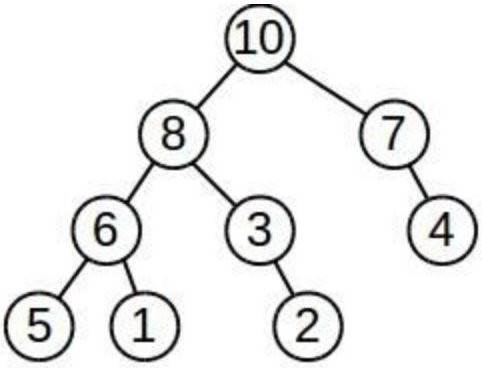
1. What is the order of the nodes visited for an in-order traversal of the following tree?



5, 6, 1, 8, 3, 2, 10, 7, 4

2. Write a high-level version of a Binary Search Tree delete method. The description should be written in either clear pseudo-code (with English explanations) or in precise, well-defined English. There should be only one interpretation for how your method should be carried out. In short the description should not be ambiguous.

```
Delete Value V from Tree T
        Let Node be the node with value V found using some search method
        If Node has 2 children
                Let R be the max node in Node's left child's subtree
                Swap R's and Node's values
                Let Node be R
        End If
        If Node has 1 child
                Replace Node with it's child
        Else
                // Node has 0 children
                Remove the Node reference from it's parent
        End If
```

3. What is the worst and average case Big O runtimes of the following **efficient** but singularly linked list operations?

	Worst	Average
Insert in sorted order	O(N)	O(N)
Delete by value	O(N)	O(N)
Insert at beginning	O(1)	O(1)
Delete beginning	O(1)	O(1)
Freeing full list	O(N)	O(N)

4. Write a Method that returns the sum of values of a given linked list. Use the given Node struct definition definition AND the function prototype.

```
struct Node {
  int value;
  Node * next;
};
int sum(Node * head);

// one liner
int sum(Node * head) {
   return (head == NULL) ? 0 : (head->value + sum(head->next));
}

// more readable
int sum(Node * head) {
   if (head == NULL)
      return 0;
   return head->value + sum(head->next);
}
```

5. Write the linked list of values from front (left) to back (right) after each of the following queue operations. Seperate elements by commas (with a space).

Operation	**Linked List**
Enqueue 5	5
Enqueue 6	5, 6
Dequeue	6
Enqueue 3	6, 3
Enqueue 4	6, 3, 4
Dequeue	3, 4
Enqueue 7	3, 4, 7

6. Finish the dequeue function for a linked list queue. The dequeue should remove from the front. Use the given Node and Queue struct definitions AND the given function prototype.

```
typedef struct Node Node;
typedef struct Queue Queue;
struct Node{
  Node * next;
  int value;
};
struct Queue{
  Node * front;
  Node * end;
};

void dequeue(Queue * qPtr);
```

```
void dequeue(Queue * qPtr) {
   if (qPtr != NULL || qPtr->front == NULL)
      return;
   Node * newFront = qPtr->front->next;
   free(qPtr->front)
   qPtr->front = newFront;
   if (qPtr->front == NULL)
      qPtr->back = NULL;
}
```

7. What value does the following post-fix expressions evaluate to?

```
5 7 + 2 3 * / 1 -
```

1

8. What value does the following post-fix expressions evaluate to?

```
1 2 3 4 5 * * * * 5 +
```

125

9. What is the post-fix representation of the following in-fix expression?

   1 - (8 - 9 * 2) + 3 * 4 / 6

   1 8 9 2 * - - 3 4 * 6 / +