CPSC 221: Algorithms and Data Structures Crash Course on Arrays

Alan J. Hu

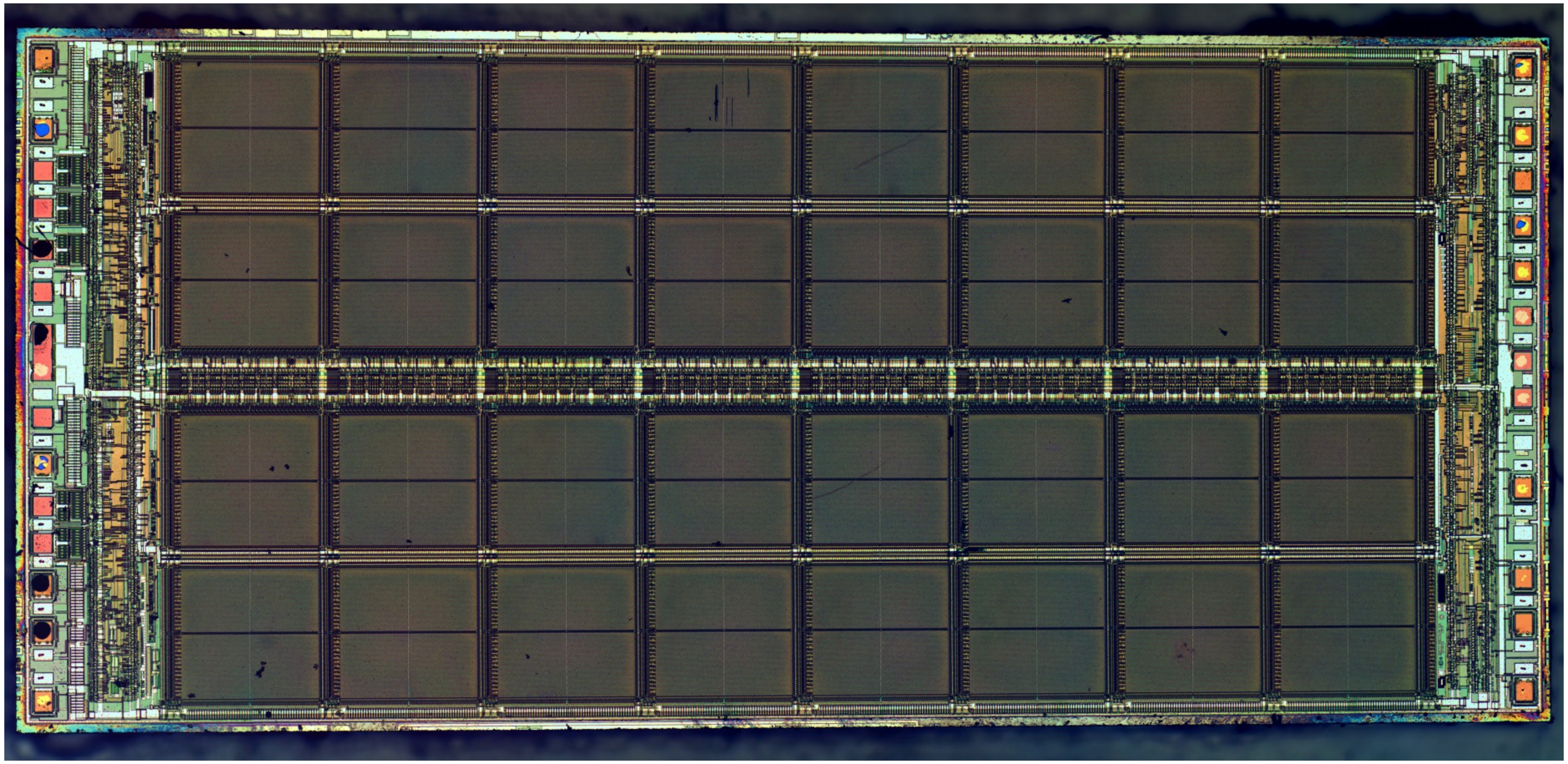
Why Arrays?

- Arrays are a very low-level data structure, that basically matches the underlying memory.
- Good: They are very efficient!
- Bad: They have unpleasant limitations.

Fact: Bits are real!

- Every bit of memory in your program is stored in an actual physical location on a silicon chip.
- These physical memory bits are organized into rectangular arrays, and you can quickly read/write any bit by giving its location as a **numerical** address.
- (Google DRAM “die photo” to see some pictures of what memory really looks like.)

Die Photo of 1Mb DRAM



Source: http://zeptobars.ru/en/read/how-to-open-microchip-asic-what-inside

Consequences of Bits Being Real

- If you know the address where your data is, you can quickly access its memory.
- If you don’t know the address, you can’t find the data easily.
- You must work to move data. You can’t just “squeeze in” some more bits between data you’ve already stored.

Arrays in C++

- These are almost the same as arrays in Java.
- Declare an array, e.g.:

  ```
  int x[10];
  ```

  creates an array of 10 ints: `x[0], x[1], …, x[9]`
- Access array elements just like any variable:

  ```
  x[0] = x[1]+x[2];
  for (int i=0; i<10; i++) x[i] = 0;
  ```
- Lots more info in book, online, etc., e.g.,

  http://www.cplusplus.com/doc/tutorial/arrays/

Arrays vs. Java's ArrayList

- **Arrays have a fixed size.** They cannot grow or shrink.
- You can't insert things or delete things from the middle of an array.
- Java provides an ArrayList class that does let you do those things. That makes programming easier.
- (But Java ArrayLists are doing things behind the scenes to make things nicer for you to program…)

Do-It-Yourself ArrayLists

- ArrayLists are nothing magical!
  - (OK, the generic <type> stuff is kind of magic.)
- It’s just a class. If we fix the type of the elements (e.g., have an ArrayList of String), you know enough to write your own version.
- But how do you allow arrays to grow?

Real-Life Analogy: Moving Homes

- A house (or condo, apartment, etc.) has a fixed size. What happens when your family grows and you need more space?

Real-Life Analogy: Moving Homes

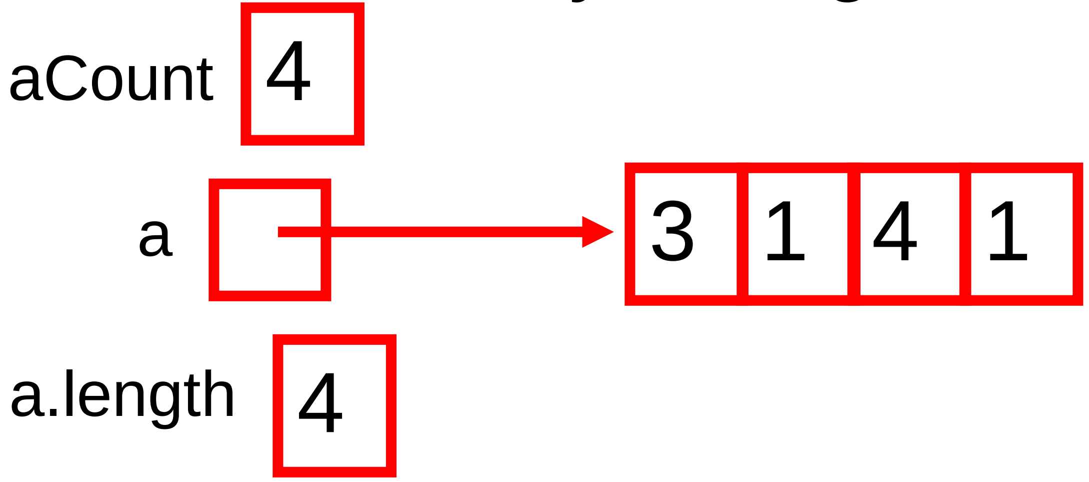
- A house (or condo, apartment, etc.) has a fixed size. What happens when your family grows and you need more space?
- Answer: You buy a bigger place, and then you pack up and move all your stuff to the new place, and get rid of your old home.

Making Your Own ArrayList

- An array has a fixed size. What happens when your list grows and you need more space?
- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.

