Method Complexity

"Divide and Conquer" is an oft touted strategy...
Split complex code up into simple sub-routines
(and sub-sub-routines)

Avoid massive, hard-to-understand methods
Particularly with complex loop & decision structures
These are very hard to understand (and to change)

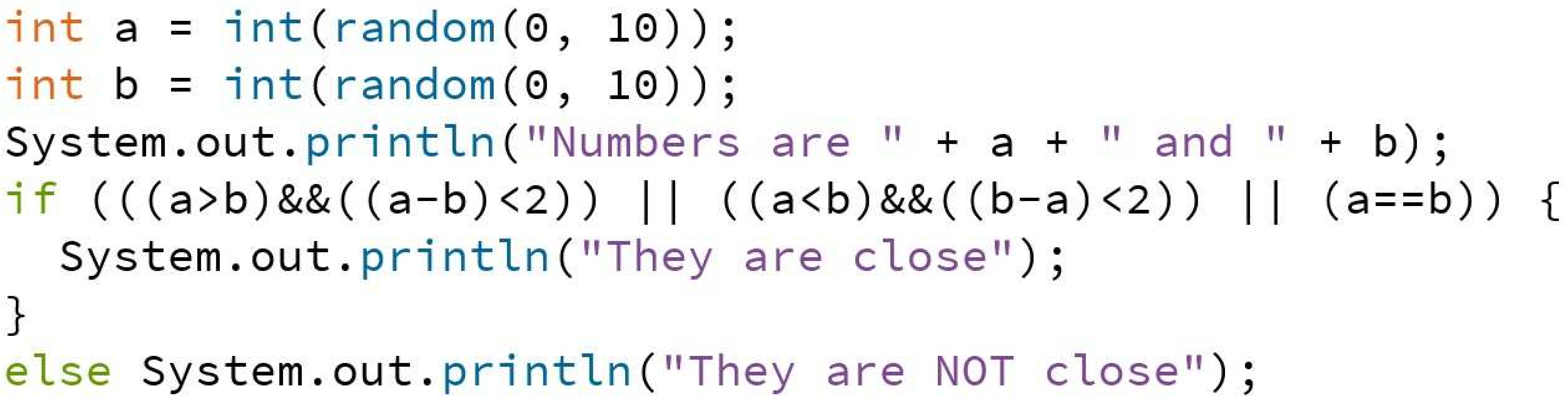
Big improvements in understandability can be
achieved by "farming out" code to suitable functions

"Farming Out" Example

Consider a function to check if two numbers are "close"

