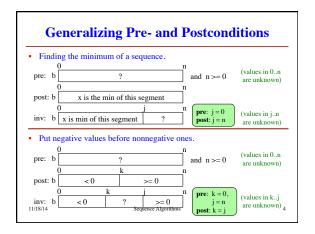
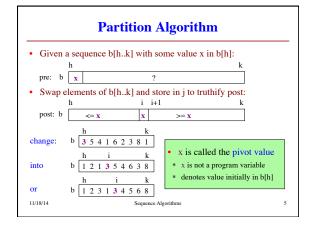


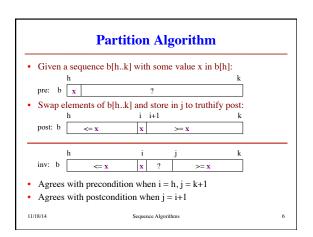
Developing Algorithms on Sequences

- Specify the algorithm by giving its precondition and postcondition as pictures.
- Draw the invariant by drawing another picture that "generalizes" the precondition and postcondition
- The invariant is true at the beginning and at the end
- The four loop design questions (memorize them)
 - 1. How does loop start (how to make the invariant true)?
 - 2. How does it stop (is the postcondition true)?
 - 3. How does the body make progress toward termination?
 - 4. How does the body keep the invariant true?

11/18/14 Sequence Algorithms 2

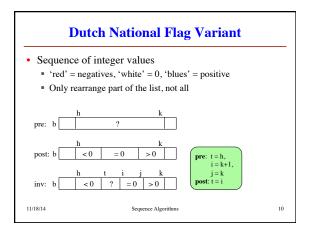






Partition Algorithm Implementation def partition(b, h, k): "Partition list b[h..k] around a pivot x = b[h]""" i = h; j = k+1; x = b[h]# invariant: b[h..i-1] < x, b[i] = x, b[j..k] >= xwhile i < i-1: if b(i+1) >= x: partition(b,h,k), not partition(b[h:k+1]) # Move to end of block. Remember, slicing always copies the list! _swap(b,i+1,j-1) We want to partition the original list **else**: # b[i+1] < x _swap(b,i,i+1) i = i + 1# post: b[h..i-1] < x, b[i] is x, and b[i+1..k] >= xreturn i Sequence Algorithms

Partition Algorithm Implementation "Partition list b[h..k] around a pivot x = b[h]""" i = h; j = k+1; x = b[h]1 2 3 1 5 0 6 3 8 # invariant: b[h..i-1] < x, b[i] = x, b[j..k] >= xwhile i < i-1: i i+1 if b(i+1) >= x: 1 3 5 0 6 3 8 # Move to end of block. _swap(b,i+1,j-1) j=j-1 1 3 0 5 6 3 8 else: # b[i+1] < x $_{ ext{swap}(b,i,i+1)}$ i = i + 1# post: b[h..i-1] < x, b[i] is x, and b[i+1..k] >= x1 2 1 0 3 5 6 3 8 return i Sequence Algorithms



```
Dutch National Flag Algorithm
def dnf(b, h, k):
                                                                           = 0
   ""Returns: partition points as a tuple (i,j)"
  t = h; i = k+1, j = k;
                                                    -1 -2 3 -1 0 0 0 6 3
  # inv: b[h..t-1] < 0, b[t..i-1] ?, b[i..j] = 0, b[j+1..k] > 0
  while t < i:
    if b[i-1] < 0:
      swap(b,i-1,t)
      t = t+1
    elif b[i-1] == 0:
    i = i-1
     else:
      swap(b,i-1,j)
      i = i-1; j = j-1
  # post: b[h..i-1] < 0, b[i..j] = 0, b[j+1..k] > 0
  return (i, j)
11/18/14
                                    Sequence Algorithms
```

```
Dutch National Flag Algorithm
def dnf(b, h, k):
                                                                     = 0
  """Returns: partition points as a tuple (i,j)"
 t = h; i = k+1, j = k;
                                                 -1 -2 3 -1 0 0 0 6 3
  # inv: b[h..t-1] < 0, b[t..i-1] ?, b[i..j] = 0, b[j+1..k] > 0
  while t < i:
    if b[i-1] < 0:
                                                 -1 -2 3 -1 0 0 0 6 3
      swap(b,i-1,t)
      t = t+1
    elif b[i-1] == 0:
                                                   -1 -2 -1 3 0 0 0 6 3
    i = i-1
     swap(b,i-1,j)
      i = i-1; j = j-1
                                                 -1 -2 -1 0 0 0 3 6 3
  # post: b[h..i-1] < 0, b[i..j] = 0, b[j+1..k] > 0
  return (i, j)
11/18/14
                                  Sequence Algorithms
```