Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Завдання №3

“d-нарні піраміди”

Виконав студент 2-го курс

Групи К-28

Гуща Дмитро Сергійович

2020

**Завдання:**

d-арні піраміди схожі на бінарні, лише їх вузли, відмінні від листя, мають не по 2, а по d дочірніх елементів. Представте d-арну піраміду у вигляді масиву (якою буде її висота для n елементів?). 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 та Алгоритм:**

***Extract\_max*:** шукаємо максимальний елемент. Піраміда незростаюча, тому найбільшим буде 1 елемент масиву. Він і буде повертатися. Далі будемо змінювати значення першого та другого елементу, а потім видалимо його. Потім порівнюємо 1 елемент з дочірніми, та також міняємо їх, якщо порушується властивість незростаючої піраміди поки елемент не стане на останнє місце

**Insert:** Вставляємо новий елемент в кінець массиву та порівнюємо його зі значенням батьківського елементу. Для вирахування індексу використаємо формулу id\_p = [і/d], і – індекс елементу, який ми вставляємо, а d – кількість дочірніх елементів.

Якщо батьківський елемент менше за новий, то змінимо їх місцями, в протилежному випадку все залишиться на своєму місці. Будемо повторювати поки батьківський елемент не буде більшим за елемент, який ми вставляємо або не стане на перше місце в масиві.

**Increase\_key:** за лінійну складність, за допомогою ключа знайдемо потрібний елемент, а потім збільшимо.

* Знаходимо елемент з ключем key за лінійний час та збільшуємо його у k разів.
* Порівнюємо цей елемент з його батьківським вузлом. Якщо батьківський елемент менший за елемент, який ми вставляємо то ми міняємо їх місцями, проте в протилежному випадку нічого не відбувається.

Повторюємо 2гий крок поки батьківський елемент не буде більшим за елемент, який ми збільшили , або поки він не став на місце верхнього вузла.

**Складність:**

**Extract\_Max:**

Час роботи алгоритму . Оскільки наша функція один раз викликає balance для 1 елементу, складність якої О.

**Incease\_key:**

Час роботи . Коли ми збільшили ключ, то перевіряємо, чи йде перевірка, чи не порушилось «незростання» піраміди. В такому випадку відбувається обмін з батьківським елементом, поки властивість незростання не буде відновлена.

**Insert :**

Час роботи алгоритму також .

Оскільки йде перевірка останього елементу з батькіськими елементами.

**Мова програмування:** С++

**Модулі програми:**

void balance(int index)  
//Допоміжна функція збалансування масиву  
void insert(int data)  
//Функція вставки елемента в піраміду  
int Extract\_max()  
//Функція повернення збалансованого масиву, без максимального елементу.  
void increase\_key(int key, int step, int index)  
//Функція збільшення вибраного елементу піраміди

**Інтерфейс користувача:** Вхідні дані вводяться з консолі і виводяться також в консоль.

**Тестові приклади:**

**d= 3**

**Input:** 71 13 12 23 6 7 8 0 6 5 5

**Результат після видалення максимального:** 23 13 12 6 6 7 8 4 1 0 5 5

**Результат після вставки «9»:** 23 13 12 9 6 7 8 4 1 0 5 5 6

**Результат після збільшення «5» :** 25 13 12 23 6 7 8 4 1 0 9 5 6

**Література**

* Лекція №5
* <https://uk.wikipedia.org/wiki/D-%D0%B0%D1%80%D0%BD%D0%B0_%D0%BA%D1%83%D0%BF%D0%B0>
* https://en.wikipedia.org/wiki/D-ary\_heap