**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

ЛАБОРАТОРНА РОБОТА №3

З курсу “Алгоритмі і складність”

Біноміальна купа (піраміда)

Виконав

Студент групи ПІ-22

Факультету комп’ютерних наук

та кібернетики

Мандзюк Дмитро

Київ 2023

**Зміст**

[Завдання 3](#_Toc128161844)

[Теорія 3](#_Toc128161845)

[Алгоритм 5](#_Toc128161846)

[Складність 9](#_Toc128161847)

[Мова програмування 9](#_Toc128161848)

[Модулі програми 9](#_Toc128161849)

[Інтерфейс користувача 10](#_Toc128161850)

[Приклади 10](#_Toc128161851)

[Висновок 13](#_Toc128161852)

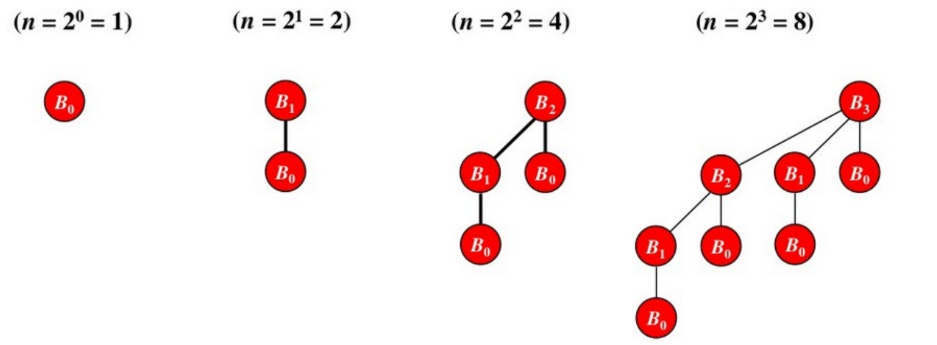
[Література 13](#_Toc128161853)

# **Завдання**

Реалізувати біноміальну піраміду для **раціональних чисел**.

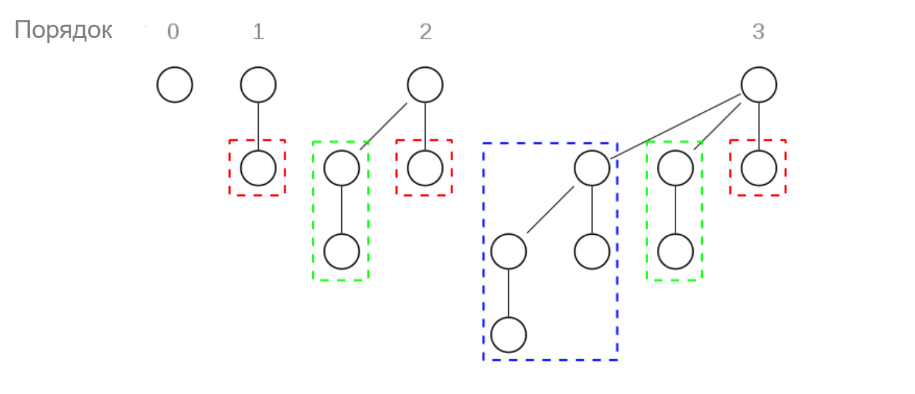
# **Теорія**

**Біноміальне дерево** - це рекурсивно задані впорядковані дерева. Біноміальне дерево B0 порядку 0 складається з єдиної вершини. Біноміальне дерево Bk порядку k складається з двох зв’язаних біноміальних дерев Bk–1: корінь одного є крайнім лівим сином іншого.



Біноміальне дерево Bk

1. має 2k вузлів;
2. має висоту k;
3. має рівно k вузлів на глибині i = 0,1...k;
4. має корінь степені k, а степінь синів менша степеня кореня; при цьому якщо синів пронумерувати зліва направо числами (k–1), (k–2), ..., 0, то i-й син є коренем біноміального дерева Bi.

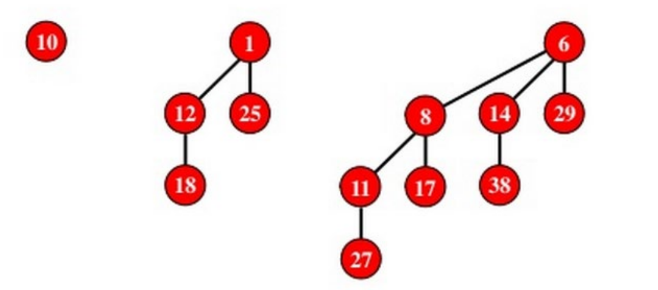


**Біноміальна піраміда** – множині біноміальних дерев, що задовольняє властивостям біноміальних пірамід:

1. Кожне біноміальне дерево у купі підпорядковується властивості неспадної купи (min-heap property): ключ вузла завжди менший або рівний ключу дитини.
2. Для будь-якого невід'ємного цілого числа k в купі існує не більше одного біноміального дерева, чий корінь має ступінь (degree) k.

Біноміальна піраміда з n вузлів складається не більше ніж з (|lg(n)| + 1) біноміальних дерев.

Якщо біноміальна піраміда має n вузлів, то одиниці в двійковому записі n відповідають порядкам біноміальних дерев, що входять до цієї піраміди.  
Наприклад, для 13 вузлів: 13 = 11012. Біноміальна піраміда складатиметься з біноміальних дерев B0, B2 та B3 з 1, 4 і 8 вузлів відповідно.

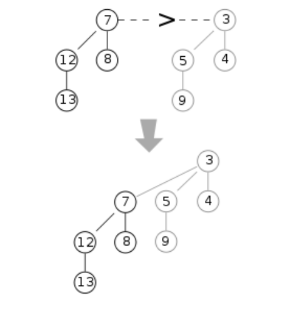


# **Алгоритм**

Створення порожнього біноміального дерева або купи очевидні.

1. Операція **link** приймає на вхід два біноміальні дерева одного порядку k. Вона зливає їх в одне, теж коректне, біноміальні дерево порядку k+1.

Алгоритм link елементарний. Вершину дерева з більшим ключем робимо новою крайньою лівою дитиною дерева з меншим ключем, враховуючи всі зв’язки, і збільшує порядок відповідного вузла.

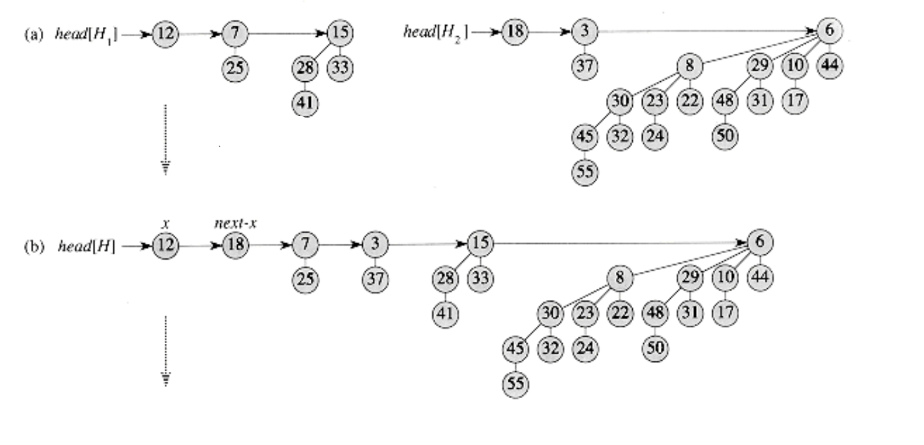


1. Операція **min** (пошуку мінімуму) приймає на вхід біноміальну купу і повертає посилання на вузол з мінімальним ключем.

Алгоритм – звичайний перебір всіх корневих вузлів в піраміді.

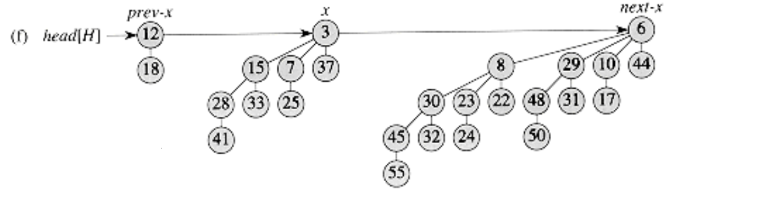
1. Операція **push\_back** (злиття) – діє аналогічно етапу злиття в сортуванні злиттям, на кожному кроці переміщаючи в результуючий список дерево меншого порядку.

Після злиття списків коренів відомо, що купа містить не більше двох коренів однакової степені, і вони стоять підряд.



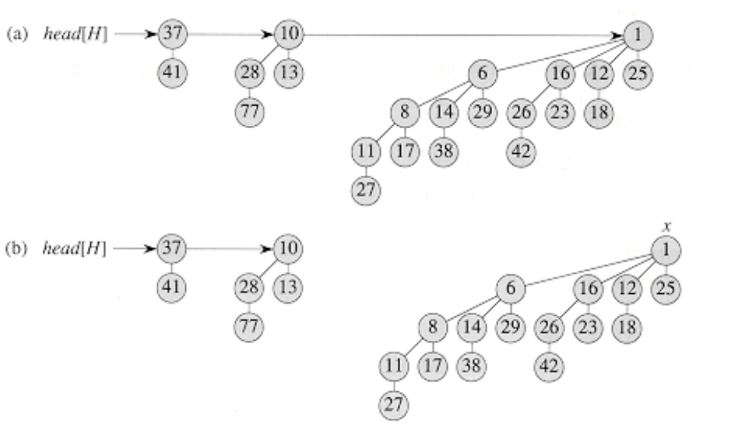
1. Операція **merge** – об’днання двох пірамід.   
   Застосовується операція **push\_back** після чого відбувається відновлення властивостей біноміальної купи через відповідну операцію **link** для сусідніх дерев з однаковою степінню.

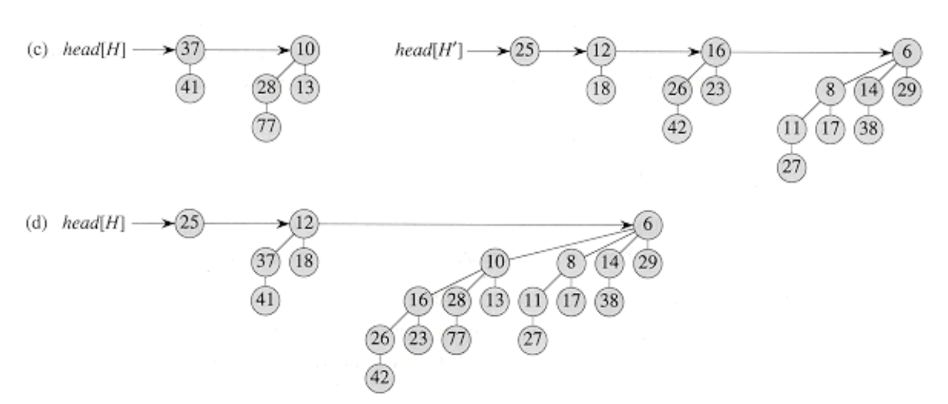
Якщо продовжити попередній приклад, то отримаємо:



1. Операція **insert** – вставка елемента у купу.  
   Алгоритм також простий – створюється біноміальна купа з лише єдиного одинарного дерева (ключ – елемент, який необхідно вставити), після чого застосовується операція **merge** на нову піраміду і піраміду, в яку вставляємо елемент.
2. Операція **extract\_min** – видалення мінімального вузла з купи.

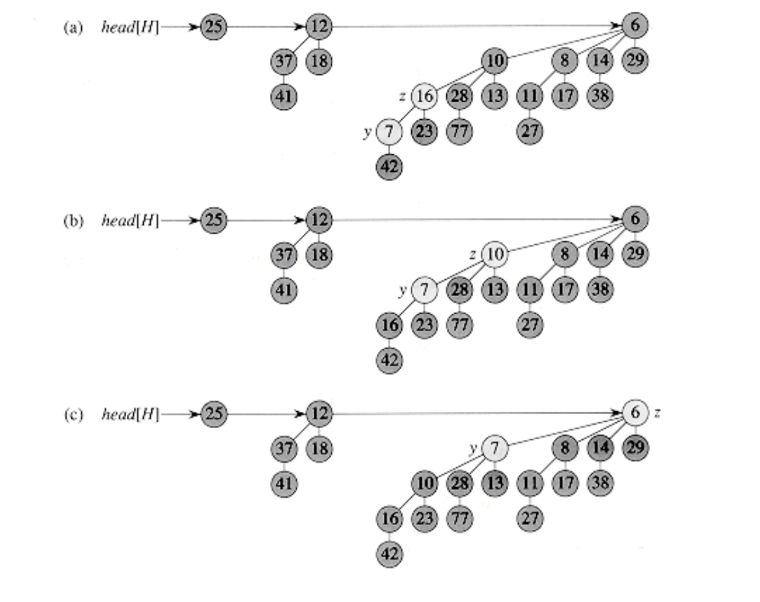
Видаляємо мінімальний корінь. Утворюємо з його синів нову біноміальну піраміду (перестановкою у зворотному порядку). Зливаємо новоутворену піраміду з вихідною.





1. Операція **decrease\_key** – зменшення ключа у вузлі купи.

Значення ключа замінюється на менше. Після цього рухаємось у напрямку кореня, обмінюючи значення, якщо порушена умова неспадаючої піраміди.



# **Складність**

Складність створення порожньої біноміальної піраміди - **O(1)**

