

1 Runtime Analysis

Find out the Big-O runtime and Tight-O runtime of the following methods:

- ```
1. int a = 0, b = 0;
 for(i = 0; i < N; i++) {
 a = a + rand();
 }

 for(j = 0; j < M; j++) {
 b = b + rand();
 }
```
- ```
2. int a = 0;
   for(i = 0; i < N; i++) {
       for(j = N; j > i; j--) {
           a = a + i + j;
       }
   }
```
- ```
3. int i, j, k = 0;
 for(i = N/2; i < N; i++) {
 for(j = 2; j <= N; j = j*2) {
 k = k + N/2;
 }
 }
```
- ```
4. int a = 0, i = N;
   while(i > 0) {
       a += i;
       i /= 2;
   }
```

2 Review

Toby wants to rule the world with an army of cats. Each cat may or may not have one parent, and may or may not have 'kitties'. Each cat that has a parent is a 'kitty' of that parent. But after implementing `copyCat`, which creates a copy of a cat and its descendants, he realizes the function contains a bug.

```
public class Cat {
    private Cat parent;
    private ArrayList<Cat> kitties;
    private String name;

    public Cat(Cat parent, String name) {
        this.name = name;
        this.kitties = new ArrayList<Cat>();
        this.parent = parent;
    }

    public Cat copyCat() {
        Cat copy = new Cat(this.parent, this.name);
        for (int i = 0; i < this.kitties.size(); i += 1) {
            copy.kitties.add(this.kitties.get(i).copyCat());
        }
        return copy;
    }
}
```

What's wrong with his `Cat` class? Drawing a box and pointer diagram may help!