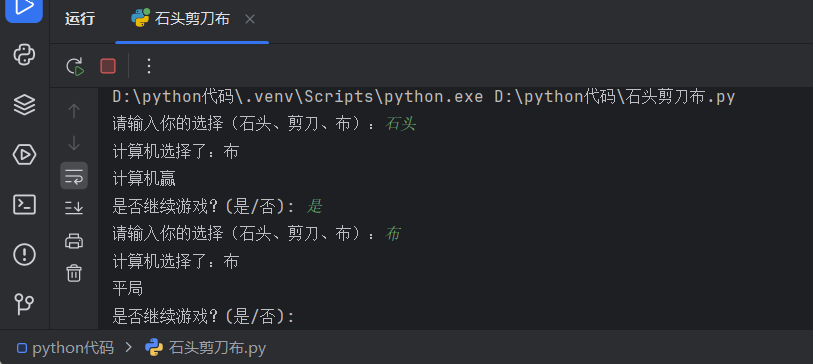
# 计算机基础——Python作业实验报告

**作业一：石头剪刀布**

一、输入输出截图

（注：这是连续运行的结果，分别测试了多种情况，确保代码的严谨正确）

1. 相关介绍

1. 用到的包

这段代码仅使用了 random 包。具体作用如下：

random.choice(): 用于从指定的序列中随机选择一个元素。在本代码中，random.choice(['石头', '剪刀', '布'])用于模拟计算机的选择。

除此之外，没有使用其他外部包或模块，代码运行依赖于标准库。

2. 设计思路

该游戏的设计思路是通过模拟用户与计算机进行“石头、剪刀、布”对战，按照以下流程实现。

①获取计算机的选择：

使用 random.choice() 从三种可能的选项（‘石头’，‘剪刀’，‘布’）中随机选择一个。

②获取玩家的选择：

玩家通过 input() 函数输入他们的选择，输入的内容必须是“石头”，“剪刀”或“布”。为了确保用户输入有效，代码中使用了一个 while 循环来验证输入是否在有效范围内。如果输入无效，则会提示用户重新输入。

③确定游戏结果：

比较玩家和计算机的选择，根据游戏规则判断胜负。规则如下：

石头击剪刀，剪刀击布，布击石头。

如果双方选择相同，则为平局。

④继续游戏的选项：

在每轮游戏结束后，用户可以选择是否继续进行游戏。如果输入“是”则继续，如果输入“否”则退出游戏。

3. 实现的功能

①get\_computer\_choice()：

返回计算机随机选择的“石头”，“剪刀”或“布”。

②get\_player\_choice()：

允许玩家输入他们的选择，并确保输入有效。若输入无效，则提示用户重新输入。

③determine\_winner(player, computer)：

根据玩家和计算机的选择，判断谁赢了，或者是否平局。判断逻辑遵循“石头、剪刀、布”的规则。

④play\_game()：

主游戏函数。它会不断执行游戏循环，直到玩家选择停止游戏。每一轮游戏中，都会显示计算机的选择并判断结果，最后询问玩家是否继续游戏。

4. 代码的执行流程

①游戏开始时，计算机通过 get\_computer\_choice() 随机选择“石头”，“剪刀”或“布”。

②玩家通过 get\_player\_choice() 输入他们的选择。输入会进行检查，确保输入是有效的。

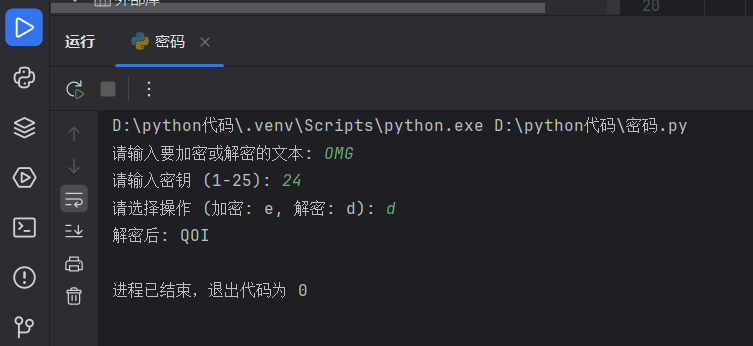
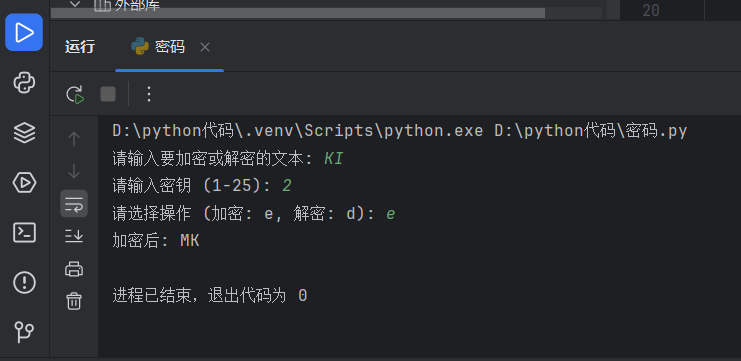
③determine\_winner() 函数会判断玩家和计算机的选择，根据石头、剪刀、布的规则确定结果。