Складність зв’язування двох біноміальних дерев одного порядку Bk-1 в біноміальне дерево Bk (дерево з коренем y "підчеплюється" до дерева з меншим ключем) - **O(1)**

Складність злиття двох біноміальних пірамід - **O(log n)**

Складність об’днання (merge) двох біноміальних пірамід - **O(log n)**

Складність пошуку мінімального вузла - **O(log n)**

Складність вставки вузла - **O(log n)**

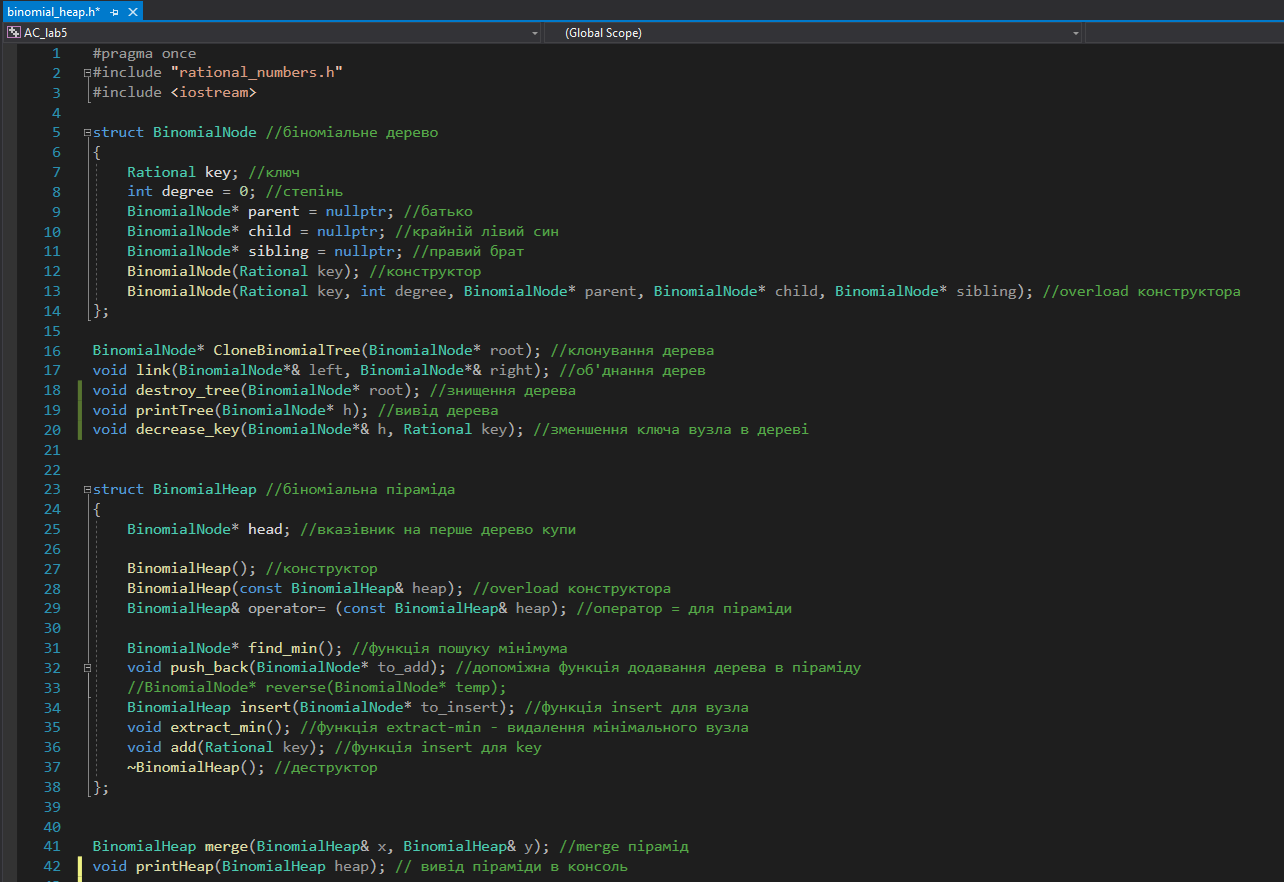
Складність вилучення мінімального вузла - **O(log n)**

