Санкт-Петербургский политехнический университет Петра Великого

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе № 4**

**Дисциплина:** Низкоуровневое программирование

**Тема:** Программирования на Си

Выполнил студент гр. 3530901/90004 Пудов Н. А.

(подпись)

Преподаватель Алексюк А.О.

(подпись)

“ ” 2021 г.

Санкт-Петербург

2021

**Бинарная куча**

Введение

Существует структура данных «Бинарная куча». В куче элементы хранятся в виде двоичного дерева, то есть у элементов есть два потомка - левый и правый. В вершине кучи находится один элемент, у него - два потомка на следующем уровне, у них, в свою очередь, по два потомка на третьем уровне (итого - 4 элемента на третьем уровне) и т. д. Уровни заполняются в порядке увеличения номера уровня, а сам уровень заполняется слева направо. У элементов последнего уровня нет ни одного потомка, возможно, что и у некоторых элементов предпоследнего уровня нет потомков. Также в куче может быть один элемент, у которого только один потомок (левый).

Выделяют два типа кучи: min heap – в корне кучи- минимальный элемент, и max-heap – в корне кучи – максимальный элемент. Я реализовал min-heap дабы потом использовать эту особенность в сортировке (но об этом позже). На рис.1. представлен пример бинарной кучи с максимальным элементом в корне(max-heap). В данном случае выполняется основное свойство кучи: Отец больше своих потомков (в более сложных случаях допускается равенство потомков с отцом, то есть наличие одинаковых элементов в куче) Отметим отдельно, что отношения между потомками не регламентированы. Другими словами, левый потомок может быть как больше, так и меньше правого и наоборот. Это некоторое отличие от привычного нам дерева, где слева меньшие элементы, а справа большие.

Индексация кучи начинается с первого элемента(корня). Левый потомок имеет индекс 2\*i+1, правый потомок 2\*i+2. i-индекс отца.

|  |
| --- |
|  |
| Рис.1. Пример бинарной кучи(max-heap) |

Описание работы кучи.

Давайте кратко опишем механизм работы кучи. Рассмотрим метод add в binaryHeap.c. Казалось бы, что сложного просто добавить элемент в кучу, которая хранится на основе обычного списка. Однако при добавлении элемента в куче нарушается основное свойство кучи (в нашем случае потомки больше отца- min-heap). Тогда наиболее разумным будет просто добавить элемент в конец списка, а далее уже последовательно проверять потомок с отцом и поднимать его вверх кучи до тех пор, пока основное свойство кучи не будет восстановлено (либо пока не достигнем начала кучи). В этом нам помогает метод bubbleUp. Сложность этой операции O(logN).

Рассмотрим ещё одну распространённую операцию- извлечение минимального элемента (метод deleteSmallestElement). Здесь всё довольно просто. Берём первый элемент (первый индекс) из нашего массива кучи. Всё, извлекли, всё прекрасно, сложность операции O(1). Но мы не учли важную особенность- после извлечения место корневого элемента должен кто-то занять, ведь свято место пусто не бывает. Давайте поставим последний элемент на место корневого. И опять возникает проблема, что и при добавлении элемента в кучи. Может нарушиться основное свойство кучи (отмечу, что может нарушится, но необязательно это случится, как и в первом случае.). Поэтому нам поможет метод продвижения элемента из корня вниз кучи до тех пор, пока основное свойство кучи не будет восстановлено (либо пока не достигнем конца кучи.) В этом нам помогает метод bubbleDown. Сложность этой операции уже O(logN).

# **СТРУКТУРА КОДА И ТЕСТИРОВАНИЕ**

Были реализованы файлы binaryHeap.c, test.c, main.c, test.h, binaryHeap.h. В первом файле реализованы основные методы работы с кучей. В файле main.c реализовано взаимодействие с файлом и работа консольного приложения. Также был написан make-file, осуществляющий сборку программы и статической библиотеки.

Вывод.

Была реализована бинарная куча на языке Си. Изучена работа с указателями и динамическим выделением памяти в Си. Разработан make-file для сборки нашей программы. Разработано консольное приложение, осуществляющее забор данных из файла, их обработка и вывод в выходной файл.