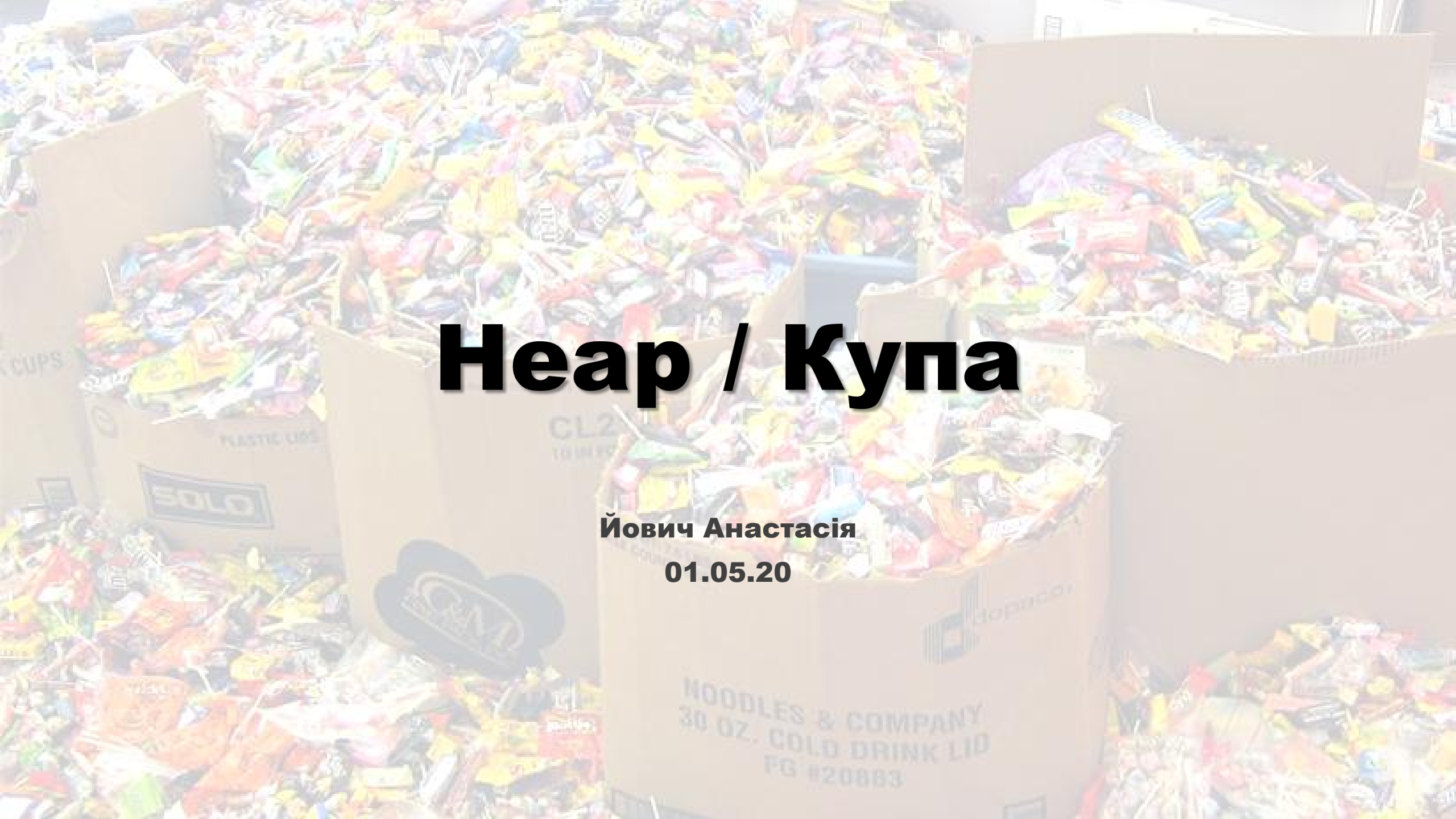


Heap / Купа

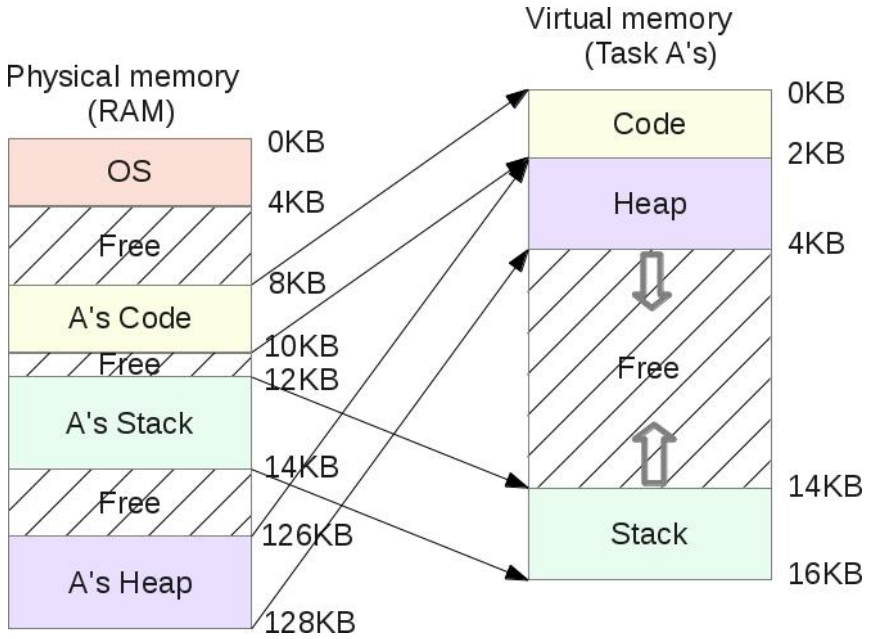
Йович Анастасія

01.05.20

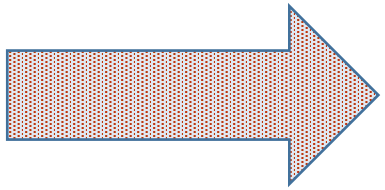
Heap (пам’ять) vs heap (структура данних)

Пам’ять

- Купа дозволяє створювати змінні, які живуть в пам’яті незалежно від функції
- Працює трохи повільніше, ніж стек
- Набагато більший об’єм пам’яті в порівнянні зі стеком
- Заповнюється від найменших адресів, до найбільших
- `int * numPtr = new int;`



