Binary Search

→ used to efficiently locate a target value within a sorted sequence of n elements
→ maintains 2 parameters, low and high, such that all candidate entries have index at least low and at most high.

$$mid = \lfloor (low + high)/2 \rfloor$$

→ if target = data[mid], stop
→ if target < data[mid], recur on first half of sequence (mid-1)
→ if target > data[mid], recur on second half of sequence (mid+1)

```
def binary_search(data, target, low, high):
    if low > high:
        return False
    else:
        mid = (low + high) / 2
        if target == data[mid]:
            return True
        elif target < data[mid]:
            // recur on left side of arr
            return binary_search(data, target, low, mid-1)
        else:
            // recur on right
            return binary-search(data, target, mid+1, high)
```

Recursive Function to Calculate Disk Usage : return bytes used by folder & descendants

```
def disk_usage(path):
    total = os.path.getsize(path)
    if os.path.isdir(path):
        for filename in os.listdir(path)
            childpath = os.path.join(path, filename)
            total += disk_usage(childpath)
print('{0:<7}'.format(total), path)
    return total
```