Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Завдання №2

“d-арна піраміда”

Виконав студент 2-го курсу

Групи К-28

Самойлич Євгеній Костянтинович

2020

**Завдання**:

Розробіть ефективні реалізації процедур Extract\_Max, Insert та Increase\_Key,

призначених для роботи з d-арною незростаючою пірамідою. Проаналізуйте час роботи цих процедур і виразіть їх в термінах n та d

**Теорія**

d-арні піраміди схожі на бінарні, лише їх вузли, відмінні від листя, мають не по 2, а по d дочірніх елементів.

**Алгоритм**

* **Extract\_Max**

Піраміда незростаюча, отже повертаємо перше значення масиву. Потім присвоюємо значення останнього елемента першому та видаляємо останній. Порівнюємо нове значення кореня піраміди з дочірніми елементами та міняємо їх місцями, якщо не виконується умова незростання. Таку дію повторюємо допоки умова незростання не буде відновлена.

* **Insert**

Записуємо елемент в кінець масиву. Визначимо батьківський елемент для доданого елементу за формулою [i-1]/d. Та виконаємо перевірку чи є батьківський елемент більшим за доданий, якщо ні, то міняємо їх місцями. Повторюємо останній крок, допоки умова незростання не буде відновлена.

* **Increase\_Key**

Знаходимо елемент *key* у масиві та збільшуємо його значення. Після збільшення ключа виконуємо перевірку, чи не порушилась умова незростання. Порівнюємо елемент з батьківським, якщо наш елемент виявився більшим, то міняємо їх місцями. Повторюємо останній крок, допоки умова незростання не буде відновлена.

**Складність**

* **Extract\_Max**

Після вилучення максимального елементу ми виконуємо перевірку з дочірніми елементами на незростання, тому складність O

* **Insert**

Порівнюємо останній елемент лише з батьківськими. Складність O.

* **Increase\_Key**

Порівнюємо змінений елемент лише з батьківськими. Складність O.

**Мова програмування**

С++

**Модулі програми**

* void restoreDown(int arr[], int len, int index, int d)

Функція, яка будує (або відновлює) незростаючу властивість d-арної піраміди.

* void insert(int arr[], int\* n, int k, int elem)

Функція, яка додає значення *elem* у піраміду.

* int extractMax(int arr[], int\* n, int d)

Функція, яка вилучає та повертає максимальне значення піраміди.

* void increaseKey(int arr[], int key, int d, int plus)

Функція, яка збільшує значення *key* на *plus*.

**Інтерфейс користувача**

Вхідні дані вводяться з консолі і виводяться також в консоль.

**Тестові приклади**

Розмірність купи: 3.

Масив значень: {24, 8, 21, 18, 3, 13, 15, 14, 0, 5, 1}

Купа після побудови: {24, 15, 21, 18, 3, 13, 8, 14, 0, 5, 1}

Купа після додавання 7: { 24, 15, 21, 18, 3, 13, 8, 14, 0, 5, 1, 7}

Купа після вилучення максимуму: {21, 15, 14, 18, 3, 13, 8, 7, 0, 5, 1}

Купа після збільшення 7 на 88: { 95, 15, 21, 18, 3, 13, 8, 14, 0, 5, 1}

**Висновки**

Extract\_Max, Insert та Insrease\_Key потребують O часу.

**Література**

* Лекція №5
* https://www.geeksforgeeks.org/k-ary-heap/