Assignment #4: 排序、栈、队列和树

Updated 0005 GMT+8 March 11, 2024

2024 spring, Complied by 同学的姓名、院系

说明：

1）The complete process to learn DSA from scratch can be broken into 4 parts:

Learn about Time complexities, learn the basics of individual Data Structures, learn the basics of Algorithms, and practice Problems.

2）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

3）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。

4）如果不能在截止前提交作业，请写明原因。

编程环境

（请改为同学的操作系统、编程环境等）

操作系统：macOS Ventura 13.4.1 (c)

Python编程环境：Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境：Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

1. 题目

05902: 双端队列

http://cs101.openjudge.cn/practice/05902/

思路：

调用deque即可

代码

```
'''
2100017810 刘思瑞
'''
from queue import deque

def imple(a,b):
```

```
    global d
    if a ==1:
        d.append(b)
    else:
        if b:
            d.pop()
        else:
            d.popleft()

n = int(input())
for i in range(n):
    num = int(input())
    d  = deque()
    for j in range(num):
        a,b = map(int,input().split())
        imple(a,b)
    if d:
        for i in d:
            print(i,end=' ')
    else:
        print('NULL',end=' ')
    print()
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
from queue import deque

def imple(a,b):
    global d
    if a ==1:
        d.append(b)
    else:
        if b:
            d.pop()
        else:
            d.popleft()

n = int(input())
for i in range(n):
    num = int(input())
    d  = deque()
    for j in range(num):
        a,b = map(int,input().split())
        imple(a,b)
    if d:
        for i in d:
            print(i,end=' ')
    else:
        print('NULL',end=' ')
    print()
```

02694: 波兰表达式

http://cs101.openjudge.cn/practice/02694/

思路:

前面是用递归写的，直接贴了上学期的递归的程序，后面第二部分用stack实现了，确实正向的思维简化了许多

代码

```
'''
刘思瑞 2100017810
'''
def calcu(calculate,i,j):
    global calcull
    if calculate[i+1] not in calcull:
        if calculate[i+2] not in calcull:
            calculate[i] =
str(eval(calculate[i+1]+calculate[i]+calculate[i+2]))
```

```
            del calculate[i+1]
            del calculate[i+1]
            i = j[-1]
            j = j[:-1]
        else:
            j.append(i)
            i = i+2
    else:
        j.append(i)
        i = i+1
    return calculate, i, j


calcull = ['+','-','*','/']
calculate = list(input().split())
i = 0
j = [0]
while True:
    calculate, i ,j = calcu(calculate,i,j)
    if len(calculate) == 1:
        break
print('%.6f' % float(calculate[0]))
```

代码运行截图

状态：Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def calcu(calculate,i,j):
    global calcull
    if calculate[i+1] not in calcull:
        if calculate[i+2] not in calcull:
            calculate[i] = str(eval(calculate[i+1]+calculate[i]+calculat
            del calculate[i+1]
            del calculate[i+1]
            i = j[-1]
            j = j[:-1]
        else:
            j.append(i)
            i = i+2
    else:
        j.append(i)
        i = i+1
    return calculate, i, j


calcull = ['+','-','*','/']
calculate = list(input().split())
i = 0
j = [0]
while True:
    calculate, i ,j = calcu(calculate,i,j)
    if len(calculate) == 1:
        break
print('%.6f' % float(calculate[0]))
```

代码

```
'''
2100017810 刘思瑞
'''
from queue import deque
def calcu():
    global m,n
    a = m.pop()
    if a in ('+','-','*','/'):
        b = n.pop()
        c = n.pop()
        n.append(str(eval(b+a+c)))
    else:
        n.append(a)

s = list(input().split())
m = deque(s)
n = deque()
while True:
```

```
    calcu()
    if not m:
        break
print('%.6f' %float(n[0]))
```

代码运行截图

状态：Accepted

源代码

```
'''
2100017810 刘思瑞
'''
from queue import deque
def calcu():
    global m,n
    a = m.pop()
    if a in ('+','-','*','/'):
        b = n.pop()
        c = n.pop()
        n.append(str(eval(b+a+c)))
    else:
        n.append(a)

s = list(input().split())
m = deque(s)
n = deque()
while True:
    calcu()
    if not m:
        break
print('%.6f' %float(n[0]))
```

24591: 中序表达式转后序表达式

http://cs101.openjudge.cn/practice/24591/

思路：

场调度算法

代码

```
def inp(s):
    import re
    s=re.split(r'([\(\)\+\-\*\/])',s)
    s=[item for item in s if item.strip()]
    return s

num = int(input())
```

```
for j in range(num):
    stack = []
    output = []
    dic = {'+':1,'-':1,'*':2,'/':2}
    for i in inp(input()):
        if not i in '+-*/()':
            output.append(i)
        else:
            if i == '(':
                stack.append(i)
            elif i == ')':
                while True:
                    if stack[-1] == '(':
                        stack.pop()
                        break
                    output.append(stack.pop())
            else:
                if not stack or stack[-1]=='(' or dic[i] > dic[stack[-1]]:
                    stack.append(i)
                else:
                    while True:
                        output.append(stack.pop())
                        if not stack or stack[-1]=='(' or dic[i] >
dic[stack[-1]]:
                            stack.append(i)
                            break
    while stack:
        output.append(stack.pop())
    print(' '.join(output))
```

代码运行截图

状态: Accepted

源代码

