Bài 7.2: Tạo và thêm phần tử vào heap

- ✓ Tạo heap
- ✓ Yêu cầu và các bước thực hiện
- ✓ Mã giả và triển khai
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

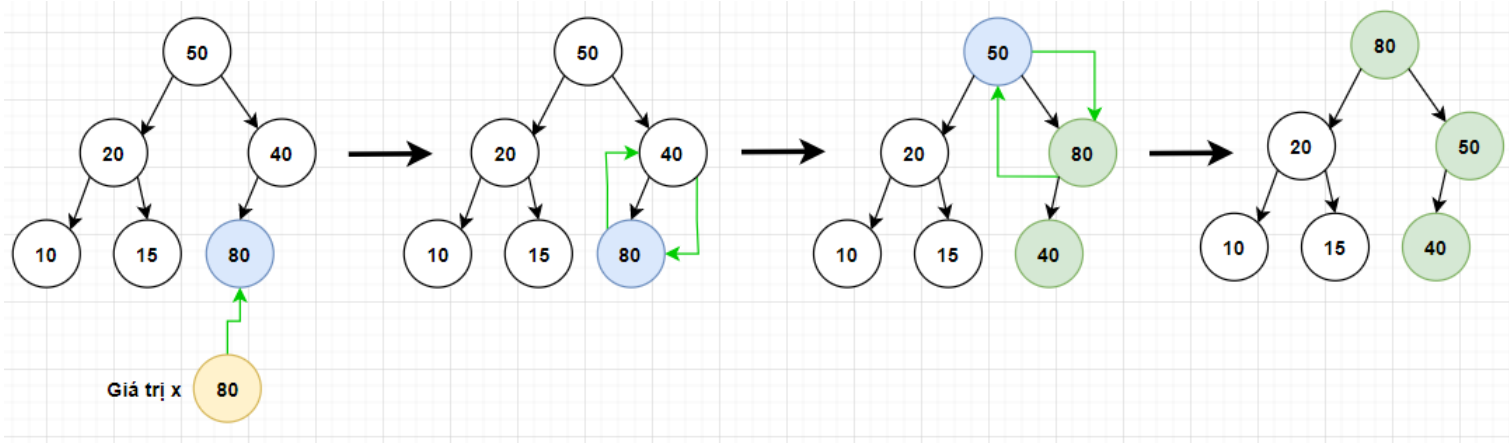
Tạo heap

- Ta tạo heap với các thành phần:
  - Mảng data để lưu dữ liệu.
  - Số phần tử hiện thời: currentSize.
  - Số phần tử tối đa: capacity.

```
template<class T> class Heap { // max heap
    T* data;
    int capacity;
    int currentSize;
public:
    // hàm tạo
    Heap(int cap = 10) { ... }
    // thêm mới phần tử vào heap
    bool add(T value) { ... }
    // sàng lên(vun)
    void siftUp(int index) { ... }
    // hàm hiển thị các phần tử trong heap
    void showElements() { ... }
    // lấy số phần tử hiện thời trong heap
    int size() { ... }
    // hàm kiểm tra xem heap có rỗng không
    bool isEmpty() { ... }
    // hàm hủy
    ~Heap() { ... }
};
```

Yêu cầu và các bước thực hiện

- Yêu cầu: Cho trước một heap và giá trị cần thêm vào heap. Hãy thêm giá trị này vào heap sao cho vẫn giữ được tính chất của heap.
- Các bước thực hiện:
  - B1: Thêm phần tử mới vào cuối heap.
  - B2: Sàng lên để tái cân bằng lại heap.
  - B3: Tăng kích thước của heap lên 1 để có chỗ chứa phần tử mới.



Thêm phần tử

- Sau đây là mã giả của thao tác thêm node vào heap:

```
// hàm thêm phần tử vào heap
// value: giá trị cần chèn
bool add(value) :
    if(currentSize < capacity): // nếu số phần tử hiện thời
        data[currentSize] = value // gán giá trị cho phần tử mới
        siftUp(currentSize)     // sàng lên
        curentSize++    // tăng số phần tử hiện thời lên 1
        return true     // trả về kết quả thêm thành công
    else:               // nếu heap đã đầy
        return false    // thông báo thêm thất bại
```

Thêm phần tử

- Sau đây là mã thật của thao tác thêm node vào heap:

```
// thêm mới phần tử vào heap
bool add(T value) {
    if (currentSize < capacity) {
        data[currentSize] = value;
        siftUp(currentSize);
        currentSize++;
        return true;
    }
    else {
        return false;
    }
}
```

Sàng lên

➢ Sau đây là mã giả và triển khai của thuật toán:

```
// thao tác sàng lên
// index: vị trí phần tử cần tráo đổi
function siftUp(index):
    parentIndex = (index - 1) / 2 // lấy vị trí node cha
    if(data[index] > data[parentIndex]): // node con > node cha
        swap(data[index], data[parentIndex]) // tráo đổi giá trị
        siftUp(parentIndex) // tiếp tục sàng lên ở vị trí node cha
```

```
// sàng lên(vun)
void siftUp(int index) {
    int parentIndex = (index - 1) / 2;
    if (data[index] > data[parentIndex]) {
        swap(data[index], data[parentIndex]);
        siftUp(parentIndex);
    }
}
```

Nội dung tiếp theo

Xóa node khỏi heap