Eliminating Tail Recursion

→ if any recursive call made from one context is the last operation in that context, with the return value of the recursive call immediately returned by the enclosing recursion

```
def binary_search_iterative(data, target):
  low = 0
  high = len(data)-1
  while low <= high:
    mid = (low + high) // 2
    if target == data[mid]:
      return True
    elif target < data[mid]:
      high = mid-1
    else
      low = mid + 1
  return False
```

→ where we make a recursive call in the original algorithm, simply replace it with high = mid-1 and continue

→ original base case low <= high in while loop; in new implementation

→ removes tail recursion by enclosing the body in loop for repetition

```
def reverse_iterative(S):
  start, stop = 0, len(S)
  while start < stop -1:
    S[start], S[stop-1] = S[stop-1], S[start]   // swap 1st and last
    start, stop = start+1, stop-1               // narrow range
```