

暨南大学本科实验报告专用纸

课程名称 python程序设计 成绩评定

实验项目名称 python作业第六章

姓名 崔嘉容 学号 2020100069 学院 网络空间安全 专业 网络空间安全

实验时间 2023 年 10 月 23 日 ~ 10 月 30 日 实验地点: 516

一、实验目的

1. 掌握使用命令行参数实现交互功能。
2. 能够熟练使用标准输入和输出函数。
3. 掌握文件输入和输出。
4. 了解图形化用户界面。

二、实验环境和设备

实验环境：操作系统-Windows10，python版本-3.11.3，开发环境-pycharm

实验设备：华为MateBook14-2020，处理器-i7-10510U，内存-16GB

三、实验内容和结果

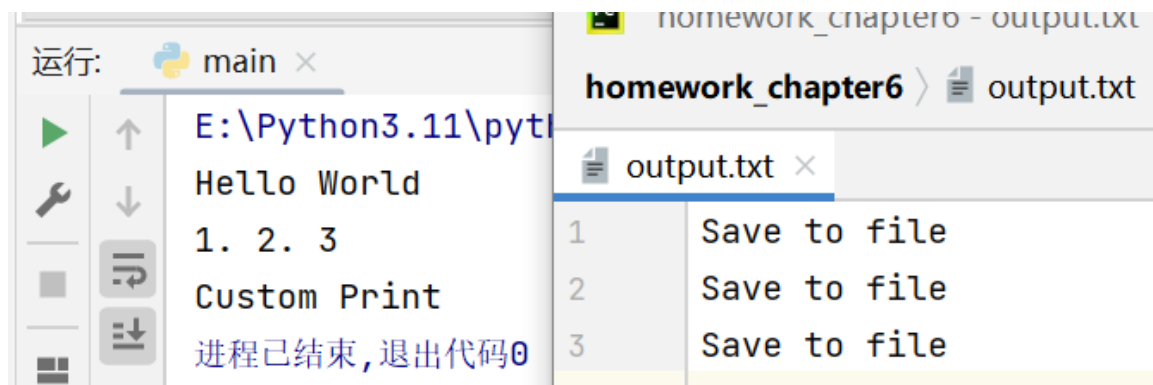
题目一：参照例6.3编写输入函数和输出函数示例程序

实验代码：

```
import sys
def my_print(*args, sep=' ', end='\n', file=None):
    output = sep.join(map(str, args)) + end
    if file is not None:
        with open(file, 'a') as f:
            f.write(output)
    else:
        sys.stdout.write(output)
        sys.stdout.flush()
my_print("Hello", "World") # 输出: Hello World
my_print(1, 2, 3, sep='. ') # 输出: 1. 2. 3
my_print("Custom", "Print", end=' ') # 输出: Custom Print
my_print("Save to file", file='output.txt') # 将输出保存到 output.txt
文件中
```

暨南大学本科实验报告专用纸(附页)

实验结果:



```
运行: main x
E:\Python3.11\pyt
Hello World
1. 2. 3
Custom Print
进程已结束,退出代码0

homework_chapter6 - output.txt
homework_chapter6 > output.txt
output.txt x
1 Save to file
2 Save to file
3 Save to file
```

实验分析和总结:

其实没有太明白“参照例6.3编写输入函数和输出函数示例程序”是要做什么,目前理解为为了实现不同格式的输出与输入.在这个实验中,我首先定义了一个名为`my_print`的函数,并为其提供了灵活的参数,包括可选的分隔符、结束符以及输出至文件的选项。

通过这个实验,学到了如何通过自定义函数来实现灵活的输出控制,这对于在特定情境下自定义输出格式非常有用。同时可以利用内置的`sys`模块来进行更底层的输出操作,使得输出函数具有了更多的灵活性。

在实验的改进方面,我认为可以进一步探索如何处理异常情况,比如当用户输入无效参数时如何提供友好的提示。此外,也可以考虑添加更多输出格式的选项,以进一步提升输出函数的灵活性。

暨南大学本科实验报告专用纸(附页)

