You will be asked about one of these algorithms on the final:

binary search
Dutch national flag
partition algorithm
insertion sort
selection sort

You **have** to know the specifications of these algorithms. Memorize them.

Then, develop an invariant from the spec and develop the loop from using 4 loopy questions. An algorithm given without the inv or an algorithm that doesn't use the inv is simply wrong.

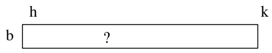
Review of Required Array Algorithms

Reasons for this:

1. Important algorithms.
2. Forces you to think in terms of specifications.
3. Forces you do learn to develop invariants.
4. Forces you to learn to use the four loopy questions in reading or developing a loop

Below are specs of the algorithms. Memorize them.
We give pre- and post-conditions as diagrams. Same for loop invariants. You can do this too.

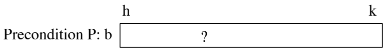
We may specify that an algorithm is working with an array b[0..b.length-1], or a segment b[h..k], or a segment b[m..n-1]. It doesn't matter; work with whatever is given.



Remember, the number of values in an array segment is given by the formula Following – First
—e.g. the number of values in b[h..k-1] is k – 1.

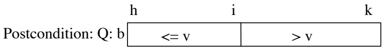
Binary search:
Vague spec: Look for v in **sorted** array segment b[h..k].

BETTER IS:



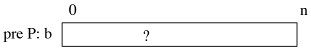
(in ascending order)

Store in i to truthify:

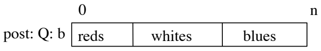


Called binary search because each iteration of the loop will cut the ? Segment in half

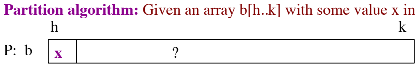
Dutch National Flag
b contains red balls, white balls, and blue balls



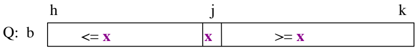
Swap the balls in b[0..n-1] so that



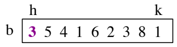
Partition algorithm: Given an array b[h..k] with some value x in b[h]:



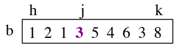
Swap elements of b[h..k] and store in j to truthify Q:



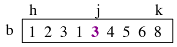
e.g. change:



into

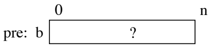


or

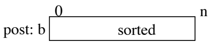


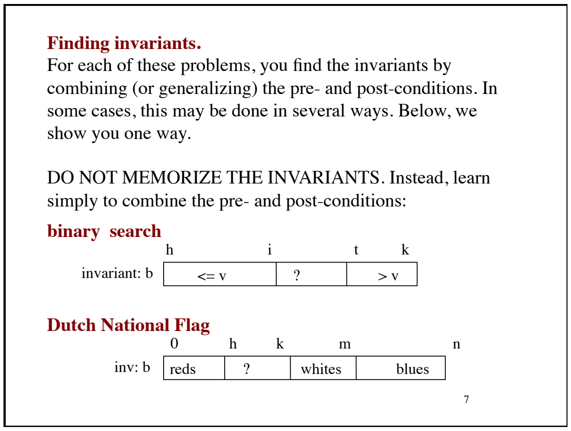
x is called the pivot value.
x is not a program variable; x just denotes the value initially in b[h].

Insertion sort AND
Selection sort



Swap values of b[0..n] so that the array looks like this:





Finding invariants.

For each of these problems, you find the invariants by combining (or generalizing) the pre- and post-conditions. In some cases, this may be done in several ways. Below, we show you one way.

DO NOT MEMORIZE THE INVARIANTS. Instead, learn simply to combine the pre- and post-conditions:

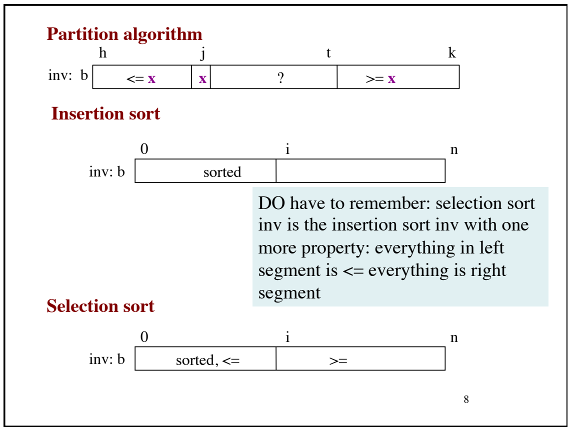
binary search

	h	i	t	k
invariant: b	<= v	?	> v	

Dutch National Flag

	0	h	k	m	n
inv: b	reds	?	whites	blues	

7



Partition algorithm

	h	j	t	k
inv: b	<= x	x	?	>= x

Insertion sort

	0	i	n
inv: b	sorted		

DO have to remember: selection sort inv is the insertion sort inv with one more property: everything in left segment is <= everything is right segment

Selection sort

	0	i	n
inv: b	sorted, <=	>=	

8