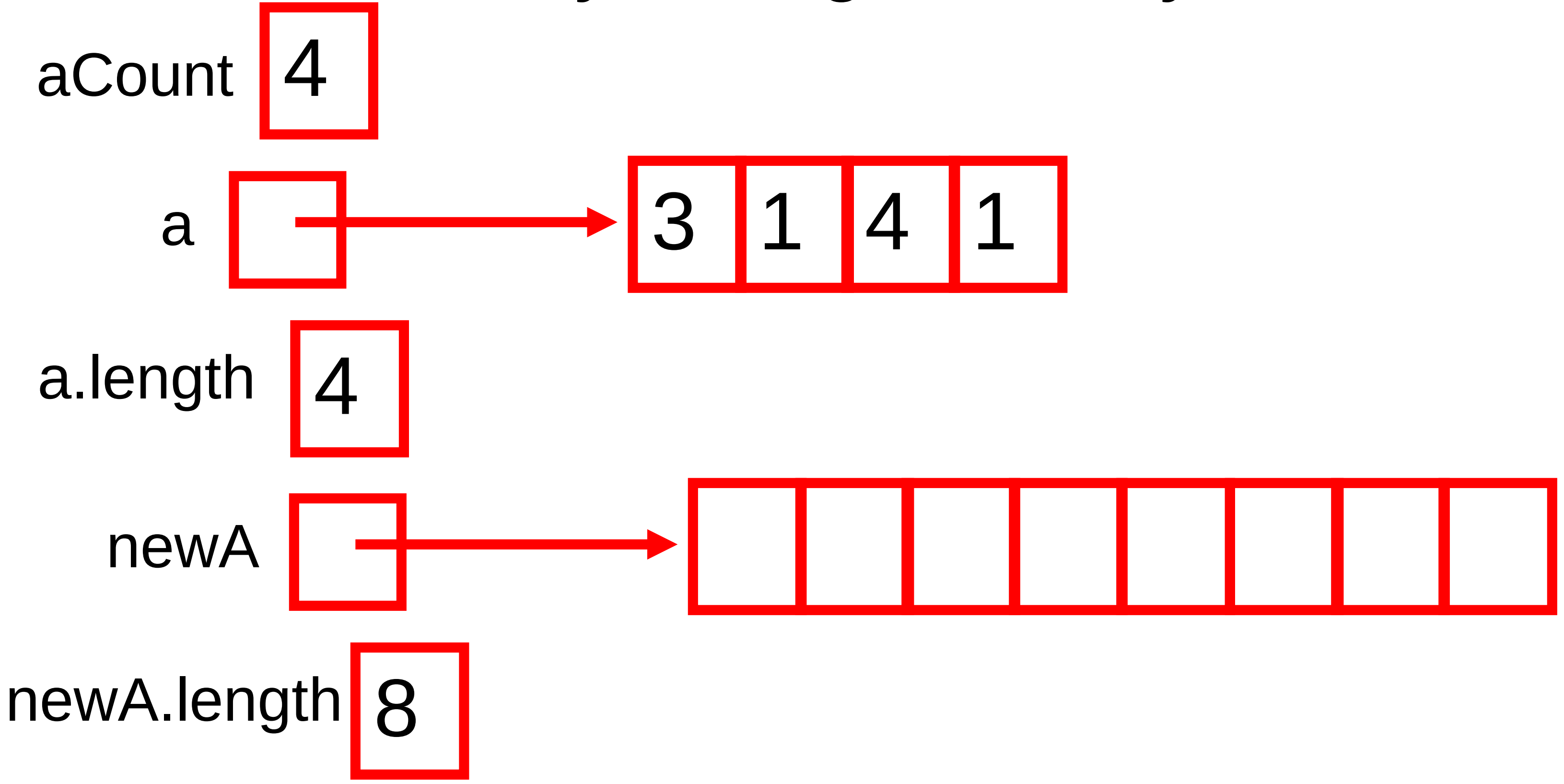
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



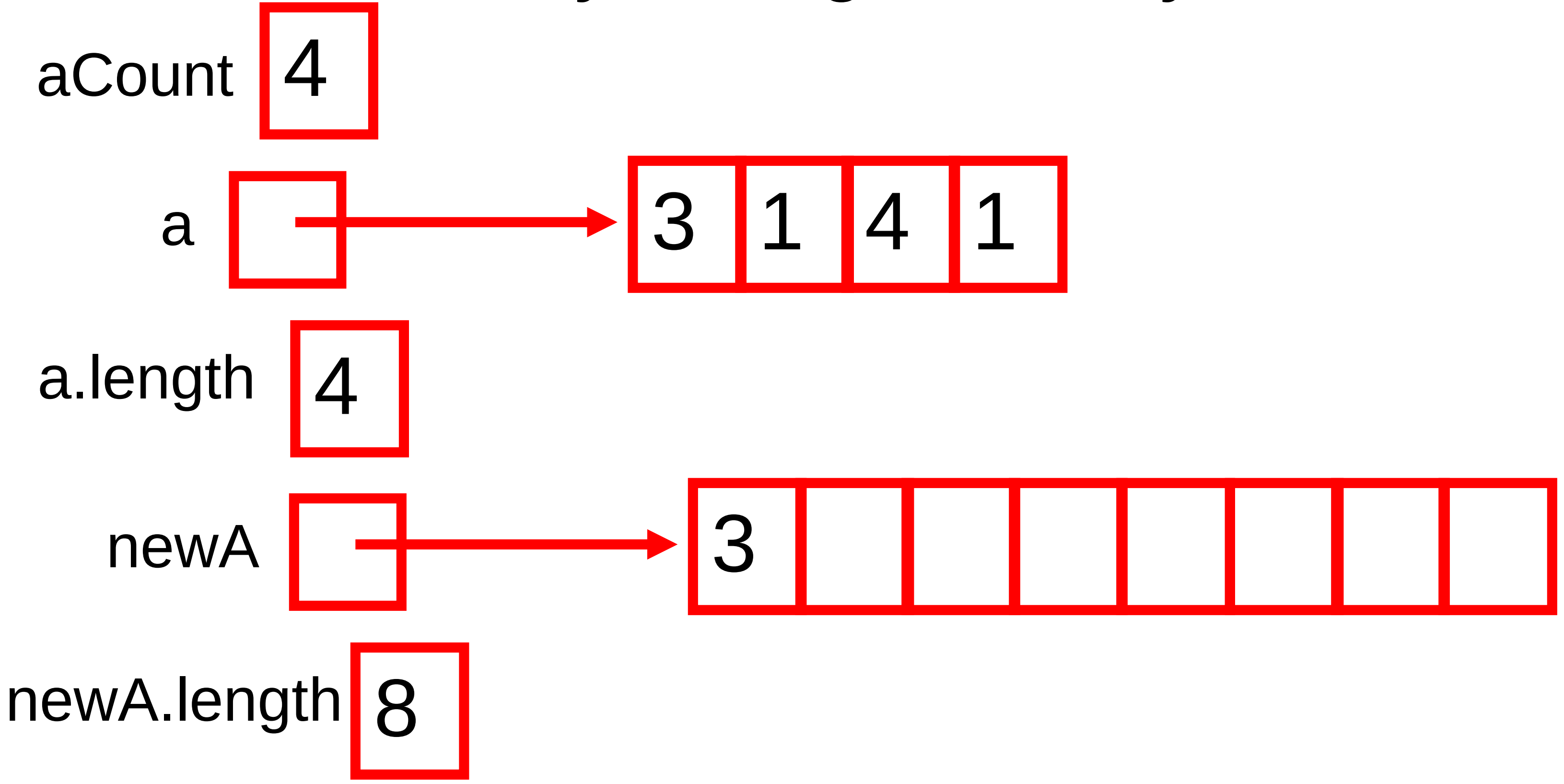
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