题目二: 参照例6.7编写运行时提示输入密码的程序。

实验代码:

```
import getpass
username = input("请输入用户名: ")
passwd = getpass.getpass("请输入密码: ")
if username == 'admin' and passwd == '123456':
    print("登录成功")
else:
    print("登录失败")
```

实验结果:

```
PS D:\学习\大四上\python_learn\homework_chapter6> python 2_password.py
请输入用户名: admin
请输入密码:
登录成功
PS D:\学习\大四上\python_learn\homework_chapter6> 
```

实验分析和总结:

在进行实验时出现无法正常调用“getpass.getpass”函数的情况，调试后发现调试时可以正常运行程序，说明程序本身没有问题，可能是集成开发环境的问题。并且调试时出现“Warning: Password input may be echoed.”这个警告，这个警告是在某些环境下（特别是在一些集成开发环境或特定的终端程序中）可能会出现。它的含义是可能会有密码输入被回显（显示在屏幕上）的风险。所以改用控制台执行程序，正常。

对程序进行修改增加了登录窗口：

代码：

```
import tkinter as tk
from tkinter import messagebox
import getpass

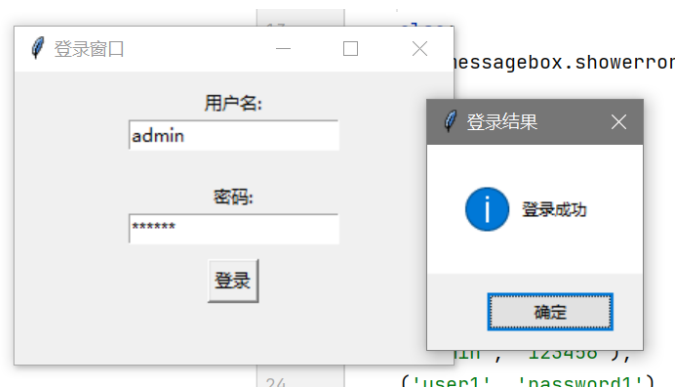
def check_login(event=None):
    username = entry_username.get()
    passwd = entry_password.get()

    if (username, passwd) in user_credentials:
```

暨南大学本科实验报告专用纸(附页)

```
messagebox.showinfo("登录结果", "登录成功")
# 登录成功后清空密码输入框
entry_password.delete(0, tk.END)
else:
    messagebox.showerror("登录结果", "登录失败")

# 创建主窗口
root = tk.Tk()
root.title("登录窗口")
root.geometry("300x200")
# 用户名和密码的列表
user_credentials = [
    ('admin', '123456'),
    ('user1', 'password1'),
    ('user2', 'password2')
]
# 创建用户名和密码的 Label 和 Entry
label_username = tk.Label(root, text="用户名:")
entry_username = tk.Entry(root)
label_password = tk.Label(root, text="密码:")
entry_password = tk.Entry(root, show="*") # 使用 show 参数将密码输入变成掩码
# 在密码输入框上绑定回车键事件
entry_password.bind('<Return>', check_login)
# 创建登录按钮
login_button = tk.Button(root, text="登录", command=check_login)
# 布局界面元素
label_username.pack(pady=(10, 0))
entry_username.pack(pady=(0, 10))
label_password.pack(pady=(10, 0))
entry_password.pack(pady=(0, 10))
login_button.pack()
# 运行主事件循环
root.mainloop()
```



暨南大学本科实验报告专用纸(附页)

题目三： 参照例6.9编写利用with语句读取并输出文本文件的程序。并尝试修改程序，由命令行第一个参数确认所输出的文本名。

实验代码：

```
import sys
import chardet
# 获取命令行参数中的文件名
file_name = sys.argv[1]
# 检测文件编码
with open(file_name, 'rb') as f:
    rawdata = f.read()
    result = chardet.detect(rawdata)
    encoding = result['encoding']
# 打开文件
with open(file_name, 'r', encoding='utf-8') as file:
    # 逐行读取并输出
    for i, line in enumerate(file, 1):
        print(f'Line {i}: {line}', end='')
```

