Duyệt cây theo chiều rộng

Tiền đề cho bài học

Để nắm rõ kiến thức của bài học này bạn cần nắm được khái niệm chung về cây nhị phân và hàng đợi. bạn có thể đọc các kiến thức này ở các đường dẫn sau:

- Cây nhị phân
- Hàng đợi (queue)

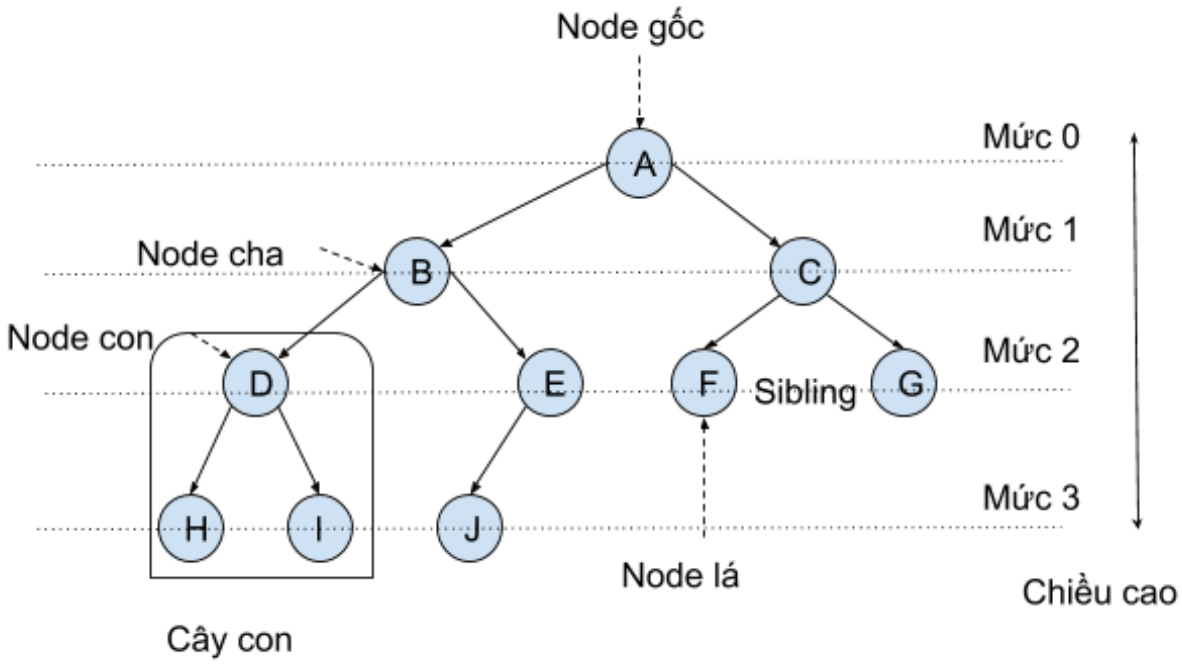
Chú ý:

Trong bài này tôi sẽ sử dụng hai từ hàng đợi và queue xen kẽ để tránh sự lặp lại câu từ khi viết.

Enqueue: Thêm một phần tử mới vào cuối hàng đợi.

Dequeue: Lấy phần tử đầu tiên ra khỏi hàng đợi.

Phương thức duyệt cây theo chiều rộng



Hình 1: Cây nhị phân

Như đã đề cập ở bài trước, chúng ta có hai cách thức duyệt cây chính là duyệt theo chiều rộng và duyệt theo chiều sâu. Ở bài này sẽ giới thiệu chi tiết hơn về cách thức duyệt cây theo chiều rộng.

Duyệt cây theo chiều rộng là cách duyệt cây theo một mức hay độ sâu nhất định trước khi duyệt tới mức tiếp theo sâu hơn.

Giả sử ta phải duyệt toàn bộ cây nhị phân như hình một và in ra dữ liệu khi ta duyệt. Các bước làm sẽ là:

- Đầu tiên ta duyệt node gốc (mức 0) và in ra dữ liệu là A

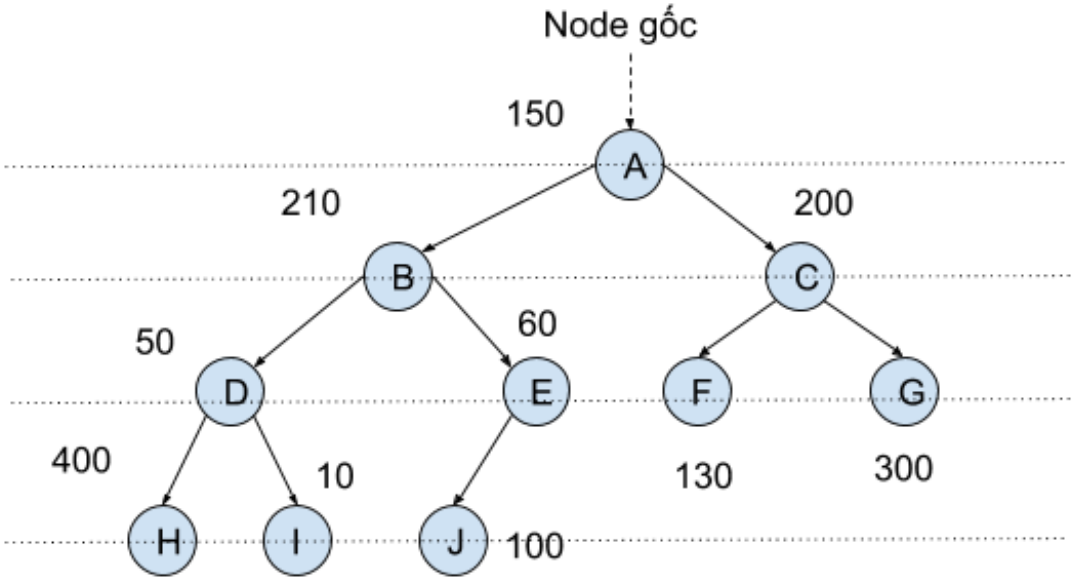
- Tiếp theo ta duyệt tới mức 1 và duyệt theo thứ tự từ trái sang phải, dữ liệu in ra là B,C
- Tiếp tục duyệt mức 2 và in ra dữ liệu theo chiều từ trái sang phải D,E,F,G
- Khi kết thúc mức 2 ta duyệt tới mức cuối cùng và dữ liệu in ra là H,I,J

Sau khi hoàn thành việc duyệt thì dữ liệu in ra sẽ là: A, B, C, D, E, F, G, H, I ,J.

Cách thức chương trình máy tính hoạt động

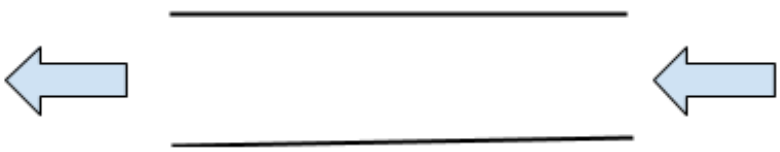
Ở trên ta đã thực hiện việc duyệt cây theo mức một cách thủ công.

Giả sử mỗi node trong cây nhị phân ở hình một có địa chỉ như sau:



Hình 2: Địa chỉ các node

Để thực hiện việc này với chương trình máy ta cần sự hỗ trợ của một hàng đợi (Trên thực tế ta có thể duyệt cây theo mức dựa chính vào mức của các node mà không cần dùng hàng đợi, nhưng trong bài này tôi sẽ tập trung vào dùng hàng đợi vì nó là cách dễ cài đặt hơn, có quy tắc chung và được dùng rộng rãi), đặt tên cho hàng đợi là bf_queue.



Hình 3. bf_queue (FIFO)

Khi mới khởi tạo, hàng đợi rỗng. Việc cần phải làm với hàng đợi để có thể duyệt cây theo mức là chứa các node vào hàng đợi và lấy các node ra khỏi hàng đợi theo một thứ tự thích hợp, chú ý rằng khi ta nói chứa một node vào trong hàng đợi có nghĩa là ta chứa địa chỉ của node đó vào trong hàng đợi, các địa chỉ được giả sử như trong hình 2.

Cụ thể các bước thực hiện với hàng đợi như sau:

Chú ý: Ở trong ví dụ này, các trạng thái của node được thể hiện băng 3 mầu:

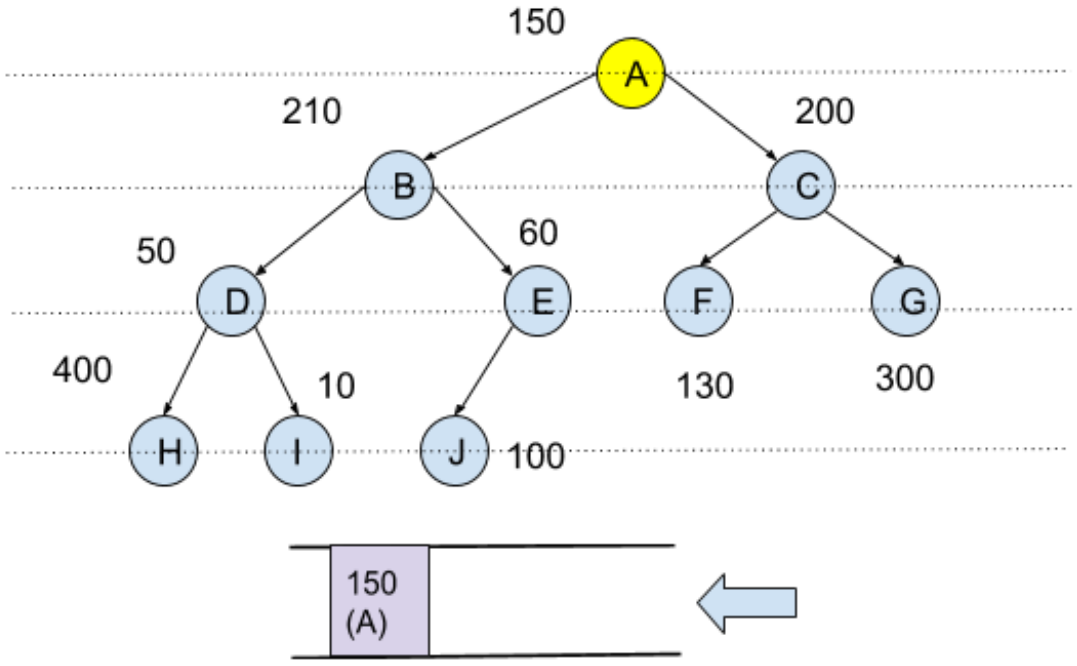
Node xanh nhạt: Node chưa được duyệt tới. Lúc này, node chưa được chứa vào trong hàng đợi.

Node mầu vàng: Node đang được duyệt hay còn gọi là node khám phá. Lúc này, node đang được chứa ở trong hàng đợi.

Node xanh lá: Node đã được duyệt xong. Lúc này, node đã được lấy ra khỏi hàng đợi.

Bước 1:

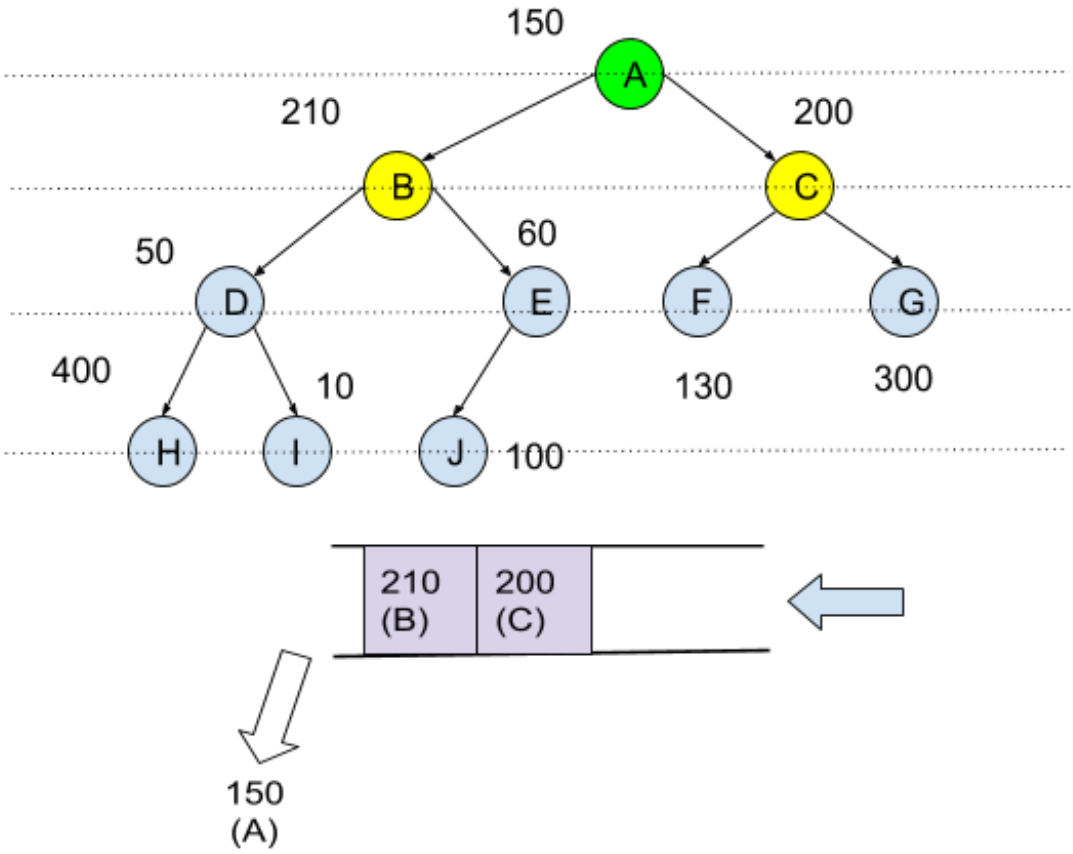
- Duyệt node gốc, đánh dấu node đó là đang được duyệt (node khám phá).
- Enqueue node gốc (A) vào hàng đợi bf_queue.



