中山大学计算机学院

人工智能

本科生实验报告

（2022 学年春季学期）

课程名称：Artificial Intelligence

教学班级	202320346	专业（方向）	计算机科学与技术
学号	21312450	姓名	林隽哲

一、 Question1: Binary Search

（1） Question

1.二分查找

给定一个 n 个元素有序的（升序）整型数组 nums 和一个目标值 target ，写一个函数 BinarySearch 搜索 nums 中的 target ，如果目标值存在返回下标，否则返回 -1 。

（2） Code

```python
def BinarySearch(nums, target):
    """
    :param nums: list[int]
    :param targe: int
    :return int
    """

    left, right = 0, len(nums) - 1

    while left <= right:
        mid = left + (right - left) // 2
        if nums[mid] == target:
            return mid
        elif nums[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1
```

（3） Test Case

```python
def BinarySearch(nums, target):
    """
    :param nums: list[int]
    :param targe: int
    :return int
    """

    left, right = 0, len(nums) - 1

    while left <= right:
        mid = left + (right - left) // 2
        if nums[mid] == target:
            return mid
        elif nums[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1

if __name__ == "__main__":
    nums = [1, 2, 3, 4, 5, 6, 7, 8, 9]
    target = 5
    print(BinarySearch(nums, target))  # 4
    target = 10
    print(BinarySearch(nums, target))  # -1
```

Print Output:

```
4
-1
```

Variables:

```
-{
    nums: -[
        1,
        2,
        3,
        4,
        5,
        6,
        7,
        8,
        9
    ],
    target: 10
}
```

二、 Question2: Matrix Calculation

（1） Question

2.矩阵加法,乘法

给定两个$n\times n$的整型矩阵 A 和 B ，写两个函数 MatrixAdd 和 MatrixMul，分别得出这两个矩阵加法和乘法的结果.

两个矩阵的数据类型为嵌套列表, 即 list[list]，且满足 len(list)==n, len(list[0])==n .

注意不要打乱原矩阵 A 和 B 中的数据.

（2） Code

```python
def MatrixAdd(A, B):
    """
    :param A: list[lsit[int]]
    :param B: list[lsit[int]]
    :return:  list[list[int]]
    """
    return [[A[i][j] + B[i][j] for j in range(len(A[0]))] for i in range(len(A))]

def MatrixMul(A, B):
    """
    :param A: list[lsit[int]]
    :param B: list[lsit[int]]
    :return:  list[list[int]]
    """
    return [[sum(A[i][k] * B[k][j] for k in range(len(A[0]))) for j in range(len(B[0]))] for i in range(len(A))]

```

（3） Test Case

```python
def MatrixAdd(A, B):
    """
    :param A: list[lsit[int]]
    :param B: list[lsit[int]]
    :return:  list[list[int]]
    """
    return [[A[i][j] + B[i][j] for j in range(len(A[0]))] for i in range(len(A))]

def MatrixMul(A, B):
    """
    :param A: list[lsit[int]]
    :param B: list[lsit[int]]
    :return:  list[list[int]]
    """
    return [[sum(A[i][k] * B[k][j] for k in range(len(A[0]))) for j in range(len(B[0]))]


if __name__ == "__main__":

    A = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
    B = [[1, 0, 0], [0, 1, 0], [0, 0, 1]]

    print(MatrixAdd(A, B))  # [[2, 2, 3], [4, 6, 6], [7, 8, 10]]
    print(MatrixMul(A, B))  # [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

Print Output:

```
[[2, 2, 3], [4, 6, 6], [7, 8, 10]]
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

Variables:

```
-{
    A: -[
       +[3 items],
       +[3 items],
       +[3 items]
    ],
    B: -[
       +[3 items],
       +[3 items],
       +[3 items]
    ]
}
```

三、 Question3: Reverse Key Value

（1） Question

3.字典遍历

给定非空字典 dict1 ，其键为姓名，值是学号. 写一个函数 ReverseKeyValue 返回另一个字典，其键是学号，值是姓名.

例如，dict1={'Alice':'001', 'Bob':'002}，则 ReverseKeyValue(dict1) 返回的结果是 {'001':'Alice', '002':'Bob'}.

（2） Code

```python
def ReverseKeyValue(dict1):
    """
    :param dict1: dict
    :return: dict
    """
    return {v: k for k, v in dict1.items()}
```

（3） Test Case

```python
def ReverseKeyValue(dict1):
    """
    :param dict1: dict
    :return: dict
    """
    return {v: k for k, v in dict1.items()}

if __name__ == "__main__":
    import random
    import string
    dict1 = {"kobayashi" : "21312450"}
    for i in range(10):
        r_name = ''.join(random.sample(string.ascii_letters + string.digits, 8))
        r_id = ''.join(random.sample(string.digits, 8))
        dict1[r_name] = r_id

    print(ReverseKeyValue(dict1))  # {'21312450': 'kobayashi'}
```

Print Output:

```
{'21312450': 'kobayashi', '07316894': '7eAMqx0E', '49672850': '3ZXPKVjz',
'86147590': 'CbJkFDv6', '01483679': 'xvuFDa3E', '38207159': 'KxCUSFW4',
'79624538': 'tHoEA9iy', '09743821': 'TzQGAMLp', '41935782': 'Dr9lGQFL',
'14938267': 'SO1xVZvU', '06837542': 'L5QKUgTR'}
```

Variables:

```
-{
    dict1: -{
        kobayashi: "21312450",
        7eAMqx0E: "07316894",
        3ZXPKVjz: "49672850",
        CbJkFDv6: "86147590",
        xvuFDa3E: "01483679",
        KxCUSFW4: "38207159",
        tHoEA9iy: "79624538",
        TzQGAMLp: "09743821",
        Dr9lGQFL: "41935782",
        SO1xVZvU: "14938267",
        L5QKUgTR: "06837542"
    },
    i: 9,
    r_name: "L5QKUgTR",
    r_id: "06837542"
}
```

四、 Question4: Student Data

（1） Question

给定 student_data.txt 文本文件, 每一行是一名学生的信息, 从左到右分别是该学生的姓名,学号, 性别和年龄, 每个属性以空格间隔. 数据类型如下:

```
name: str # 姓名
stu_num: str # 学号
gender: str # 性别, "M"为男性, "F"为女性
age: int # 年龄
```

编写 StuData 类, 须有以下方法:

- 构造函数(即 __init__), 以文件名(str 类型, 带 .txt 后缀)为输入, 读取文件中的学生信息, 存储到类成员 data 中. data 的数据类型为 list , 其中每一个学生的信息以列表方式存储. 例如, 读入一行学生信息"Aaron 243 M 18", 则 data 变为

  ```
  [["Aaron", "243", "M", 18]]
  ```

  再读入学生信息"Eric 249 M 19", 则 data 变为

  ```
  [["Aaron", "243", "M", 18], ["Eric", "249", "M", 19]]
  ```

- AddData 方法, 以单个学生的信息作为输入, 存储到 data 中. 调用该方法的参数形式为学生属性的4个关键字实参. 例如, 执行 self.AddData(name="Bob", stu_num="003", gender="M", age=20) 后, data 变为

  ```
  [["Aaron", "243", "M", 18], ["Eric", "249", "M", 19], ["Bob", "003", "M", 20]]
  ```

- SortData 方法, 以学生某个属性(str 类型, 是 'name', 'stu_num', 'gender', 'age' 的其中之一)作为输入, 将 data 按该属性从小到大排序. 可以假定不会输入非学生属性的字符串. 例如, 执行 self.Sort('stu_num') 后, data 的学生信息按学号从小到大排序, 变为

  ```
  [["Bob", "003", "M", 20],["Aaron", "243", "M", 18], ["Eric", "249", "M", 19]]
  ```

- ExportFile 方法, 以导出的文件名(str 类型, 带 .txt 后缀)为输入, 新建一个 txt 文件, 将 data 中的数据按当前列表顺序导出到该文件内, 格式同原 student_data.txt 文本文件, 即"姓名 学号 性别 年龄". 例如, 调用 self.ExportFile('new_stu_data.txt') , 则将 data 中数据导出 new_stu_data.txt 文件.

提示

1. 学号信息数据类型为 str 而不是 int . 可以假定学号都由3个0-9数字组成.
2. 可以假设每个学生有且只有这4个属性, 且不会缺省.
3. 可以在类中编写其他辅助方法, 也可以在同一个代码文件中编写其他函数或类供自己调用.
4. 本次作业中, 类方法的输入参数名可自定义, 但参数数据类型需保证测试程序正常运行. 请不要更改类方法的名.
5. 调试代码时请将 student_data.txt 文件与代码文件放到同一文件夹中, 以避免不必要的bug.

