In [7]:
```
def mergesort(array):
    """ Returns a sorted version of `array` """
    print(f"Merge sorting {array}")
    if len(array) <= 1:
        return array
    half = len(array) // 2
    # recursively sort first and second half
`    result = [None for _ in array]
    j = k = 0
    for i in range(len(array)):
        # result[i] = min(first_half[j:], second_half[k:])
        if j < len(first_half) and \
            (k >= len(second_half) or first_half[j] < second_half[k]):
            result[i] = first_half[j]
            j += 1
        else:
            result[i] = second_half[k]
            k += 1
    print(f"Returning sorted {result}")
    return result
```

In [8]:
```
mergesort("algorithms")
```

```
Merge sorting algorithms
Merge sorting algor
Merge sorting al
Merge sorting a
Merge sorting l
Returning sorted ['a', 'l']
Merge sorting gor
Merge sorting g
Merge sorting or
Merge sorting o
Merge sorting r
Returning sorted ['o', 'r']
Returning sorted ['g', 'o', 'r']
Returning sorted ['a', 'g', 'l', 'o', 'r']
Merge sorting ithms
Merge sorting it
Merge sorting i
Merge sorting t
Returning sorted ['i', 't']
Merge sorting hms
Merge sorting h
Merge sorting ms
Merge sorting m
Merge sorting s
Returning sorted ['m', 's']
Returning sorted ['h', 'm', 's']
Returning sorted ['h', 'i', 'm', 's', 't']
Returning sorted ['a', 'g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']
```

Out[8]:
```
['a', 'g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']
```

In [12]:
```
def quicksort(array):
    print(f"quicksorting {array}")
    if len(array) <= 1:
        return array
    pivot = array[0]
    first_half = []
    second_half = []
    for i in range(1, len(array)):
        if array[i] < pivot:
            first_half.append(array[i])
        else:
            second_half.append(array[i])
    print(f"split: {first_half} {pivot} {second_half}")
    first_half_sorted = quicksort(first_half)
    second_half_sorted = quicksort(second_half)
    return first_half_sorted + [pivot] + second_half_sorted
```

In [13]:
```
quicksort("algorithms")
```

```
quicksorting algorithms
split: [] a ['l', 'g', 'o', 'r', 'i', 't', 'h', 'm', 's']
quicksorting []
quicksorting ['l', 'g', 'o', 'r', 'i', 't', 'h', 'm', 's']
split: ['g', 'i', 'h'] l ['o', 'r', 't', 'm', 's']
quicksorting ['g', 'i', 'h']
split: [] g ['i', 'h']
quicksorting []
quicksorting ['i', 'h']
split: ['h'] i []
quicksorting ['h']
quicksorting []
quicksorting ['o', 'r', 't', 'm', 's']
split: ['m'] o ['r', 't', 's']
quicksorting ['m']
quicksorting ['r', 't', 's']
split: [] r ['t', 's']
quicksorting []
quicksorting ['t', 's']
split: ['s'] t []
quicksorting ['s']
quicksorting []
```

Out[13]:
```
['a', 'g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']
```

In [14]:
```
quicksort("aghilmorst")
```

```
quicksorting aghilmorst
split: [] a ['g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']
quicksorting []
quicksorting ['g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']
split: [] g ['h', 'i', 'l', 'm', 'o', 'r', 's', 't']
quicksorting []
quicksorting ['h', 'i', 'l', 'm', 'o', 'r', 's', 't']
split: [] h ['i', 'l', 'm', 'o', 'r', 's', 't']
quicksorting []
quicksorting ['i', 'l', 'm', 'o', 'r', 's', 't']
split: [] i ['l', 'm', 'o', 'r', 's', 't']
quicksorting []
quicksorting ['l', 'm', 'o', 'r', 's', 't']
split: [] l ['m', 'o', 'r', 's', 't']
quicksorting []
quicksorting ['m', 'o', 'r', 's', 't']
split: [] m ['o', 'r', 's', 't']
quicksorting []
quicksorting ['o', 'r', 's', 't']
split: [] o ['r', 's', 't']
quicksorting []
quicksorting ['r', 's', 't']
split: [] r ['s', 't']
quicksorting []
quicksorting ['s', 't']
split: [] s ['t']
quicksorting []
quicksorting ['t']
```

Out[14]: `['a', 'g', 'h', 'i', 'l', 'm', 'o', 'r', 's', 't']`

In [ ]: