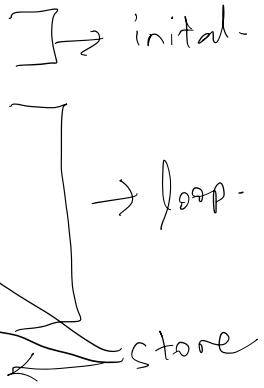
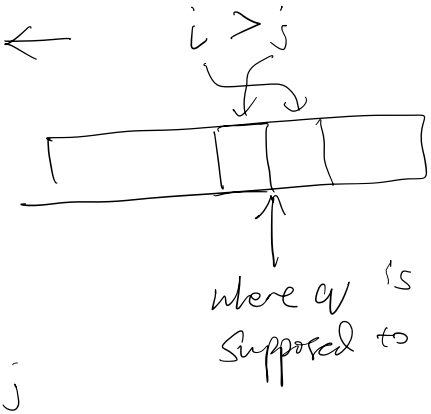
```
long search(long list[], long i, long j, long q) {
  if (i > j) {
    return -1;
  }
  long mid = (i+j)/2;
  if (list[mid] == q) {
    return mid;
  }
  if (list[mid] > q) {
    return search(list, i, mid-1, q);
  }
  return search(list, mid+1, j, q);
}
```



The idea is to copy parts to the loop, replacing recursion.

Invariant is that

@ return (-1) ← i > j



i-1 or j

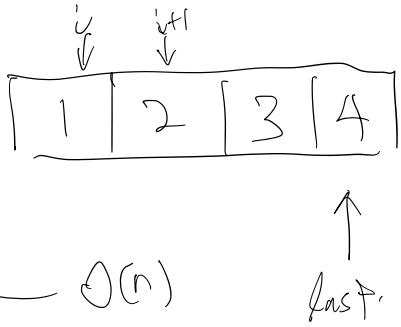
```
void bubble_pass(size last, long a[]) {
  for (size_t i = 0; i < last; i += 1) {
    if (a[i] > a[i+1]) {
      swap(a, i, i+1);
    }
  }
}

void bubble_sort(size_t n, long a[]) {
  for (size_t last = n - 1; last > 0; last -= 1) {
    bubble_pass(last, a);
  }
}
```

O(n)

O(n)

Consider.

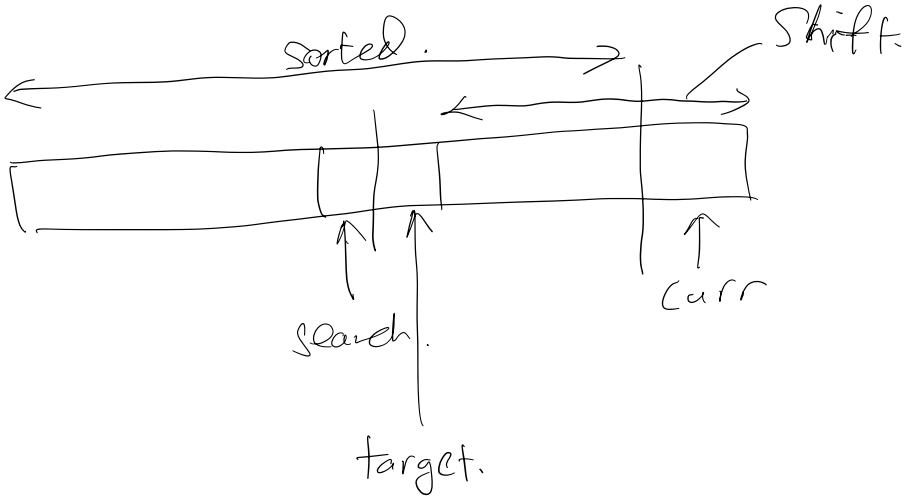


Still $O(n^2)$ even no swaps done.

→ When can we stop checking?

→ bubble pass need to run at least once.

$$n < \text{Runtime} < n^2$$



comparisons.

Before $O(n^2)$

After $O(n \log(n))$ ← comparisons only in Binary search!