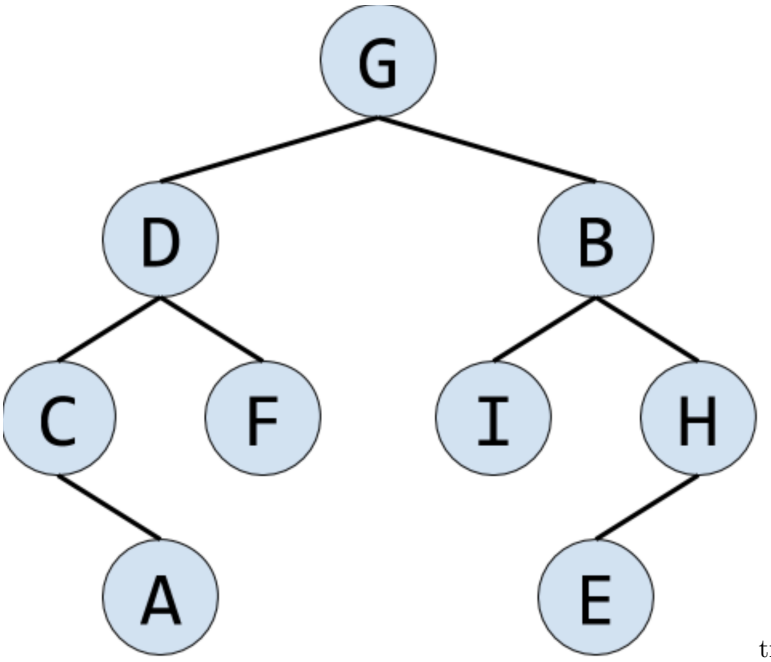
CS 61BL Summer 2021

Tree Traversals and Quiz 6: Friday July 16, 2021

1 Traverse the World

For the next four parts, we will refer to the following tree.



tree.png

In your answer for each of the traversals, **please specify your answer using uppercase letters with no spaces in between them**. For example if you think the traversal order was A, B, C, D, then your answer should be ABCD.

Additionally assume that we should process the children from left to right for traversals who can visit children in an arbitrary order. E.g. in the tree above, at the root G we should process D before B.

(a) What is the pre-order traversal for the tree above?

(b) What is the post-order traversal for the tree above?

(c) What is the in-order traversal for the tree above?

(d) What is the breadth first / level order traversal for the tree above?

2 Stacks

Suppose we have a stack and the following sequence of operations is executed. Before any operations are executed the stack contains B,D,E,F with B being on the top of the stack and F being on the bottom of the stack.

```
// starting stack state B,D,E,F
pop
pop
push A
push G
pop
push C
pop
```

(a) What is left in the stack after the following calls have been made?

Please specify your answer using uppercase letters with no spaces in between them ordered from top to bottom. For example if you think the remaining elements in order are A, B, C, D, with A on top and D on the bottom then your answer should be ABCD.

(b) What are the elements popped off of the stack?

Please specify your answer using uppercase letters with no spaces in between them ordered from the first item popped to the last item popped. For example if you think the elements popped in order were A, B, C, D, with A being popped first and D being popped last then your answer should be ABCD.

3 Disjoint Sets

Suppose we execute the following `connect` calls on a WeightedQuickUnion without path compression. Note that if we connect two sets of equal weight, by convention we make the set whose root has a smaller number the child of the other. We use the convention that if an element is the root of the set, its array value is the weight of the set negated.

```
1 connect(0, 1);
2 connect(2, 3);
3 connect(9, 5);
4 connect(5, 7);
5 connect(7, 1);
6 connect(4, 2);
7 connect(3, 1);
```

Here, fill in the value in each index of the resulting underlying array of the Weighted Quick Union object.