Дисциплина «Алгоритмы решения прикладных задач»
Рабочая тетрадь 3.
Двоичная куча. Очередь с приоритетом. Пирамидальная сортировка

Теоретический материал

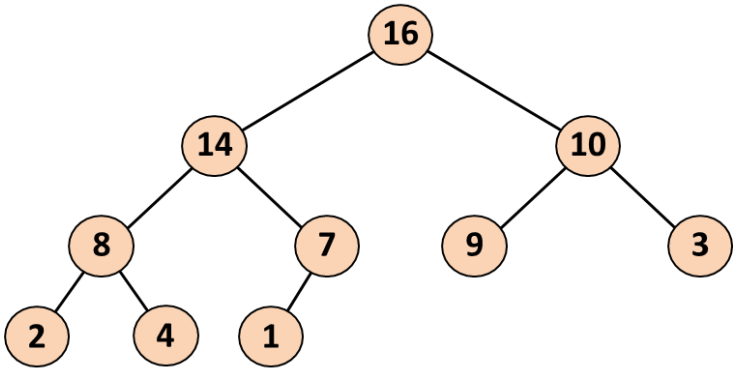
Куча (англ. heap) — это специализированная структура данных типа дерево, которая удовлетворяет свойству кучи: если B является узлом-потомком узла A, то ключ(A) ≥ ключ(B). Из этого следует, что элемент с наибольшим ключом всегда является корневым узлом кучи, поэтому иногда такие кучи называют max-кучами (в качестве альтернативы, если сравнение перевернуть, то наименьший элемент будет всегда корневым узлом, такие кучи называют min-кучами). Не существует никаких ограничений относительно того, сколько узлов-потомков имеет каждый узел кучи, хотя на практике их число обычно не более двух.

Кучи обычно реализуются в виде массивов, что исключает наличие указателей между её элементами.

Над кучами обычно проводятся следующие операции:

- найти максимум или найти минимум: найти максимальный элемент в max-куче или минимальный элемент в min-куче, соответственно
- удалить максимум или удалить минимум: удалить корневой узел в max- или min-куче, соответственно
- увеличить ключ или уменьшить ключ: обновить ключ в max- или min-куче, соответственно
- добавить: добавление нового ключа в кучу.
- слияние: соединение двух куч с целью создания новой кучи, содержащей все элементы обеих исходных.

Реализация двоичной кучи на основе массива



16	14	10	8	7	9	3	2	4	1				

- Корень дерева храниться в ячейке $H[0]$ – максимальный элемент
- Индекс родителя узла *i*: $Parent\,(i\,) = \,[(i\text{-}1)/2]$
- Индекс левого дочернего узла: $Left\,(i) = 2i + 1$
- Индекс правого дочернего узла: $Right\,(i\,) = 2i + 2$

Очередь с приоритетом

Очередь с приоритетом — это абстрактный тип данных, описывающий структуру данных, в которой каждый фрагмент обладает определенным приоритетом. В отличие от очереди, освобождающей элементы по принципу «первым вошел, первым вышел», очередь с приоритетом обслуживает элементы в соответствии с приоритетом: сначала она удаляет данные с наивысшим приоритетом, затем данные с последующими по рангу приоритетами (или, наоборот, начинает с наименьших значений).

Очередь с приоритетом поддерживает две обязательные операции — добавить элемент и извлечь максимум (минимум). Предполагается, что для каждого элемента можно вычислить его приоритет — действительное число.

Основные методы, реализуемые очередью с приоритетом, следующие:

insert(ключ, значение) — добавляет пару (ключ, значение) в хранилище;

extract_minimum() — возвращает пару (ключ, значение) с минимальным значением ключа, удаляя её из хранилища.

При этом меньшее значение ключа соответствует более высокому приоритету.

В некоторых случаях более естественен рост ключа вместе с приоритетом. Тогда второй метод можно назвать **extract_maximum()**.

В качестве примера очереди с приоритетом можно рассмотреть список задач работника. Когда он заканчивает одну задачу, он переходит к очередной — самой приоритетной— то есть выполняет операцию извлечения максимума. Начальник добавляет задачи в список, указывая их приоритет, то есть выполняет операцию добавления элемента.

Одной из популярных реализаций очереди с приоритетом является <u>куча</u>.

Задание 1

Задача:

На вход программе подается массив, элементы которого расположены в произвольном порядке. Создать из массива минимальную или максимальную двоичную кучу (см. для примера слайды 4 -10 лекции).

Решение:

Ответ:

Задание 2

Задача:

Создать класс «Двоичная куча» (максимальная куча). Реализовать для класса конструктор, формирующий кучу из произвольного массива, конструктор по умолчанию (создание пустой кучи). Для класса реализовать методы: поиск максимума, удаление максимума, добавление нового элемента в кучу, слияние двух куч[1]

Решение:

Ответ:

Задание 3

Задача:

Задача о связывании канатов. Дан список (массив) канатов различной длины, которые требуется связать, по два за раз, в порядке, который приведет к наименьшим затратам. Затраты на соединение двух канатов равны сумме их длин, а общие затраты равны сумме соединения всех канатов. На выходе программа должна выдать порядок связывания канатов и суммарные затраты. Решить задачу на основе использования двоичной кучи (см. слайды 28-29 лекции)

Решение:

Ответ:

[1] Слияние куч – задание со «звездочкой»

Задание 4*

Задача:

Напишите функцию, которая принимает двоичное дерево в качестве параметра и возвращает *true*, если это двоичная куча (максимальная или минимальная), и *false*, если нет.

Решение:

Ответ:

Задание 5*

Задача:

Реализовать алгоритм пирамидальной сортировки массива (слайды 19-25 лекции)

Решение:

Ответ:

Задание 6*

Задача:

Реализовать очередь с приоритетом на базе двоичной кучи. На основе очереди с приоритетом сделать todo list[2], для которого пользователь либо добавляет новое задание, имеющее название (одно слово) и приоритет (целое число), либо запрашивает задание с максимальным приоритетом из списка дел и делает его (помечает как выполненное, выводит за пределы очереди). Предусмотреть возможность редактирования задания.

Решение:

Ответ:

[2] «ToDo list» — приложение, которое позволяет пользователям добавлять, редактировать и удалять задачи, над которыми они хотят работать, а также отмечать задачи как выполненные, не удаляя их.