中山大学计算机学院

人工智能

本科生实验报告

课程名称：Artificial Intelligence

学号	22336327	姓名	庄云皓

一、 实验题目

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.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.

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'}.

二、 实验内容

二分查找

算法原理：对于一个有序递增序列，选序列中间的数字与目标值比较，如果相等则返回答案，否则如果中间值小于目标值，则排除中间数字及其左的数字；如果中间值大于目标值，则排除中间数字及其左的数字。

注意 while 中的条件是 low<=high 而非 low<high。我们使用 low < high 时，如果找到目标元素，循环可能会在执行完 low++操作后立即结束，这样下一次循环的 low 值就会比 high 小，导致

跳过了一次数组元素的访问，从而可能漏掉了一些情况（尤其是当目标元素是数组的最后一个元素时）

因为

关键代码展示：

```
    while(low <= high):
        mid = int(low +(high - low)>>1)#注意索引要是整数
        if(target < nums[mid]):
            high = mid - 1
        elif(target > nums[mid]):
            low = mid + 1
        else:
            return mid
    return -1
```

创新点&优化：使用 mid = int(low +(high - low)//2)防止溢出

若其中有重复元素：返回最左边的查找到的元素的下标

```
    while(low <= high):
        mid = int(low +(high - low)//2)#注意整除
        if(target < nums[mid]):
            high = mid - 1
        elif(target > nums[mid]):
            low = mid + 1
        else:
            while( mid > low and nums[mid-1]==nums[mid]):
                mid = mid - 1
            return mid
    return -1
```

对开始时就数组为空或查找元素不存在进行判断：

```
if nums == None or nums[0]>target or nums[high]<target:
        return -1
```

矩阵加法

算法原理：

矩阵加法是矩阵中每个元素相加，矩阵乘法是$C_{i,j} = \sum_{k=1}^{n} A_{i,k}B_{k,j}$

先初始化 C 矩阵为全 0 的矩阵

通过遍历矩阵中每个元素按照对应的规则实现。

关键代码展示：

```
def MatrixAdd(A, B):

    C = [ [0 for _ in range(len(A[0]))] for _ in range(len(A))]
    for i in range(len(A)):
        for j in range(len(A[0])):
```

```
            C[i][j] = A[i][j] + B[i][j]
    return C


def MatrixMul(A, B):
    n = len(A)
    C = [ [0 for _ in range(n)] for _ in range(n)]
    for i in range(n):
        for j in range(n):
            for k in range (n):
                C[i][j] += A[i][k] * B[k][j]#k
    return C
```

创新点&优化：在矩阵加法中可采用列表推导式改进，使代码看上去更加简洁。

```
C=[[(A[i][j]+B[i][j]) for j in range(len(A[0]))] for i in range(A)]
```

当不是 n*n 矩阵时可以加上判断条件看看是否能相乘

```
if(len(A[0])!=len(B)):
        print("不能相乘")
        return None
```

```
    C = [ [0 for _ in range(len(B[0]))] for _ in range(len(A))]
    for i in range(len(A)):
        for j in range(len(B[0])):
            for k in range (len(B)):
                C[i][j] += A[i][k] * B[k][j]#k
```

字典遍历

算法原理：如下面的代码所示，将列表从 dict1 通过 dict1.items()函数中取出 key,value 并倒过来形成一个 generator object，再用将其转化为 dict

关键代码展示：

```
inv = dict([val,key] for key,val in dict1.items())#括号的位置
```

参考链接[5]还有另外实现的方式：

```
inv = dict(zip(dict1.values(),dict2.keys()))
```

Zip 函数对象中对应的元素打包成一个个元组，然后返回由这些元组组成的列表，最后再用 dict 函数将其转化为字典。

三、 实验结果及分析

1.1 二分查找

```python
if __name__ == "__main__":
    nums = list(range(1,100000,2))
    #nums = [1,2,51,51,51,51,78,90]
    target = 51
    print("the index is:",BinarySearch(nums, target))
    target = 10
    print("the index is:",BinarySearch(nums, target))
    """

    :param nums: list[int]

    :param target: int

    :return: int
```

问题 输出 调试控制台 终端 端口 注释

```
the index is: 25
the index is: -1
PS D:\Data\ai2024\第一周\Code>
```

1.2 矩阵相乘相加

```python
if __name__ == "__main__":
    r = c = 3
    A = [[1,2,3],[4,5,6],[7,8,9]]
    B = [[1,2,3],[4,5,6],[7,8,9]]
    # l1=[[9,9,9],[9,9,9]]
    # l2=[[1,1],[1,1],[1,1]]
    print(MatrixAdd(A, B))
    print(MatrixMul(A, B))
    #print(MatrixMul(l1, l2))
```

问题 输出 调试控制台 终端 端口 注释

```
PS D:\Data\ai2024> d:; cd 'd:\Data\ai2024'; & 'd:\Apps\anaconda3\python.exe' 'c:\Users\ykylcxlx\.vscode\extensions\ms-python.debugpy-2024.3.10542132-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '8016' '--' 'd:\Data\ai2024\字典遍历.py'
None
PS D:\Data\ai2024> d:; cd 'd:\Data\ai2024'; & 'd:\Apps\anaconda3\python.exe' 'c:\Users\ykylcxlx\.vscode\extensions\ms-python.debugpy-2024.3.10542132-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '8546' '--' 'd:\Data\ai2024\字典遍历.py'
None
PS D:\Data\ai2024> d:; cd 'd:\Data\ai2024'; & 'd:\Apps\anaconda3\python.exe' 'c:\Users\ykylcxlx\.vscode\extensions\ms-python.debugpy-2024.3.10542132-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '8923' '--' 'd:\Data\ai2024\矩阵加法乘法.py'
[[2, 4, 6], [8, 10, 12], [14, 16, 18]]
[[30, 36, 42], [66, 81, 96], [102, 126, 150]]
PS D:\Data\ai2024>
```

1.3 字典遍历

```python
if __name__ == "__main__":
        dict1 ={'Alice':'001', 'Bob':'002'}
        print(ReverseKeyValue(dict1))
```

问题 输出 调试控制台 终端 端口 注释

```
PS D:\Data\ai2024> python -u "d:\Data\ai2024\字典遍历.py"
{'001': 'Alice', '002': 'Bob'}
PS D:\Data\ai2024>
```

四、 参考资料

[1]【二分查找】详细图解_二分查找法流程图-CSDN 博客
[2]二分法查找及有重复值的二分法_重复有序数组二分查找-CSDN 博客
[3]算法 | 【二分查询】进阶与优化 ——区间查询、递归查询、0.618 优化... - 我叫 RT - 博客园 (cnblogs.com)
[4]Python 列表实现矩阵的创建、输入输出、转化转置、加减乘运算并设计一个矩阵计算器 GUI 界面_python 的列表可以进行矩阵运算吗-CSDN 博客
[5]Python 将字典中的键值对反转方法_py 字典颠倒键值-CSDN 博客