Структура данних



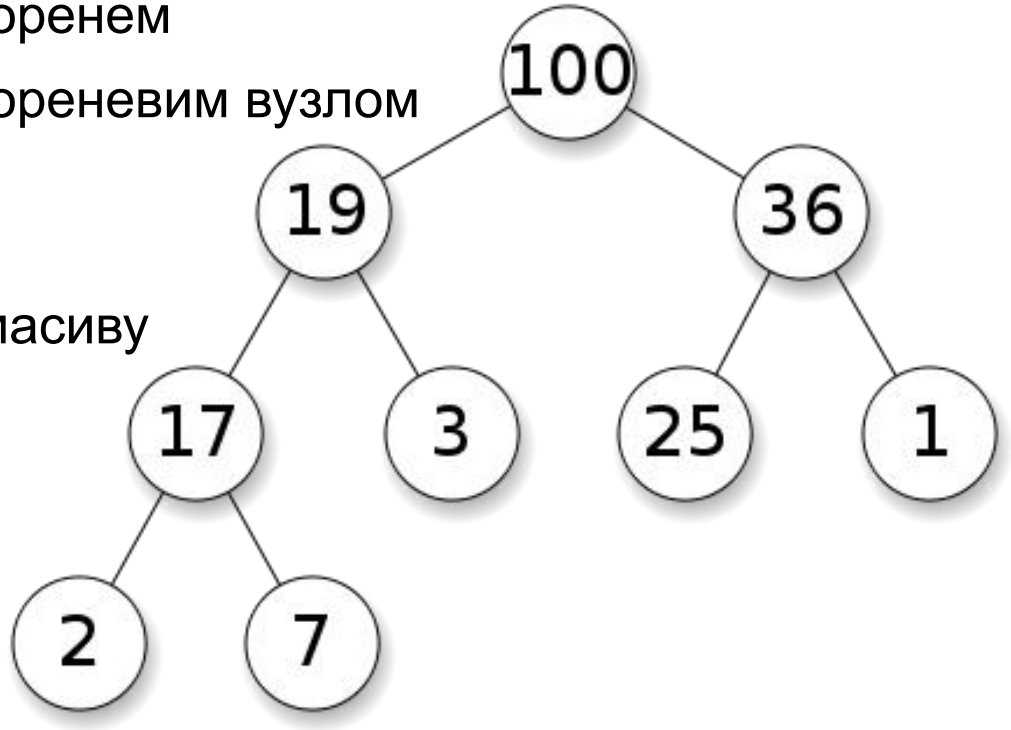
наступний слайд

Купа

- Структура типу дерево, яка задовільняє умові:

якщо B є вузлом-нащадком вузла A, то ключ(A) ≥ ключ(B)

- Елемент з найбільшим ключем завжди є коренем
- Елемент з найбільшим ключем завжди є кореневим вузлом
- Базові операції з купою такі:
  - підтримка основної властивості купи
  - побудова купи з невпорядкованого масиву
  - сортування купи
  - видалення найменшого елемента
  - отримання найбільшого елемента
  - додавання елемента



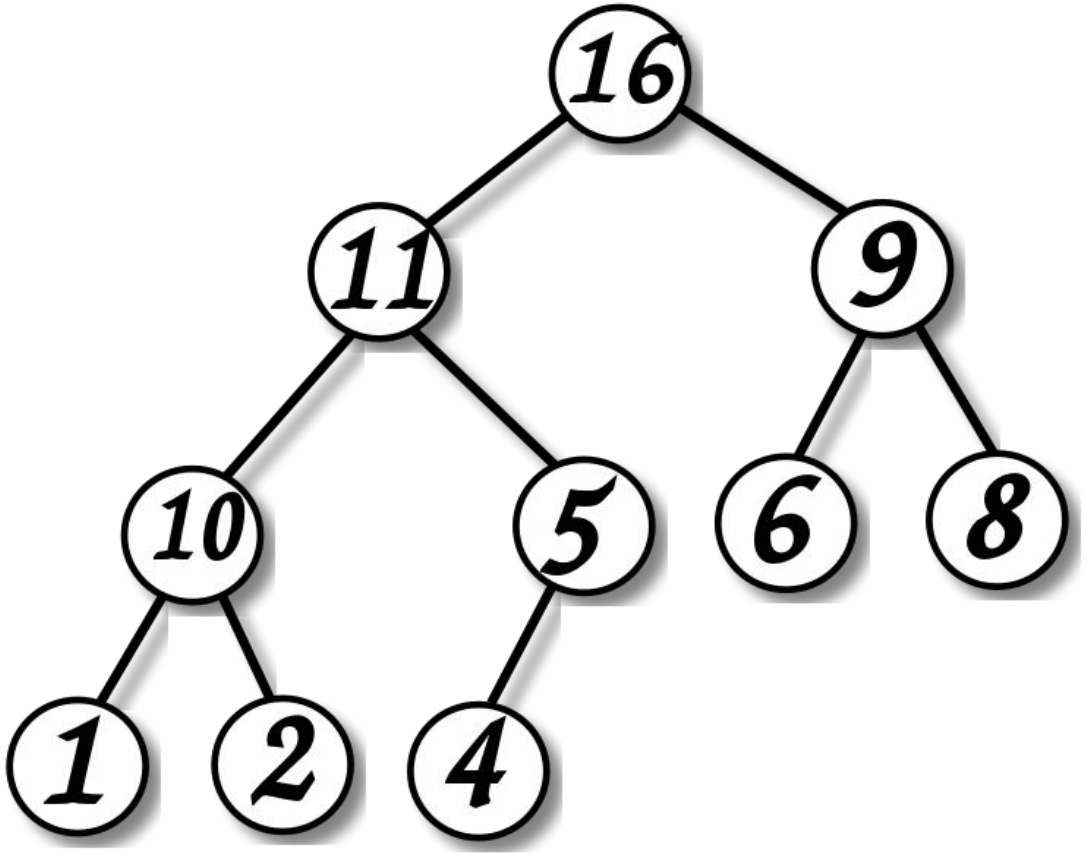
Бінарна купа

У кожного вузла максимум 2 нащадки

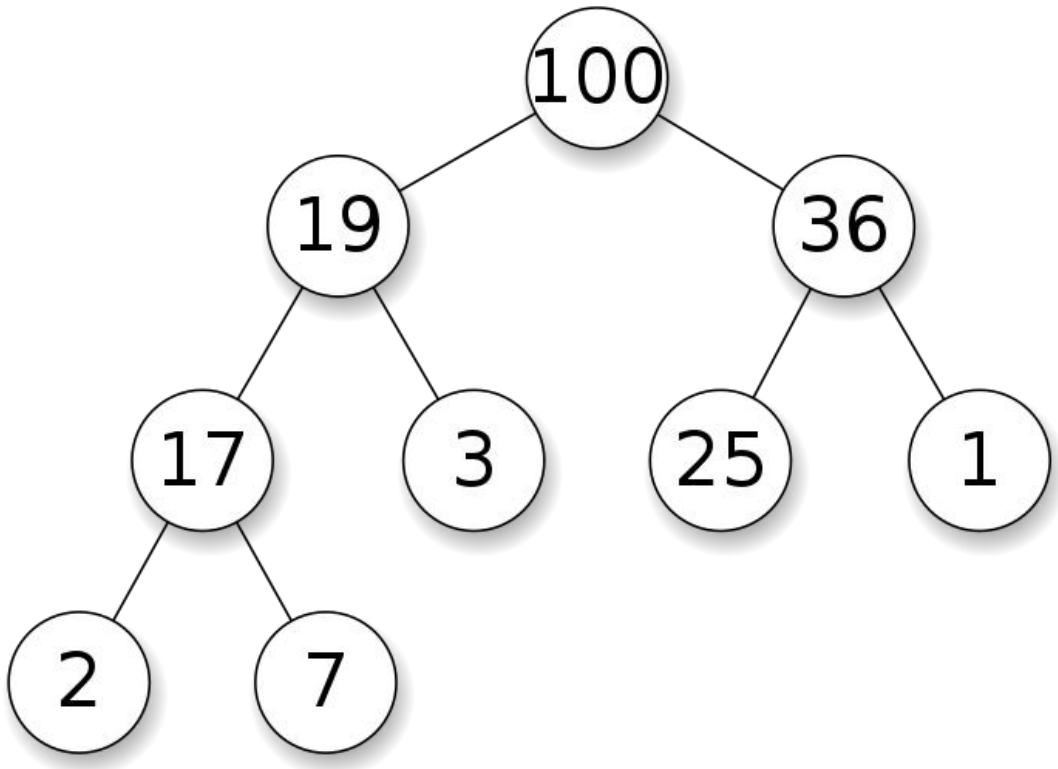
1. Значення в будь-якій вершині не менше, ніж значення в нащадків.

2. Дерево є повним (complete tree)

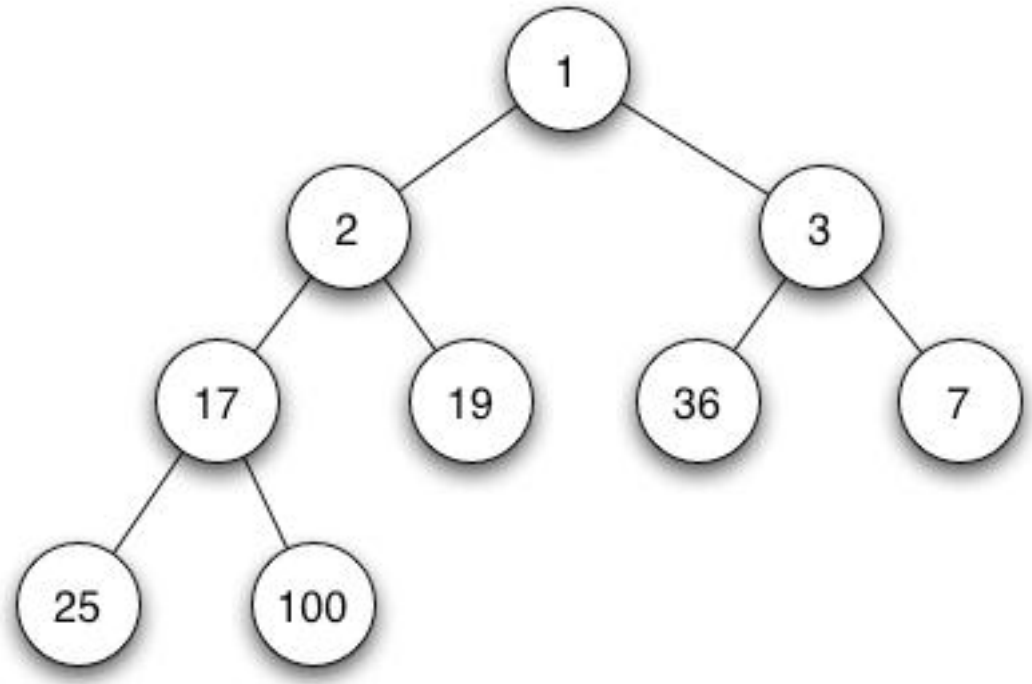
3. Останній слой заповнюється зліва направо без дірок



