深 圳 大 学 实 验 报 告

课程名称：数据挖掘

实验项目名称：实验二 python 编程快速上手

学院：计算机与软件学院

专业：数学与计算机实验班

指导教师：王祎乐

报告人：詹耿羽 学号：2023193026 班级：数计

实验时间：2025 年 3 月 14 日

实验报告提交时间：2025.3.16

教务处制

实验目的与要求：

1. 熟悉 python 基本语法。
2. 掌握 python 编程基础。

试验环境：

操作系统： Win11
编译器：Visual Studio Code

实验内容及过程：

完成以下试题：

题目 1：有两个磁盘文件 A 和 B,各存放一行字母,要求把这两个文件中的信息合并 (按字母顺序排列), 输出到一个新文件 C 中。

注：运行以上程序前，你需要在脚本执行的目录下创建 test1.txt、test2.txt 文件

• 完整代码：

```
if __name__ == '__main__':
    str1 = open("C:\\Users\\詹耿羽\\Desktop\\test1.txt").read()  # 读取文件text1.txt的内容
    str2 = open("C:\\Users\\詹耿羽\\Desktop\\test2.txt").read()  # 读取文件text2.txt的内容
    f = open("C:\\Users\\詹耿羽\\Desktop\\test3.txt", 'w')  # 以写模式打开文件text3.txt

    # 合并字符串并进行排序：
    # 排序规则：1. 按字符的小写字母顺序排序（忽略大小写）
    #           2. 按字符的大小写顺序（大写字母排在小写字母之前）
    f.write(''.join(sorted(str1 + str2, key=lambda x: (x.lower(), x))))  # 合并str1和str2并排序后写入文件
    f.close()  # 关闭文件，确保写入完成

    print(u'txt3写入成功')  # 输出'文件写入成功'的提示信息
```

• 解释

key=lambda x: (x.lower(), x) 是排序的关键部分。这里使用了 lambda 函数来定义排序的规则。

x.lower()：首先按字符的小写形式进行排序，忽略大小写。

x：如果小写字母相同，再根据字母的原始形式进行排序，这意味着大写字母会排在小写字母之前。

• 实验过程与结果

图 test1

图 test2

结果：

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/1.py
txt3写入成功
PS C:\Users\詹耿羽\Desktop\数据挖掘代码>
```

图 test3

题目 2：创建一个名为 names 的空列表，往里面按顺序添加 Lihua、Rain、Jack、Xiuxiu、Peiqi 和 Black 共 6 个元素。再分别按照如下 3 个要求对 names 进行操作（每个操作不相关）：

• 完整代码与结果

```
实验1 > 2.py > ...
1   name = []  # 创建一个空的列表 name
2
3   name.append('Lihua')  # 将 'Lihua' 添加到列表 name 中
4   name.append('Rain')   # 将 'Rain' 添加到列表 name 中
5   name.append('Jack')   # 将 'Jack' 添加到列表 name 中
6   name.append('Xiuxiu') # 将 'Xiuxiu' 添加到列表 name 中
7   name.append('Peiqi')  # 将 'Peiqi' 添加到列表 name 中
8   name.append('Black')  # 将 'Black' 添加到列表 name 中
9
10  print(name)  # 打印整个列表 name 的内容
11
```

```
['Lihua', 'Rain', 'Jack', 'Xiuxiu', 'Peiqi', 'Black']
```

1. 往 names 列表里 Black 前面插入一个 Blue，后面插入 White，输出 names 列表；

• 完整代码与结果

```python
name = []  # 创建一个空的列表 name

name.append('Lihua')  # 将 'Lihua' 添加到列表 name 中
name.append('Rain')   # 将 'Rain' 添加到列表 name 中
name.append('Jack')   # 将 'Jack' 添加到列表 name 中
name.append('Xiuxiu') # 将 'Xiuxiu' 添加到列表 name 中
name.append('Peiqi')  # 将 'Peiqi' 添加到列表 name 中
name.append('Black')  # 将 'Black' 添加到列表 name 中

name.insert(name.index('Black'), 'Blue')  # 找到 'Black' 的索引位置，然后在该位置之前插入 'Blue'
name.insert(name.index('Black') + 1, 'White')  # 找到 'Black' 的索引位置，然后在该位置之后插入 'White'
print(name)  # 打印修改后的列表
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/2.py
['Lihua', 'Rain', 'Jack', 'Xiuxiu', 'Peiqi', 'Blue', 'Black', 'White']
```

2. 把 names 列表中 Xiuxiu 的名字替换成“秀秀”，并输出 names 列表；

• 完整代码与结果