④显示计算机的选择和游戏结果。

⑤询问玩家是否继续游戏，如果玩家选择“是”，则继续进行下一轮；如果选择“否”，则退出游戏。

**作业二：凯撒密码**

一、输入输出截图



二、相关介绍

1. 涉及的包

这段代码并没有使用外部的Python库或包，仅仅依赖了Python内建的函数和方法，主要用到了以下内容：

ord()：用于获取字符的Unicode码（ASCII码）。

chr()：将Unicode码（整数）转换为字符。

isalpha()：用于检查字符是否是字母（即是否为 A-Z 或 a-z）。

islower() 和 isupper()：检查字符是否是小写字母或大写字母。

2. 设计思路

设计思路可以分为以下几个部分。

①函数功能：caesar\_cipher函数是核心，它接收文本、位移量以及加密或解密的标志参数，根据位移量对字母进行位移，并返回加密或解密后的文本。

②处理字母与非字母字符：该算法仅对字母进行加密或解密，其他字符（如空格、标点符号等）保持不变，直接添加到结果中。

③加密和解密的区分：通过传入encrypt=True来决定是执行加密（加密时按正位移量处理），还是解密（解密时按负位移量处理）。位移量（shift）经过shift % 26处理后，确保其在0到25的范围内，从而避免无效的位移。

④循环处理字符：遍历输入文本中的每个字符，检查它是否是字母，若是字母则进行加密或解密操作，若不是字母则原样保留。

⑤处理小写与大写字母：考虑到字母的大小写（islower()和isupper()），分别对大写字母和小写字母进行不同的Unicode码范围判断，确保字符按正确的范围回绕。

3. 实现的功能

3.1 加密和解密

通过凯撒密码的规则，将文本中的每个字母移动一定的位数。位移的值由shift参数决定。

①加密：将每个字母的Unicode码加上shift，并确保加密后的字符仍然在字母表范围内。如果超出了‘z’或‘Z’的范围，则通过回绕机制（加减26）使其重新回到字母表内。

②解密：解密操作通过将shift设置为负值来实现，即相当于“反向”加密。

3.2 输入与输出

程序会：

①接收用户输入的文本。

②接收位移量（密钥）并进行合法性校验（确保shift在1到25之间，虽然代码中没有明确做这一点）。

③接收用户选择是执行加密还是解密操作。

④根据用户选择调用相应的加密或解密函数。

⑤输出加密或解密后的文本。

3.3 处理非字母字符

对于文本中的非字母字符（如空格、标点符号等），该算法不会对其进行任何加密或解密处理，而是原样保留。例如：

输入：“Hello, World!”

如果加密位移量为3，输出将会是：“Khoor, Zruog!”

3.4 保证循环位移

对于字母的加密和解密，程序通过检查字符的Unicode值是否超出字母的范围（‘a’到‘z’或‘A’到‘Z’），并进行回绕处理，从而避免了字符“溢出”或“错误”的情况。

4. 代码的完整流程

初始化一个空字符串：用来保存加密或解密后的结果。

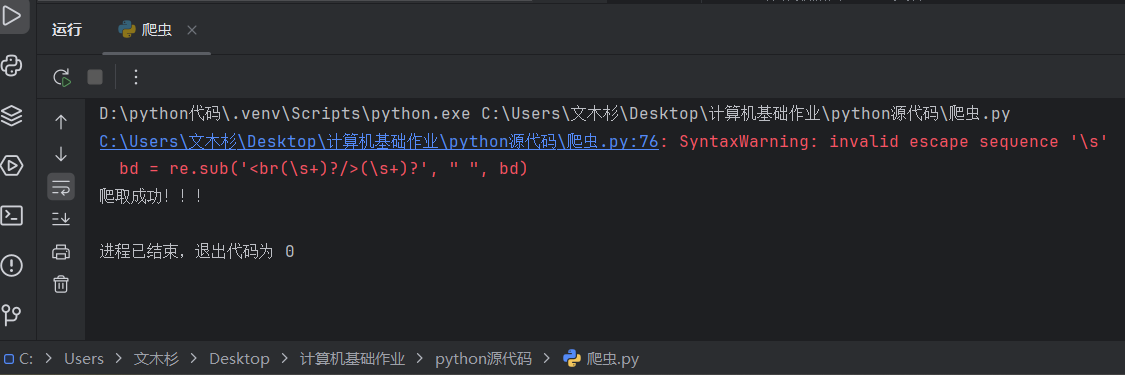
遍历每个字符：判断字符是否是字母，如果是字母，进行加密或解密；否则，直接将字符添加到结果字符串中。

