Отчет по лабораторной работе 2

Основные структуры данных

Дата: 10-10-2025
Семестр: 3 курс 1 полугодие
Группа: ПИЖ-б-о-23-1
Дисциплина: Анализ сложности алгоритмов
Студент: Пурас М.Р.

Цель работы

Изучить понятие и особенности базовых абстрактных типов данных (стек, очередь, дек, связный список) и их реализаций в Python. Научиться выбирать оптимальную структуру данных для решения конкретной задачи, основываясь на анализе теоретической и практической сложности операций.

Теоретическая часть

В работе рассматриваются следующие структуры данных:

- **Список (list)** в Python: реализация динамического массива
- **Связный список (Linked List)**: цепочка узлов с ссылками
- **Стек (Stack)**: LIFO (Last-In-First-Out) структура
- **Очередь (Queue)**: FIFO (First-In-First-Out) структура
- **Дек (Deque)**: двусторонняя очередь

Практическая часть

Выполненные задачи

- [x] Задача 1: Реализовать класс LinkedList
- [x] Задача 2: Анализ производительности операций
- [x] Задача 3: Сравнительный анализ структур данных
- [x] Задача 4: Решение практических задач

Ключевые фрагменты кода

Реализация связного списка

```
def insert_at_start(self, data: Any) -> None:
    """Вставка элемента в начало списка. Сложность O(1)."""
    new_node = Node(data)
    if self.head is None:
        self.head = new_node
        self.tail = new_node
    else:
        new_node.next = self.head
        self.head = new_node
```

Проверка сбалансированности скобок

```
def check_brackets(expression: str) -> bool:
    stack: List[str] = []
    brackets = {')': '(', ']': '[', '}': '{'}
    for char in expression:
        if char in brackets.values():
            stack.append(char)
        elif char in brackets:
            if not stack or stack.pop() != brackets[char]:
                return False
    return not stack
```

Результаты выполнения

Пример работы программы

```
Сравнение вставки в начало (1000 элементов):
List: 0.02781 секунд
LinkedList: 0.02662 секунд

Сравнение операций очереди (1000 операций dequeue):
List (pop(0)): 0.01471 секунд
Deque (popleft()): 0.00677 секунд

Выражение '((( )))' сбалансировано: True
Выражение '((( ))]' сбалансировано: False
Слово 'радар' является палиндромом: True
Слово 'привет' является палиндромом: False
Печатается: doc1.pdf
Печатается: doc2.pdf
Печатается: doc3.pdf
Все задачи выполнены.
```

Тестирование

- [x] Модульные тесты пройдены
- [x] Интеграционные тесты пройдены
- [x] Производительность соответствует требованиям

Выводы

1. Связный список эффективен для вставки в начало, тогда как list для этой операции неэффективен
2. Deque из модуля collections показал себя как эффективная структура для реализации очереди
3. Выбор структуры данных существенно влияет на производительность операций

Ответы на контрольные вопросы

1. **В чем преимущества и недостатки связного списка по сравнению с массивом?**
   Преимущества: вставка и удаление в начале за O(1), динамический размер.
   Недостатки: доступ по индексу за O(n), больше памяти из-за хранения ссылок.

2. **Почему для реализации очереди лучше использовать deque, а не list?**
   Потому что операции добавления и удаления с обоих концов в deque выполняются за O(1), а в list удаление из начала (pop(0)) имеет сложность O(n).