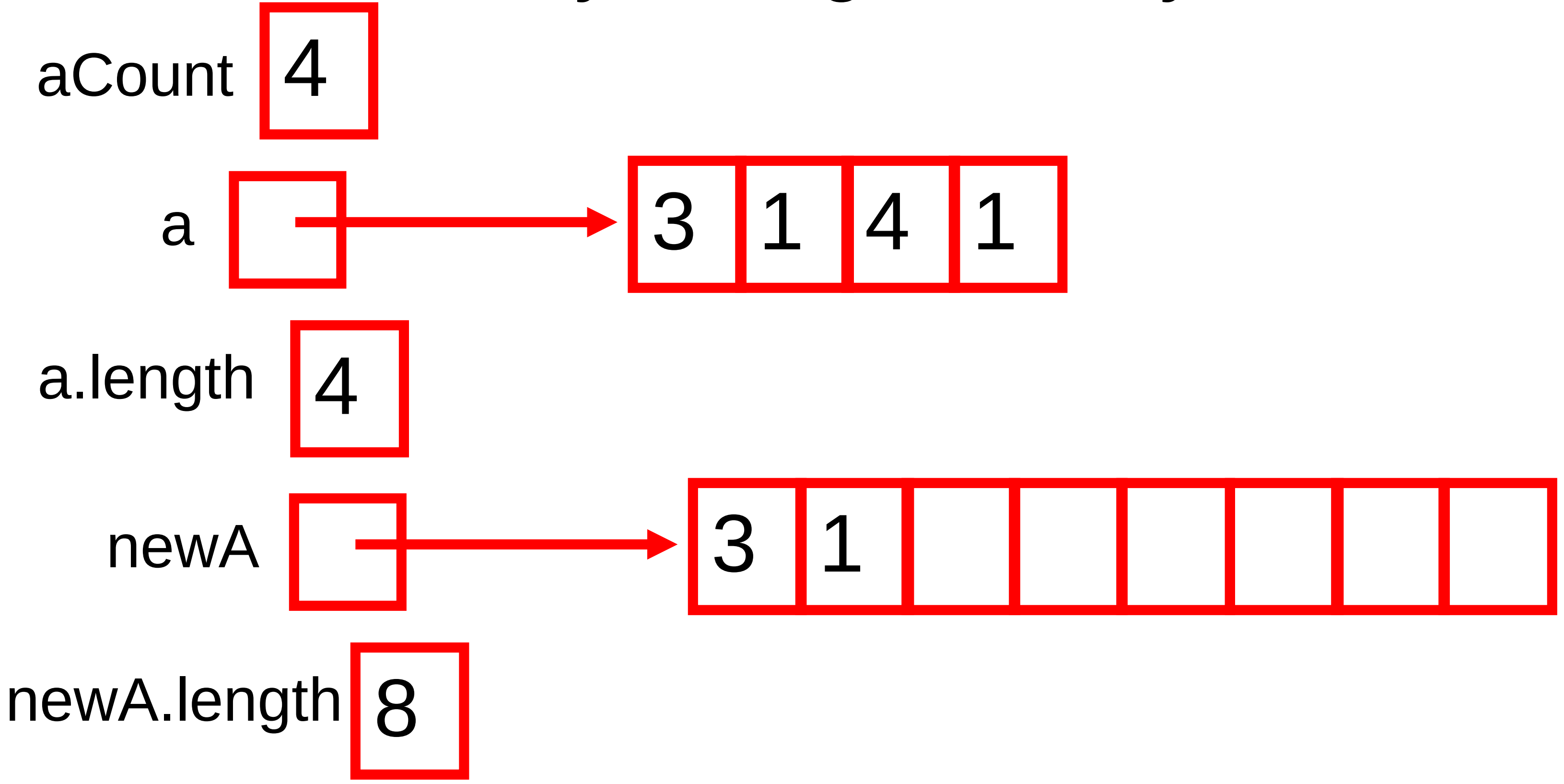
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



