CS240: Data Structures And Algorithms

Nathan Sprague

Spring 2024

LinkedList or ArrayList?

```
import java.util.ArrayList;
import java.util.LinkedList;

public class ListDemo {

  public static void main(String[] args) {

    LinkedList<String> list = new LinkedList<String>();

    for (int i = 0; i < 500000; i++) {
      list.add(0, "A");
    }

    for (int i = 0; i < 500000; i++) {
      list.remove(0);
    }

  }
}
```

- What is the result of changing the code snippet to use an ArrayList instead of a LinkedList?

```
for (int i = 0; i < 500000; i++) {
     list.add(0, "A");
}
for (int i = 0; i < 500000; i++) {
     list.remove(0);
}
```

A) No change in correctness or execution time. No one notices you made the change.

B) An ArrayList will not work correctly in this case. You've broken the code! You're Fired!

C) The new implementation is slightly slower than the original implementation.

D) The new implementation is slightly faster than the original implementation.

E) The new implementation is MUCH faster than the original version. Nice Job! Promoted on your first day!

F) The new implementation is MUCH slower than the original version. You're fired!

Abstract Data Types and Data Structures

- “An abstract data type (ADT) is the specification of a data type within some language, independent of an implementation. The interface for the ADT is defined in terms of a type and a set of operations on that type. An ADT does not specify how the data type is implemented. These implementation details are hidden from the user of the ADT and protected from outside access.”
- “A data structure is the implementation for an ADT. In an object-oriented language, an ADT and its implementation together make up a class.”

(OpenDSA 1.2.1)

Java Collections

- Naming convention for Java Collection types: ArrayList
  - Array – Coded using Arrays "under the hood"
  - List – Implements the List Interface, which represents the list abstract data type.

    An ordered collection (also known as a sequence). The user of this interface has precise control over where in the list each element is inserted. The user can access elements by their integer index (position in the list), and search for elements in the list

List, ArrayList, LinkedList

- Java interfaces can be thought of as ADT's*:
  - List, Map, Set, Queue, Deque
- Java collection classes are data structures:
  - ArrayList
  - LinkedList

*Some authors would argue that an ADT is an abstract mathematical description that is not tied to any particular language construct.

Two Steps Back: CS261

- X86 Assembly language example :

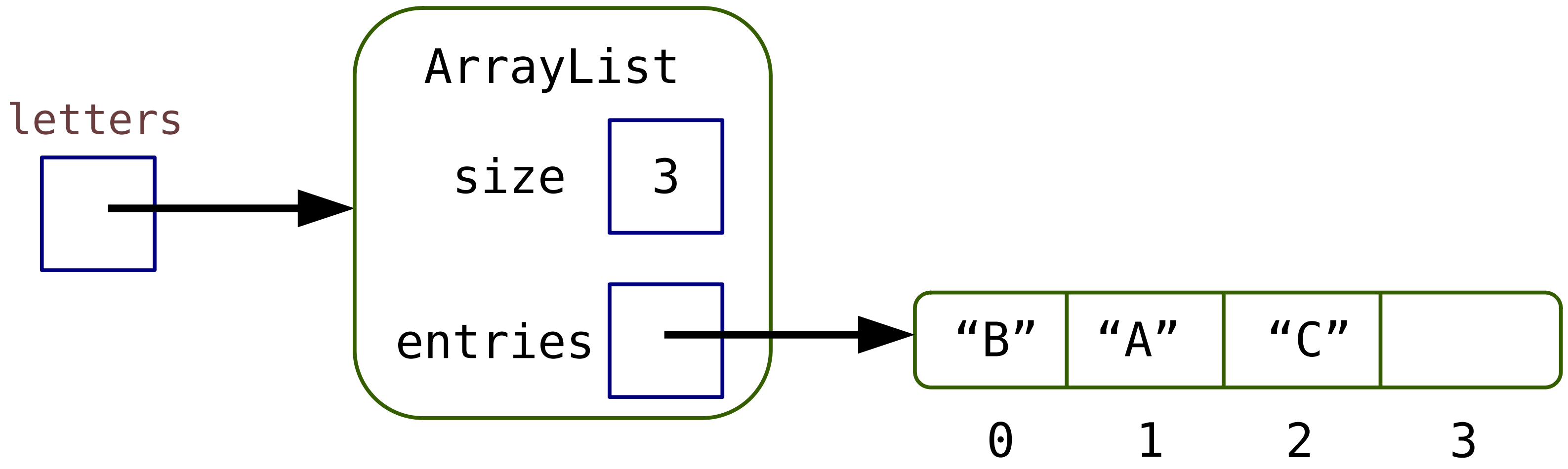
```
mov (%ebx), %eax
add %eax, %eax
mov %eax, (%ebx)
```

- The structure of memory...

