Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Динамические списки.»

Выполнили:

студенты группы 23ВВВ4

Брагин А.М.

Зарубин Я.Д

Герасимов К.А

Приняли:

Деев М.В.

Юрова О.В.

Пенза 2024

class Node:

def \_\_init\_\_(self, value, priority=0):

self.value = value

self.priority = priority

self.next = None

class PriorityQueue:

def \_\_init\_\_(self):

self.head = None

def add(self, value, priority):

new\_node = Node(value, priority)

if self.head is None or self.head.priority < priority:

new\_node.next = self.head

self.head = new\_node

else:

current = self.head

while current.next and current.next.priority >= priority:

current = current.next

new\_node.next = current.next

current.next = new\_node

def remove(self):

if self.head is None:

print("Очередь пуста")

return

removed = self.head

self.head = self.head.next

return removed.value

def display(self):

current = self.head

if not current:

print("Очередь пуста")

return

while current:

print(f"{current.value} (priority {current.priority})", end=" -> ")

current = current.next

print("None")

class Queue:

def \_\_init\_\_(self):

self.head = None

self.tail = None

def enqueue(self, value):

new\_node = Node(value)

if self.tail:

self.tail.next = new\_node

self.tail = new\_node

if self.head is None:

self.head = new\_node

def dequeue(self):

if self.head is None:

print("Очередь пуста")

return

removed = self.head

self.head = self.head.next

if self.head is None:

self.tail = None

return removed.value

def display(self):

current = self.head

if not current:

print("Очередь пуста")

return

while current:

print(f"{current.value}", end=" -> ")

current = current.next

print("None")

class Stack:

def \_\_init\_\_(self):

self.head = None

def push(self, value):

new\_node = Node(value)

new\_node.next = self.head

self.head = new\_node

def pop(self):

if self.head is None:

print("Стек пуст")

return

removed = self.head

self.head = self.head.next

return removed.value

def display(self):

current = self.head

if not current:

print("Стек пуст")

return

while current:

print(f"{current.value}", end=" -> ")

current = current.next

print("None")

# Тестирование PriorityQueue

print("--- Priority Queue ---")

priority\_queue = PriorityQueue()

priority\_queue.add("Task1", 1)

priority\_queue.add("Task2", 3)

priority\_queue.add("Task3", 2)

priority\_queue.display()

priority\_queue.remove()

priority\_queue.display()

# Тестирование Queue

print("--- Queue ---")

queue = Queue()

queue.enqueue("Item1")

queue.enqueue("Item2")

queue.enqueue("Item3")

queue.display()

queue.dequeue()

queue.display()

# Тестирование Stack

print("--- Stack ---")

stack = Stack()

stack.push("Element1")

stack.push("Element2")

stack.push("Element3")

stack.display()

stack.pop()

stack.display()