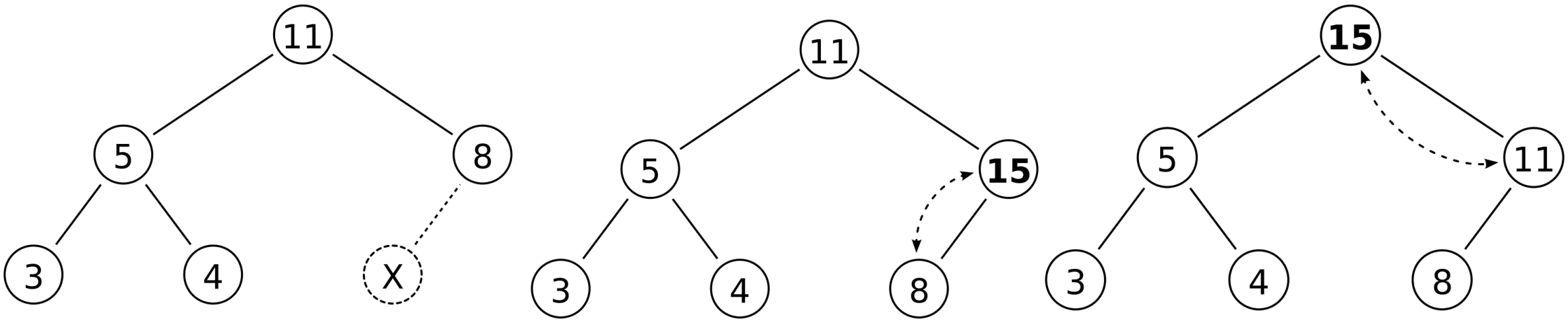
Binary max heap



Binary min heap

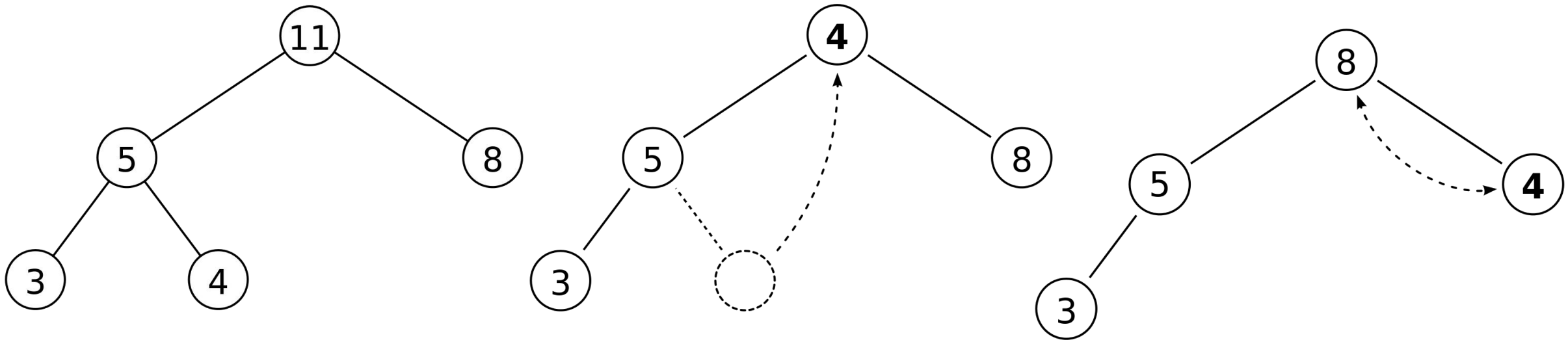


Вставка елементу в купу



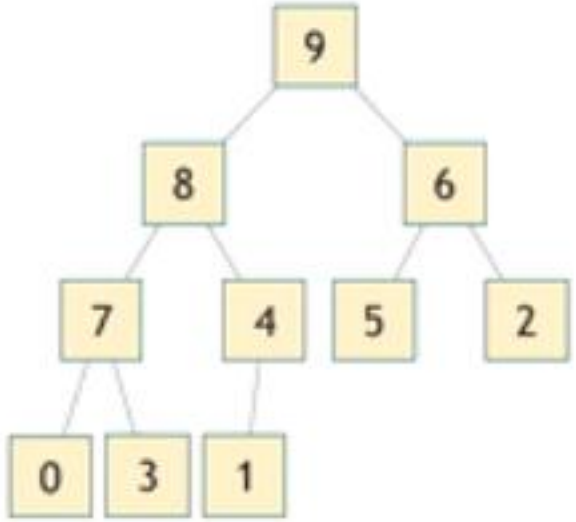
Складність операції залежить від висоти дерева, тому загалом це O(lg n)

Видалення елементу з купи



В найгіршому випадку O(lg n)

Купа в stl



Купа може бути реалізована за допомогою std::vector

std::make_heap

```
std::vector<int> numbers = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};

std::make_heap(begin(numbers), end(numbers));

for (int number : numbers)
{
    std::cout << number << '  ';
}
```

Output:

9 8 6 7 4 5 2 0 3 1

Черга з пріоритетами (priority queue)

Черга з пріоритетами - це черга, де важливо не хто останнім став в чергу (порядок додавання в чергу не грає ролі), а хто важливіший. При додаванні до черги указується пріоритет елементу, а при взятті елемента з черги обрається елемент з найбільшим пріоритетом.

Зазвичай реалізується на основі heap.


Дякую за увагу!