САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №4

по курсу «Алгоритмы и структуры данных»

Тема: Стек, очередь, связанный список

Вариант 20

Выполнил:

Смирнов Георгий Валерьевич

К3139

Проверил:

Афанасьев А.В.

Санкт-Петербург

2024 г.

# Содержание отчета

[**Содержание отчета 2**](#_gjdgxs)

[**Задачи по варианту 3**](#_30j0zll)

[Задача №2. Очередь. 3](#_1fob9te)

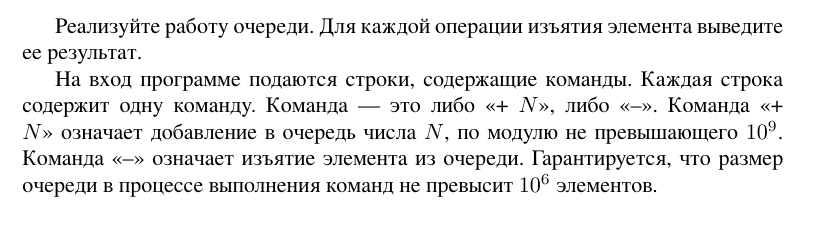
[Задача №4.Скобочная последовательность. 5](#_3znysh7)

[Задача №6. Очередь с минимумом 8](#_iqgs9j8tu785)

[Задача №8. Постфиксная запись 8](#_tepa4zvxmump)

# Задачи по варианту

## Задача №2. Очередь.



**Код программы**

from collections import deque

def process\_queue(commands):

queue = deque()

results = []

for command in commands:

if command.startswith('+'):

\_, number = command.split()

queue.append(int(number))

elif command == '-':

if queue:

results.append(queue.popleft())

return results

# Чтение данных из файла input.txt

with open('input.txt', 'r') as file:

lines = file.readlines()

# Первая строка содержит количество команд

M = int(lines[0].strip())

# Остальные строки содержат команды

commands = [line.strip() for line in lines[1:M+1]]

# Обработка команд

output = process\_queue(commands)

# Запись результатов в файл output.txt

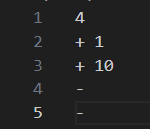
with open('output.txt', 'w') as file:

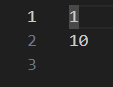
for result in output:

file.write(f"{result}\n")

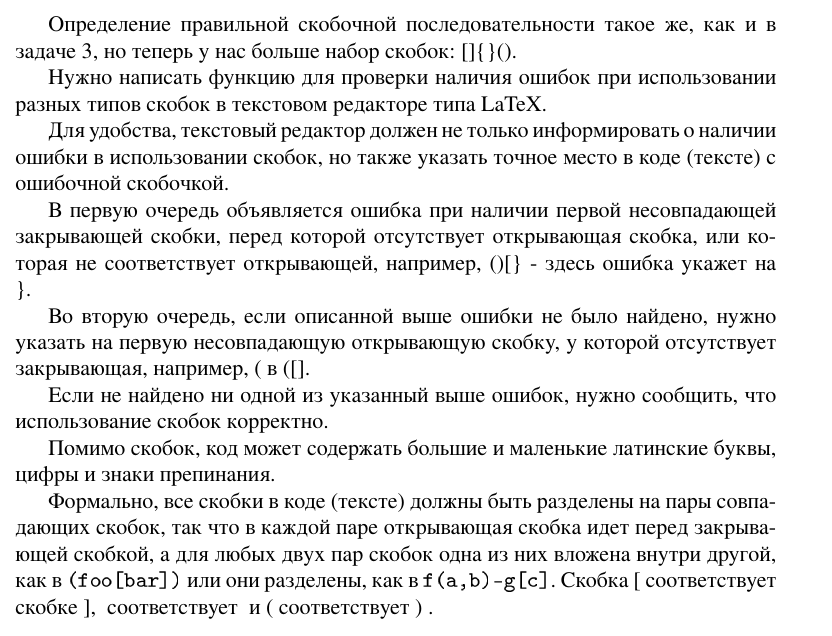
Результат работы кода на примерах из текста задачи:

1)





## Задача №4.Скобочная последовательность.



**Код программы**

def check\_brackets(s):

stack = []

bracket\_map = {')': '(', ']': '[', '}': '{'}

opening\_brackets = set(bracket\_map.values())

for i, char in enumerate(s):

if char in opening\_brackets:

stack.append((char, i + 1))

elif char in bracket\_map:

if stack and stack[-1][0] == bracket\_map[char]:

stack.pop()

else:

return i + 1

if stack:

return stack[0][1]

return "Success"

# Чтение данных из файла input.txt

with open('input.txt', 'r') as file:

s = file.readline().strip()

# Проверка скобочной последовательности

result = check\_brackets(s)

# Запись результата в файл output.txt

with open('output.txt', 'w') as file:

file.write(f"{result}\n")

Результат работы кода на примерах из текста задачи:

1)





