JavaScript Asymptotic Notations

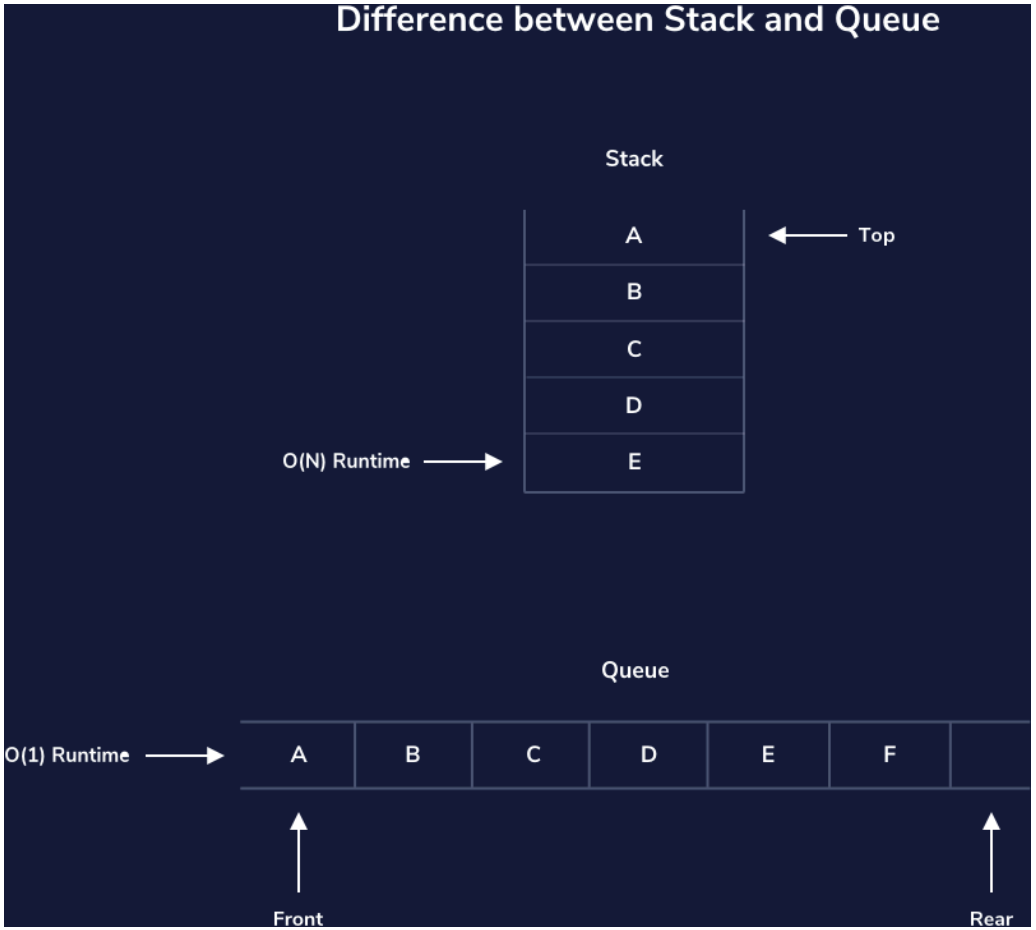
Analyzing Runtime

The speed of an algorithm can be analyzed by using a while loop. The loop can be used to count the number of iterations it takes a function to complete.

```
def half(N):
  count = 0
  while N > 1:
    N = N//2
    count += 1
  return count
```

Queue Versus Stack

A `Queue` data structure is based on First In First Out order. It takes O(1) runtime to retrieve the first item in a `Queue`. A `Stack` data structure is based on First In Last Out order. Therefore, it takes O(N) runtime to retrieve the first value in a `Stack` because it is all the way at the bottom.



Max Value Search in List

The big-O runtime for locating the maximum value in a list of size N is O(N). This is because the entire list of N members has to be traversed.

```
# O(N) runtime
def find_max(linked_list):
  current = linked_list.get_head_node()
  maximum = current.get_value()
  while current.get_next_node():
    current = current.get_next_node()
    val = current.get_value()
    if val > maximum:
      maximum = val
  return maximum
```

Bubble Sort with Linked List

Bubble Sort is the simplest sorting algorithm for a list. For every element in the list, it compares it with its subsequent neighbor and swaps them if they are in descending order. Each pass of the swap takes O(N). Since there are N elements in the list, it would take N*N swaps. The Big O runtime would be O(N^2).

Print