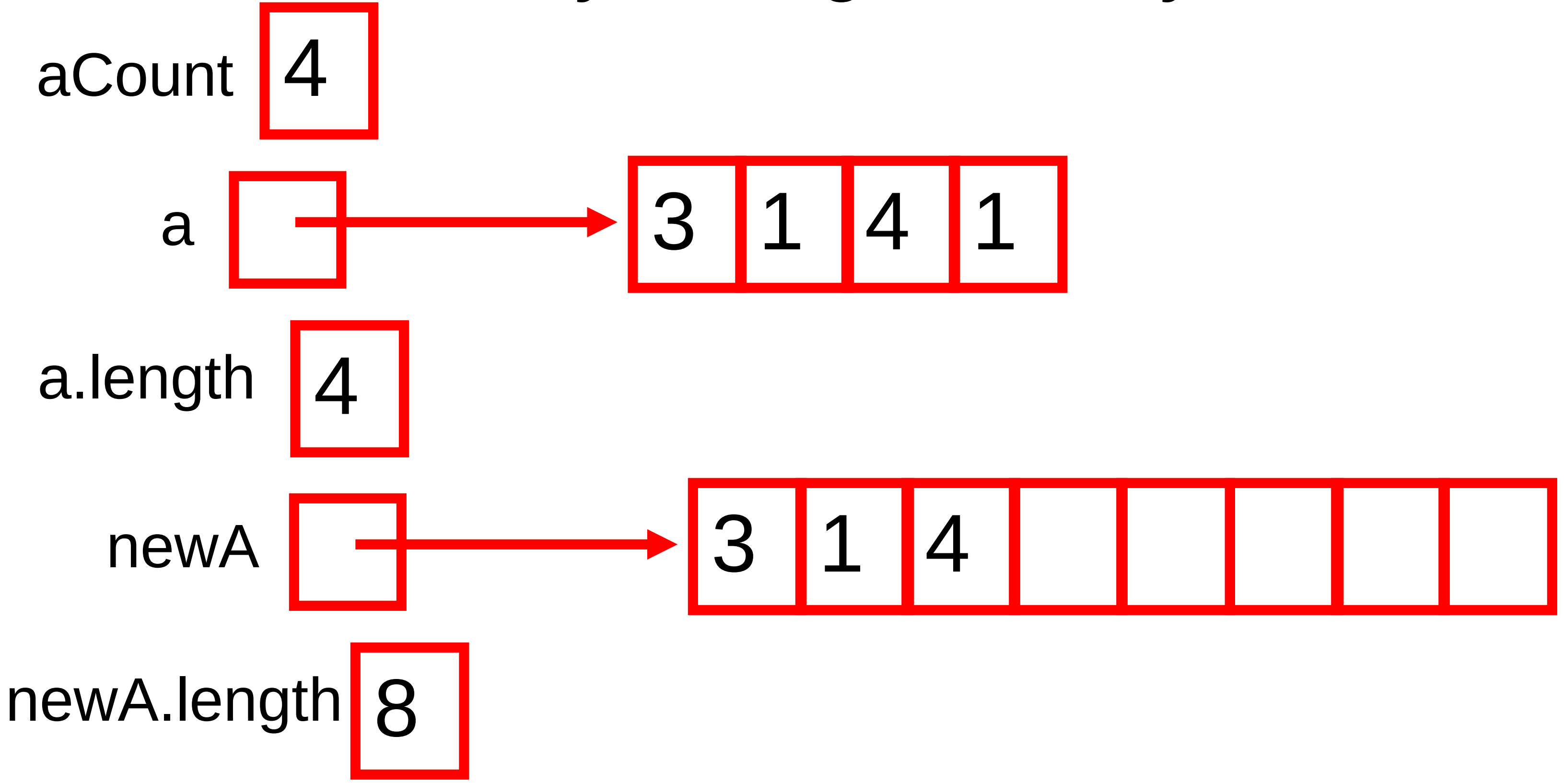
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