Hình 4: Duyệt bước 1

Bước 2:

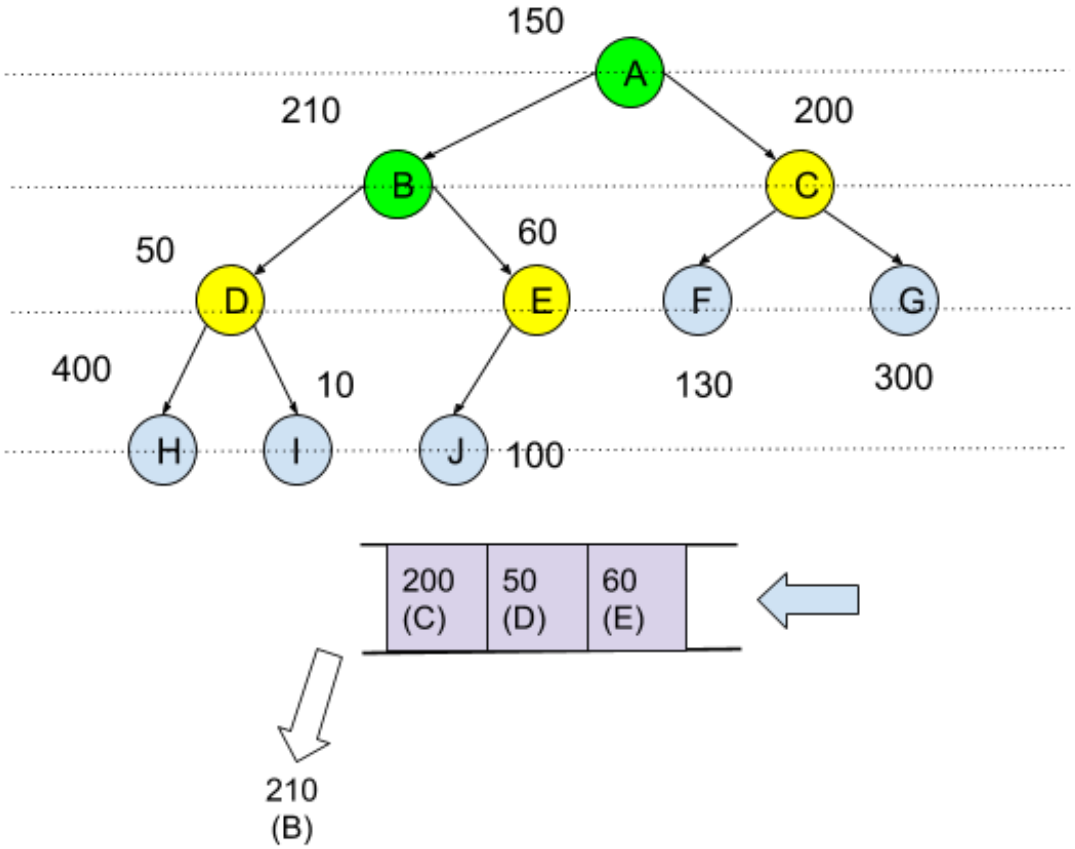
- Kiểm tra queue, queue không rỗng, lấy (dequeue) phần tử đầu tiên (A) ra khỏi queue. Phần tử này chuyển về trạng thái đã duyệt.
- Enqueue các con (B, C) của phần tử vừa dequeue (A) vào bf_queue.
- In ra dữ liệu của node A.



Hình 5: Duyệt bước 2

Bước 3:

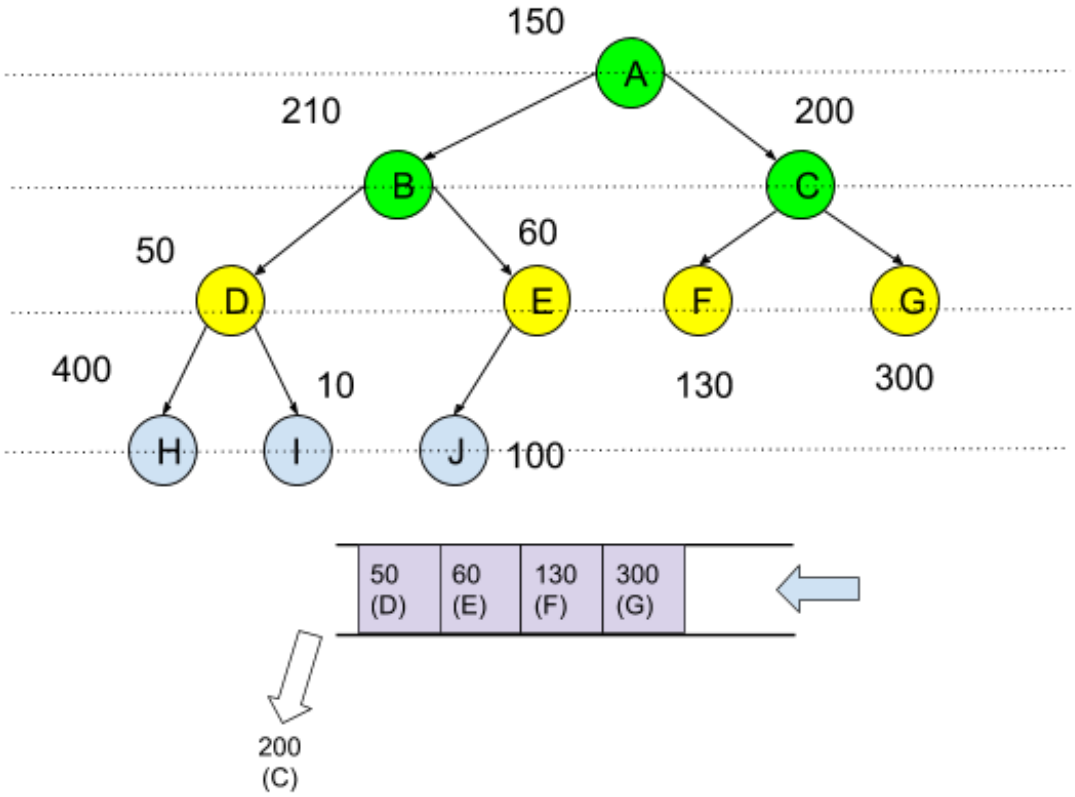
- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (B) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- Enqueue các con (D, E) của phần tử vừa được dequeue (B) vào bf_queue.
- In ra dữ liệu node B.



