Resizing Array Computations

Stack: resizing-array implementation

- Q. How to grow array?
- A. If array is full, create a new array of twice the size, and copy items.

"repeated doubling"

Array accesses to insert first N = 2^i items. $N + (2 + 4 + 8 + \ldots + N) \sim 3N$

1 array access per push

k array accesses to double to size k

(ignoring cost to create new array)

Stack: resizing-array implementation

- Q. How to grow array?
- A. If array is full, create a new array of twice the size, and copy items.

"repeated doubling"

- Array accesses to insert first N = 2^i items. $N + (2 + 4 + 8 + \ldots + N) = 3N-2$

1 array access per push

k array accesses to double to size k

(ignoring cost to create new array)

- Array accesses to insert first N = 2^i items. $N + (4 + 8 + 16 + \ldots + 2N) = 5N-4$

1 array access per push

k array accesses to double to size k

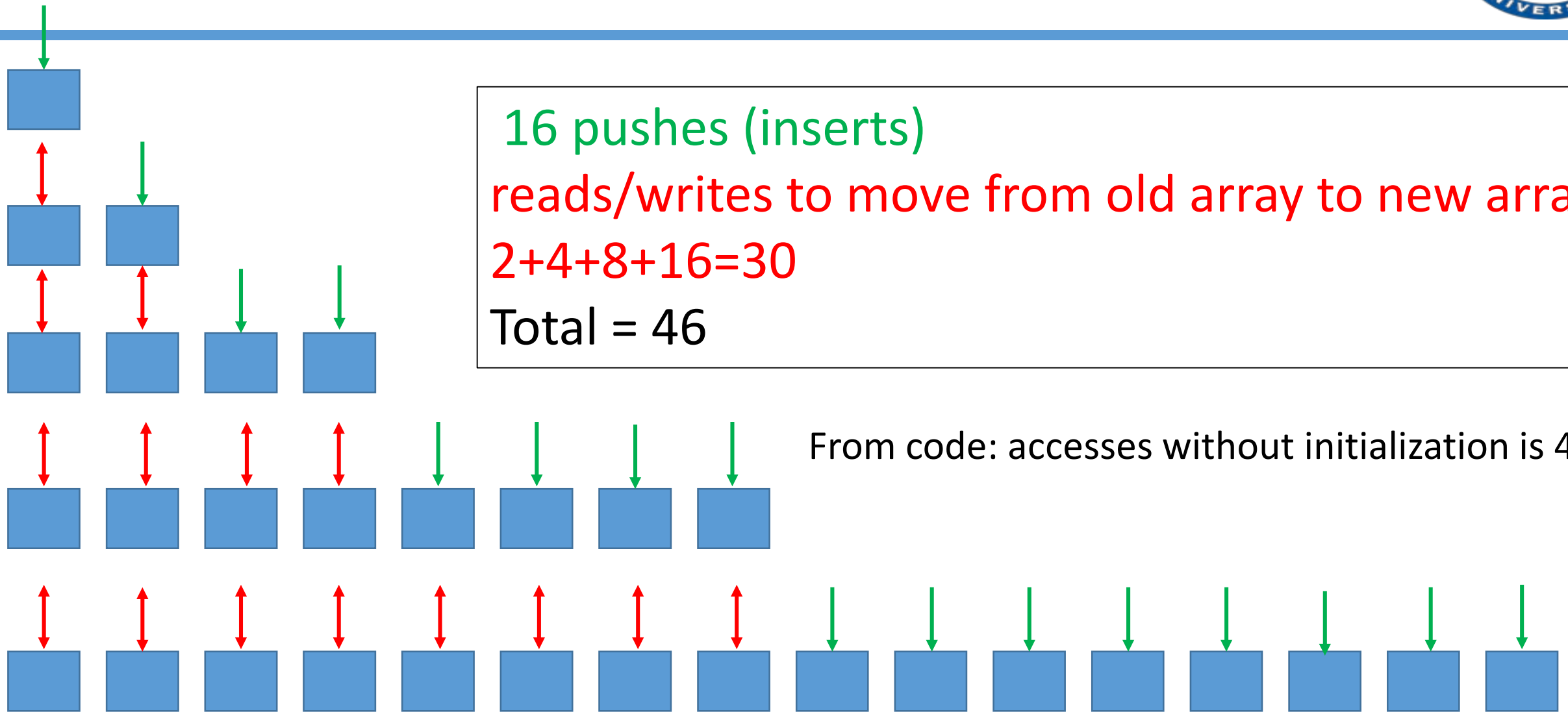
(including cost to create new array)