位移处理：根据加密或解密的标志，对字母进行位移处理，确保位移后的字符仍然处于字母范围内。

返回处理结果：返回加密或解密后的字符串。

**作业三：爬虫脚本**

一、输入输出截图



具体结果见附件“豆瓣电影Top250前125的相关信息.xlsl”

二、相关介绍

1. 所用的包

①BeautifulSoup (来自 bs4 库)：

用于解析 HTML 文档并提取需要的内容。BeautifulSoup 提供了简便的 API 来遍历和操作 HTML 元素。

②re（正则表达式）：

用于通过正则表达式对网页源代码进行模式匹配，从中提取特定的数据（如链接、评分、图片等）。

③urllib.request 和 urllib.error：

用于发送 HTTP 请求并获取网页内容。urllib.request 提供了获取网页的功能，urllib.error 用于处理请求中的错误（如网络连接失败或网页不可用）。

④openpyxl：

用于操作 Excel 文件，能够创建、修改和保存 .xlsx 格式的文件。在这段代码中，openpyxl 用于创建 Excel 文件并将抓取到的数据写入表格中。

1. 设计思路

该爬虫程序的设计流程如下。

①自定义网页链接：

豆瓣电影的 Top 250 页面按 25 部电影分为 10 页。在这段代码中，手动提供了 5 个页面的链接，目标是抓取前 125 部电影的数据。

②获取网页数据：

程序通过 urllib.request 发送请求，获取网页内容，并通过 BeautifulSoup 解析 HTML 页面。

③提取信息：

使用正则表达式提取每部电影的具体信息，如电影链接、图片链接、影片中文名、外文名、评分、评价人数、电影概况、导演和演员等信息。

④保存数据到 Excel 文件：

所有抓取到的数据被存储在一个列表中，最终通过 openpyxl 将这些数据写入 Excel 表格，以便用户查看。

1. 功能实现

该程序的主要功能是抓取豆瓣电影 Top 250 中前 125 部电影的相关信息，并将其保存为一个 Excel 文件。具体实现如下。

①getdata(url\_list)：

·该函数接受一个包含多个豆瓣电影链接的列表（url\_list），然后循环遍历每个链接，获取网页内容并解析。

·使用正则表达式提取每部电影的详细信息（链接、图片链接、电影名称、评分、评价人数等）。

·把每部电影的相关信息作为一个列表，最终将所有电影的信息汇总到一个大的列表 datalist。

②savedata(datalist, savepath)：

·该函数接受提取的数据 datalist 和保存路径 savepath，将数据写入到一个 Excel 文件中。

·首先，创建一个新的工作簿并设置工作表标题为“豆瓣电影Top250前125的相关信息”。

·然后，写入表头（列名）和数据（每部电影的相关信息）。

·最后，将工作簿保存到指定的路径。

③geturl(url)：

·该函数用于向豆瓣电影页面发送 HTTP 请求并获取响应。它设置了一个模拟浏览器的 User-Agent，以防止被网站拦截。

·如果请求成功，返回网页的 HTML 内容；如果请求失败，捕获错误并打印错误信息。

④主函数 (main)：

·这是程序的入口点，首先定义了包含 5 个网页链接的列表 custom\_urls。

·调用 getdata(custom\_urls) 获取数据，并将结果传递给 savedata(datalist, savepath) 以保存到 Excel 文件。

1. 功能概述

该程序实现了以下功能：

①从多个网页获取豆瓣电影 Top 250 的前 125 部电影的信息。

②提取每部电影的详细数据：包括电影详情链接、图片链接、中文名、外文名、评分、评价数、概况和相关信息。

③将抓取到的数据保存为 Excel 文件，以便于用户查看和分析。

1. 代码中主要逻辑的详细解释

①提取数据：

正则表达式在这里用于从 HTML 中匹配和提取相关字段。

例如：

findLink 用来匹配电影详情链接。

findImgSrc 用来匹配图片链接。

findTitle 用来匹配电影的标题，分别提取中文名和外文名。

findRating 用来提取电影的评分。

findJudge 用来提取评价人数。

findInq 用来提取电影的简短概述。

findBd 用来提取电影的导演、演员等相关信息。

②Excel 文件存储：

openpyxl 库提供了一个简单的 API 来创建、修改和保存 Excel 文件。在此程序中，首先创建一个新的工作簿，设置表头，然后将每部电影的详细信息按行存储到 Excel 表格中，最终保存文件。