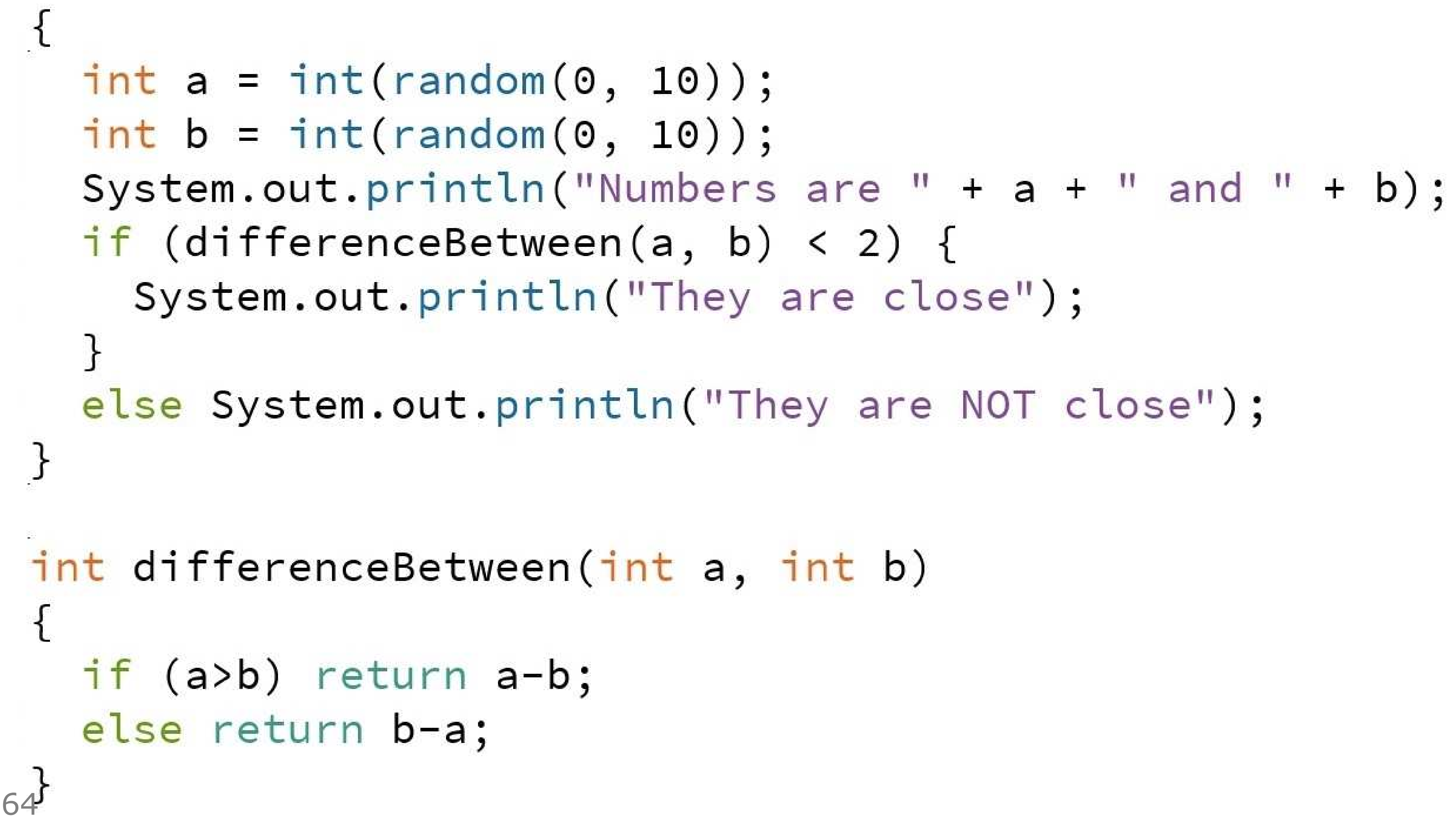
(e.g. 1 and 2 are close, 1 and 8 are not)

A first attempt might look something like this:



```
int a = int(random(0, 10));
int b = int(random(0, 10));
System.out.println("Numbers are " + a + " and " + b);
if (((a>b)&&((a-b)<2)) || ((a<b)&&((b-a)<2)) || (a==b)) {
  System.out.println("They are close");
}
else System.out.println("They are NOT close");
```

A better solution ?



```
{
  int a = int(random(0, 10));
  int b = int(random(0, 10));
  System.out.println("Numbers are " + a + " and " + b);
  if (differenceBetween(a, b) < 2) {
    System.out.println("They are close");
  }
  else System.out.println("They are NOT close");
}

int differenceBetween(int a, int b)
{
  if (a>b) return a-b;
  else return b-a;
}
```