（2） Code

```python
class StuData:

    __slots__ = ['data']

    def __init__(self, filename: str) -> None:

        self.data = []

        with open(filename, 'r') as f:
            for line in f:
                data_line = line.strip().split(' ')
                data_line[3] = int(data_line[3])
                self.data.append(data_line)

    def __str__(self) -> str:
        return "\t\t".join(['name', 'stu_num', 'gender', 'age']) + '\n' + '\n'.join(['\t\t'.join([str(i) for i in line]) for line in self.data]) + '\n'

    def AddData(self, name: str, stu_num: str, gender: str, age: int) -> None:
        self.data.append([name, stu_num, gender, age])

    def SortData(self, attr: str) -> None:
        # if you want to throw an exception when the key is not in the dictionary,
        # you can use dict instead of defaultdict
        from collections import defaultdict
        key_map = defaultdict(lambda: 0, {'name': 0, 'stu_num': 1, 'gender': 2, 'age': 3})
        self.data.sort(key=lambda x: x[key_map[attr]])

    def ExportFile(self, filename: str) -> None:
        with open(filename, 'w') as f:
            for line in self.data:
                f.write(' '.join([str(i) for i in line]) + '\n')
```

（3） Test Case

```
class StuData:
                f.write(' '.join([str(i) for i in line]) + '\n')


if __name__ == "__main__":
    import os

    filename = os.path.join(os.path.dirname(__file__), 'student_data.txt')

    if not os.path.exists(filename):
        import random
        import string
        with open(filename, 'w') as f:
            for i in range(1000):
                random_name = ''.join(random.sample(string.ascii_letters, 5))
                random_stu_num = ''.join(random.sample(string.digits, 8))
                random_gender = random.choice(['F', 'M'])
                random_age = random.randint(18, 25)
                f.write(f'{random_name} {random_stu_num} {random_gender} {random_age}\n')

    stu = StuData(filename)

    stu.SortData('age')

    stu.ExportFile(os.path.join(os.path.dirname(__file__), 'student_data_sorted.txt'))

    print(stu)
```

Print Output:

name	stu_num	gender	age
Leo	092	M	17
Aaron	243	M	18
Sam	230	F	18
Eric	249	M	19
Alex	812	M	19
Ruth	942	M	19
Cynthia	920	F	19
Beryl	091	F	20

Variables:

```
-{
    StuData: -{
        py/type: "__main__.StuData"
    },
    filename: +"d:\Program\Code\2_Computer_Science\Artificial_Intelligence\experiment\
    stu: -{
        py/object: "__main__.StuData",
        data: +[8 items]
    }
}
```

五、 算法原理

在 Question1 中使用了简单的函数递归调用实现了折半查找的逻辑，其余题目中只涉及到基本的 Python 的基础语法知识。

六、 创新点&优化

出于平日的习惯，本次实验的代码中一些简单的列表或字典生成均使用了生成器。
在 StuData 的编写中，__slots__属性可以用于去除生成对象中的冗余属性，有利于节省内存。
而在类中的 SortData 函数中，我选择使用 defaultdict 实现当函数出现错误 attr 输入时的缺省处理。若想让函数直接抛出异常，直接使用普通的 dict 即可。

七、 思考题

课堂练习题：**猜数字游戏：**

```
In [1]: import random

In [2]: num = random.randint(0, 100)

In [3]: num
Out[3]: 11

In [4]: while True:
   ...:     in_num = int(input("input number:"))
   ...:     if in_num == num:
   ...:         print("YOU WIN!")
   ...:         break
   ...:     print(("too big" if in_num > num else "too small"))
   ...:
input number:10
too small
input number:12
too big
input number:11
YOU WIN!
```

课堂练习题：**一行代码求出 1 到 100 的和：**

```
PS C:\Users\KOBAYASHI> ipython
Python 3.10.9 | packaged by Anaconda, Inc. | (main, Mar  1 2023, 18:18:15) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: sum(range(1, 101))
Out[1]: 5050
```

八、 参考资料：无