- Contiguous vs. Linked Structures...

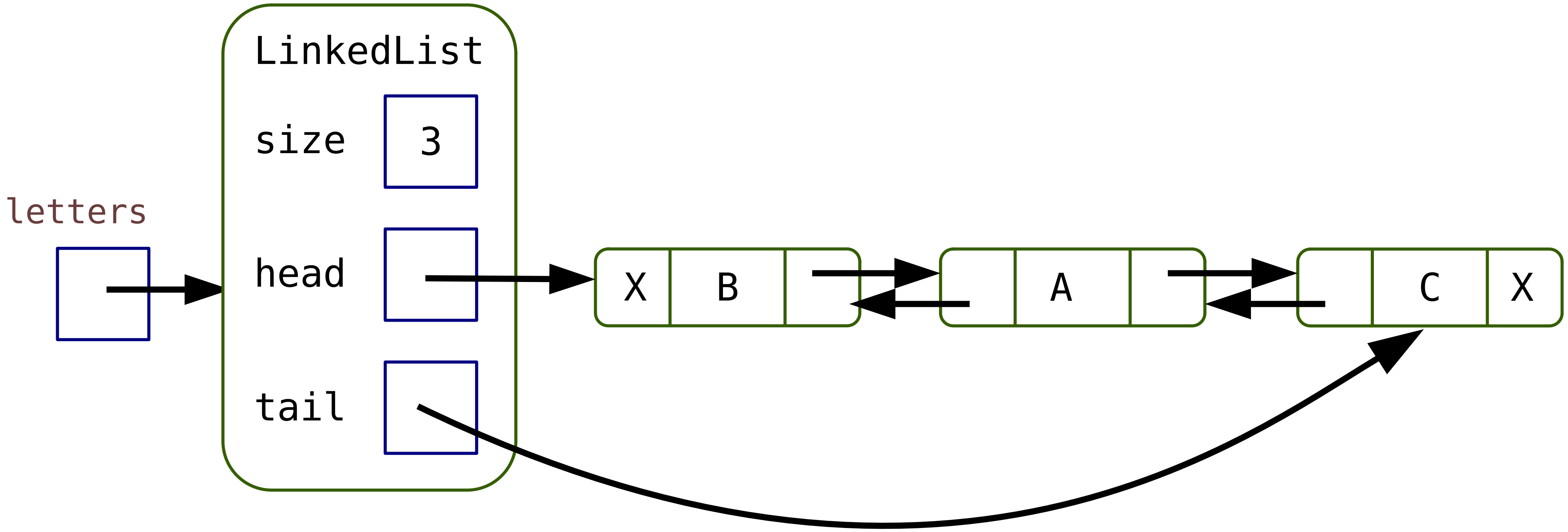
ArrayList…

```
ArrayList<String> letters = new ArrayList<>();
letters.add("A");
letters.add(0, "B");
letters.add("C");
```



Doubly-Linked Lists

```java
LinkedList<String> letters = new LinkedList<>();
letters.add("A");
letters.add(0, "B");
letters.add("C");
```



- (Java's LinkedList class is “doubly-linked”)

Key Goals For This Course

- Selecting appropriate ADT's
  - Based on the logical requirements of the problem: How will we need to interact with our data

Key Goals For This Course

- Selecting appropriate ADT's
  - Based on the logical requirements of the problem: How will we need to interact with our data
- Selecting appropriate Data Structures
  - How can we organize our data so that we can interact with it efficiently in both time and space.

Key Goals For This Course

- Selecting appropriate ADT's
  - Based on the logical requirements of the problem: How will we need to interact with our data
- Selecting appropriate Data Structures
  - How can we organize our data so that we can interact with it efficiently in both time and space.
- Algorithm Analysis
  - How can we determine the performance characteristics of our code before we write it

Back to Linked List vs. ArrayList…

- LinkedList CRUSHED ArrayList in our earlier code snippet. Which, if any, of these will be much *faster for an ArrayList* (assuming a large number of entries)?

```
A) list.get(0);
B) list.get(list.size() - 1);
C) list.get(list.size() / 2);
D) None of the above.
```

For Friday

- Do the posted reading and quizzes for Friday (and today)
- Sign into Canvas and log into Piazza
- Complete the course survey + 1Pic1Par
- Read the syllabus carefully
- If you plan to bring a laptop, make sure it has Java 17 and a working Java IDE.