Hình 6: Duyệt bước 3

Bước 4:

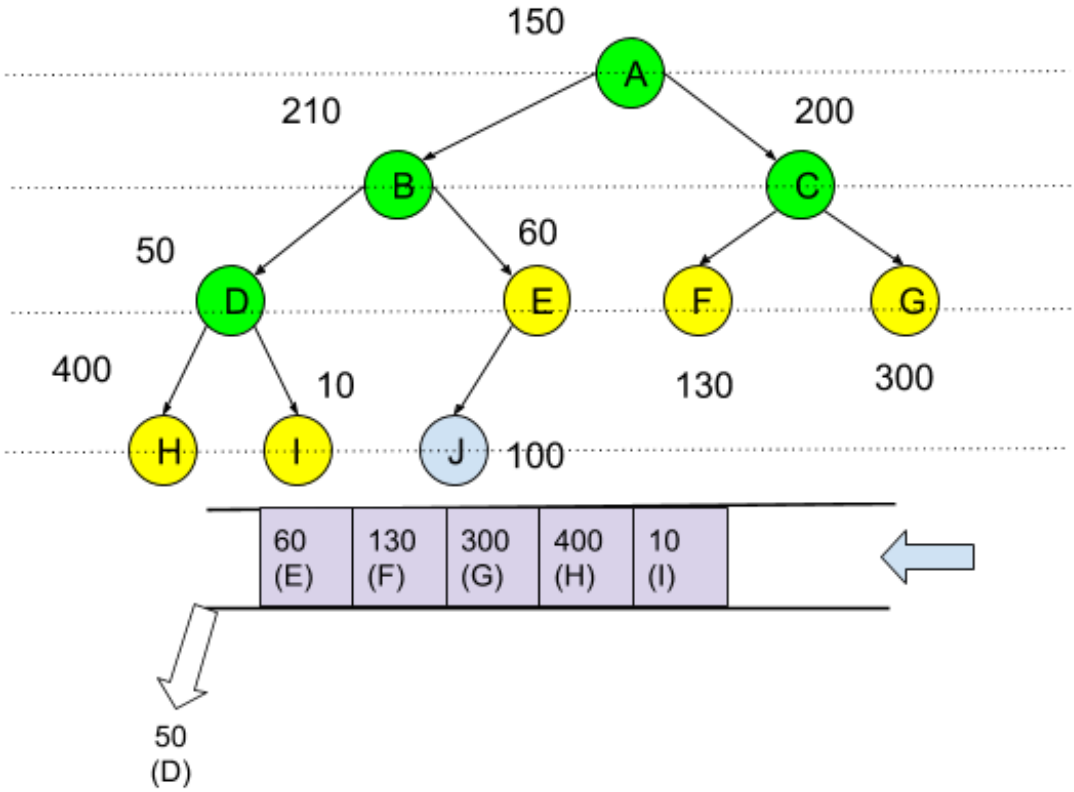
- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (C) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- Enqueue các con (F, G) của phần tử vừa được dequeue (C) vào bf_queue.
- In ra dữ liệu node C.



Hình 7: Duyệt bước 4

Bước 5:

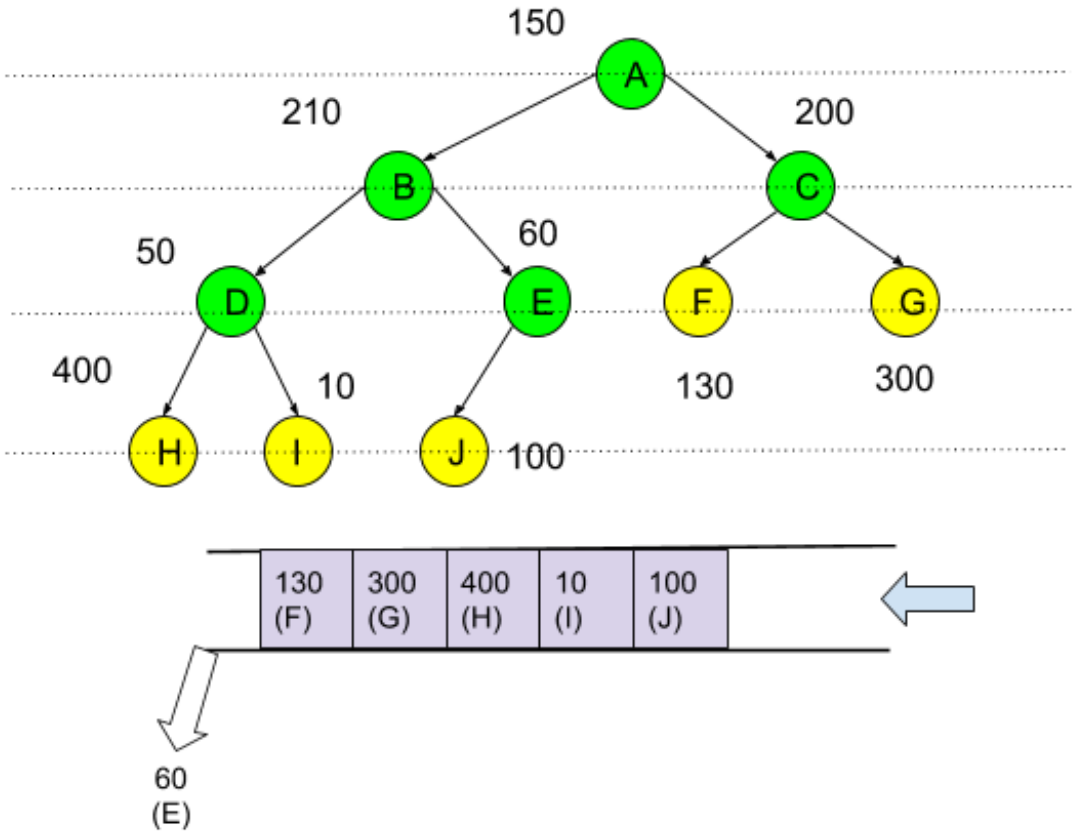
- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (D) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- Enqueue các con (H, I) của phần tử vừa được dequeue (D) vào bf_queue.
- In ra dữ liệu node D.



Hình 8: Duyệt bước 5

Bước 6:

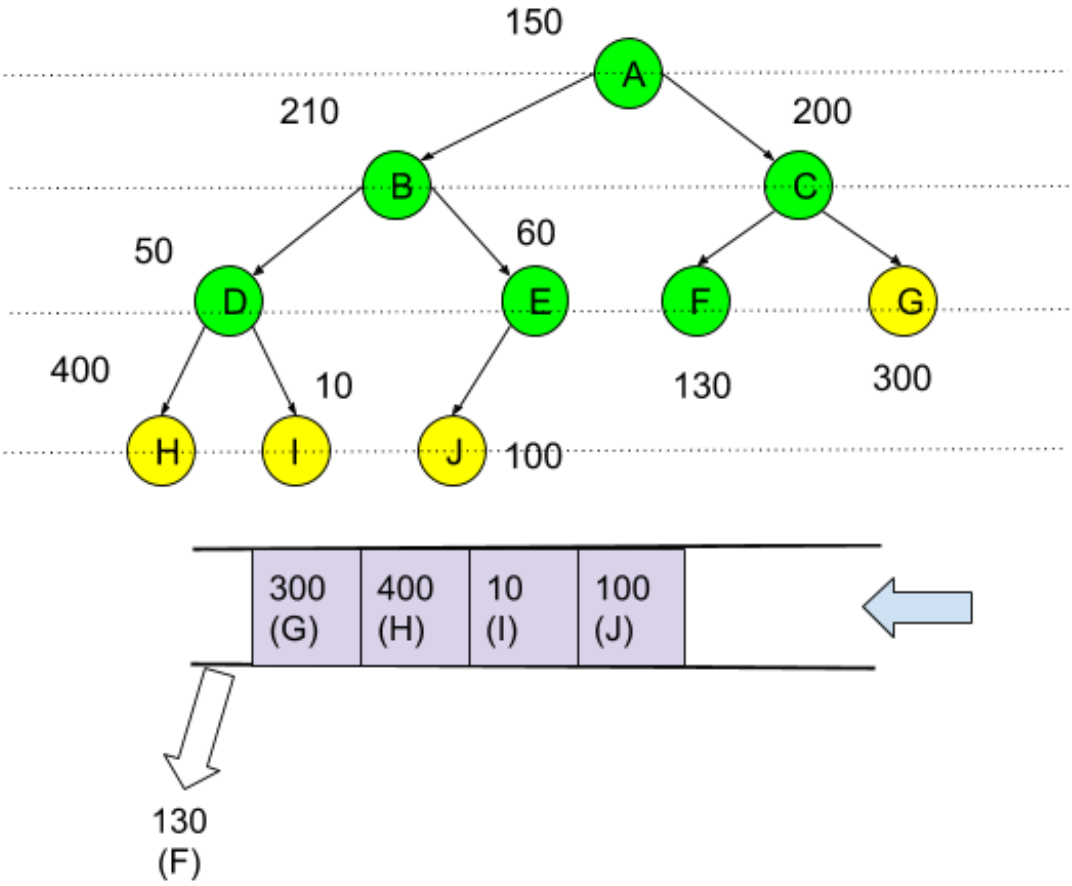
- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (E) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- Enqueue các con (J) của phần tử vừa được dequeue (E) vào bf_queue.
- In ra dữ liệu node E.