实验结果：

```
PS D:\学习\大四上\python_learn\homework_chapter6> python 3_read.py file1.txt
Line 1: This is a test.
Line 2: Python是一种面向对象、解释型计算机程序设计语言。
Line 3: 由Guido van Rossum于1989年发明。
PS D:\学习\大四上\python_learn\homework_chapter6>
```

实验分析和总结：

最初遇到这个错误是因为在尝试使用默认的编码（通常是 gbk）读取文本文件时，遇到了无法识别的字符。于是添加了尝试使用 chardet 库来自动检测；先使用 chardet 库来检测文件的编码，然后根据检测结果来打开文件。这样可以避免因编码不匹配而导致的错误。

```
Traceback (most recent call last):
  File "D:\学习\大四上\python_learn\homework_chapter6\3_read.py", line 8, in <module>
    content = file.read()
              ^^^^^^^^^^^
UnicodeDecodeError: 'gbk' codec can't decode byte 0xaf in position 25: illegal multibyte sequence
```

同时在使用 with open(...) as ... 语句时，Python 会在代码块执行完毕后自动关闭文件，不需要显式调用 close() 方法。这是 with 语句的一个特性，它会在代码块执行结束后自动清理资源。

暨南大学本科实验报告专用纸(附页)

题目四: 参照例6.12编写标准输出流重定向示例程序。并尝试修改程序，从命令行第一个参数中获取n的值，然后将0~n、0~n的2倍值、2的0~n次幂的列表打印输出到out.log文件中。

实验代码:

```
import sys
# 获取命令行参数中的 n 值
n = int(sys.argv[1])
# 保存原始的标准输出流
original_stdout = sys.stdout
# 打开一个文件作为新的标准输出流
with open('out.log', 'w') as f:
    sys.stdout = f
    # 输出需要重定向的内容
    values = list(range(n + 1))
    values_times_2 = [2 * i for i in values]
    powers_of_2 = [2 ** i for i in values]
    print(values)
    print(values_times_2)
    print(powers_of_2)
# 恢复原始的标准输出流
sys.stdout = original_stdout
```

实验结果:

```
PS D:\学习\大四上\python_learn\homework_chapter6> python 4_outlog.py 5
PS D:\学习\大四上\python_learn\homework_chapter6> type out.log
[0, 1, 2, 3, 4, 5]
[0, 2, 4, 6, 8, 10]
[1, 2, 4, 8, 16, 32]
```

实验分析和总结:

一开始一直出现这个报错，这个错误是由于在运行脚本时没有提供足够的命令行参数引起的，后使用终端运行程序时传递了一个整数作为参数。“python your_script.py 5”成功实现

```
E:\Python3.11\python.exe D:\学习\大四上\python_learn\homework_chapter6\4_outlog.py
Traceback (most recent call last):
  File "D:\学习\大四上\python_learn\homework_chapter6\4_outlog.py", line 4, in <module>
    n = int(sys.argv[1])
          ~~~~~^
IndexError: list index out of range
```

暨南大学本科实验报告专用纸(附页)

题目五: 参照例6.18编写过滤器示例程序, 将来自于标准输入中位于指定范围的值写入到标准输出。

实验代码:

```
import tkinter as tk
from tkinter import messagebox

def filter_values():
    min_value = entry_min.get()
    max_value = entry_max.get()

    try:
        min_value = int(min_value)
        max_value = int(max_value)
        if min_value > max_value:
            messagebox.showwarning("警告", "下限不能大于上限")
            return
    except ValueError:
        messagebox.showerror("错误", "请输入有效的整数")
        return

    result_text.configure(state=tk.NORMAL)
    result_text.delete(1.0, tk.END)

    values = input_entry.get().split()

    for value in values:
        try:
            value = int(value)
            if min_value <= value <= max_value:
                result_text.insert(tk.END, str(value) + '\n')
        except ValueError:
            pass

    result_text.configure(state=tk.DISABLED)

# 创建主窗口
root = tk.Tk()
root.title("过滤器")
root.geometry("400x300") # 设置窗口大小

# 创建用户输入部分
label_min = tk.Label(root, text="下限:")
```

