**Київський національний університет імені Тараса Шевченка**

**Факультет комп’ютерних наук та кібернетики**

Кафедра інформаційних систем

Алгоритми та складність

Лабораторний проект № 3

d-арні піраміди

**Звіт**

**Виконала:**

студентка групи К-28

Гожда Марія Олександрівна

**Київ-2018**

**d-арні піраміди**

1. **Умова 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.

1. **Основні поняття.**

*Піраміда – частково упорядкована структура даних, яку можна розглядати як d-арне дерево з певними властивостями:*

*• Дерево заповнене на всіх рівнях крім, можливо, останнього.*

*• Останній рівень заповнюється зліва направо до вичерпання елементів.*

*• Ключі у вузлах певним чином упорядковані.*

*• d-арна піраміда може бути ефективно реалізована у вигляді масиву шляхом запису її елементів згори донизу зліва направо.*

*• Висота d-арної піраміди для n елементів дорівнює*

1. **Опис, аналіз та основні модулі програми.**

* int Parent (int i) Визначає індекс батька і-го вузла
* int Son(int parent, int k) Визначає індекс k-го сина у вузлі, батьком якого є parent
* void Max\_Heapify(vector<int> &A, int i)

**Час роботи**: О(d\*)

Підтримує властивість незростання піраміди;

На вхід приймає масив А та індекс вузла і;

Вважається, що дерева з коренями Son(i, k) (k =1-d) –

незростаючі піраміди, але елемент A[i] може порушувати

цю властивість. Значення A[i] просувається вниз, поки відповідне дерево з

цим коренем не стане незростаючою пірамідою.

* int Extract\_Max(vector<int> &A)

**Час роботи**: О(d\*)

На вхід приймає масив А;

Повертає і вилучає максимальне значення в піраміді;

За побудовою піраміди максимальне значення знаходиться у корені, воно вилучається, на його місце ставиться останній елемент масива і викликається процедура Max\_Heapify(A, 0)

* void Increase\_Key(vector<int> &A, int i, int key)

**Час роботи**: О()

На вхід приймає масив А, позицію і в піраміді та ключ key;

Змінює ключ елемента з індексом і на новий ключ key(за умови що значення нового ключа не менше за старе);

Після заміни ключа йде перевірка, чи не порушилась властивість незростання. В такому випадку відбувається обмін з батьківським елементом, поки властивість не відновиться.

* void Insert(vector<int> &A, int key)

**Час роботи**: О()

На вхід приймає масив А та ключ key;

Вставляє в піраміду новий вузол зі значенням key;

1. **Реалізація алгоритму.**

Реалізовано на С++

1. **Інтерфейс програми.**

Вхідні дані прописані в срр файлі, а результати виводяться в консоль.

1. **Використані структури даних.**

Вектор std::vector<>;

1. **Тестовий приклад.**

29 26 28 27 10 25 20 19 17 15

After Increase\_Key for i = 6, key = 42

42 29 28 27 10 25 26 19 17 15

After Insert for key = 31

42 29 28 31 10 25 26 19 17 15 27

max = 42

After Extract\_Max

31 29 28 27 10 25 26 19 17 15

1. **Використані джерела.**

1. *Лекція №5*

2. *http://study.sfu-kras.ru/DATA/docs/ProgramTheory/recurs/s\_sort.htm*