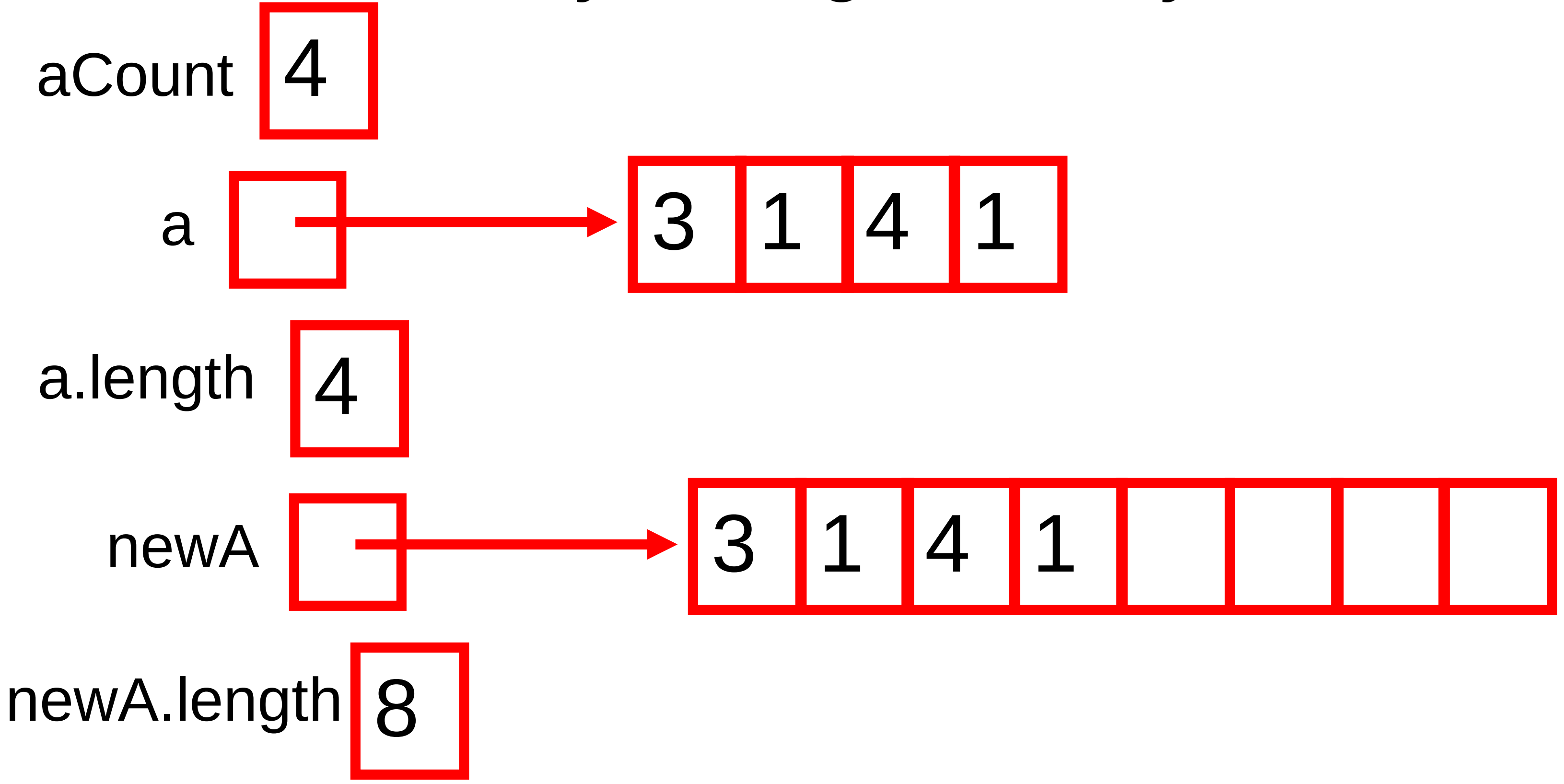
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