Measuring Method Complexity

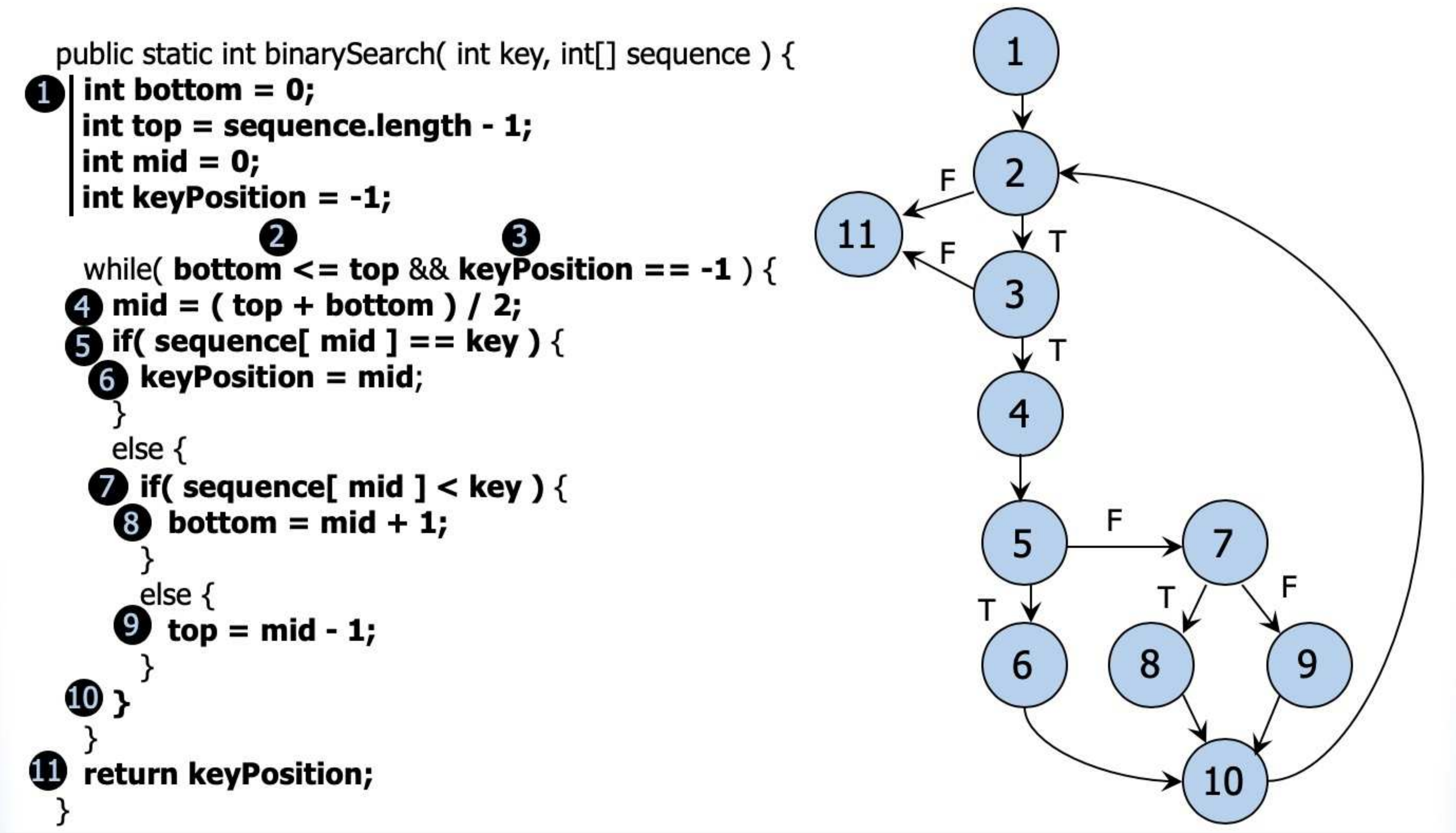
To control something, we need to measure it

(Otherwise we don't know if it is good or bad !)

Various different approaches are possible:

- Measuring the length of methods (in lines)
- Counting the number of parameters passed in
- Depth of nesting (number of levels of indentation)
- "Cyclomatic Complexity" is a popular measure...

Cyclomatic Complexity: Flow Graph



```
public static int binarySearch( int key, int[] sequence ) {
❶ int bottom = 0;
  int top = sequence.length - 1;
  int mid = 0;
  int keyPosition = -1;

            ❷                 ❸
  while( bottom <= top && keyPosition == -1 ) {
 ❹ mid = ( top + bottom ) / 2;
 ❺ if( sequence[ mid ] == key ) {
  ❻ keyPosition = mid;
    }
    else {
  ❼ if( sequence[ mid ] < key ) {
   ❽ bottom = mid + 1;
      }
      else {
   ❾ top = mid - 1;
      }
 ❿ }
  }
⓫ return keyPosition;
}
```

Calculating Cyclomatic Complexity

CC can be calculated from the flow graph:

$$CC = (E - N) + 2$$

Where E is the number of edges
And N the number of nodes

Don't worry about too much about this !
You won't be asked to calculate CC
It's just presented here for illustration

The main message: CC is a measure of complexity !