Складність зменшення ключа у вузлі - **O(log n)**

# **Мова програмування**

С++

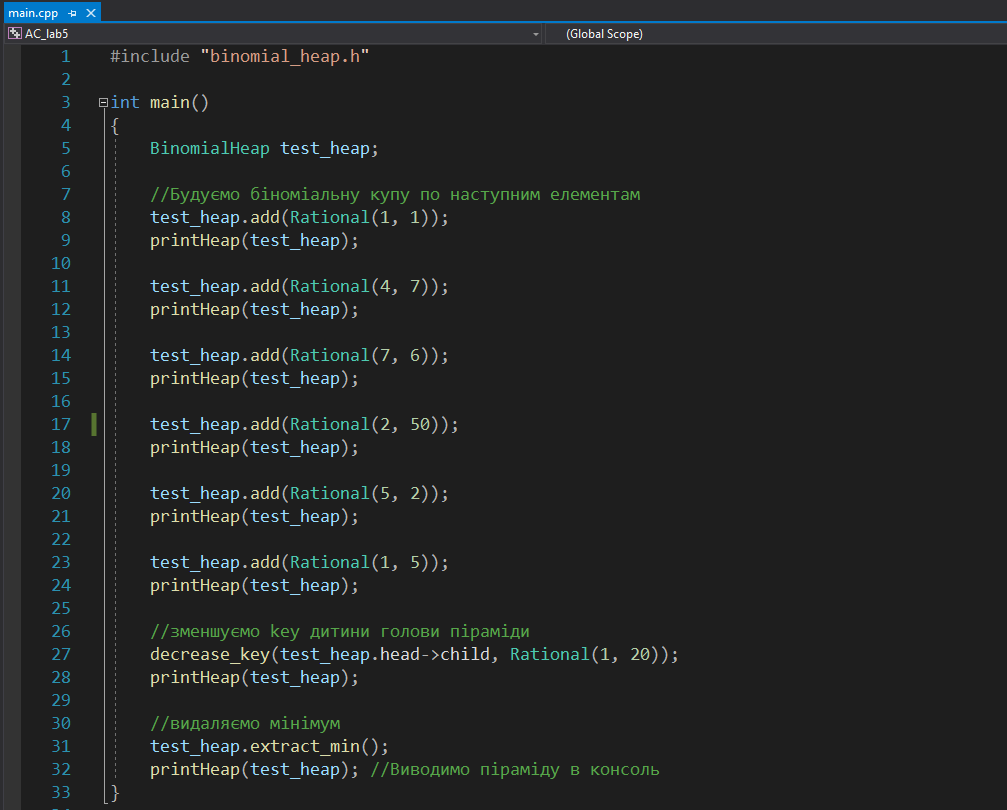
# **Модулі програми**

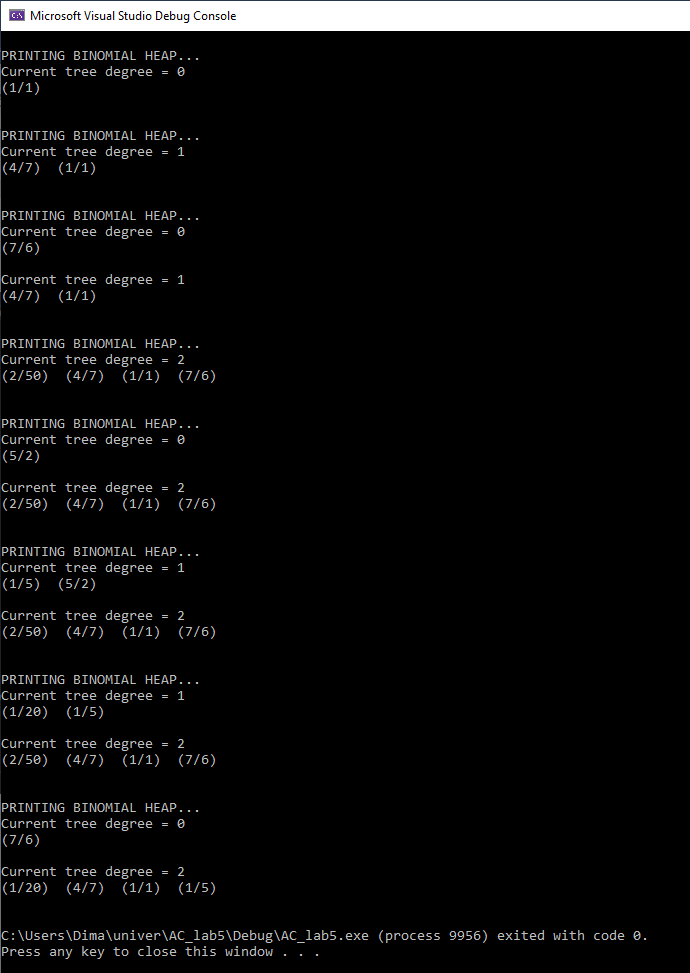


# **Інтерфейс користувача**

Вхідні данні задаються в коді програми (а саме в наведеній нижче функції int(main)), а результат виводиться у консоль.

# **Приклади**





Програма успішно створила піраміду по заданим ключам, вивила її в консоль на кожному етапі, зменшила вказаний ключ і видалила мінімум.

# **Висновок**

Біноміальна піраміда доволі цікава структура. Кожну піраміду можна представити двійковим числом, а якщо розглянути всі операції, що можна виконати над нею, і проаналізувати зміни, що відбулися з цим двійковим представленням, то можна помітити, що ці операції схожі на шкільне алгоритми додавання в стовпчик тільки для двійкових чисел.

В реальному житті цю структуру використовують для реалізації черг з пріоритетами – наприклад в планувальнику операційної системи: процес з найбільшим пріоритетом виконується першим; або для розбиття множин: множина з найбільшою кількістю елементів розбивається першою. Також за допомогою неї можна швидко шукати мінімуми. В цілому, існують деякі специфічні типи задач, в яких доволі зручно використовувати біноміальну піраміду

# **Література**

1. Лекція з предмету «Алгоритми та складність» 6
2. <https://www.geeksforgeeks.org/binomial-heap/>
3. <https://habr.com/ru/articles/135232/>