**Оглавление**

1. [Введение](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5)……….…………………………………………………………………………3
2. [Основная часть](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D0%B0%D1%8F-%D1%87%D0%B0%D1%81%D1%82%D1%8C)
   1. [Постановка задачи](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%9F%D0%BE%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0-%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B8)……………………………………………………………….4
   2. [Описание решения](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%9E%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5-%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F)……………………………………………………………….4
3. [Эксперименты](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34" \l "%D0%AD%D0%BA%D1%81%D0%BF%D0%B5%D1%80%D0%B8%D0%BC%D0%B5%D0%BD%D1%82%D1%8B)
   1. [Описание тестов](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%9E%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5-%D1%82%D0%B5%D1%81%D1%82%D0%BE%D0%B2)………………………………………………………………….5
   2. [Результаты тестов](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%A0%D0%B5%D0%B7%D1%83%D0%BB%D1%8C%D1%82%D0%B0%D1%82%D1%8B-%D1%82%D0%B5%D1%81%D1%82%D0%BE%D0%B2)………………………………………………………………..5
4. [Заключение](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%97%D0%B0%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)……………………………………………………………………………….6
5. [Приложение](https://chatgpt.com/c/67702a31-a75c-8003-93c8-daf94e4c3b34#%D0%9F%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)………………………………………………………………………………7

**Введение**

Цель работы — проектирование и реализация эффективной структуры данных для предложенной задачи

В ходе работы было выполнено:

1. Реализована структура данных для хранения элементов с весами и выполнения операций взвешенного случайного выбора.
2. Проведено тестирование производительности структуры данных.
3. Выполнена проверка точности работы структуры в условиях различных распределений данных.

**Постановка задачи**

Задача заключается в разработке структуры данных, которая позволяет эффективно выполнять следующие операции:

1. Добавить элемент  с положительным вещественным весом  в контейнер (если он уже там лежит – заменить вес).
2. Удалить элемент  из контейнера.
3. Получить элемент случайным образом в соответствии с вероятностями

где - текущее количество элементов в контейнере.

**Описание решения**

Для реализации структуры данных была выбрана модифицированная версия красно-чёрного дерева. В каждом узле хранится дополнительная информация — общий вес всех узлов в поддереве, включая сам узел. Это позволяет эффективно выполнять операции взвешенного случайного выбора.

**Ключевые аспекты реализации:**

**Добавление элемента (add):**

Процесс добавления элемента основан на стандартной реализации вставки в красно-чёрное дерево. Однако, при спуске по дереву, ко всем посещаемым узлам добавляется вес нового элемента, чтобы поддерживать корректное значение суммарного веса в поддеревьях.

**Удаление элемента (delete):**

Процесс удаления элемента основан на стандартной реализации для красно-чёрного дерева. После удаления узла выполняется обратный обход от удаленного узла до корня дерева с корректировкой суммарных весов, уменьшаемых на значение веса удалённого элемента.

**Выбор элемента (get):**

Выбор элемента осуществляется путём спуска по дереву, где на каждом шаге принимается решение о переходе к левому, правому потомку или выборе текущего узла, исходя из их весов. Для этого генерируется случайное значение *threshold* в пределах от 0 до суммарного веса текущего узла. Решение принимается следующим образом:

* Если *threshold* меньше веса левого потомка, осуществляется переход к левому потомку.
* Если *threshold* находится в диапазоне веса текущего узла (включая левый потомок), возвращается текущий элемент.
* В противном случае, переход осуществляется к правому потомку.

**Тестирование**

**Описание тестов**

Были проведены два набора тестов:

1. **Тест производительности:** Измерение времени выполнения операций add, get, delete при добавлении, удалении и выборке большого количества элементов.
2. **Тест точности:** Проверка отклонения частотности выборки элементов от ожидаемой теоретической вероятности в зависимости от веса.

**Результаты тестов**

**Тест производительности**

|  |  |  |
| --- | --- | --- |
| **Операция** | **Кол-во элементов** | **Время на операцию** |
| Добавление (add) | 10,000 | 0.07s |
| Выборка (get) | 10,000 | 0.02s |
| Удаление (delete) | 10,000 | 0.03 |
| Добавление (add) | 100,000 | 1s |
| Выборка (get) | 100,000 | 2.3s |
| Удаление (delete) | 100,000 | 0.42 |
| Добавление (add) | 1000,000 | 12s |
| Выборка (get) | 1000,000 | 3.6s |
| Удаление (delete) | 1000,000 | 3.8 |

**Тест точности**

|  |  |  |
| --- | --- | --- |
| **Тест** | **Количество операций get** | **Отклонение** |
| Test1 | 1000 | 24.51% |
|  | 10,000 | 7.74% |
|  | 100,000 | 2.13% |
| Test2 | 1000 | 17.02% |
|  | 10,000 | 5.51% |
|  | 100,000 | 1.83% |
| Test3 | 1000 | 26.05% |
|  | 10,000 | 7.64% |
|  | 100,000 | 3.43% |
| Test4 | 1000 | 18.20% |
|  | 10,000 | 6.92% |
|  | 100,000 | 3.67% |

**Test1: Добавление элементов с уникальными весами**  
Добавляются элементы с ключами от 1 до 100. Каждый ключу присваивается вес, равный значению ключа. После добавления всех элементов проверяется корректность дерева, а также корректность выборки с учётом весов.

**Test2: Добавление и удаление нечётных элементов**  
Добавляются элементы с ключами от 1 до 100. Однако, для нечётных ключей происходит их немедленное удаление, а их веса заменяются на 0. Этот тест проверяет корректность удаления элементов, обновления весов и работы дерева после этих операций.

**Test3: Обновление весов для добавляемых элементов**  
Добавляются элементы с ключами от 1 до 100. Каждому элементу сначала присваивается вес, равный ключу, а затем он обновляется до значения, равного удвоенному ключу. Это проверяет корректность работы обновления весов в дереве.

**Test4: Смешанный тест с добавлением, обновлением и удалением**  
Этот тест сочетает в себе действия из тестов 2 и 3. Сначала элементы добавляются в дерево с весом, равным ключу. Затем их вес обновляется до удвоенного значения. После этого для нечётных ключей элементы удаляются, а их веса заменяются на 0. Этот тест проверяет, как дерево справляется с комплексными операциями обновления и удаления одновременно.

**Выводы**

* Время выполнения операций соответствует ожидаемой сложности .
* Отклонение выборки от теоретических значений уменьшается с увеличением количества операций.

**Заключение**

В результате работы реализована эффективная структура данных на базе красно-чёрного дерева.