ArrayList

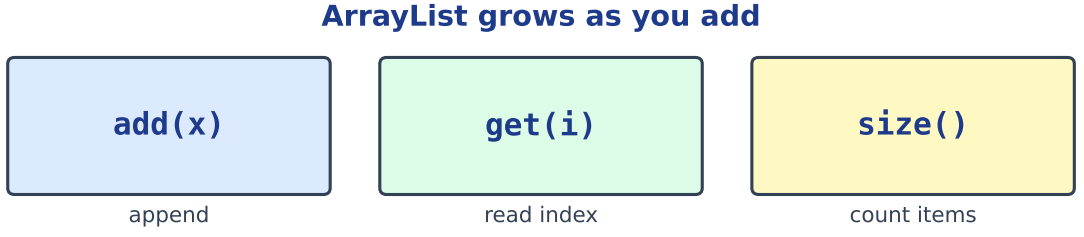
Java Reference

ArrayList basics

An `ArrayList` is a resizable 可变大小 list — it grows and shrinks as you add or remove items. It stores objects, so use a wrapper 包装类 type like `Integer` (not `int`). The `<Integer>` part is a generic 泛型 type. Key methods: `.add(x)`, `.get(i)`, `.set(i, x)`, `.size()`, `.remove(i)`.

```
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        ArrayList<Integer> nums = new ArrayList<Integer>();
        nums.add(10);
        nums.add(20);
        nums.add(30);
        System.out.println(nums.size());   // 3
        System.out.println(nums.get(1));   // 20
        nums.set(0, 99);
        System.out.println(nums);          // [99, 20, 30]
    }
}
```



Exam tip: ArrayList<Type> needs a type; indices start at 0 like arrays

ArrayList: add, get, size on a resizable list

ArrayList algorithms & the remove bug

`.remove(i)` shifts 移动 every later element one place left. If you remove while counting `i` **up**, you skip the next element. Fix: loop **backwards**, or don't increment `i` after a remove.

```
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        ArrayList<Integer> nums = new ArrayList<Integer>();
        for (int n : new int[]{4, 7, 4, 9, 4}) nums.add(n);
        // Remove every 4 - loop backwards so removals don't skip items.
        for (int i = nums.size() - 1; i >= 0; i--) {
            if (nums.get(i) == 4) nums.remove(i);
        }
        System.out.println(nums);   // [7, 9]
    }
}
```

Common mistakes

- `ArrayList` uses `.size()`, `.get(i)` and `.add(...)` — not the `[]` you use on arrays.
- Removing items while looping forward by index skips the next item (the remove bug). Loop backwards, or use an iterator.
- Store objects, not primitives: use `ArrayList<Integer>`, and Java auto-boxes `int` values.