For these computations, N must be a power of 2

Assumptions

- What to count
  - In Resize()
    - For cost including creation of array (5N+4) - when creating the new array of size – 2 * current length
      - Add capacity to accesses
    - When moving an item from the old (small) array to the new (larger) array
      - Add two to accesses (one for the read from old and one for write to new)
  - In push()
    - Add one for each item pushed
- What not to count
  - The first creation of the array
    - Equation starts with empty array

$N + (2 + 4 + 8 + \ldots + N)$=3N-2: N = 16, 3N-2 = 46



16 pushes (inserts)

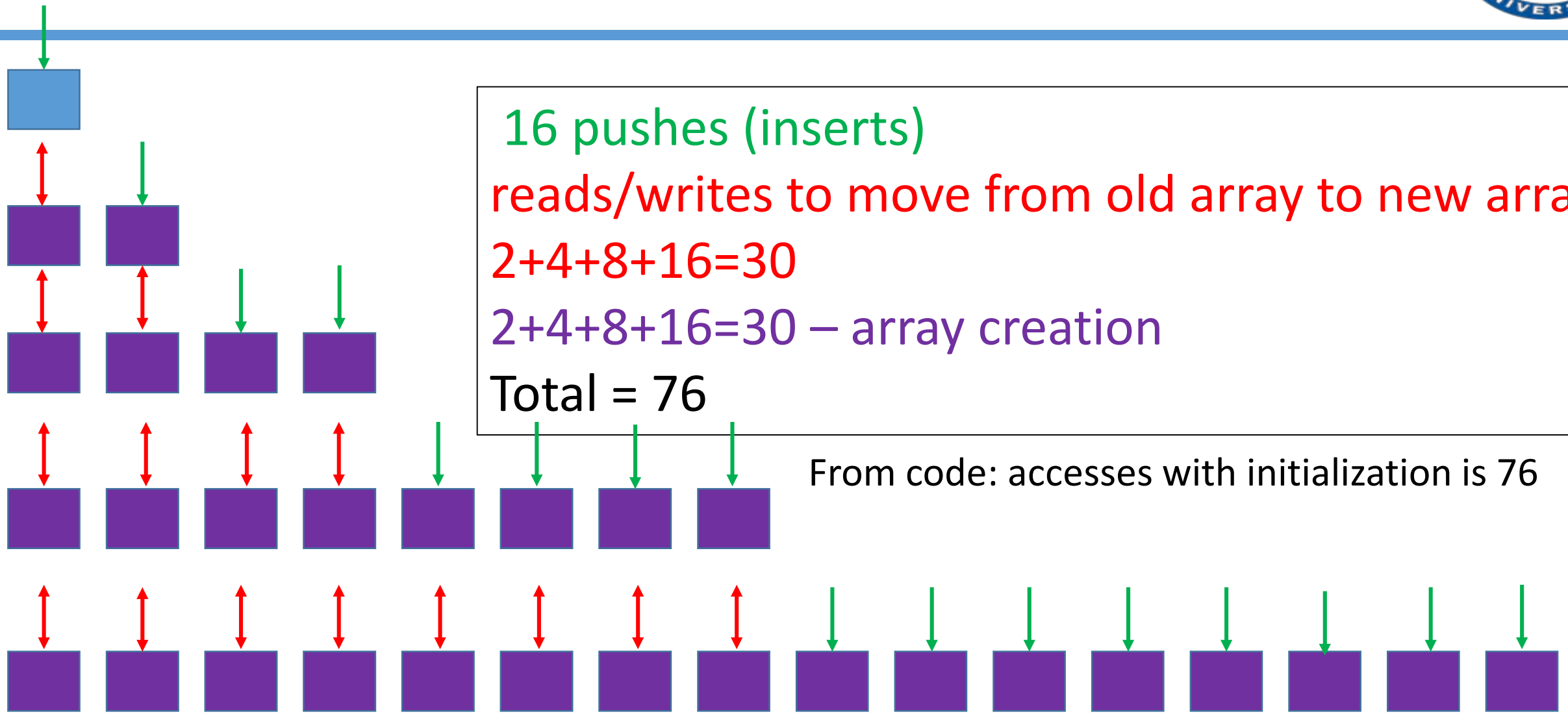
reads/writes to move from old array to new array

2+4+8+16=30

Total = 46

From code: accesses without initialization is 46

$N + (4 + 8 + 16 + \ldots + 2N) = 5N\text{-}4$: N = 16, 5N-4 = 76



16 pushes (inserts)

reads/writes to move from old array to new array

2+4+8+16=30

2+4+8+16=30 – array creation

Total = 76

From code: accesses with initialization is 76