Hình 9: Duyệt bước 6

Bước 7:

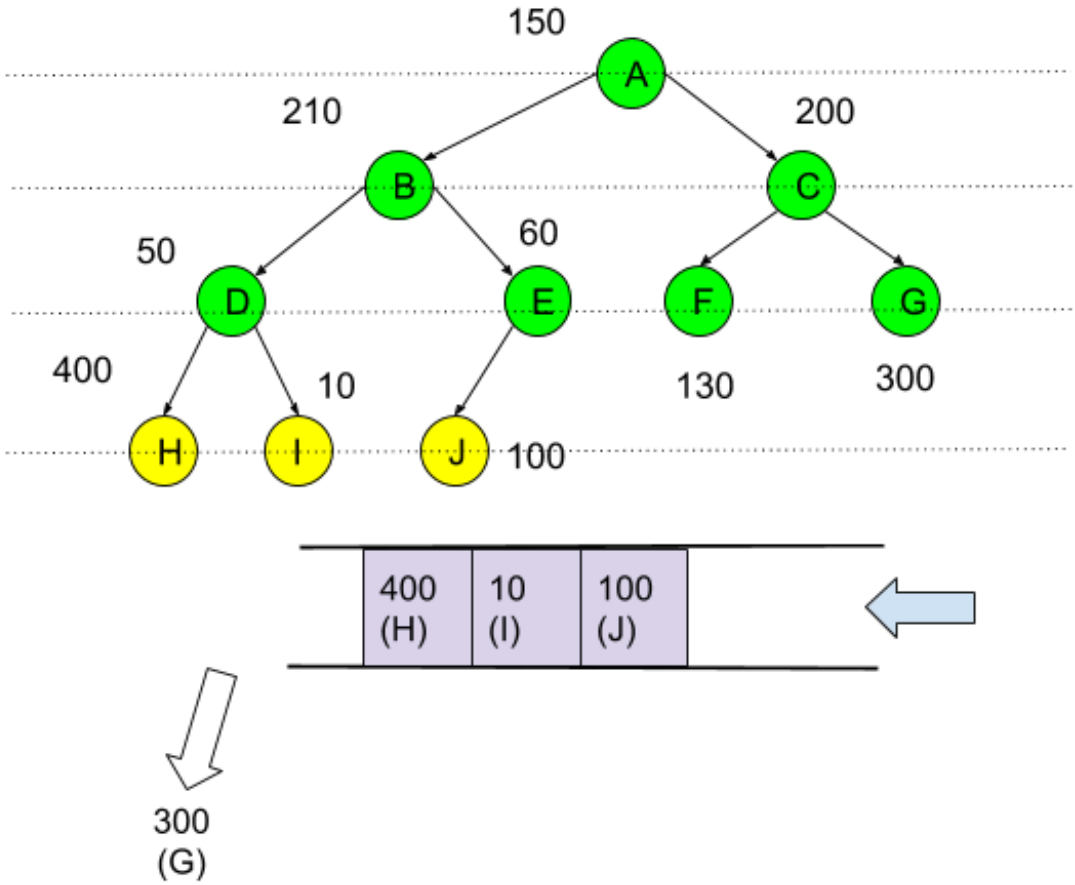
- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (F) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- F là node lá, không có con nên không cần enqueue.
- In ra dữ liệu node F.