```python
name = []  # 创建一个空的列表 name

name.append('Lihua')  # 将 'Lihua' 添加到列表 name 中
name.append('Rain')   # 将 'Rain' 添加到列表 name 中
name.append('Jack')   # 将 'Jack' 添加到列表 name 中
name.append('Xiuxiu') # 将 'Xiuxiu' 添加到列表 name 中
name.append('Peiqi')  # 将 'Peiqi' 添加到列表 name 中
name.append('Black')  # 将 'Black' 添加到列表 name 中

name.insert(name.index('Black'), 'Blue')  # 找到 'Black' 的索引位置，然后在该位置之前插入 'Blue'
name.insert(name.index('Black') + 1, 'White')  # 找到 'Black' 的索引位置，然后在该位置之后插入 'White'
name[name.index('Xiuxiu')] = '秀秀'  # 将列表中 'Xiuxiu' 替换为 '秀秀'
print(name)  # 打印修改后的列表
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实
['Lihua', 'Rain', 'Jack', '秀秀', 'Peiqi', 'Blue', 'Black', 'White']
```

3. 创建新列表[1,2,3,4,2,5,6,2]，将新列表元素追加到 names 列表末尾，并输出 names 列表；取出 names 列表中索引 2-10 的元素，步长为 2，打印所取出的元素。

• 完整代码与结果

```
name.insert(name.index('Black'), 'Blue')  # 找到 'Black' 的索引位置，然后在该位置之前插入 'Blue'
name.insert(name.index('Black') + 1, 'White')  # 找到 'Black' 的索引位置，然后在该位置之后插入 'White'
name[name.index('Xiuxiu')] = '秀秀'  # 将列表中 'Xiuxiu' 替换为 '秀秀'
numList = [1, 2, 3, 4, 2, 5, 6, 2]  # 创建一个包含数字的列表 numList
name.extend(numList)  # 将 numList 中的所有元素添加到 name 列表的末尾
print(name)  # 打印合并后的列表
temp = name[2:11:2]  # 切片操作，从索引 2 到 11 的元素，每隔 2 个取一个
print(temp)  # 打印切片后的列表
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/2.py
['Lihua', 'Rain', 'Jack', '秀秀', 'Peiqi', 'Blue', 'Black', 'White', 1, 2, 3, 4, 2, 5, 6, 2]
['Jack', 'Peiqi', 'Black', 1, 3]
```

• 总结

代码展示了如何在列表中添加、插入、替换元素，如何将一个列表扩展到另一个列表，如何进行切片操作，及如何访问和打印列表内容。

题目 3：定义一个字典。其中键是字符串，描述清单中的物品，值是一个整型值，说明玩家有多少该物品。例如，字典值{ 'arrow': 12, 'gold coin': 42,'rope': 1, 'torch': 6, 'dagger': 1} 。写一个名为displayInventory()的函数，参数是字典，打印输出物品个数和物品名称，并统计物品总数量。（输出格式参考下图）

```
Inventory:
12 arrow
42 gold coin
1 rope
6 torch
1 dagger
Total number of items: 62
```

• 完整代码与结果

实验1 > 3.py > ...

```
def displayInventory(mp: map):
    print('Inventory:')
    for item in mp.items():
        print(item[1], item[0])
    print('Total number of items:', sum(mp.values()))

if __name__ == '__main__':
    mp = {'rope': 1, 'torch': 6, 'gold coin': 42,
          'dagger': 1, 'arrow': 12}
    displayInventory(mp)
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/3.py
Inventory:
1 rope
6 torch
42 gold coin
1 dagger
12 arrow
Total number of items: 62
```

• 总结

mp.items()返回字典 mp 中所有的键值对（物品和数量）。

for item in mp.items()遍历字典中的每一个键值对，其中 item 是一个元组，包含两个元素：

item[0]是物品名（字典的键）。

item[1]是物品的数量（字典的值）。

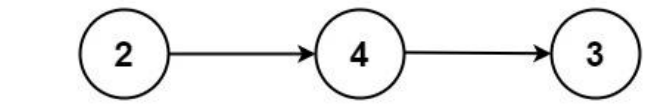
print(item[1], item[0])打印出物品的数量和物品名。注意：输出的顺序是先打印数量，再打印物品名。

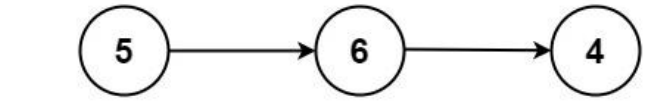
题目 4：给你两个非空的链表，表示两个非负的整数。它们每位数字都是按照逆序的方式存储的，并且每个节点只能存储 一位数字。

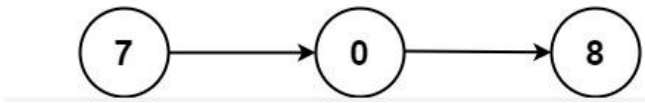
请你将两个数相加，并以相同形式返回一个表示和的链表。

你可以假设除了数字 0 之外，这两个数都不会以 0 开头。

示例 1：







存储结构：