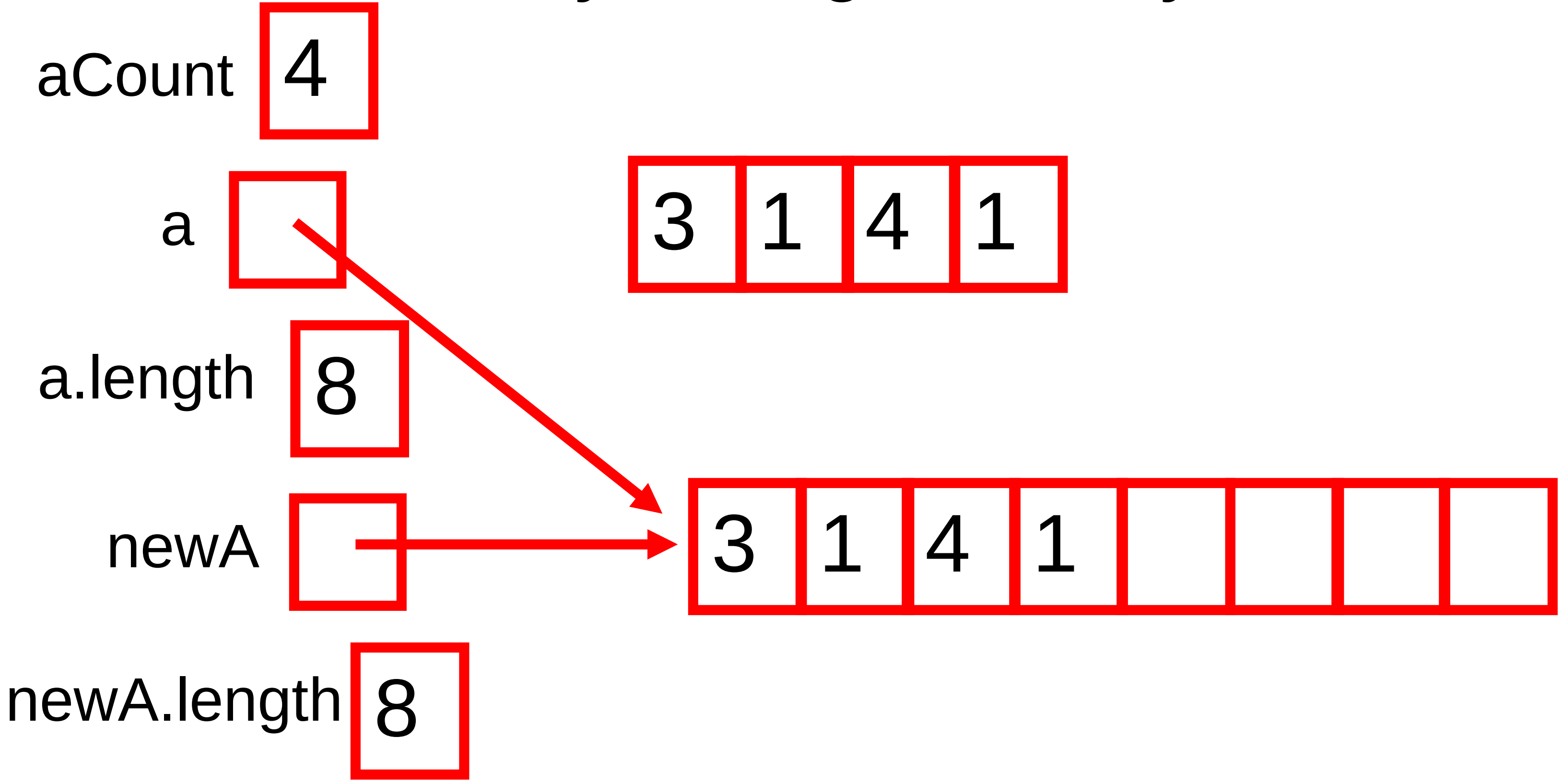
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



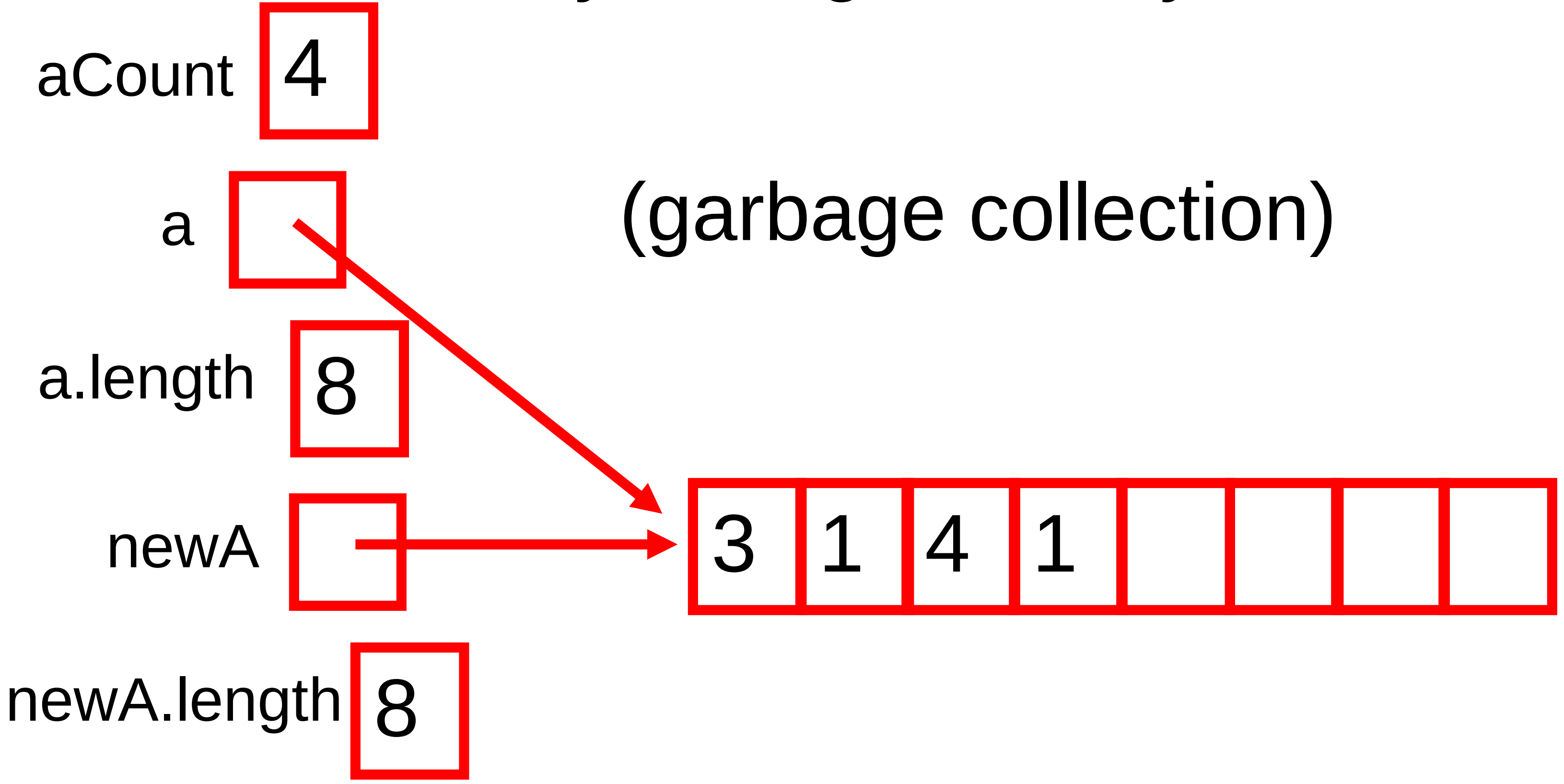
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



