Decrease-Conquer Binary Search

3 14 27 31 39 **42** 55 70 74 81 85 93 98
△

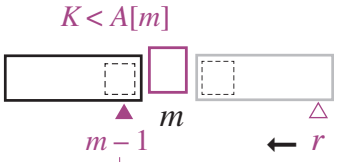
Classic top-down

Exercise
Write the sequence of lengths of searched lists in each iteration (code and generate the log below).

```
? [0 ?], m = ? (?)
6 [0 5], m = 2 (27)
3 [3 5], m = 4 (39)
1 [5 5], m = 5
```

Algorithm *BinarySearch*
Input $A[0..n-1]$ sorted in ascending order, key K
Output key index in A if found, -1 otherwise

1: $l \leftarrow 0,\ r \leftarrow n-1$
2: **while** $l \leq r$ **do**
3: $\quad m \leftarrow \lfloor \frac{l+r}{2} \rfloor$
4: $\quad$ **if** $K = A[m]$ **then return** m
5: $\quad$ **else if** $K < A[m]$ **then** $r \leftarrow m-1$
6: $\quad$ **else** $l \leftarrow m+1$
7: **return** -1



3-way key-comp counts once regardless of its runtime cost (why?)

Binary Search Analysis

Quiz
What's a suitable basic operation?

➪ **Choice of basic operation**

Quiz
Specify inputs leading to best and worst cases for efficiency. Which keys in example cause each?

➪ **Count dependance**

- ✎ Best case, efficiency?
- ✎ Worst case

A **tail recursion** involving one recursive call to a smaller instance, typical of decrease-conquer, is often easy to specify.

Exercise
Compare to definition-based recursive version.

Design clues

A simple recursive plan

Algorithm *Factorial* (n, val)
1: **if** $n = 0$ **then**
2: **return** val
3: $val \leftarrow val \times n$ — $f(n)$
4: **return** *Factorial* ($n - 1$, val)

Binary Search Efficiency

📖 Appx B ➪ Smoothness rule

Exercise
Write a recursive pseudo-code for binary search.

➪ Initial condition: $C(n = 1)$

Quiz
Guess an $f(n)$ for insertion sort. Write a recurrence. (Ans. last slide).

➪ Worst-case recurrence

- ✎ Solve for $n = 2^k$, $k = 1, 2, \ldots$
- ✎ Use smoothness rule to extend result

Exercise
Show that $n\log_2 n$ is smooth.

➪ Smooth functions

➪ Average-case efficiency

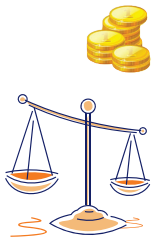
Decrease-by-a-Constant-Factor Design Pattern

Quiz
State concisely why both binary search and the fake-coin problem may not be considered divide-conquer. **Hint**: any of 3 reasons good enough.

⇨ Principle

Use mid-point key to eliminate half of ordered list

⇨ Fake-coin problem



- ✎ n-Coin underweight-fake version
- ✎ Brute-force thinking
- ✎ Decrease-by-half solution

Quiz
What's the initial condition in the recurrence?

Quiz (Previous Slide)
Recall, insertion sort does at most $n-1$ comps per iteration therefore worst case $f(n) = n-1$, init cond 0 at $n=1$ (no comps when reduced to 1 element), try it in *WolframAlpha*.

Test 1 Review
Proper Pseudocode

Pseudocode should not lead to wrong results due to mis-statement.

⇨ Statement vs. correctness

Input specs typically essential in how steps are stated.

⇨ Specify legal input instances

❶ Inputs in terms of parameters ❷ used in steps, limitations or ❸ preconditions

Components of statement

A search may return true/false (or 0/1) or an index in an array, or a pointer in a linked list or the found item itself.

⇨ Specify expected results

❶ Output items or effects, ❷ specs (how) if needed

User must be able to unambiguously follow intended steps even if the algorithm is flawed.

⇨ Elucidate structure to clarify logic

Iterations, conditionals, indent to show nesting