Python程序设计课程实验报告

暨南大学本科实验报告专用纸(附页)

```
entry_min = tk.Entry(root)
label_max = tk.Label(root, text="上限:")
entry_max = tk.Entry(root)
input_entry = tk.Entry(root, width=30)

# 提示用户输入多个值
label_input = tk.Label(root, text="请输入多个值（以空格分隔）:")

# 创建“过滤”按钮
filter_button = tk.Button(root, text="过滤", command=filter_values)

# 创建输出部分
result_text = tk.Text(root, height=10, width=30, state=tk.DISABLED)

# 布局界面元素
label_min.grid(row=0, column=0, padx=5, pady=5)
entry_min.grid(row=0, column=1, padx=5, pady=5)
label_max.grid(row=0, column=2, padx=5, pady=5)
entry_max.grid(row=0, column=3, padx=5, pady=5)
label_input.grid(row=1, column=0, columnspan=4, padx=5, pady=5)
input_entry.grid(row=2, column=0, columnspan=4, padx=5, pady=5)
filter_button.grid(row=3, column=0, columnspan=4, padx=5, pady=5)
result_text.grid(row=4, column=0, columnspan=4, padx=5, pady=5)

# 运行主事件循环
root.mainloop()
```

实验结果:



实验分析和总结:

为了实现过滤器，初步的想法是用户可以指定上下限，对用户的输入进行过滤，输出符合用户要求的数值。① 单次输入多个数值，通过提示用户使用空格分隔数字实现；② 确保上下限合法，如果输入下限大于上限，对用户进行警告；③ 对输出部分进行只读的设定，不能修改过滤后输出结果。

暨南大学本科实验报告专用纸(附页)

题目六: 将file1.txt文件中的每行按逆序方式输出到file2.txt中。文件内容如下, 请同学们自行创建file1.txt

file1.txt

This is a test.

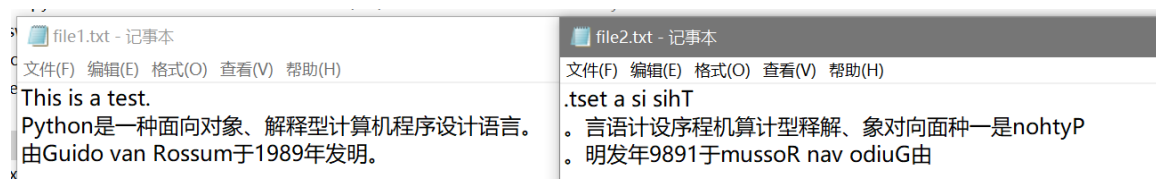
Python是一种面向对象、解释型计算机程序设计语言。

由Guido van Rossum于1989年发明。

实验代码:

```
with open('file1.txt', 'r', encoding='utf-8') as file1, open('file2.txt', 'w', encoding='utf-8') as file2:
    for line in file1:
        reversed_line = line.strip()[::-1]
        file2.write(reversed_line + '\n')
```

实验结果:



实验分析和总结:

在未指定文件的编码格式的时候程序发生报错“UnicodeDecodeError: 'gbk' codec can't decode byte 0xaf in position 25: illegal multibyte sequence”, 这个错误提示表明, 在读取 file1.txt 文件时, Python 遇到了一个无法用 'gbk' 编码解码的字节。

然后试图以二进制模式打开文件, 并使用适当的编码进行处理。

```
# 读取file1.txt的内容, 并将每行逆序写入file2.txt
with open('file1.txt', 'rb') as file1, open('file2.txt', 'w', encoding='utf-8') as file2:
    for line in file1:
        reversed_line = line.strip()[::-1].decode('utf-8', errors='ignore') # 将每行内容逆序
        file2.write(reversed_line + '\n') # 写入到file2.txt
```

发现反转后file2.txt文件内容为乱码, 查阅资料后发现需要指定file1和file2的编码格式, 指定编码格式为“utf-8”后逐行反转成功, 并且没有出现乱码。

在逆序的时候, 首先使用 strip() 去掉行尾的换行符, 然后将其逆序, 使用“line.strip()[::-1]”即可。