Hình 10: Duyệt bước 7

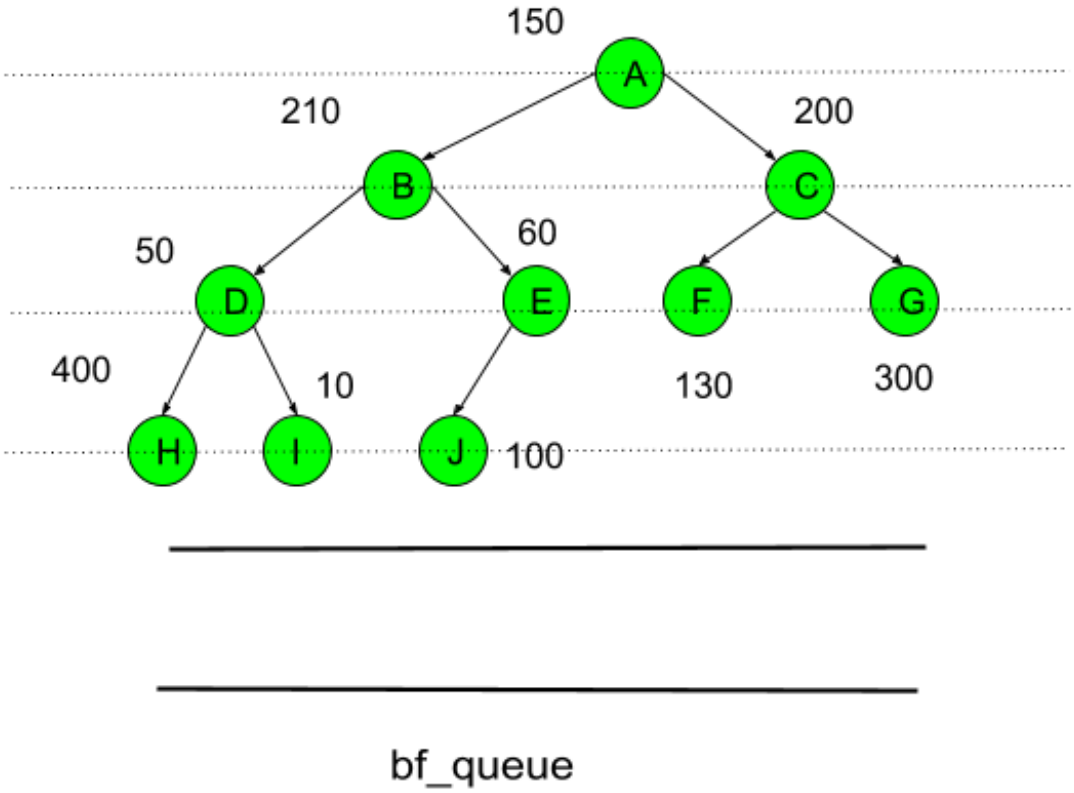
Bước 8:

- Kiểm tra queue, queue không rỗng, lấy phần tử đầu tiên (G) ra khỏi queue. Phần tử này chuyển sang trạng thái đã duyệt.
- G là node lá, không có con nên không cần enqueue.
- In ra dữ liệu node G.



Hình 11: Duyệt bước 8

Thực hiện các bước tương tự như bước 8 cho tới khi hàng đợi rỗng thì ta dừng việc duyệt lại (điều kiện để dừng việc duyệt chính là khi hàng đợi rỗng). Khi đó toàn bộ cây đã được duyệt và dữ liệu in ra theo thứ tự: A, B, C, D, E, F, G, H.



Hình 12: Trạng thái cây và hàng đợi sau khi kết thúc duyệt

Cách thực hiện duyệt cây theo mức với C++

Thuật toán: Giống với những gì tôi đã giải thích ở trên. Nói chung thuật toán bao gồm 3 bước chính.

1. Tạo một queue rỗng bf_queue.
2. Enqueue node gốc vào bf_queue,
3. Vòng lặp while khi queue không rỗng
    1. Bắt đầu duyệt cây, gán node tạm current băng phần tử đầu tiên của queue.
    2. Dequeue phần tử đầu tiên từ bf_queue.
    3. In ra dữ liệu của node current.
    4. Enqueue con trái và con phải của current vào bf_queue.

Mã nguồn:

```
/*
Cấu tạo một node bất kỳ trong cây bao gồm ba phần
Phần dữ liệu data
Phần liên kết tới con bên trái
phần liên kết tới con bên phải
*/
struct Node {
 int data;
 Node *left;
 Node *right;
};

/* Hàm in ra các node trong cây theo mức */
void printLevelOrder(Node *root) {
 if(root == NULL) return;
 queue<Node*> bf_queue;
```

```
 bf_queue.push(root);
 /* Vòng lặp kết thúc khi queue rỗng */
 while(!bf_queue.empty()) {
  Node* current = bf_queue.front();
  bf_queue.pop(); /* Lấy phần tử đầu tiên ra khỏi queue */
  cout<<current->data<<", ";
                /* Enqueue con trái và con phải của current vào bf_queue.*/
  if(current->left != NULL) {
                    bf_queue.push(current->left);
                }
  if(current->right != NULL) {
                    bf_queue.push(current->right);
                 }
 }
}
```

Các bạn có thể lấy toàn bộ mã nguồn của chương trình chạy được để kiểm thử trong đường dẫn này (https://gitlab.com/thevngeek/basic-data-structure/blob/master/duyetcaytheochieurong.cpp).

Tài liệu tham khảo

1. https://gist.github.com/mycodeschool/9507131
2. https://en.wikipedia.org/wiki/Tree_traversal#Breadth-first_search_2

Có điều gì thắc mắc hay muốn góp ý cho bài viết bạn vui lòng để lại bình luận ở phía bên dưới. Tôi sẽ trả lời nhanh nhất có thể để giải đáp các thắc mắc của các bạn.