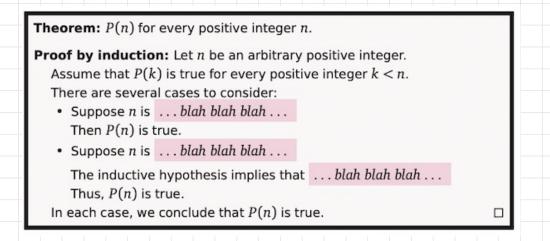
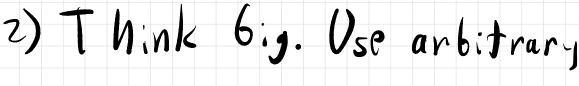
Proots by induction:

1) Write a (good) template,





(prssibly large) values of n.

Reduce to a smaller value

is you can.

3) Look for holes (base cases).

4) Rewrite everything so its easier to read tfollow.

Don'):

-assame only for k=n-1 -just assume n + prore

for nt)

Kp cursion:

Roduce large instances

of some problem A to

smaller instances of the

Same problem A.

It you can't reduce, solve the base case directly.

(don't need to le too clover, usually)

Mergesort von Neumann '45

Input: An array All.in]

ot things to sort by

pairwise comparisons

(integers, characters, etc.)

Goal: Rearrange A's chements so $A(I) \in A[2] \in ... \in A[n]$ 1) Divide orray into two subarrays of roughly equal size.

2) Recarsively morge sort the two subarrays. (magic) 3) Mergo the two conted subarrays quickly.

> Input: SORTINGEXAMPL SORTIN|GEXAMP Divide: L INORSTGEXAMP Recurse Left: L INORSTAEGLMP Recurse Right: Х AEGILMNOPRSTX Merge:

It n=1, lo nothing instead.

Merge:) Use smaller Sirst member two halves ot the sorted! 2) Recarsively sort what remains of the subarrays. -Sever elements total -so This works (by induction) assumes ACI ... m J+A[m+1... > are-sorted!" Merge(A[1..n], m): $i \leftarrow 1; j \leftarrow m+1$ for $k \leftarrow 1$ to nif j > nMergeSort(A[1..n]): $B[k] \leftarrow A[i]; i \leftarrow i+1$ if n > 1else if i > m $m \leftarrow \lfloor n/2 \rfloor$ $B[k] \leftarrow A[j]; j \leftarrow j+1$ MergeSort(A[1..m]) ((Recurse!)) else if A[i] < A[j]MERGESORT(A[m+1..n]) ((Recurse!)) $B[k] \leftarrow A[i]; i \leftarrow i+1$ Merge(A[1..n], m)else $B[k] \leftarrow A[j]; j \leftarrow j+1$ for $k \leftarrow 1$ to n $A[k] \leftarrow B[k]$

Correctness:

Thm: Assuming that Merge (B[1., l], k) sorts Bis B(InkJ&B(k+1,1) are sorted Merge Sort (A(1...) sorts A.

(notation note: C(n+1., n] is empty)

sorts D whonever ocn.

-if $n \in J$, A is sorted $\int -0.W$, $m \in n \neq n - (m \neq I) \neq I \in n$ *Cotherwise*

By assumption (IH) Merge Sort (A(1...m)) Morge Sort (A(m+1...n)) sort

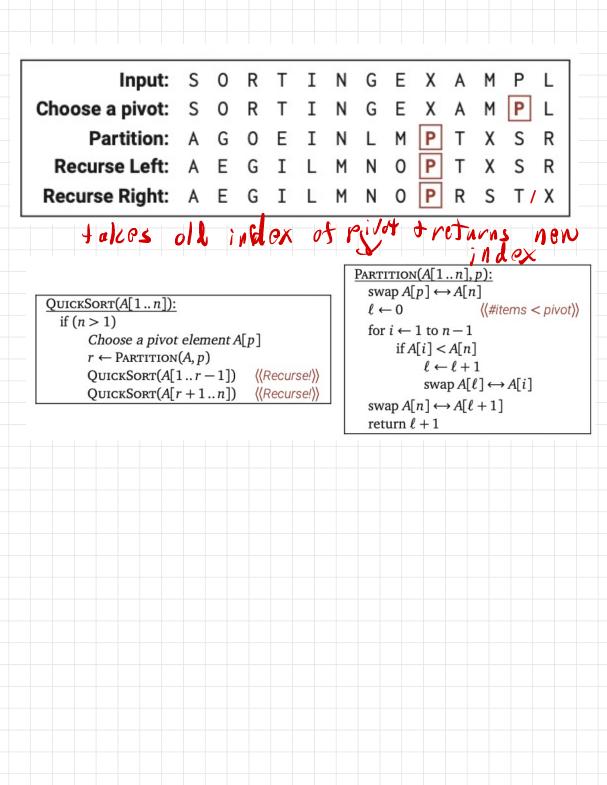
Merge does merge the now-south Subarrays.

Quicksont Houre'59

D Choose a pilot element from the array.

2) Partion array into 3 subarrays stoved in this order: D Elements smaller than pilot. 2) Just the pivot 3) Elements larger than givot.

3) Recursively sort 1)+3)



Divide - and - conquer Divide given instance to create one or more independent smaller instances ot the same problem. 2) Delegate smaller instances to Recursion Fairy. 3) Combine solutions for smaller instances. If instance cannot be divided solve as a base case.