# Assignment #4: 排序、栈、队列和树

Updated 0005 GMT+8 March 11, 2024

2024 spring, Complied by 尹柚鑫 光华管理学院 2100015878

**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：Windows11

Python编程环境：jupter notebook

# 1. 题目

## 05902: 双端队列

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

耗时：20,mins

思路：利用python自带的队列，进行简单的操作即可

代码

```
#
from collections import deque
class MyDeque:
    def __init__(self):
        self.deque = deque()

    def append_right(self, item):
        self.deque.append(item)
```

```
    def pop_left(self):
        return self.deque.popleft()

    def pop_right(self):
        return self.deque.pop()

    def is_empty(self):
        return len(self.deque) == 0

    def size(self):
        return len(self.deque)

    def __iter__(self):
        return iter(self.deque)
n=int(input())
for group in range(n):
    number=int(input())
    queue=MyDeque()

    for i in range(number):
        operation=input().split()
        op_type=int(operation[0])

        if op_type==1:
            queue.append_right(int(operation[1]))
        elif op_type==2:
            if int(operation[1])==0:
                queue.pop_left()
            elif int(operation[1])==1:
                queue.pop_right()
    if queue.size()==0:
        print("NULL")
    else:
        print(' '.join(map(str, queue)))
```

代码运行截图 ==（至少包含有"Accepted"）==

#44269775提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque
class MyDeque:
    def __init__(self):
        self.deque = deque()

    def append_right(self, item):
        self.deque.append(item)

    def pop_left(self):
        return self.deque.popleft()

    def pop_right(self):
        return self.deque.pop()

    def is_empty(self):
        return len(self.deque) == 0

    def size(self):
        return len(self.deque)

    def __iter__(self):
        return iter(self.deque)
n=int(input())
for group in range(n):
    number=int(input())
```

基本信息

#: 44269775
题目: 05902
提交人: 尹柚鑫(2100015878)
内存: 3672kB
时间: 43ms
语言: Python3
提交时间: 2024-03-17 16:12:43

# 02694: 波兰表达式

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

耗时：10mins

思路：使用栈的思想，将输入倒序读入栈，如果是数字，则压入栈，如果是运算符，则栈弹出两个数做运算，再把结果压入栈，最后栈只有一个数值，就是最终结果

代码

```
#
lis=input().split()[::-1]
operator=['+','-','*','/']
stack=[]
for i in lis:
    if i in operator:
        a=stack.pop()
        b=stack.pop()
        if i == '+':
            c=a+b
        elif i=='*':
            c=a*b
        elif i=='/':
            c=a/b
        elif i=='-':
            c=a-b
        stack.append(c)
    else:
        stack.append(float(i))
d="{:.6f}".format(stack.pop())
print(d)
```

代码运行截图 ==（至少包含有"Accepted"）==

**#44304694提交状态**

查看 提交 统计 提问

状态: Accepted

源代码

```
lis=input().split()[::-1]
operator=['+','-','*','/']
stack=[]
for i in lis:
    if i in operator:
        a=stack.pop()
        b=stack.pop()
        if i == '+':
            c=a+b
        elif i=='*':
            c=a*b
        elif i=='/':
            c=a/b
        elif i=='-':
            c=a-b
        stack.append(c)
    else:
        stack.append(float(i))
d="{:.6f}".format(stack.pop())
print(d)
```

基本信息

#: 44304694
题目: 02694
提交人: 尹柚鑫(2100015878)
内存: 3532kB
时间: 23ms
语言: Python3
提交时间: 2024-03-19 20:59:46

## 24591: 中序表达式转后序表达式

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

耗时：30mins

思路：使用栈，数字直接加入输出字符串，左括号则入栈，右括号则出栈至上一个左括号，运算符则将前面运算等级更高的运算符出栈，之后入栈

不熟练的地方在于，如何处理多位数，除了使用正则表达式，不知道有没有其他方法

代码

```
#
import re
def value(op):
    if op == '+' or op == '-':
        return 1
    elif op == '*' or op == '/':
        return 2
    return 0

def tokenize(expression):
    # 使用正则表达式匹配数字和运算符
    tokens = re.findall(r'\d+\.\d+|\d+|\D', expression)
    # 去除多余的空格
    tokens = [token.strip() for token in tokens if token.strip()]
    return tokens

def intopost(exp):
    exp=tokenize(exp)
    postexp=''
    stack=[]
    a=['(',')','+','*','/','-']
    for i in exp:
```

```
        if i not in a:
            postexp=postexp+' '+i
        elif i=='(':
            stack.append(i)
        elif i==')':
            while stack[-1]!='(':
                p=stack.pop()
                postexp+=' '+p
            stack.pop()
        else:
            while stack and value(stack[-1])>=value(i):
                p=stack.pop()
                postexp+=' '+p
            stack.append(i)
    while stack:
        p=stack.pop()
        postexp+=' '+p
    result=postexp.split('.')
    result=[i.strip() for i in result]
    result='.'.join(result)
    return result.strip()

n=int(input())
for _ in range(n):
    string=input().strip()
    postexp=intopost(string)
    print(postexp)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

**#44306307提交状态**

查看 提交 统计 提问

状态: Accepted

源代码

```
import re
def value(op):
    if op == '+' or op == '-':
        return 1
    elif op == '*' or op == '/':
        return 2
    return 0

def tokenize(expression):
    # 使用正则表达式匹配数字和运算符
    tokens = re.findall(r'\d+\.\d+|\d+|\D', expression)
    # 去除多余的空格
    tokens = [token.strip() for token in tokens if token.strip()]
    return tokens

def intopost(exp):
    exp=tokenize(exp)
    postexp=''
    stack=[]
    a=['(',')','+','*','/','-']
    for i in exp:
        if i not in a:
```

基本信息

#: 44306307
题目: 24591
提交人: 尹柚鑫(2100015878)
内存: 3816kB
时间: 38ms
语言: Python3
提交时间: 2024-03-19 22:19:00

## 22068: 合法出栈序列

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

耗时：25mins

思路：利用栈判断是否为合理的序列

代码

```
#
def is_valid_pop_sequence(origin, output):
    if len(origin) != len(output):
        return False  # 长度不同，直接返回False

    stack = []
    bank = list(origin)

    for char in output:
        # 如果当前字符不在栈顶，且bank中还有字符，则继续入栈
        while (not stack or stack[-1] != char) and bank:
            stack.append(bank.pop(0))

        # 如果栈为空，或栈顶字符不匹配，则不是合法的出栈序列
        if not stack or stack[-1] != char:
            return False

        stack.pop()  # 匹配成功，弹出栈顶元素

    return True  # 所有字符都匹配成功

# 读取原始字符串
origin = input().strip()

# 循环读取每一行输出序列并判断
while True:
    try:
        output = input().strip()
        if is_valid_pop_sequence(origin, output):
            print('YES')
        else:
            print('NO')
    except EOFError:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

状态: Accepted

源代码

```
def is_valid_pop_sequence(origin, output):
    if len(origin) != len(output):
        return False  # 长度不同，直接返回False

    stack = []
    bank = list(origin)

    for char in output:
        # 如果当前字符不在栈顶，且bank中还有字符，则继续入栈
        while (not stack or stack[-1] != char) and bank:
            stack.append(bank.pop(0))

        # 如果栈为空，或栈顶字符不匹配，则不是合法的出栈序列
        if not stack or stack[-1] != char:
            return False

        stack.pop()  # 匹配成功，弹出栈顶元素

    return True  # 所有字符都匹配成功

# 读取原始字符串
origin = input().strip()

# 循环读取每一行输出序列并判断
while True:
```

基本信息

| | |
|---|---|
| #: | 44307640 |
| 题目: | 22068 |
| 提交人: | 尹柚鑫(2100015878) |
| 内存: | 3596kB |
| 时间: | 26ms |
| 语言: | Python3 |
| 提交时间: | 2024-03-19 23:43:36 |

## 06646: 二叉树的深度

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

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

## 02299: Ultra-QuickSort

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

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

## 2. 学习总结和收获

==如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。==

1.from collections import deque
class MyDeque:
  def **init**(self):
    self.deque = deque()

```
def __iter__(self):
    return iter(self.deque)
```

自定义队列，如果想让它可以迭代（方便使用map），要在类里面加一个定义迭代的函数

2.可以灵活考虑遍历列表的是顺序还是逆序，比如02694: 波兰表达式用逆序读入就很好

3.处理多位数的输入：

def tokenize(expression):

  tokens = re.findall(r'\d+.\d+|\d+|\D', expression)

  tokens = [token.strip() for token in tokens if token.strip()]
  return tokens

栈，队列，树，这一部分的题目写起来很奇妙，经常会有不易想到的做法，要多加练习这部分的题目