```
def inp(s):
    import re
    s=re.split(r'([\(\)\+\-\*\/])',s)
    s=[item for item in s if item.strip()]
    return s

num = int(input())
for j in range(num):
    stack = []
    output = []
    dic = {'+':1,'-':1,'*':2,'/':2}
    for i in inp(input()):
        if not i in '+-*/()':
            output.append(i)
        else:
            if i == '(':
                stack.append(i)
            elif i == ')':
                while True:
                    if stack[-1] == '(':
                        stack.pop()
                        break
                    output.append(stack.pop())
            else:
                if not stack or stack[-1]=='(' or dic[i] > dic[stack[-1]
                    stack.append(i)
                else:
                    while True:
                        output.append(stack.pop())
```

22068: 合法出栈序列

http://cs101.openjudge.cn/practice/22068/

思路:

代码

```
'''
2100017810 刘思瑞
'''
def is_rational(s,m):
    stack = []
    if len(m) != len(s):
        return False
    while True:
        if not m:
```

```
            return True
        yemp = m.pop()
        while (not stack or stack[-1] != yemp) and s:
            stack.append(s.pop(0))
        if not stack or stack[-1] != yemp:
            return False
        stack.pop()

s = [i for i in input().strip()]
out = {True:'YES',False:'NO'}
while True:
    try:
            m = [i for i in input().strip()]
            print(out[is_rational(s[::],m[::-1])])
    except EOFError:
        break
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
def is_rational(s,m):
    stack = []
    if len(m) != len(s):
        return False
    while True:
        if not m:
            return True
        yemp = m.pop()
        while (not stack or stack[-1] != yemp) and s:
            stack.append(s.pop(0))
        if not stack or stack[-1] != yemp:
            return False
        stack.pop()

s = [i for i in input().strip()]
out = {True:'YES',False:'NO'}
while True:
    try:
            m = [i for i in input().strip()]
            print(out[is_rational(s[::],m[::-1])])
    except EOFError:
        break
```

06646: 二叉树的深度

http://cs101.openjudge.cn/practice/06646/

思路：

递归，第一个递归更简便，但是写完第二个才想到

代码

```
'''
2100017810 刘思瑞
'''
from queue import deque
def search(tree,i):
    if i == -2:
        return 0
    return max(search(tree,tree[i][0]),search(tree,tree[i][1]))+1
tree = []
n = int(input())
for i in range(n):
    a,b = map(int,input().split())
    tree.append([a-1,b-1])
print(search(tree,0))
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
from queue import deque
def search(tree,i):
    if i == -2:
        return 0
    return max(search(tree,tree[i][0]),search(tree,tree[i][1]))+1
tree = []
n = int(input())
for i in range(n):
    a,b = map(int,input().split())
    tree.append([a-1,b-1])
print(search(tree,0))
```

代码

```
'''
2100017810 刘思瑞
'''
leng = []
```

```
def search(tree,lens,start):
    global leng
    if start == -2:
        leng.append(lens)
        return
    for i in tree[start]:
        search(tree,lens+1,i)
    return

tree = []
n = int(input())
for i in range(n):
    a,b = map(int,input().split())
    tree.append([a-1,b-1])
search(tree,0,0)
print(max(leng))
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
leng = []
def search(tree,lens,start):
    global leng
    if start == -2:
        leng.append(lens)
        return
    for i in tree[start]:
        search(tree,lens+1,i)
    return

tree = []
n = int(input())
for i in range(n):
    a,b = map(int,input().split())
    tree.append([a-1,b-1])
search(tree,0,0)
print(max(leng))
```

02299: Ultra-QuickSort

http://cs101.openjudge.cn/practice/02299/

思路:

归并排序

代码

```python
def mergeSort(arr):
    import math
    if(len(arr)<2):
        return arr , 0
    middle = math.floor(len(arr)/2)
    left, inv_left = mergeSort(arr[:middle])
    right, inv_right = mergeSort(arr[middle:])
    merged, inv_merge = merge(left, right)
    return merged, inv_left + inv_right + inv_merge
def merge(left,right):
    result = []
    count = 0
    i,j = 0,0
    while i < len(left) and j < len(right):
        if left[i] <= right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1
            count += len(left) - i
    result += left[i:]
    result += right[j:]

    return result, count

while True:
    n = int(input())
    if n == 0:
        break
    arr = []
    for i in range(n):
        arr.append(int(input()))
    __, times = mergeSort(arr)
    print(times)
```

代码运行截图

状态：Accepted

源代码

```
def mergeSort(arr):
    import math
    if(len(arr)<2):
        return arr , 0
    middle = math.floor(len(arr)/2)
    left, inv_left = mergeSort(arr[:middle])
    right, inv_right = mergeSort(arr[middle:])
    merged, inv_merge = merge(left, right)
    return merged, inv_left + inv_right + inv_merge
def merge(left,right):
    result = []
    count = 0
    i,j = 0,0
    while i < len(left) and j < len(right):
        if left[i] <= right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1
            count += len(left) - i
    result += left[i:]
    result += right[j:]

    return result, count

while True:
    n = int(input())
    if n == 0:
```

2. 学习总结和收获

本周实验课太多，暂时没有做每日选做，以后补上