` 方法读取该文件的全部内容并打印。

```
Hello
Python
File
Operation
```

2. 对于上述 `info.txt` 文件，使用 `readline()` 方法逐行读取并打印每一行内容。
3. 对于上述 `info.txt` 文件，使用 `readlines()` 方法读取所有行，存储到列表中，然后遍历列表打印每一行。

3. 文件写入操作

1. 编写代码，向 `new.txt` 文件中写入 “Welcome to file writing!”，然后查看文件内容。
2. 向已存在的 `info.txt` 文件（内容如前面所示）中追加内容 “New Line”，然后读取文件查看结果。

4. 上下文管理器（with 语句）

1. 使用 `with` 语句以只读模式打开 `test.txt` 文件，读取其内容并打印，观察是否需要手动关闭文件。
2. 使用 `with` 语句向 `data.txt` 文件中写入 “Using with statement for writing.”，然后查看文件内容。

5. 文件操作综合案例

1. 编写程序，实现以下功能：
    - 提示用户输入一些内容。
    - 将用户输入的内容写入到 `user_input.txt` 文件中。
    - 然后读取该文件的内容并打印。
2. 编写程序，统计 `article.txt` 文件（假设文件存在，包含若干文本）的行数、单词数（以空格分隔为单词），并打印统计结果。

二、Python 模块

1. 模块的导入与使用

1. 导入 `math` 模块，使用该模块计算 `2` 的平方根，并打印结果。
2. 导入 `random` 模块中的 `randint` 函数，生成一个 1 到 10 之间的随机整数并打印。

3. 使用 `from...import` 语句导入 `datetime` 模块中的 `datetime` 类，获取当前日期和时间并打印。

2. 自定义模块

1. 创建一个名为 `my_module.py` 的模块，在其中定义一个函数 `greet(name)`，功能是打印 “Hello, name!”。然后在另一个 Python 文件中导入该模块，调用 `greet` 函数，传入自己的名字。
2. 在 `my_module.py` 模块中再定义一个变量 `PI = 3.14159`，然后在导入该模块的文件中，打印 `PI` 的值。

3. 包（Package）

1. 创建一个名为 `my_package` 的包，在包内创建两个模块 `module1.py` 和 `module2.py`。在 `module1.py` 中定义函数 `func1()`，打印 “Function from module1”；在 `module2.py` 中定义函数 `func2()`，打印 “Function from module2”。然后在包外的 Python 文件中，分别导入这两个模块并调用对应的函数。
2. 在 `my_package` 包中创建 `__init__.py` 文件，在其中定义变量 `version = "1.0"`，然后在导入包的文件中，打印 `my_package.version`。

4. 常用标准库模块

1. 使用 `os` 模块，获取当前工作目录并打印。
2. 使用 `sys` 模块，打印 Python 解释器的版本信息。
3. 编写一个函数，计算给定生日距离今天还有多少天，以及年龄。
4. 创建一个抽奖程序，从10个参与者中随机选择3个获奖者（不能重复）。
5. 编写一个函数，生成包含大写字母、小写字母、数字和特殊字符的随机密码，长度为12位。
6. 编写一个程序，计算圆的面积、球的体积和三角形的面积。

5. 综合案例

1. 创建一个模块 `calculator.py`，其中包含 `add(a, b)`、`subtract(a, b)`、`multiply(a, b)`、`divide(a, b)` 四个函数，分别实现两数的加、减、乘、除运算。然后创建一个主程序文件，导入该模块，提示用户输入两个数和一个操作符（`+`、`-`、`*`、`/`），根据操作符调用对应的函数并打印结果。
2. 利用 `os` 模块，编写程序，遍历当前目录下的所有文件，打印每个文件的名称和大小。