```
class ListNode(object):
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next
```

输入：l1 = [2,4,3], l2 = [5,6,4]

输出：[7,0,8]

解释： 342 + 465 = 807.

• 完整代码：

```
class ListNode(object):
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def addTwoNumbers(l1: ListNode, l2: ListNode) -> ListNode:
    dummy_head = ListNode(0)
    current = dummy_head
    carry = 0

    while l1 or l2 or carry:
        val1 = l1.val if l1 else 0
        val2 = l2.val if l2 else 0
        total = val1 + val2 + carry
        carry = total // 10
        current.next = ListNode(total % 10)

        current = current.next
        if l1:
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next

# 辅助函数：将输入的数字列表转换为链表
def to_linked_list(lst):
    head = current = ListNode(lst[0])
    for num in lst[1:]:
        current.next = ListNode(num)
        current = current.next
    return head

# 辅助函数：将链表转换为输出的数字列表
def to_list(node):
    result = []
    while node:
        result.append(node.val)
        node = node.next
    return result

```

```
# 主函数：接收用户输入并输出结果
def main():
    # 获取用户输入
    l1 = list(map(int, input("（以空格分隔）l1 = ").split()))
    l2 = list(map(int, input("（以空格分隔）l2 = ").split()))

    # 转换为链表
    l1_linked = to_linked_list(l1)
    l2_linked = to_linked_list(l2)

    # 计算链表之和
    result = addTwoNumbers(l1_linked, l2_linked)

    # 转换结果为列表并输出
    print("结果链表为:", to_list(result))

# 运行主函数
if __name__ == "__main__":
    main()
```

• 解释：

addTwoNumbers 函数是实现链表相加的函数。输入是两个链表，输出是它们相加后的结果链表。

to_linked_list(lst) 将一个列表转换成链表，用于将输入的数字列表转化为链表。

to_list(node) 将链表转换回列表，用于输出结果链表。

ListNode 类是一个链表节点的类，用来表示每个链表的节点。每个节点包含一个 val（值）和一个指向下一个节点的指针 next。

• 结果：

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/5.py
（以空格分隔）l1 = 2 4 3
（以空格分隔）l2 = 5 6 4
结果链表为: [7, 0, 8]
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/5.py
（以空格分隔）l1 = 1 2 3
（以空格分隔）l2 = 4 5 6
结果链表为: [5, 7, 9]
```

题目 5：给定一个字符串，请你找出其中不含有重复字符的最长子串的长度。

示例 1:

输入: s = "abcabcbb"

输出: 3

解释: 因为无重复字符的最长子串是 "abc"，所以其长度为 3。

示例 2:
输入: s = "bbbbb"
输出: 1
解释: 因为无重复字符的最长子串是 "b"，所以其长度为 1。

• 完整代码与结果

```python
def lengthOfLongestSubstring(s: str) -> int:
    char_map = {}
    left = 0
    max_length = 0

    for right in range(len(s)):
        if s[right] in char_map and char_map[s[right]] >= left:
            left = char_map[s[right]] + 1
        char_map[s[right]] = right
        max_length = max(max_length, right - left + 1)

    return max_length

# 输入
s = input("请输入一个字符串: ")

# 输出
print("最长无重复子串的长度为:", lengthOfLongestSubstring(s))
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/4.py
请输入一个字符串: abcabcbb
最长无重复子串的长度为: 3
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/4.py
请输入一个字符串: bbbbb
最长无重复子串的长度为: 1
```

```
PS C:\Users\詹耿羽\Desktop\数据挖掘代码> & C:/miniconda3/python.exe c:/Users/詹耿羽/Desktop/数据挖掘代码/实验1/4.py
请输入一个字符串: asasas
最长无重复子串的长度为: 2
PS C:\Users\詹耿羽\Desktop\数据挖掘代码>
```

• 总结

lengthOfLongestSubstring 函数，接受一个字符串 s 作为输入，返回一个整数——最长无重复子串的长度。

通过调用 lengthOfLongestSubstring(s)函数，并打印出返回的最长无重复子串的长度。

实验收获：

在掌握了 Python 编程语言的基本操作之后，我为接下来的实验和深入学习奠定了坚实的基础。这不仅为进行更高级的编程任务提供了必要的工具，还让我能够开始探索数据处理和挖掘任务的可能性。

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。