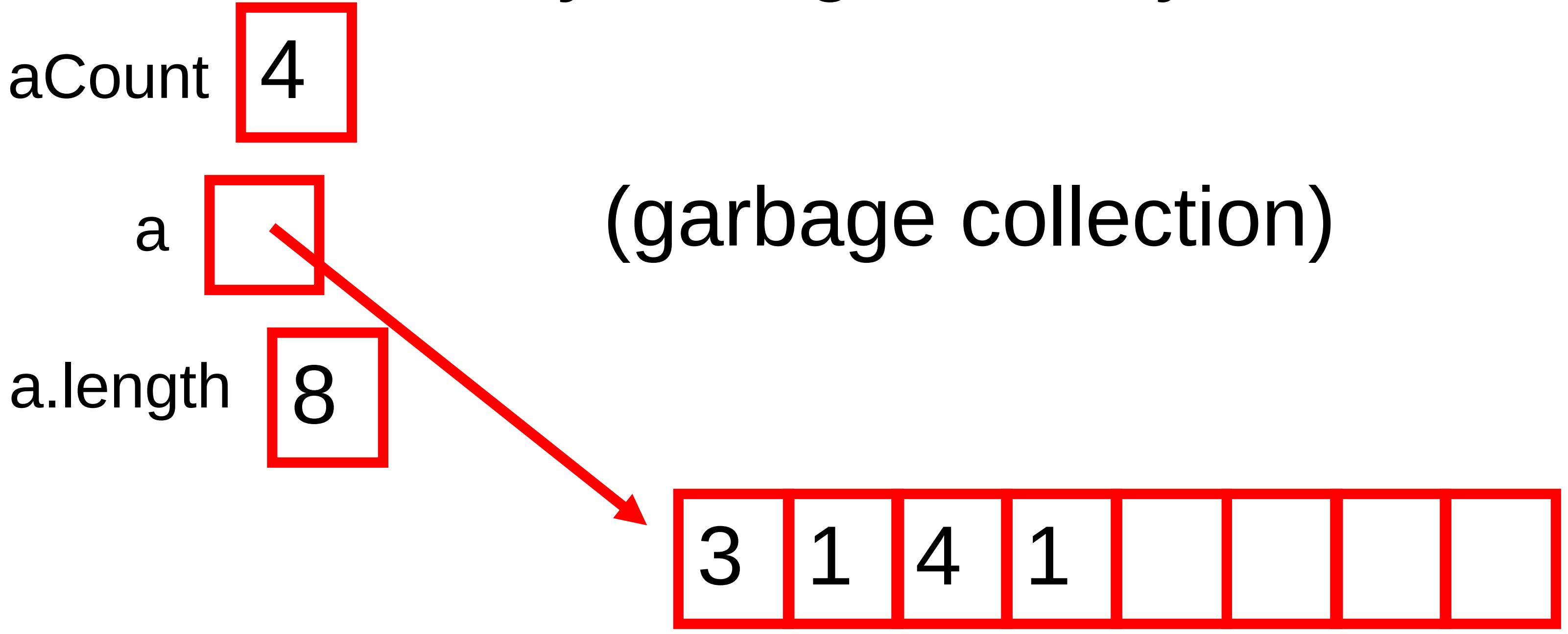
Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



Making Your Own ArrayList

- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.



(garbage collection)