Stackless Quicksort

Henry Chen

scotchka.github.io

Quicksort

To sort a list:

1) Pick an element - call it the pivot.

2) Partition list such that lesser elements are to the left, greater to the right. The pivot is now at the correct position.

3) The left and right segments remain unsorted.

4) Repeat from 1) on unsorted segments until none remain.

Quicksort is recursive

```
def qsort(lst, start=0, end=None):
    if end is None:
        end = len(lst)

    if end - start < 2:
        return

    pivot_position = partition(lst, start, end)

    qsort(lst, start, pivot_position)
    qsort(lst, pivot_position + 1, end)
```

Tip: when doing recursion, DO NOT think about the call stack!

scotchka.github.io

Recursion -> iteration + stack

```
def qsort_iterative(lst):
    stack = [(0, len(lst))]

    while stack:
        start, end = stack.pop()

        pivot_position = partition(lst, start, end)

        if pivot_position - start > 0:
            stack.append((start, pivot_position))

        if end - (pivot_position + 1) > 0:
            stack.append((pivot_position + 1, end))
```

scotchka.github.io

Leslie Lamport: a stack (LIFO) is not necessary

To sort a list:

1) Pick an element - call it the pivot.

2) Partition list such that lesser elements are to the left, greater to the right. The pivot is now at the correct position.

3) The left and right segments remain unsorted.

4) Repeat from 1) on unsorted segments until none remain.

scotchka.github.io

LIFO -> arbitrary order

```
def qsort_stackless(lst):
    not_sorted = {(0, len(lst))}

    while not_sorted:
        start, end = not_sorted.pop()

        pivot_position = partition(lst, start, end)

        if pivot_position - start > 0:
            not_sorted.add((start, pivot_position))

        if end - (pivot_position + 1) > 0:
            not_sorted.add((pivot_position + 1, end))
```

scotchka.github.io

Why didn’t I think of this?

- Recursion is implementation
- We tend to focus on implementation
- Idea is not implementation
- We should focus on ideas

More details -> scotchka.github.io