Лабораторная работа 11. Стеки

Стек (англ. stack) — абстрактный тип данных с доступом к элементам по принципу LIFO (Last In — First Out, «последний пришел — первый вышел»). В стеке добавление элемента (push — «положить в стек») и удаление (pop — «извлечь из стека») происходит с одного конца — «вершины» стека.

Создание стека на основе списка

```
stack = []

stack.append('eat')

stack.append('sleep')

stack.append('code')

print(stack)  # ['eat', 'sleep', 'code']

print(stack.pop())  # 'code'

print(stack.pop())  # 'sleep'

print(stack.pop())  # 'eat'
```

Использование deque как стека

```
from collections import deque

stack = deque()

stack.append('eat')

stack.append('sleep')

stack.append('code')

print(stack)  # deque(['eat', 'sleep', 'code'])

print(stack.pop())  # 'code'

print(stack.pop())  # 'sleep'

print(stack.pop())  # 'eat'
```

Пример работы методов deque

```
from collections import deque

stack = deque()

stack.append(1)

stack.append(2)

stack.append(3)

print(stack)  # deque([1, 2, 3])

print(stack.pop())  # 3

print(stack)  # deque([1, 2])
```

```
stack.extend([4, 5])

print(stack)  # deque([1, 2, 4, 5])
```

Задачи:

1. Напишите функцию, которая принимает строку и возвращает ее в обратном порядке, используя стек.
2. Напишите функцию, которая принимает строку из скобок (), {}, [] и проверяет, корректно ли они сбалансированы, используя стек.
3. Реализуйте функцию, которая удаляет все элементы из стека, меньшие заданного значения x
4. Напишите функцию, которая проверяет, является ли стек симметричным, то есть одинаковым, если читать его слева направо и справа налево.
5. Реализуйте функцию, которая принимает постфиксное выражение (напр. "5 1 2 + 4 * + 3 -") и вычисляет его значение, используя стек.

Постфиксное выражение — это форма записи арифметического выражения, в которой операнды (числа) записываются перед знаками операций (операторами). Такой формат также известен как обратная польская нотация (Reverse Polish Notation, RPN). В постфиксной записи операции выполняются без использования скобок, так как порядок вычислений однозначен.
Например:

```
(3 + 4) * 2
В постфиксной записи:
3 4 + 2 *
```