2)





3)





4)





5)





6)



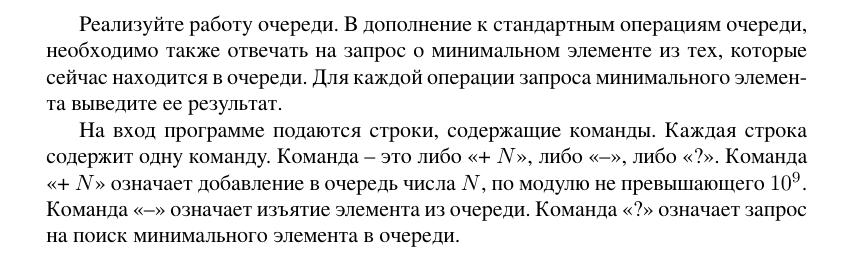


7)





## Задача №6. Очередь с минимумом



from collections import deque

def process\_queue(commands):

queue = deque()

min\_queue = deque()

results = []

for command in commands:

if command.startswith('+'):

\_, number = command.split()

number = int(number)

queue.append(number)

while min\_queue and min\_queue[-1] > number:

min\_queue.pop()

min\_queue.append(number)

elif command == '-':

if queue:

removed = queue.popleft()

if removed == min\_queue[0]:

min\_queue.popleft()

elif command == '?':

if min\_queue:

results.append(min\_queue[0])

return results

# Чтение данных из файла input.txt

with open('input.txt', 'r') as file:

lines = file.readlines()

# Первая строка содержит количество команд

M = int(lines[0].strip())

# Остальные строки содержат команды

commands = [line.strip() for line in lines[1:M+1]]

# Обработка команд

output = process\_queue(commands)

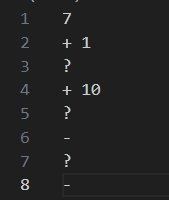
# Запись результатов в файл output.txt

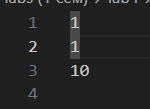
with open('output.txt', 'w') as file:

for result in output:

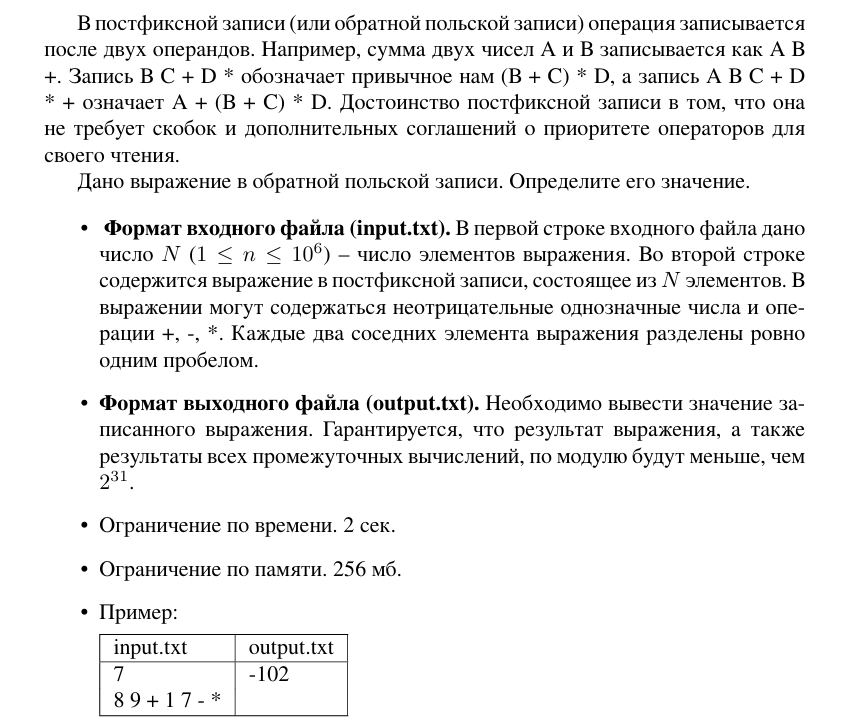
file.write(f"{result}\n")

Результат работы кода на примерах из текста задачи:





## Задача №8. Постфиксная запись



def evaluate\_postfix(expression):

stack = []

for token in expression:

if token.isdigit() or (token[0] == '-' and token[1:].isdigit()):

stack.append(int(token))

else:

b = stack.pop()

a = stack.pop()

if token == '+':

stack.append(a + b)

elif token == '-':

stack.append(a - b)

elif token == '\*':

stack.append(a \* b)

return stack[0]

# Чтение данных из файла input.txt

with open('input.txt', 'r') as file:

n = int(file.readline().strip())

expression = file.readline().strip().split()

# Вычисление значения выражения

result = evaluate\_postfix(expression)

# Запись результата в файл output.txt

with open('output.txt', 'w') as file:

file.write(f"{result}\n")

Результат работы кода на примерах из текста задачи:

