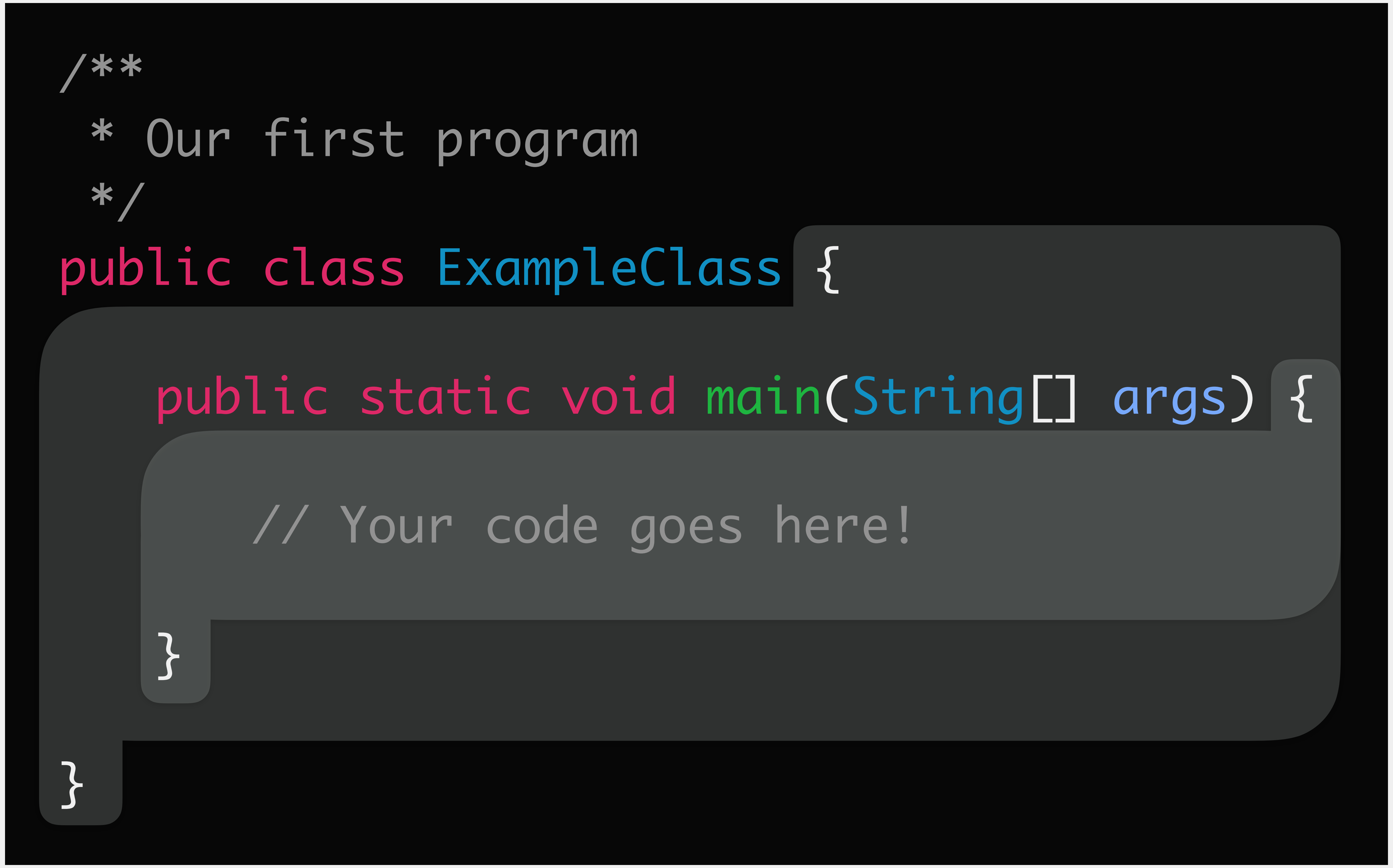
Loops

Program Structure: Code Blocks



```
/**
 * Our first program
 */
public class ExampleClass {

    public static void main(String[] args) {

        // Your code goes here!

    }

}
```

Defined by opening and closing curly bracket (e.g., { & })

Can be nested

- innermost opening curly bracket matches innermost closing curly bracket
- can nest conditionals, loops

Scope and Variables

scope defines the region of code where a variable can be used

Scope is defined by code blocks

- variables declared inside a code block are *local* to that block
- variables anywhere inside that block (even nested blocks!)
- variables **cannot** be used outside that block

Control Flow in Programs

Last week, how to make decisions about whether or not to execute code

This week, how to make decisions about whether to execute code again

Example

- previously saw how we could use conditionals to calculate a single person's age
- *loops* will allow us to repeat that same code for multiple people

Why Loops?

Often want to repeat code zero or more times

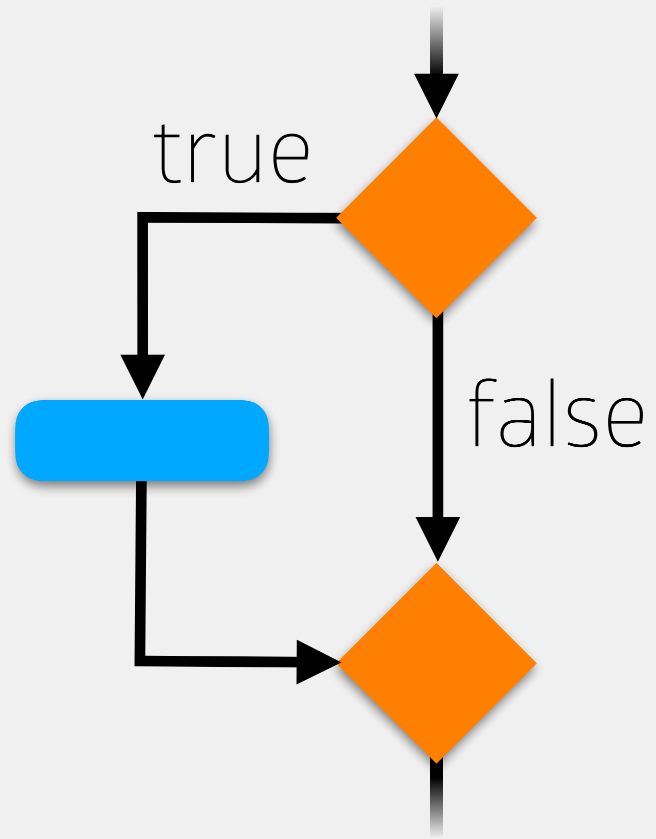
Two options

- copy and paste code multiple times
- use a loop

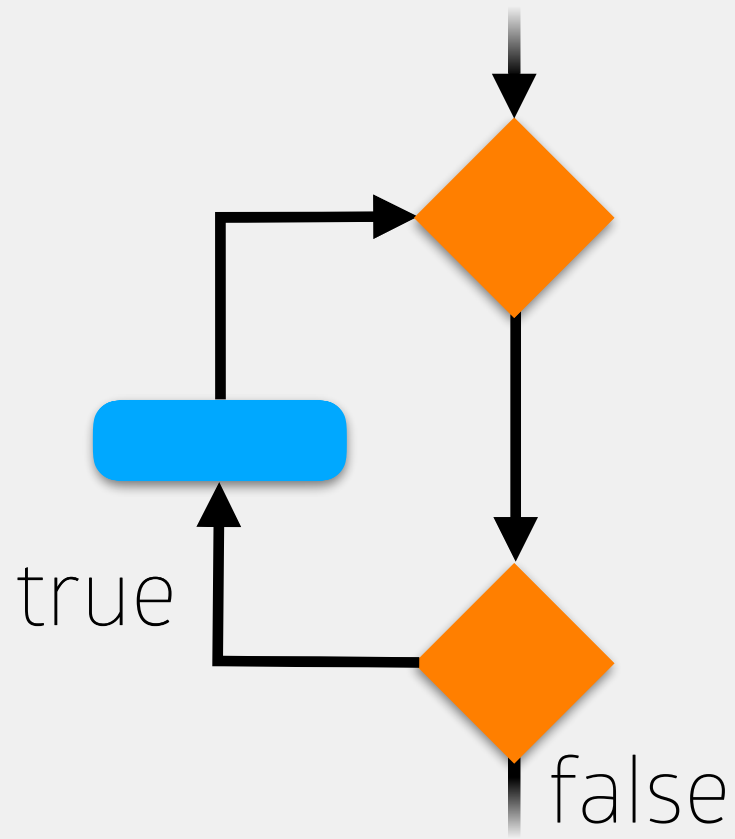
What problems arise with the first option?

Loops

Loops allow us to repeat one or more statements while some boolean condition is true, and stop when the condition is false

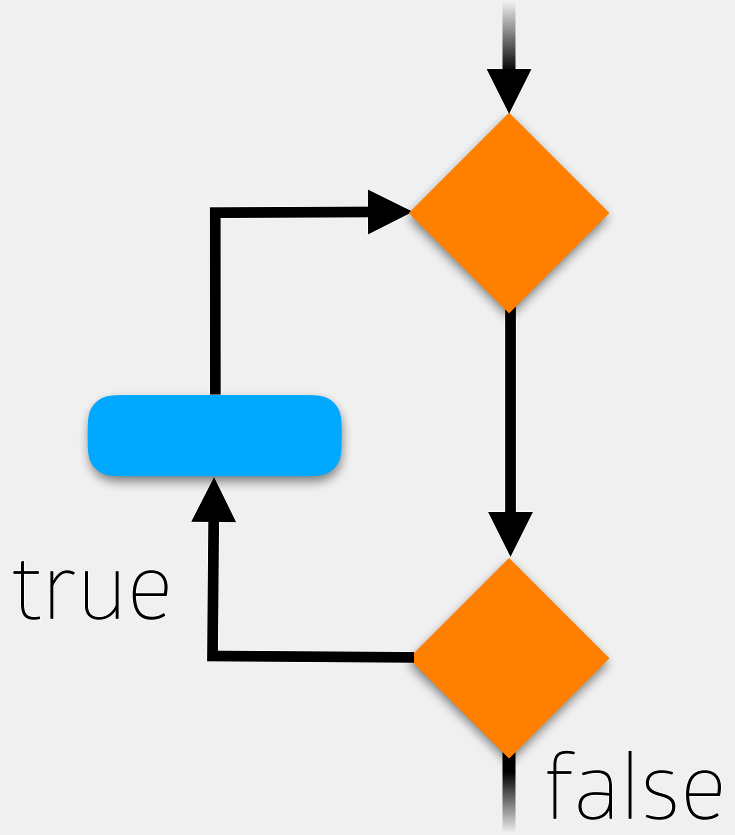


conditional statements

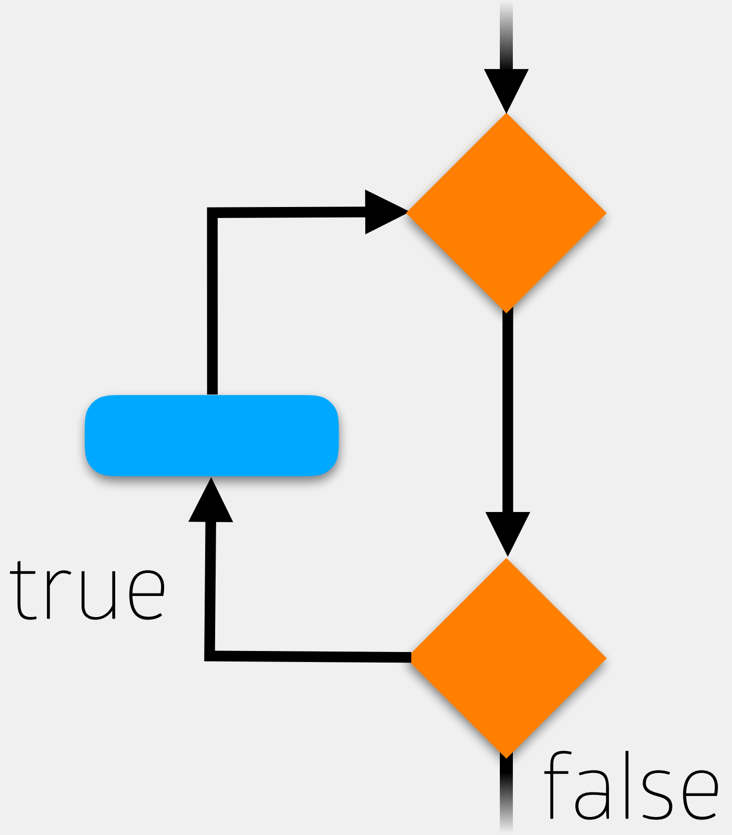


loops

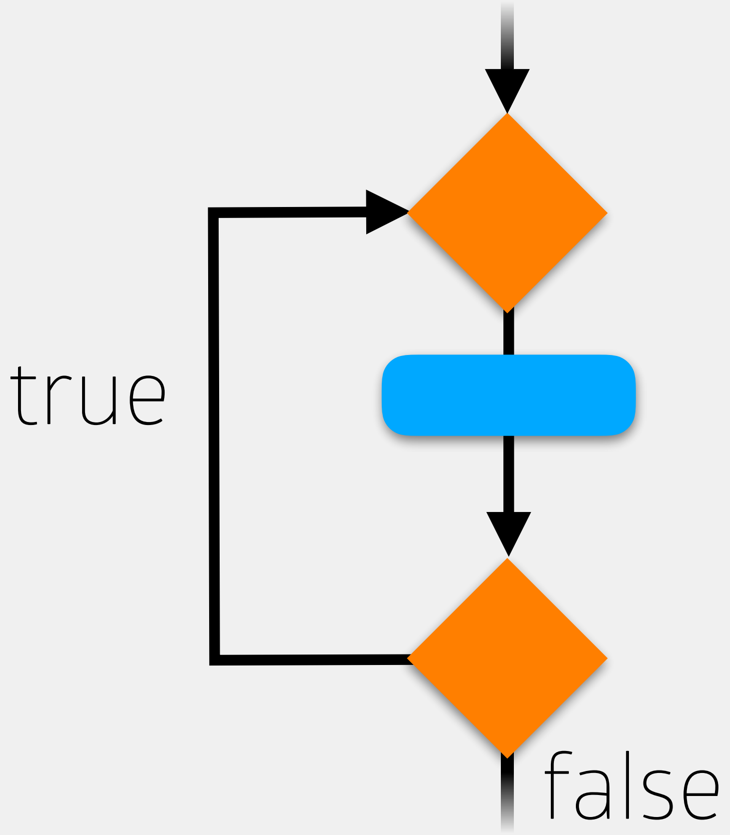
Types of Loops



while
loops

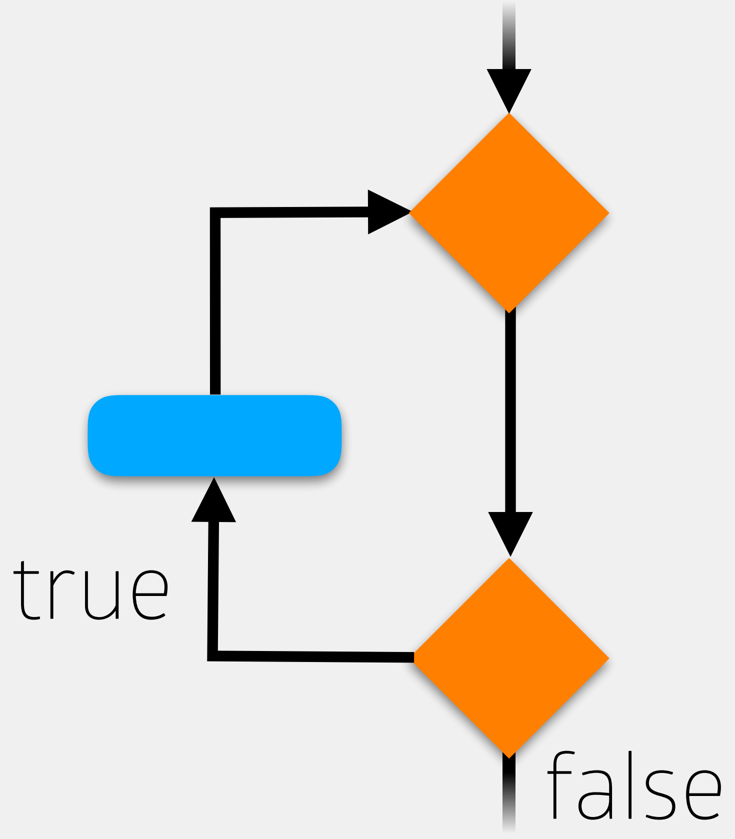


for
loops

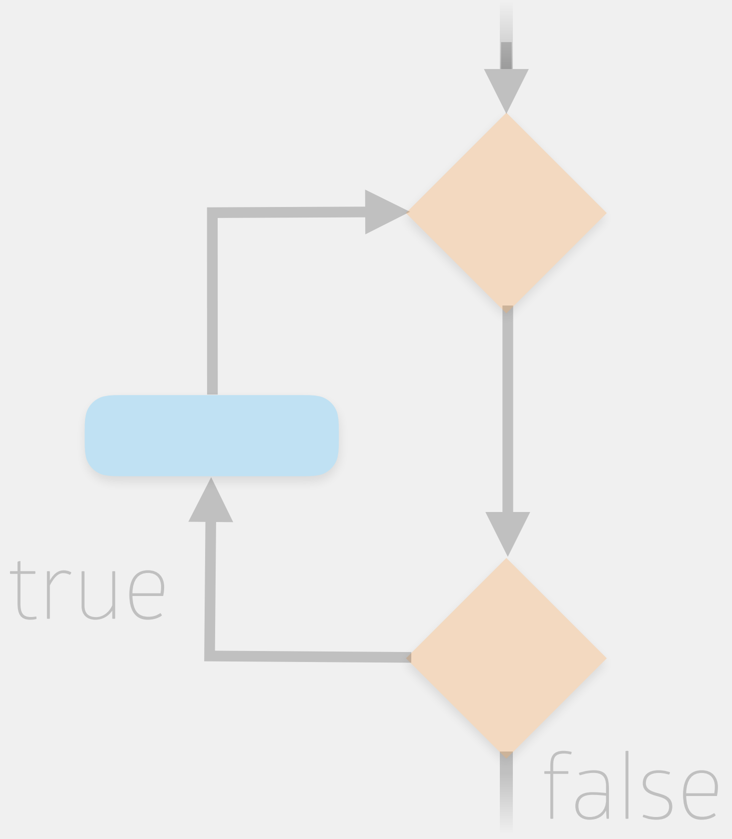


do-while
loops

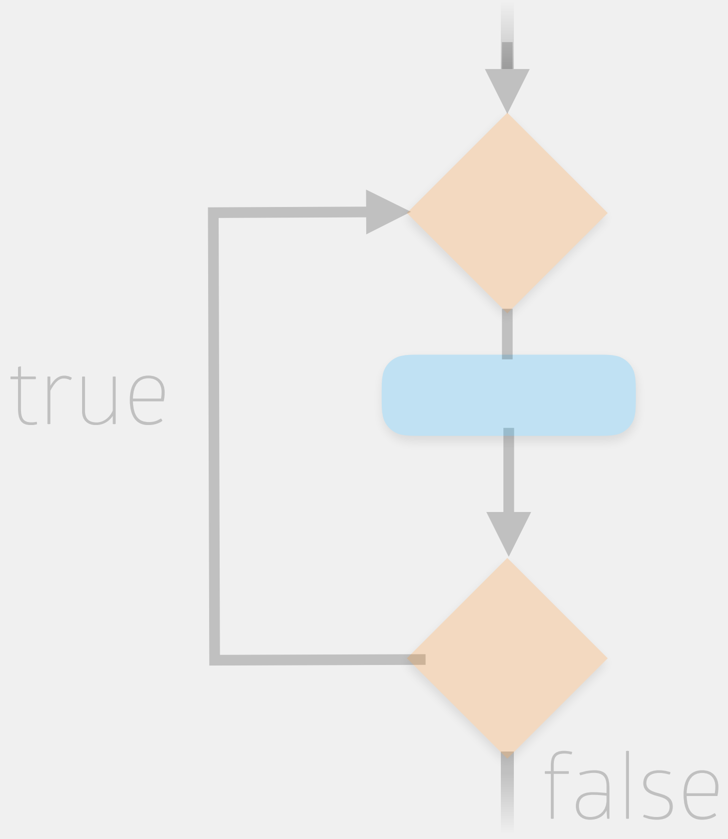
Types of Loops



while loops



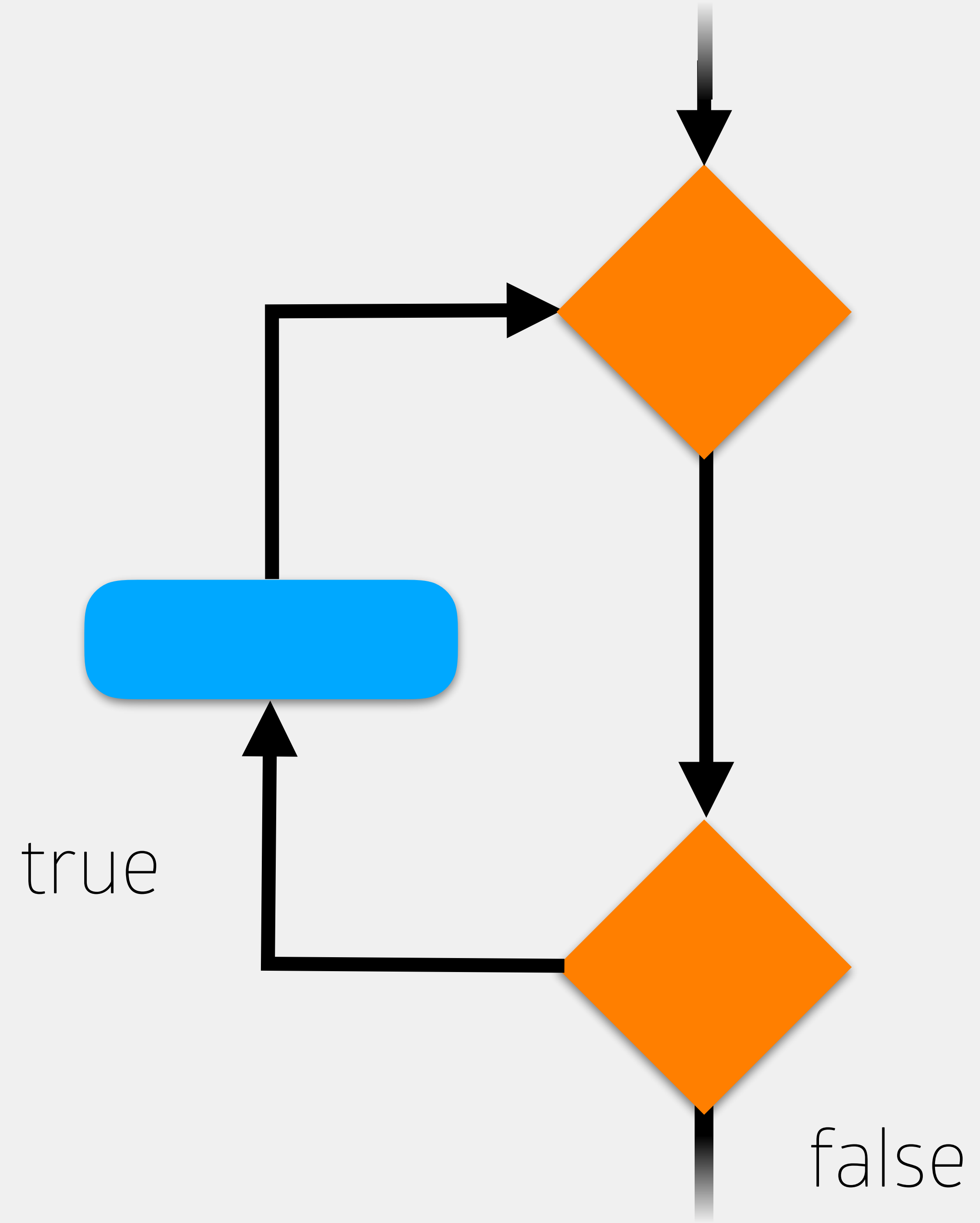
for loops



do-while loops

While Loops

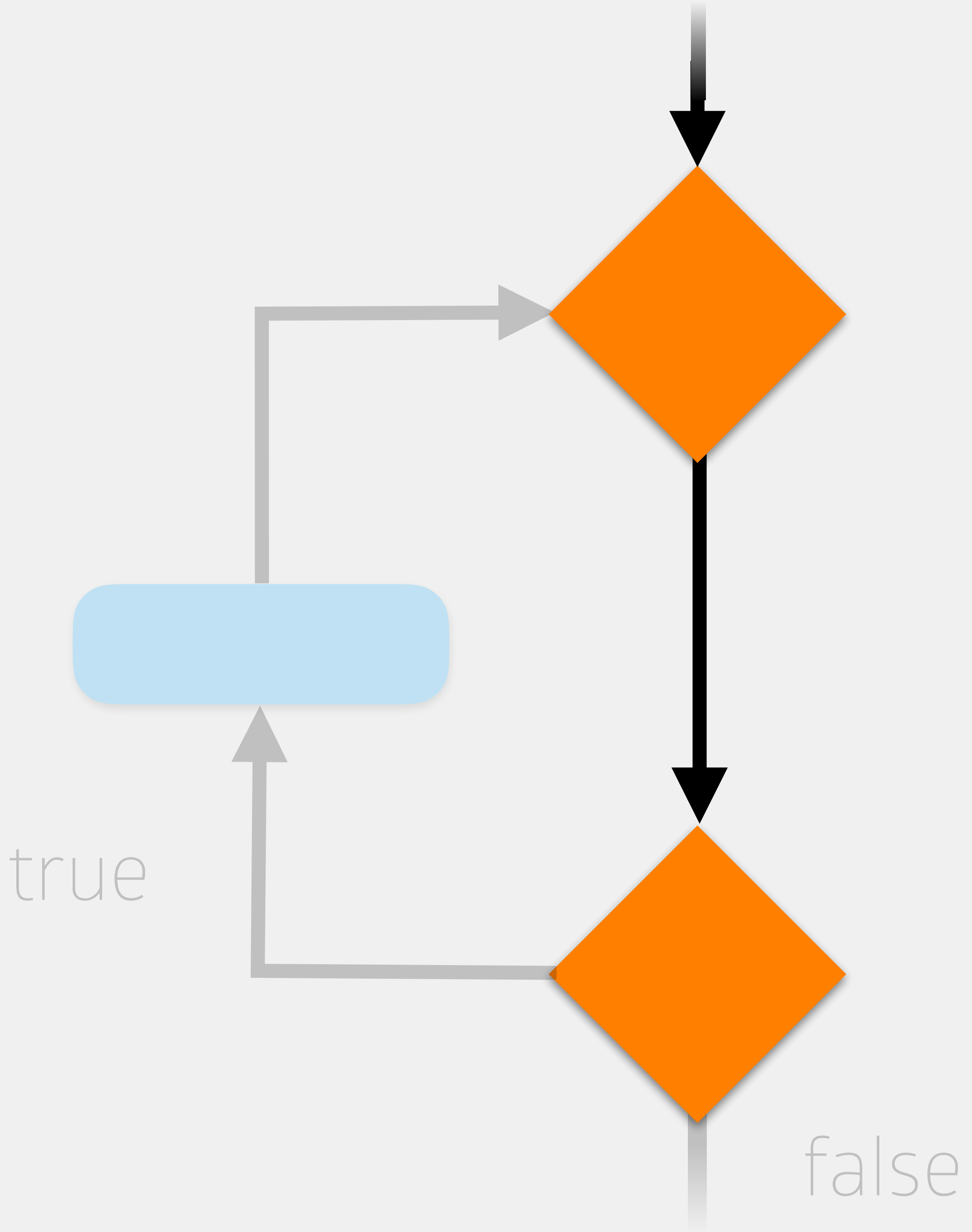
While the condition is true, execute the statements inside the loop



```
while (<boolean expression>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

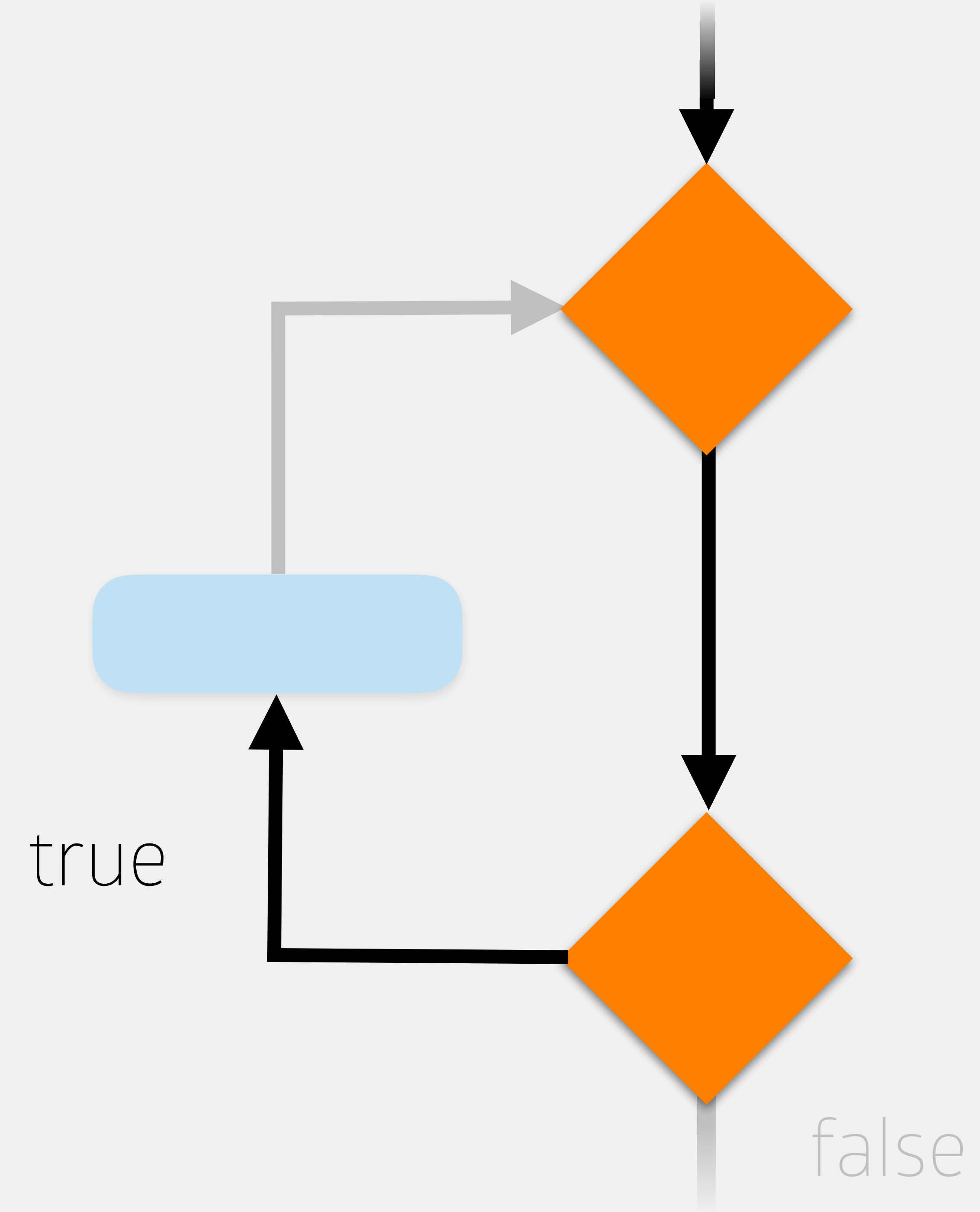
is this true?

memory

value (int)
5

While Loops

While the condition is true, execute the statements inside the loop



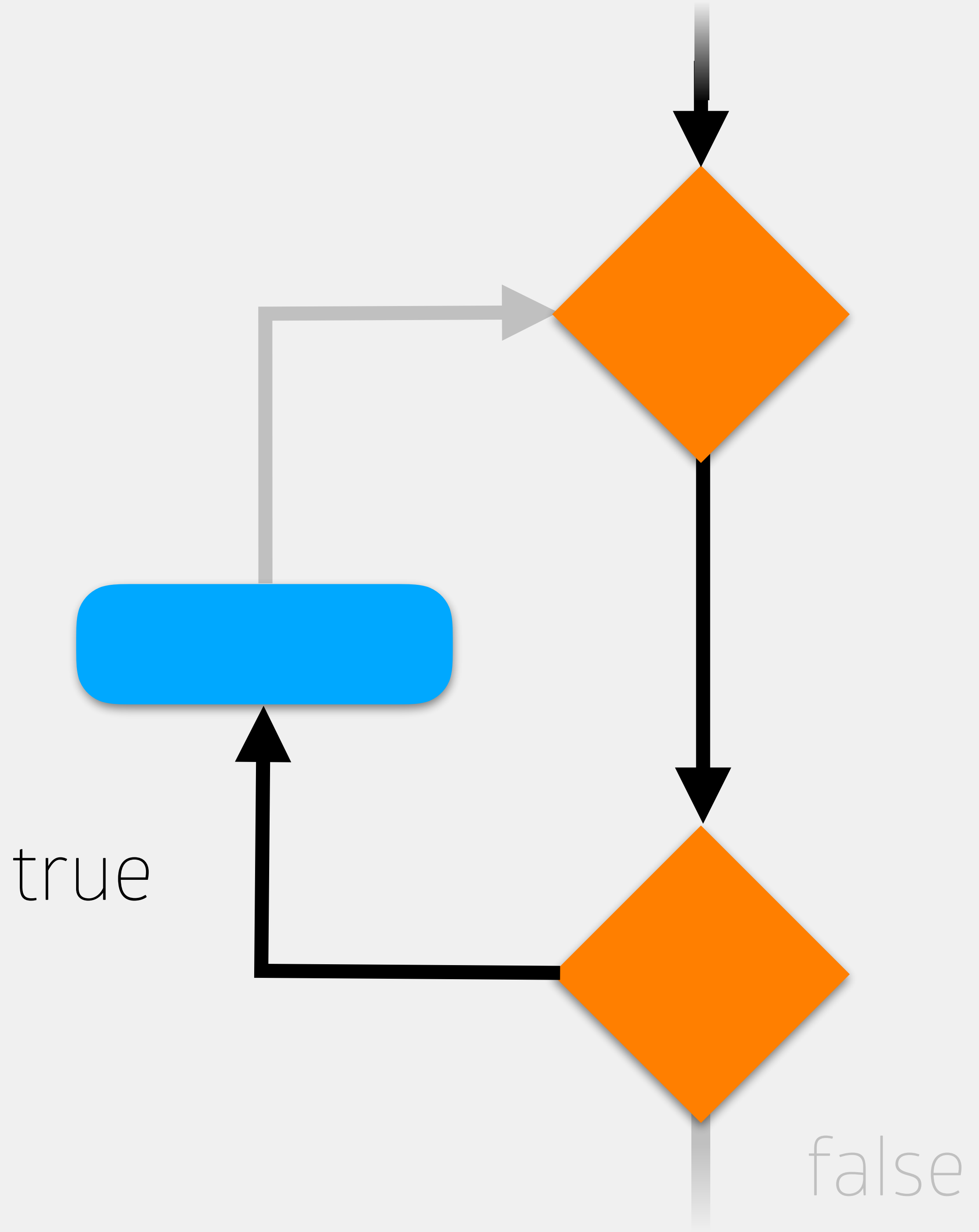
```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

memory

value (int)
5

While Loops

While the condition is true, execute the statements inside the loop



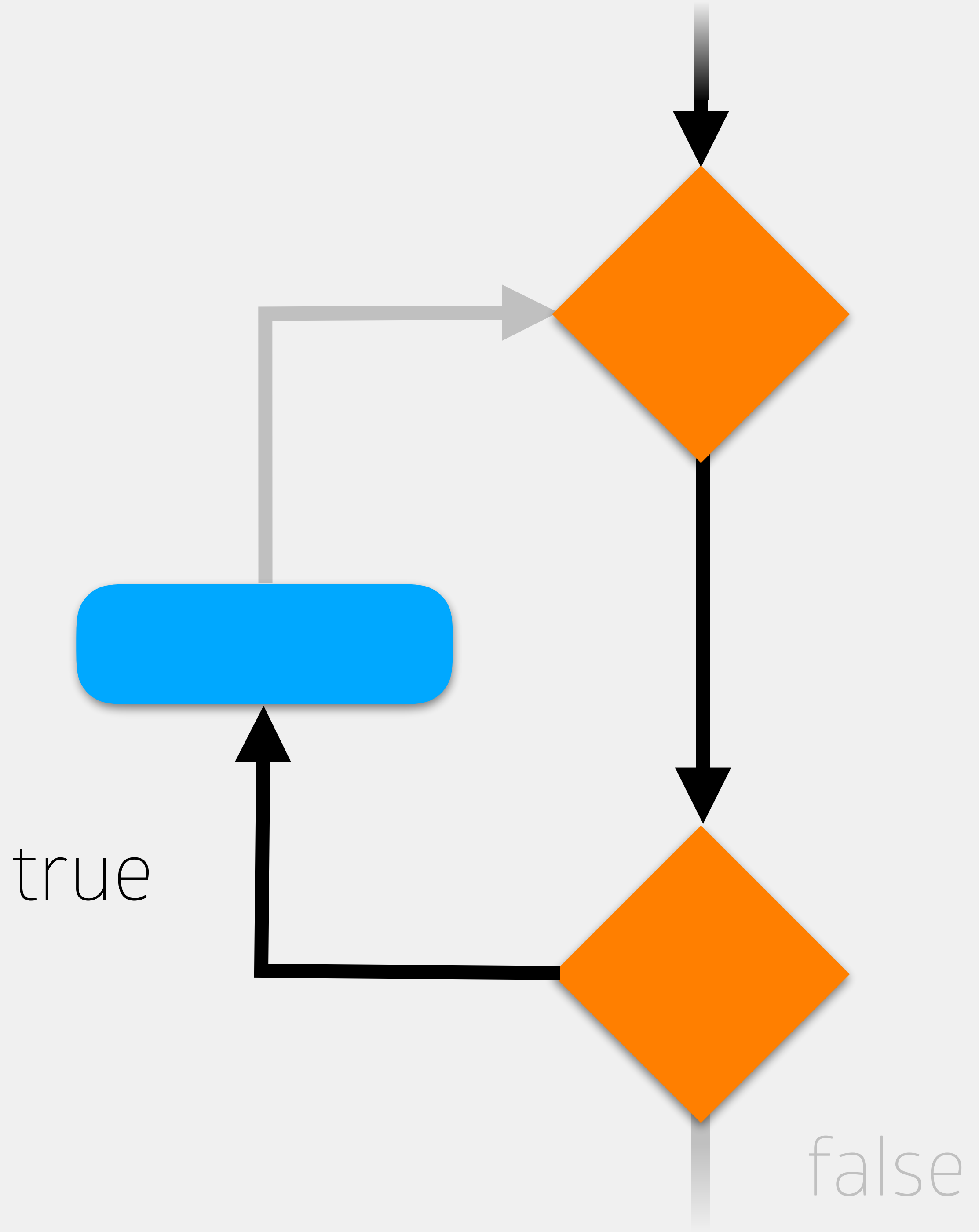
```
int value = 5;

while (value < 8) {

  > System.out.println(value);
    value++;

}

//code to execute after while loop
```

memory

value (int)
5

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
  > value++;

}

//code to execute after while loop
```

```
5
```

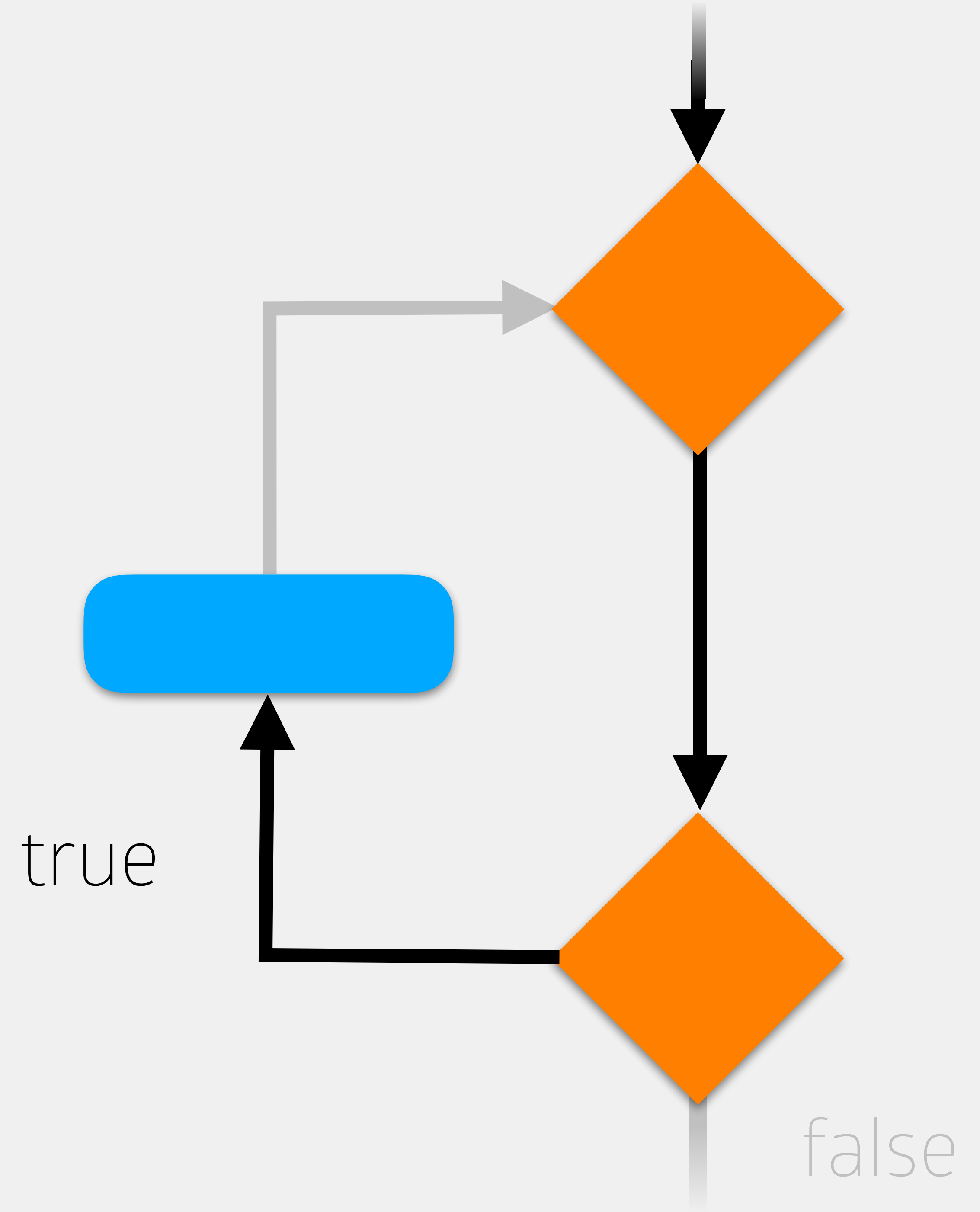
memory

value (int)

5

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
    value++;
  >
}

//code to execute after while loop
```

```
5
```

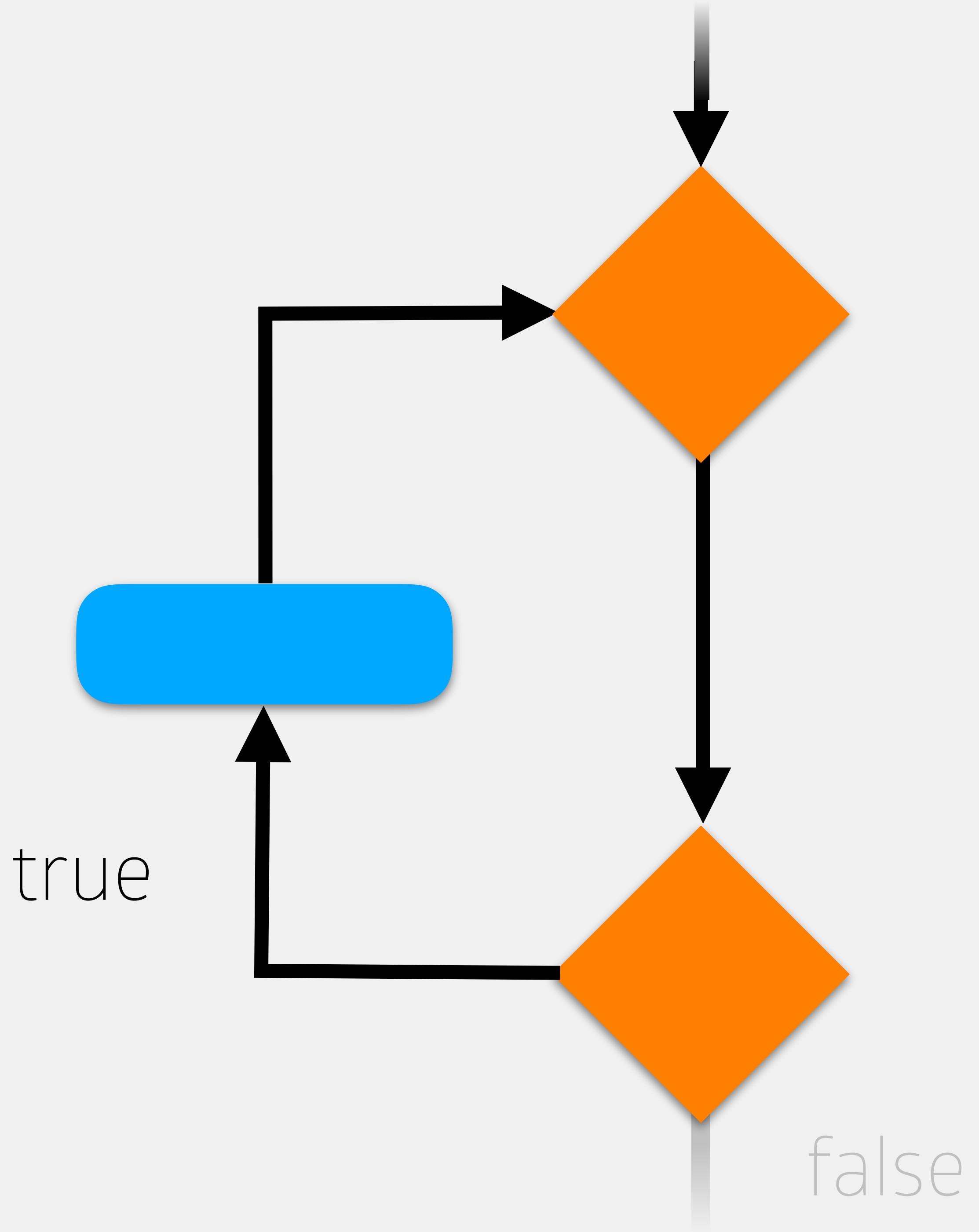
memory

value (int)

6

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

```
5
```

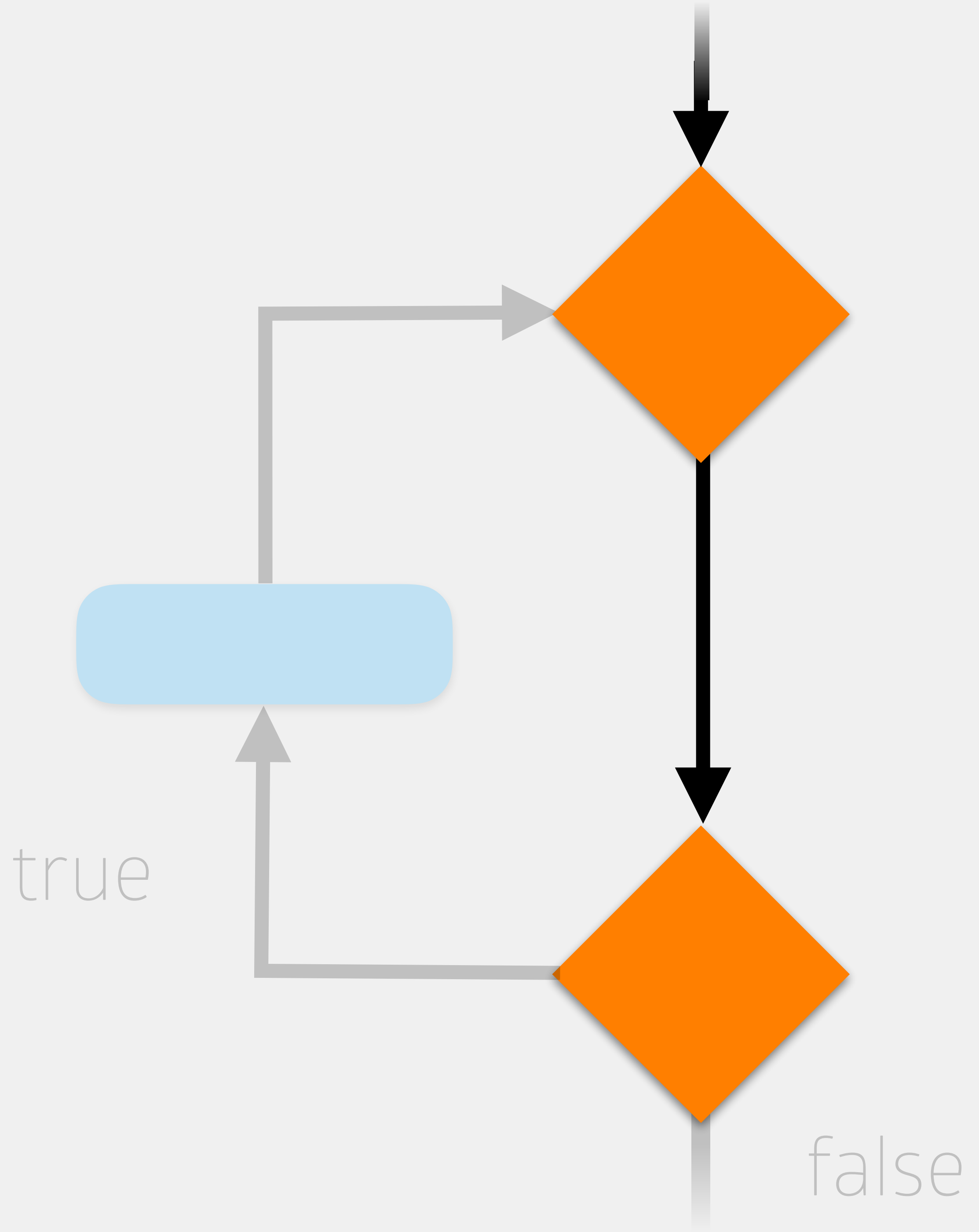
memory

value (int)

6

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;
> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

is this true?

memory

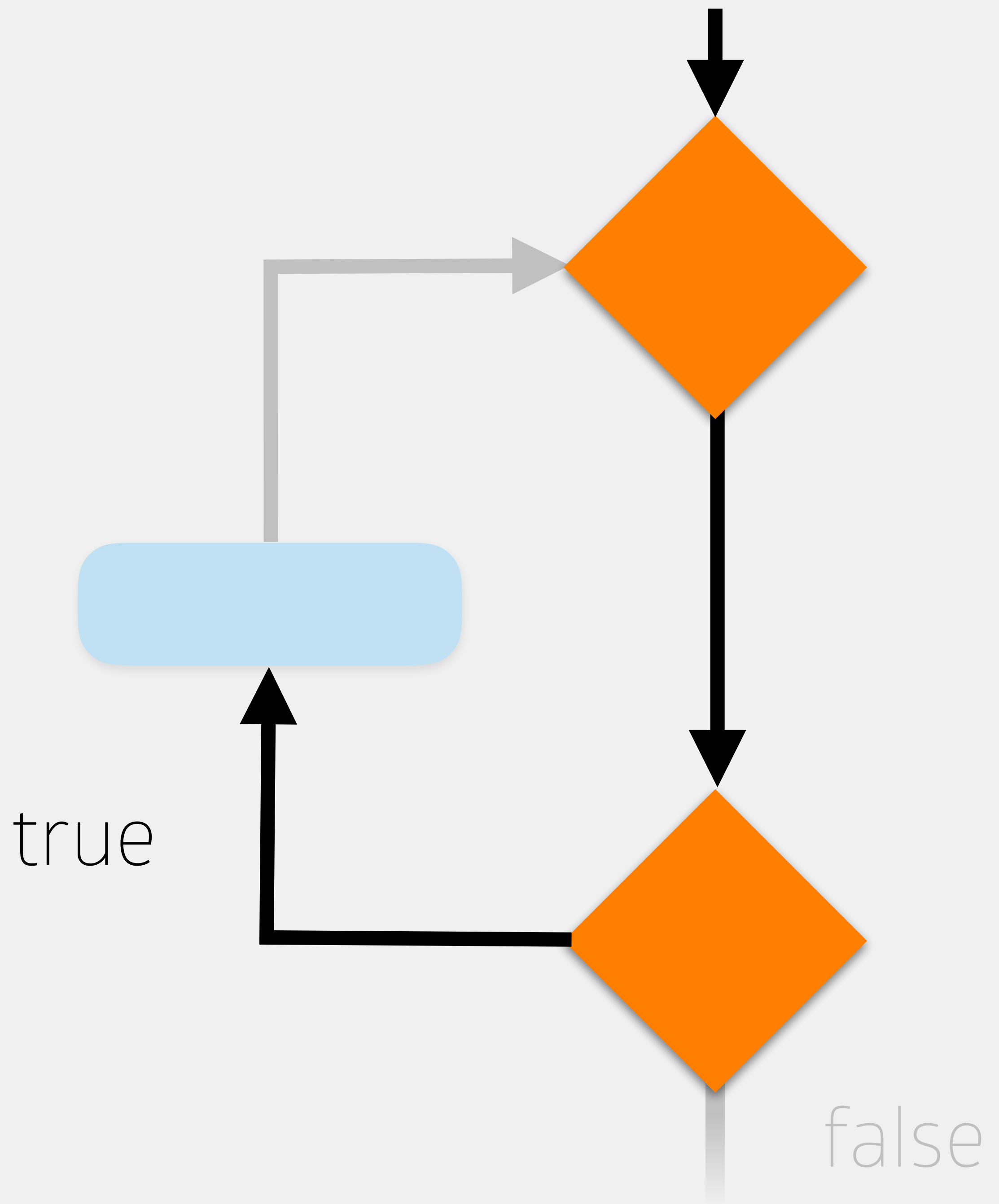
value (int)

6

```
5
```

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

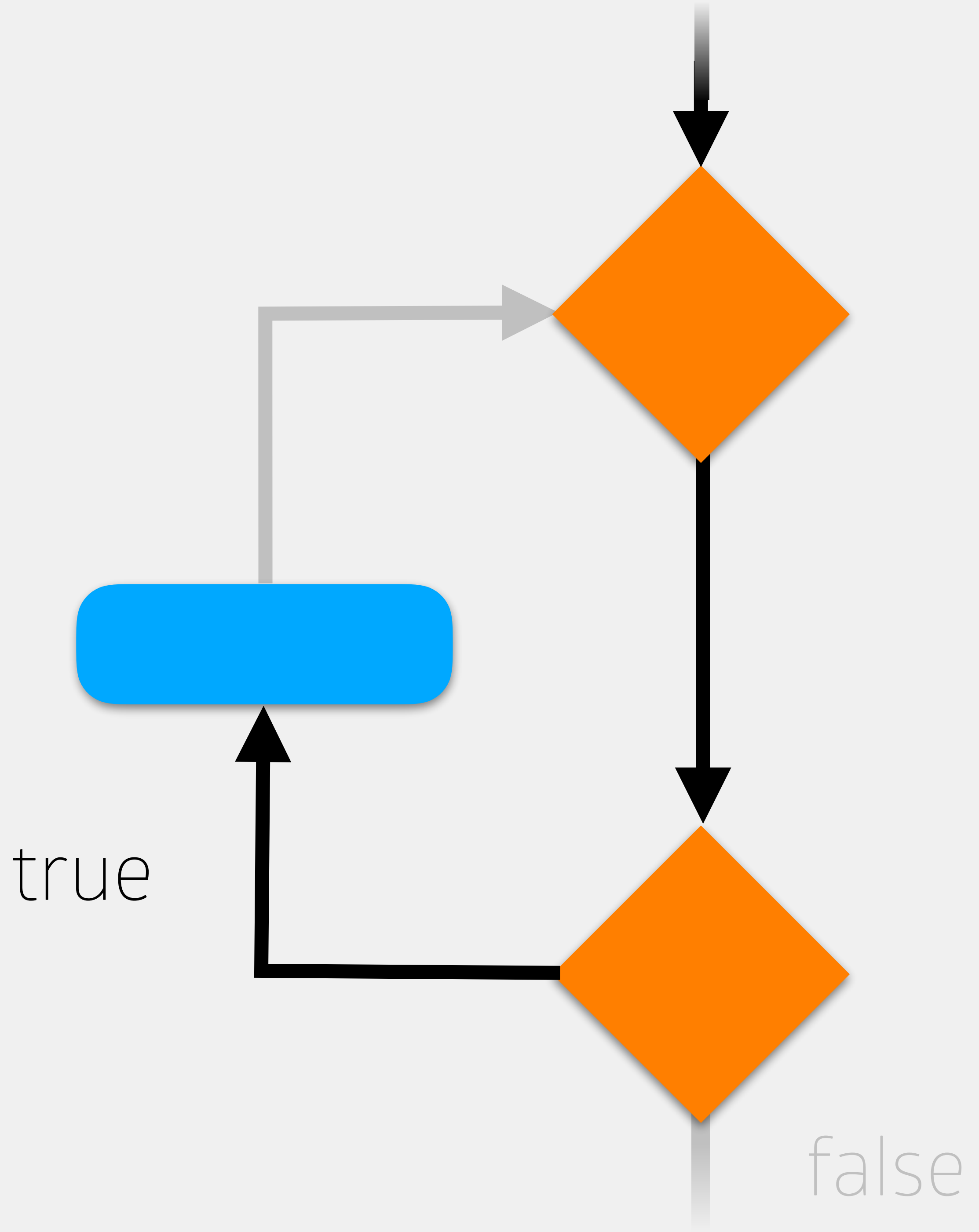
```
5
```

memory

value (int)
6

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

  > System.out.println(value);
    value++;

}

//code to execute after while loop
```

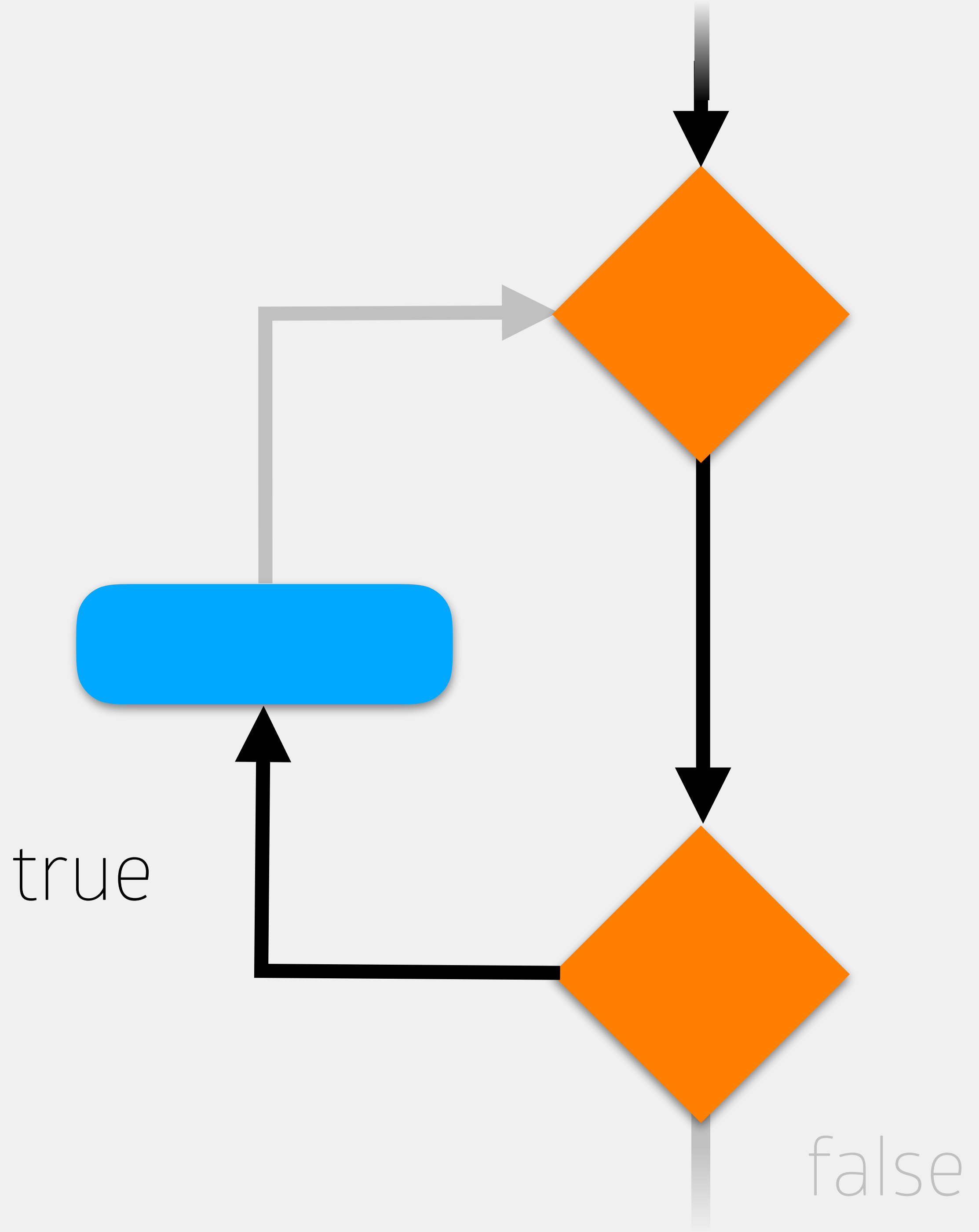
```
5
```

memory

value (int)
6

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
  > value++;

}

//code to execute after while loop
```

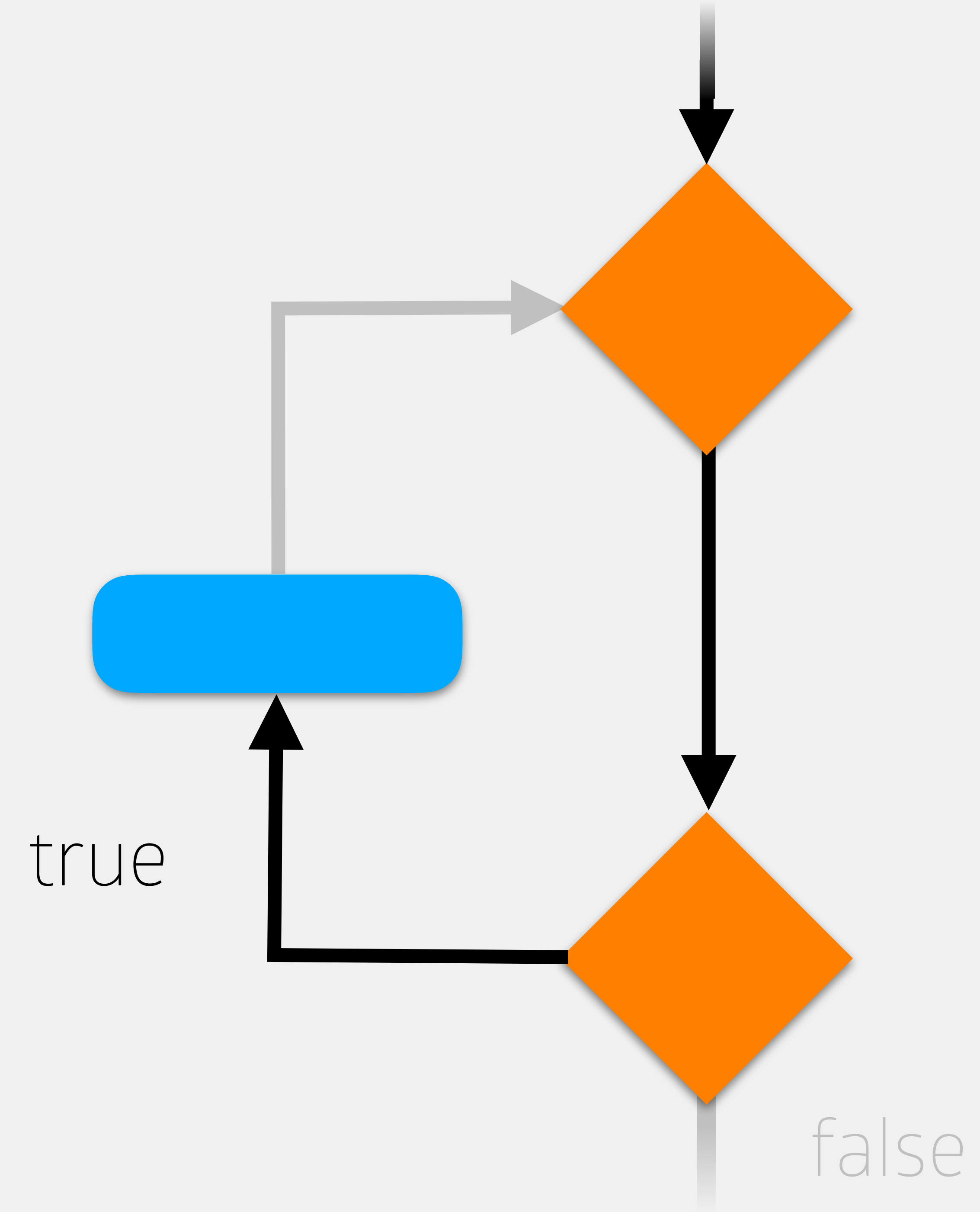
```
5
6
```

memory

value (int)
6

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
    value++;
  >
}

//code to execute after while loop
```

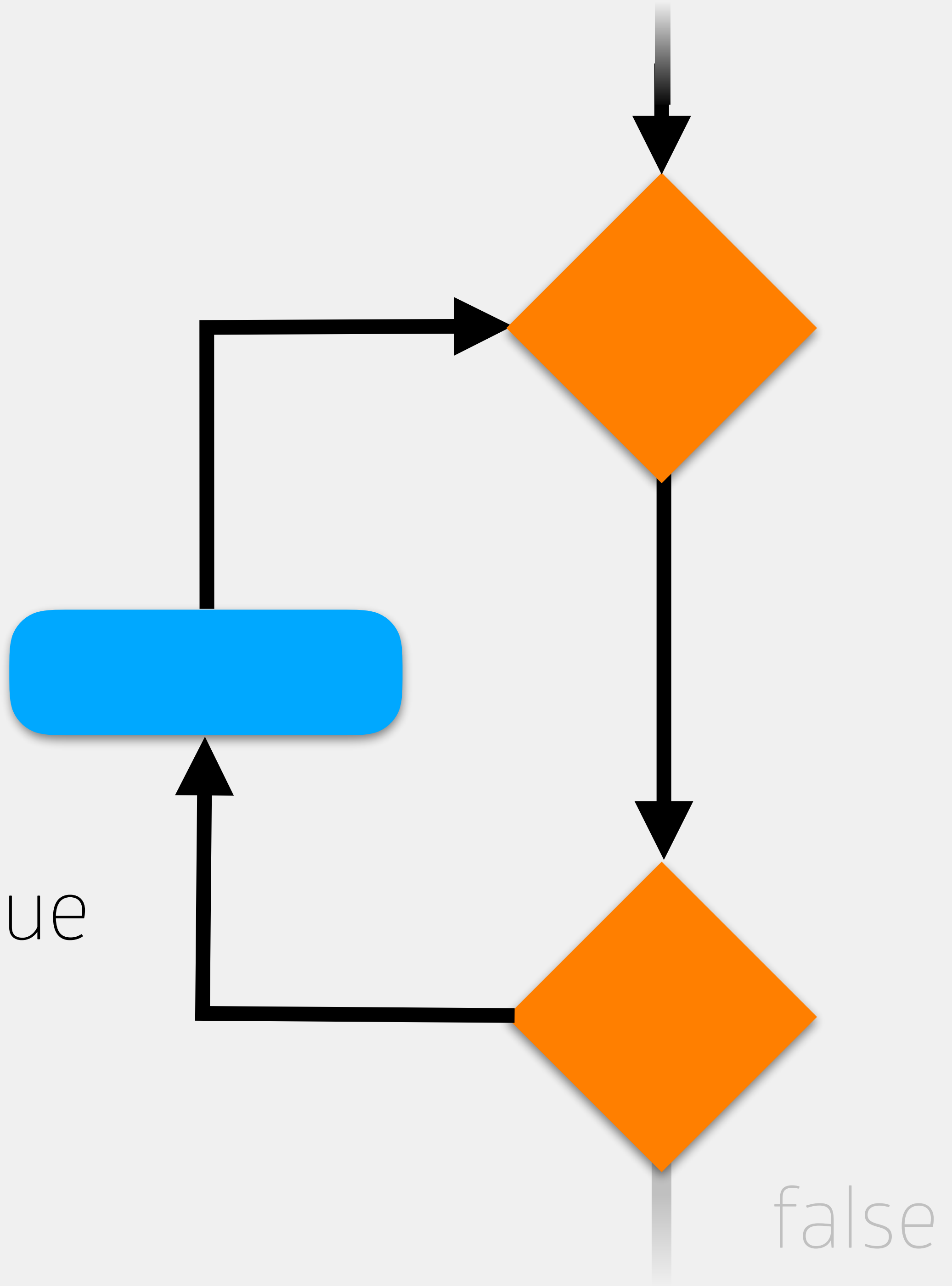
```
5
6
```

memory

value (int)
7

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

```
5
6
```

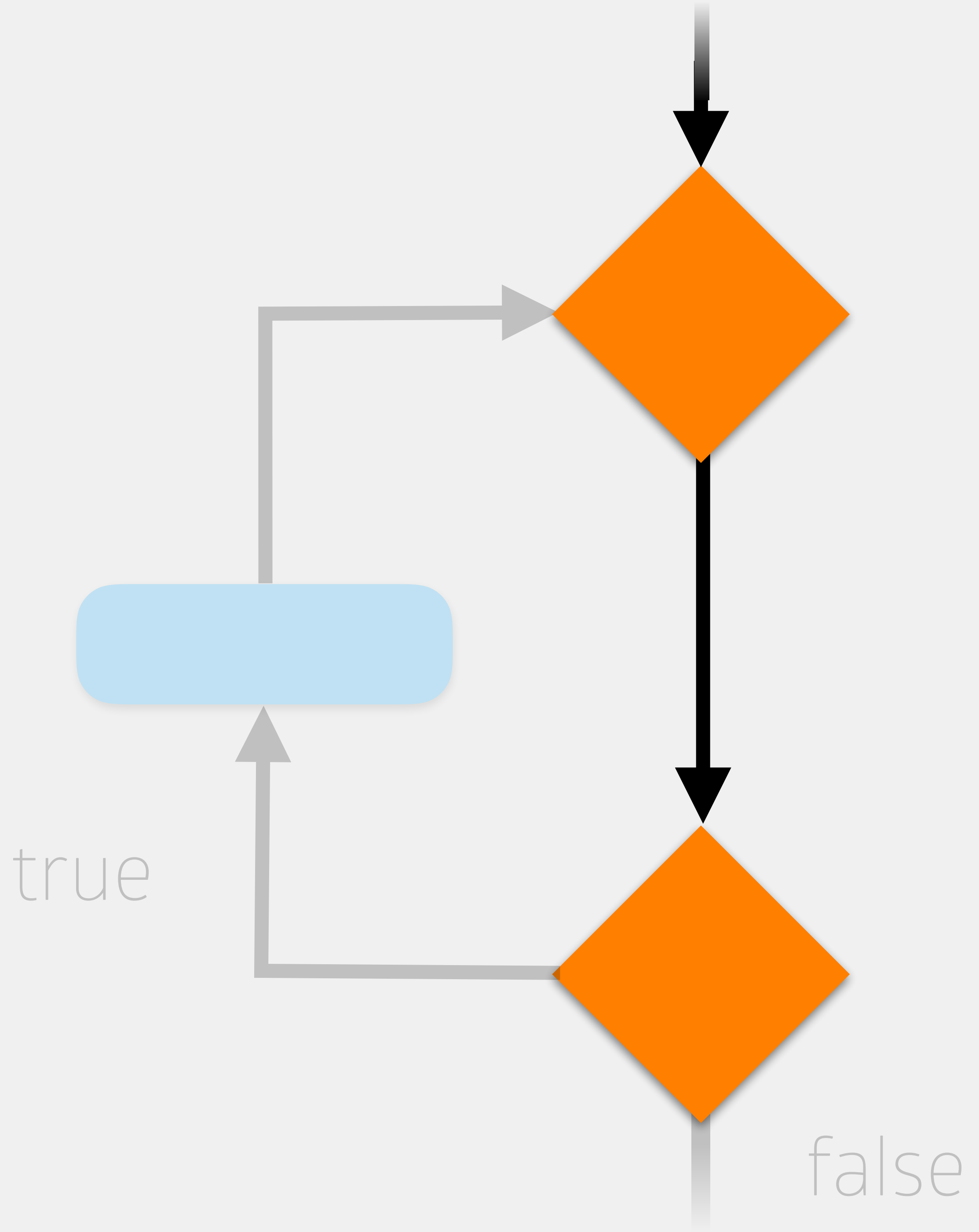
memory

value (int)

7

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;
> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

is this true?

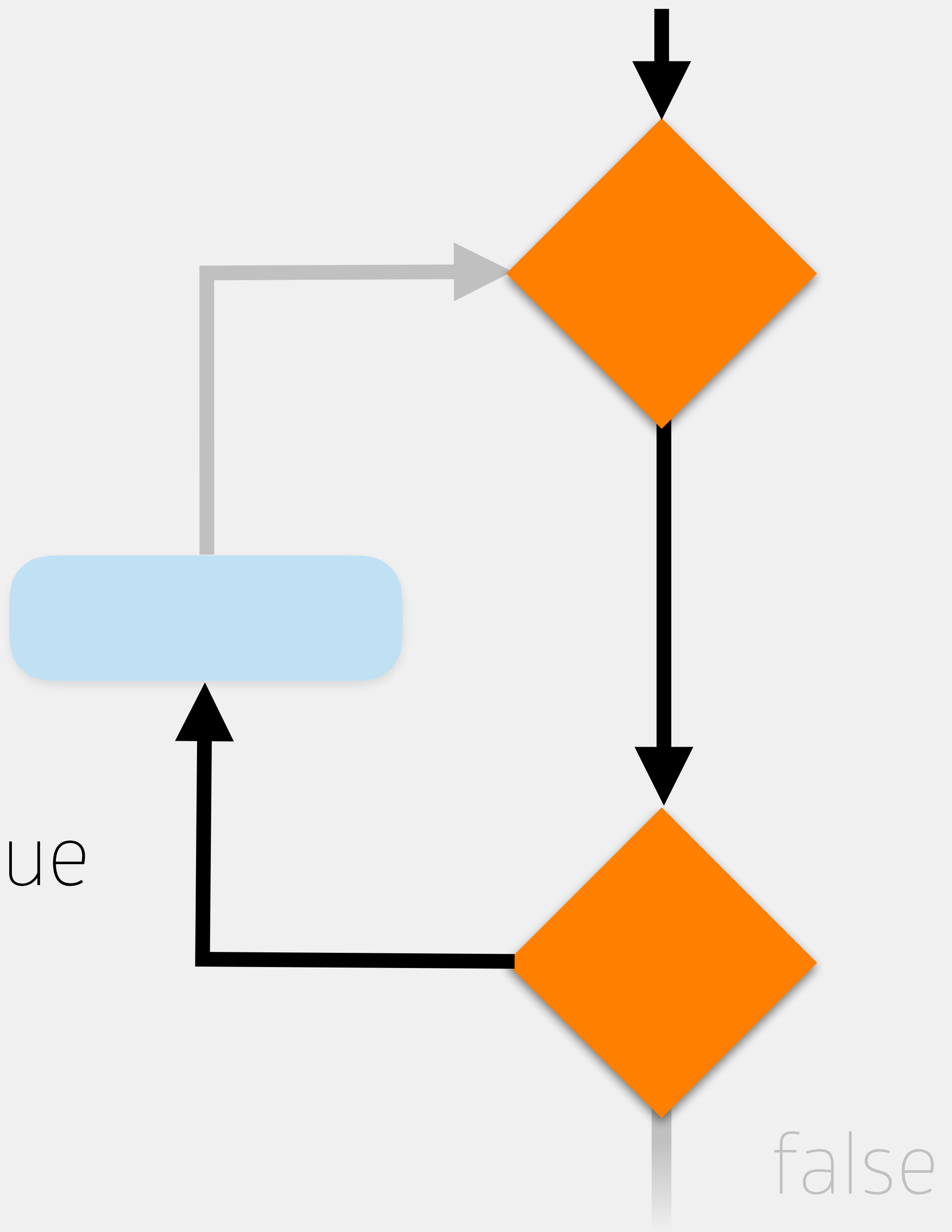
```
5
6
```

memory

value (int)
7

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

```
5
6
```

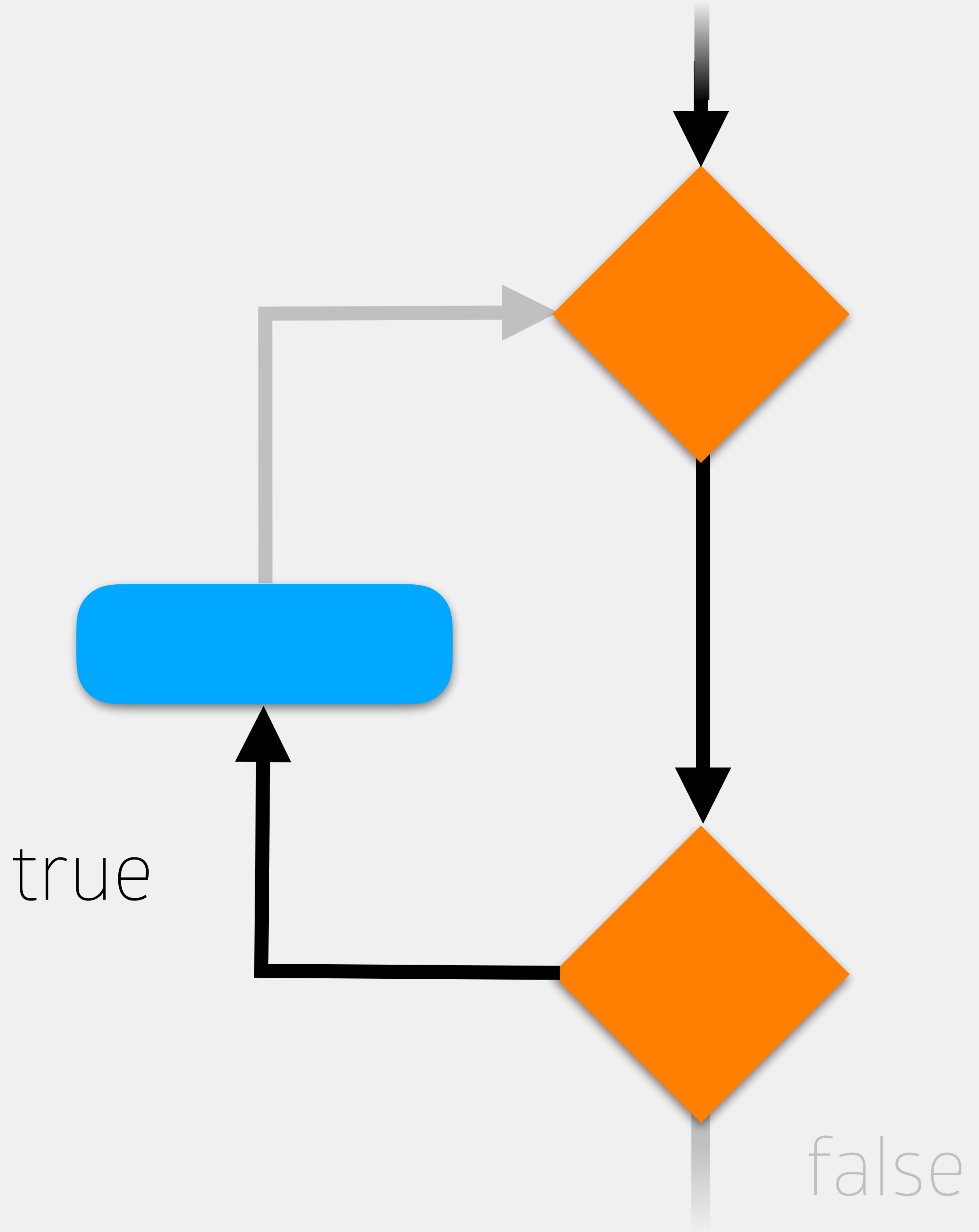
memory

value (int)

7

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

  > System.out.println(value);
    value++;

}

//code to execute after while loop
```

```
5
6
```

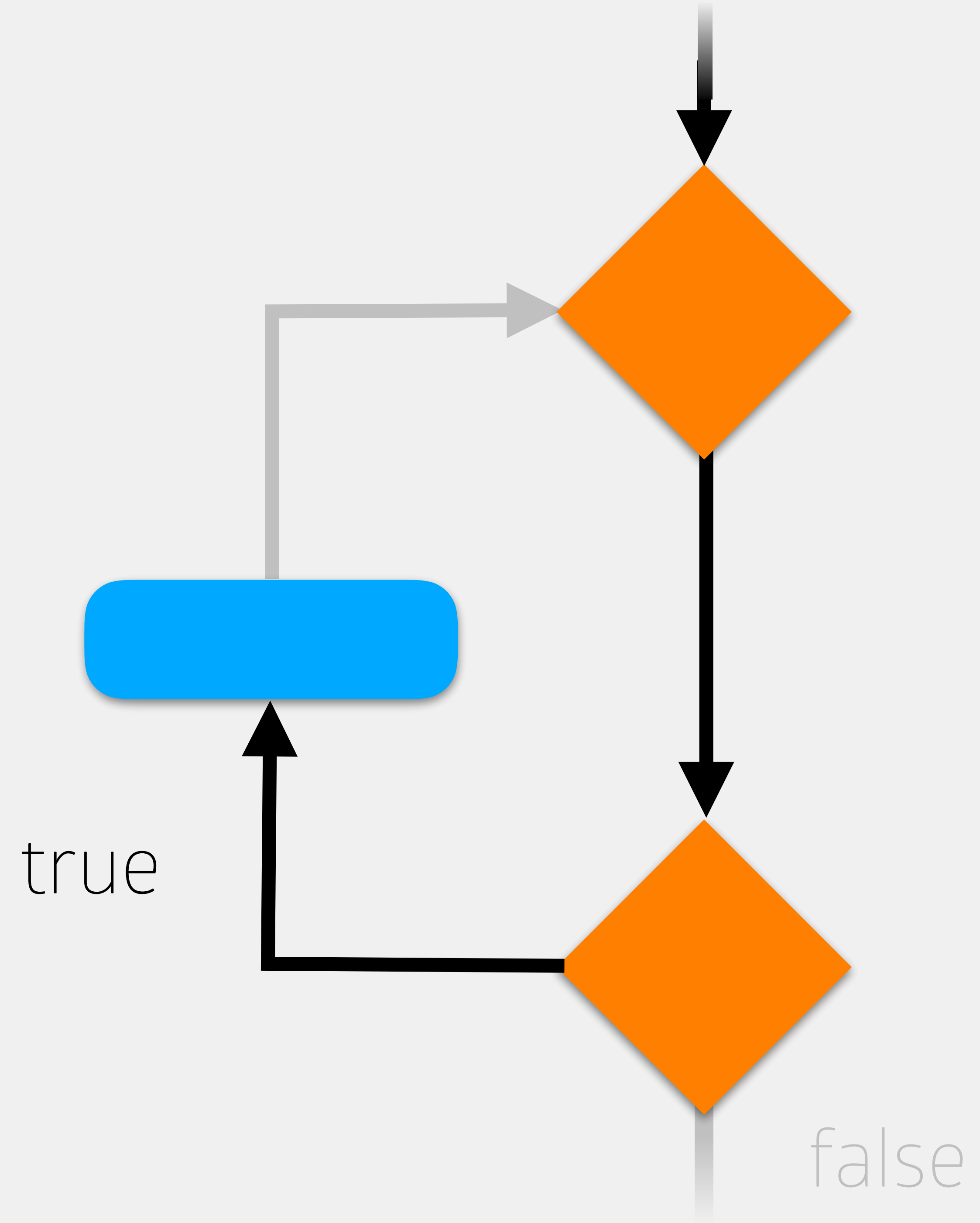
memory

value (int)

7

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
  > value++;

}

//code to execute after while loop
```

memory

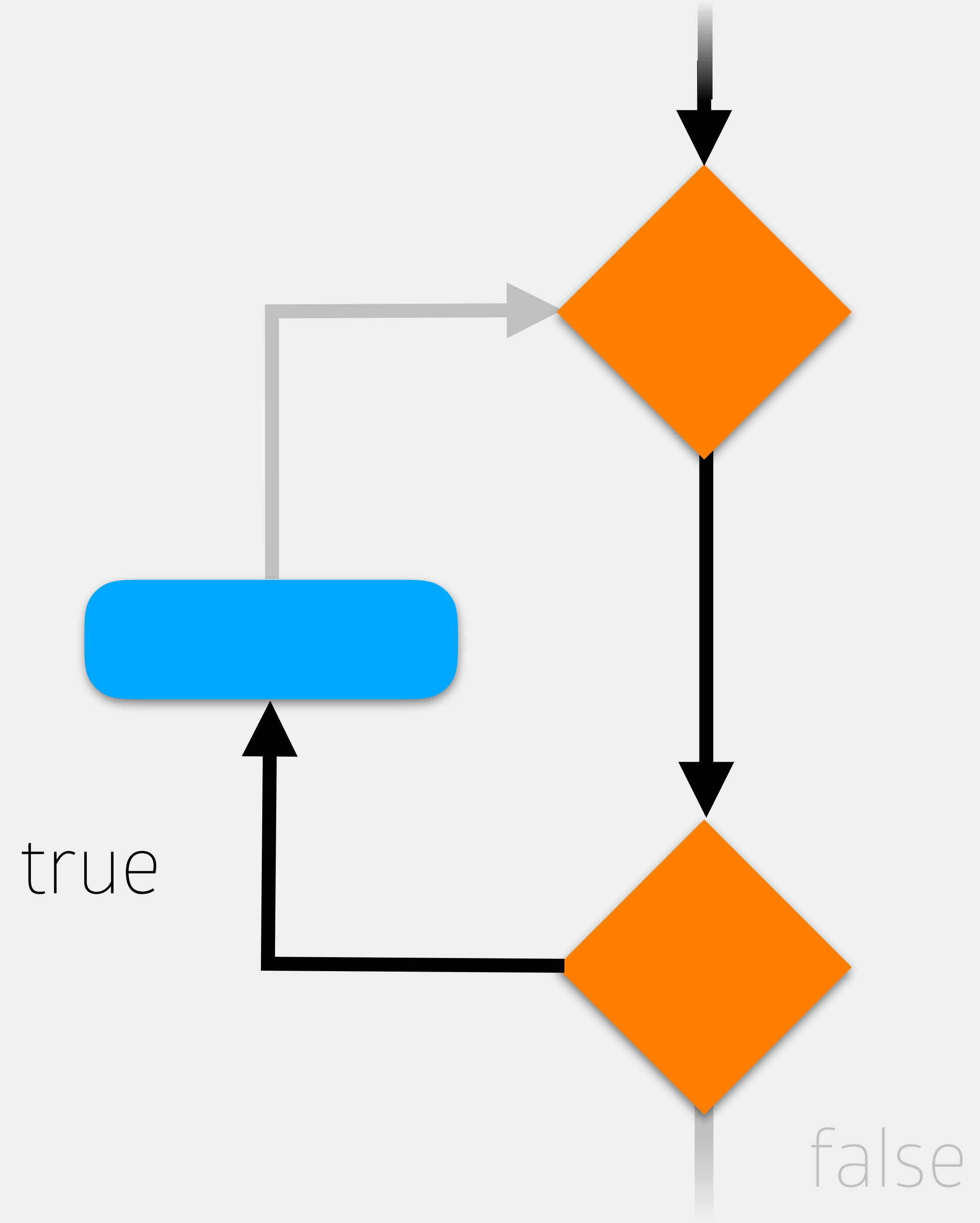
value (int)

7

```
5
6
7
```

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
    value++;
  >
}

//code to execute after while loop
```

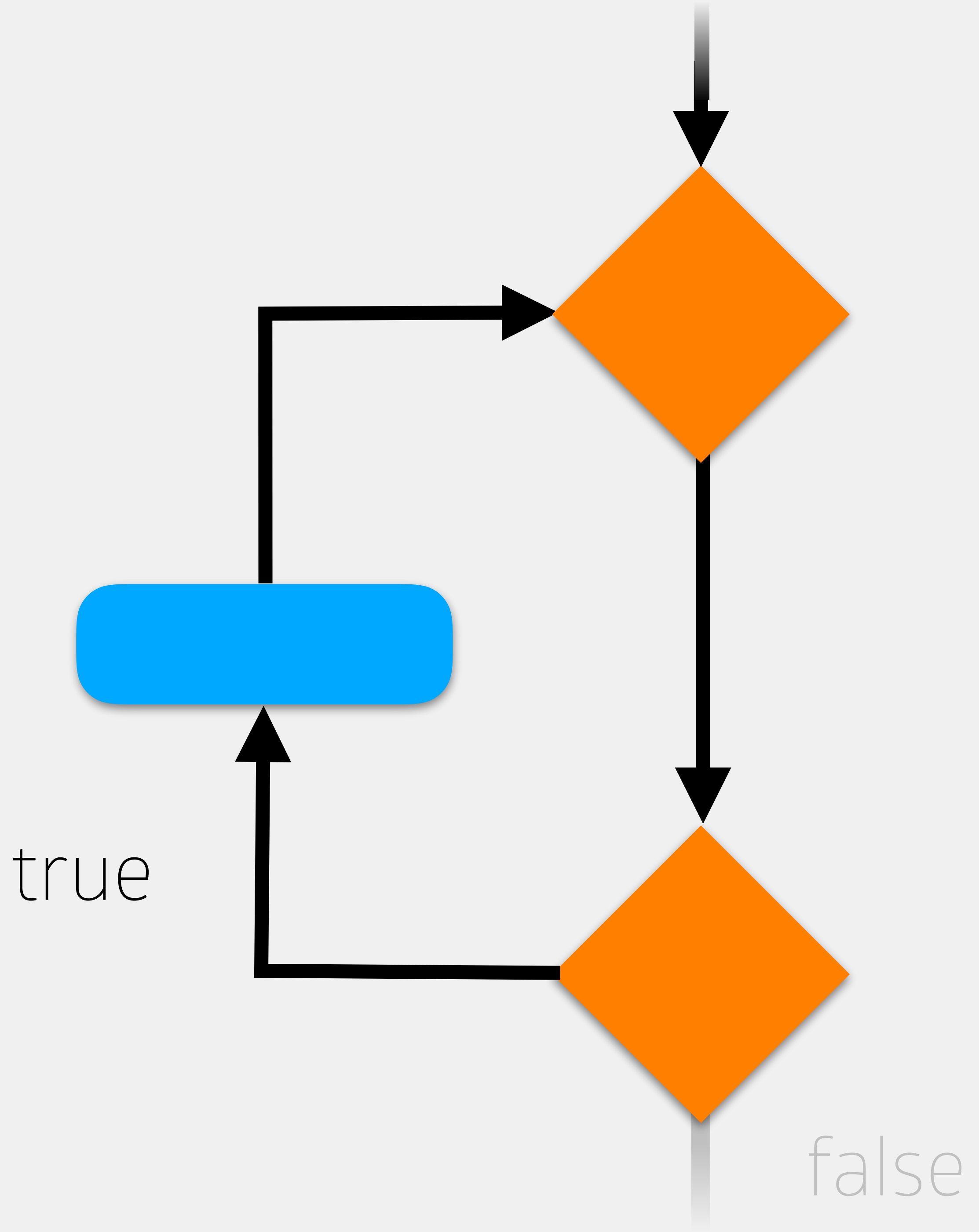
```
5
6
7
```

memory

value (int)
8

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;

> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

```
5
6
7
```

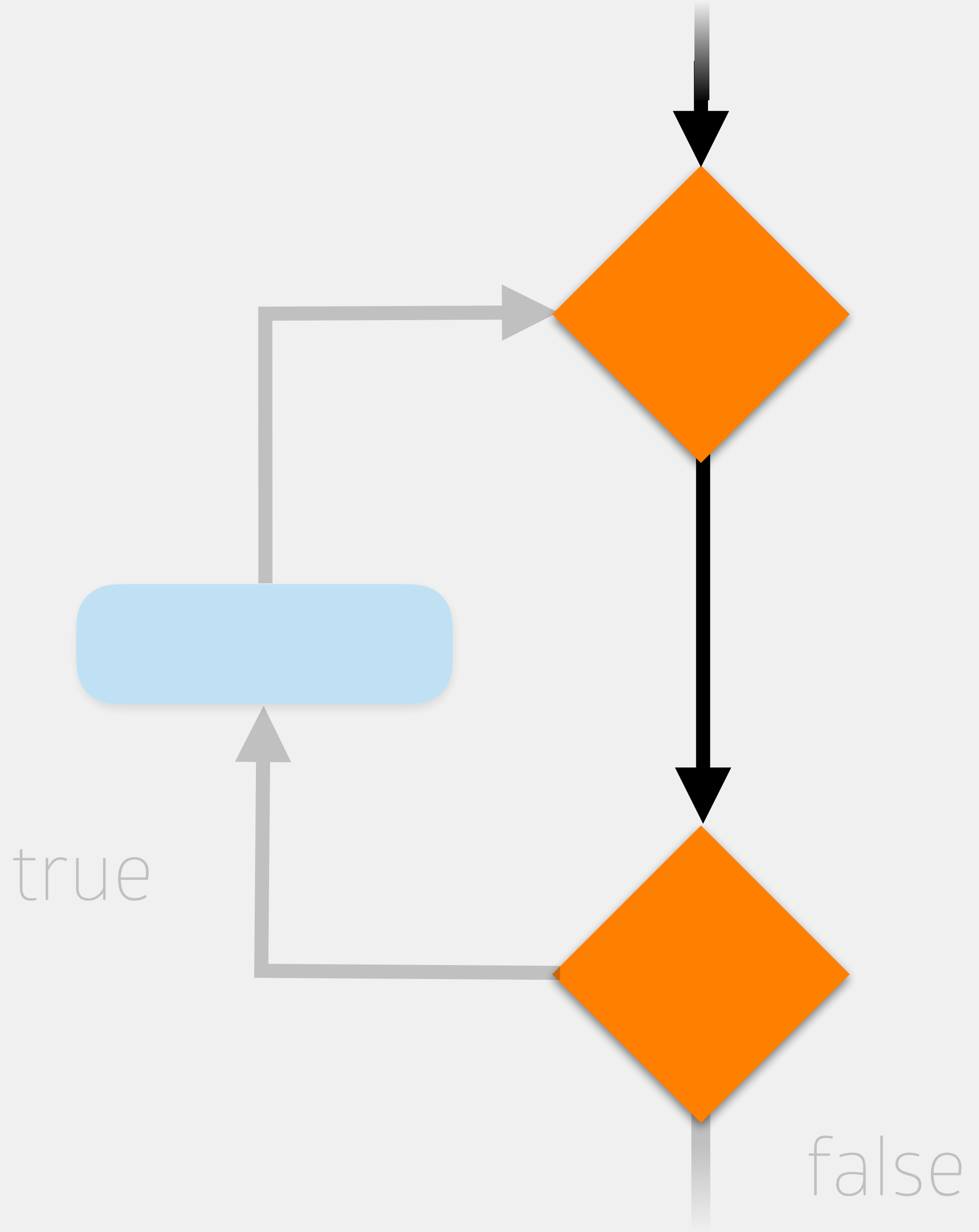
memory

value (int)

8

While Loops

While the condition is true, execute the statements inside the loop



```
  int value = 5;
> while (value < 8) {

      System.out.println(value);
      value++;

  }

  //code to execute after while loop
```

is this true?

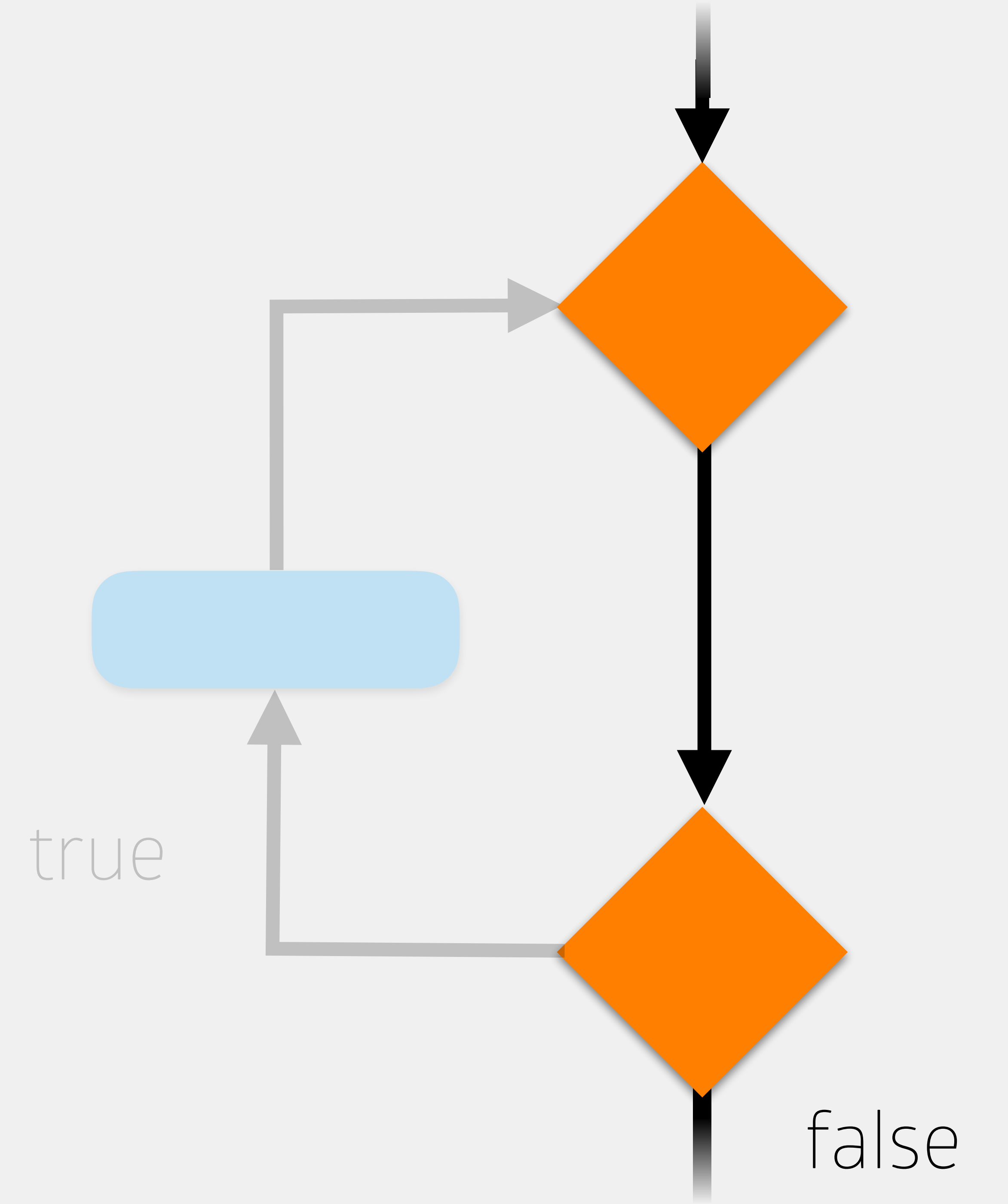
```
5
6
7
```

memory

value (int)
8

While Loops

While the condition is true, execute the statements inside the loop



```
int value = 5;

while (value < 8) {

    System.out.println(value);
    value++;

}

//code to execute after while loop
>
```

```
5
6
7
```

memory

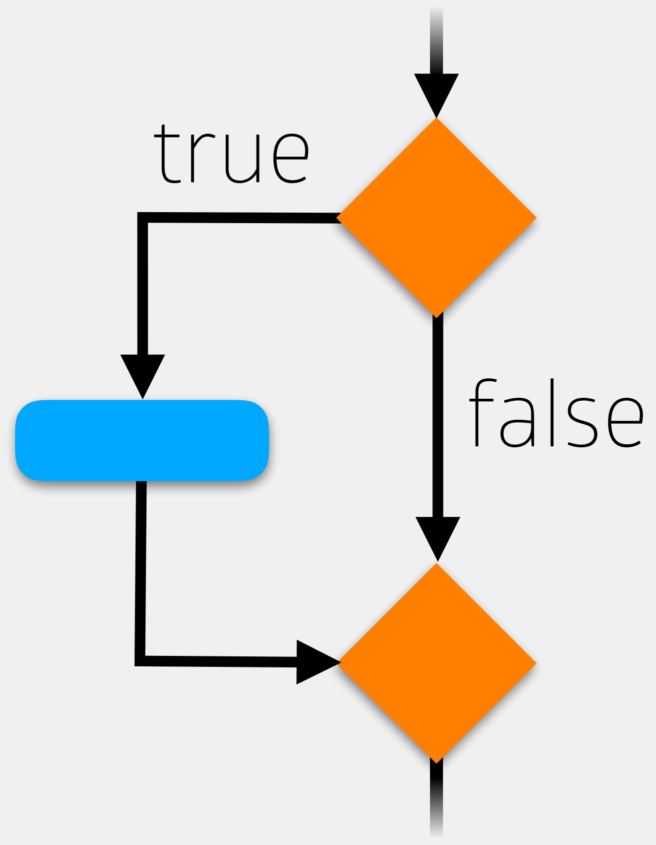
value (int)

8

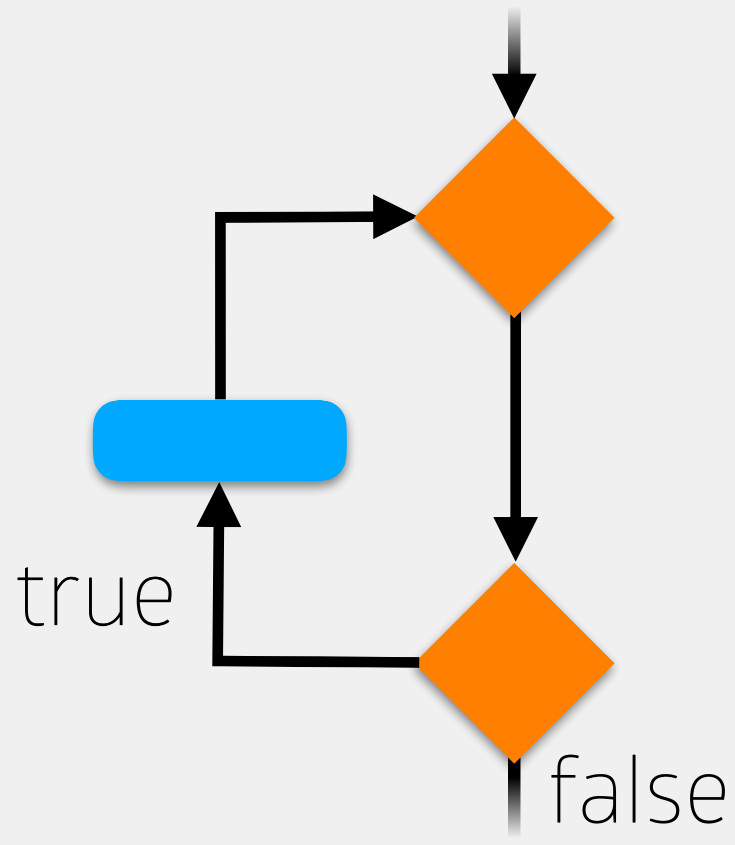
Loops vs Conditionals

Conditionals decide whether or not to execute a block of code once

Loops decide whether or not to execute a block of code multiple times



conditional statements



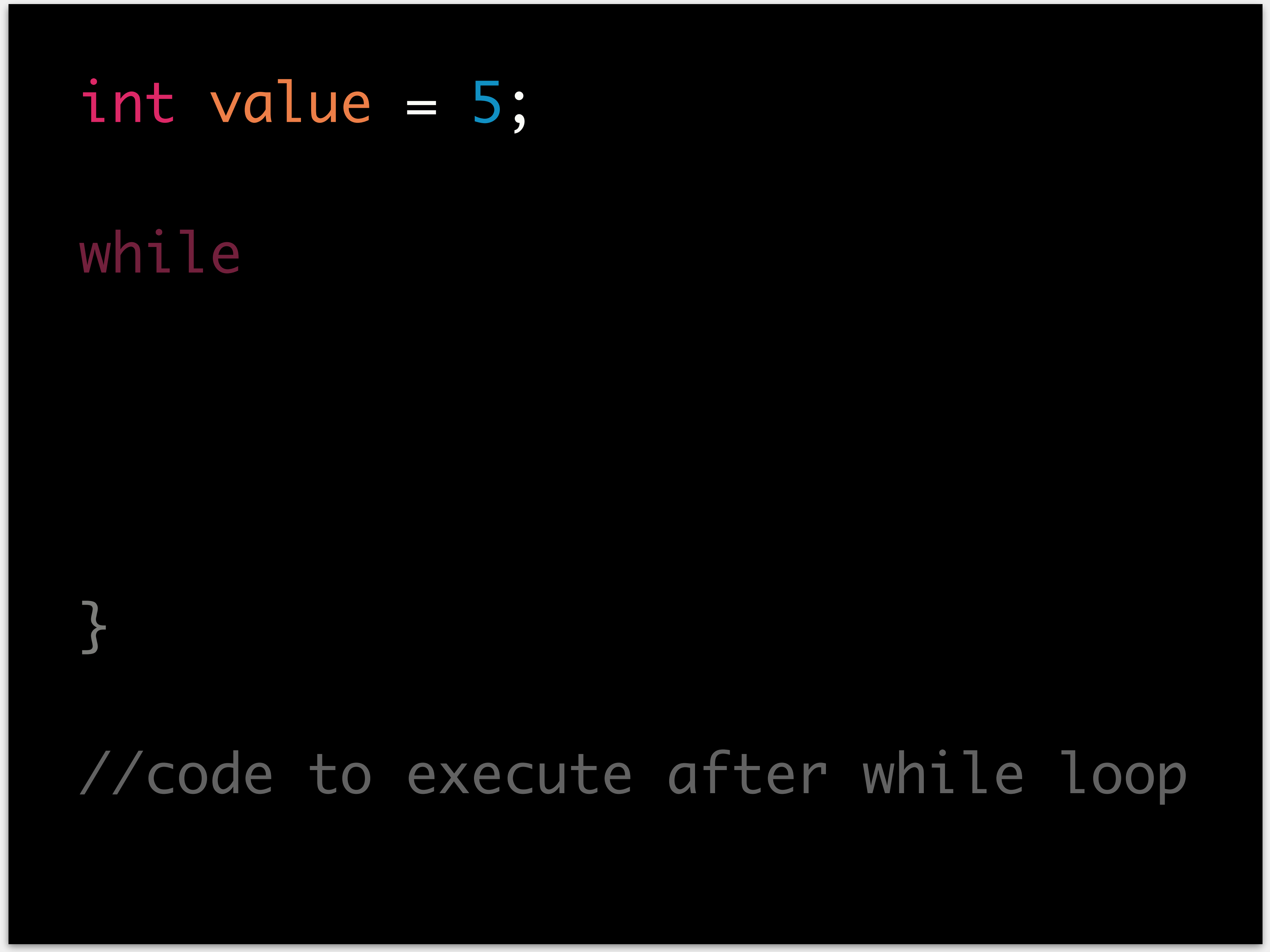
loops

Parts of a Loop

```
int value = 5;

while (value < 8) {

    System.out.println(value);
    value++;

}

//code to execute after while loop
```

Every loop has four parts

Parts of a Loop



```
int value = 5;

while



}

//code to execute after while loop
```

Every loop has four parts

- initialization
    - set up a variable that will control the loop

Parts of a Loop

```
int

while  value < 8




}

//code to execute after while loop
```

Every loop has four parts

- initialization
  - set up a variable that will control the loop
- condition
  - a boolean expression to control when the loop stops

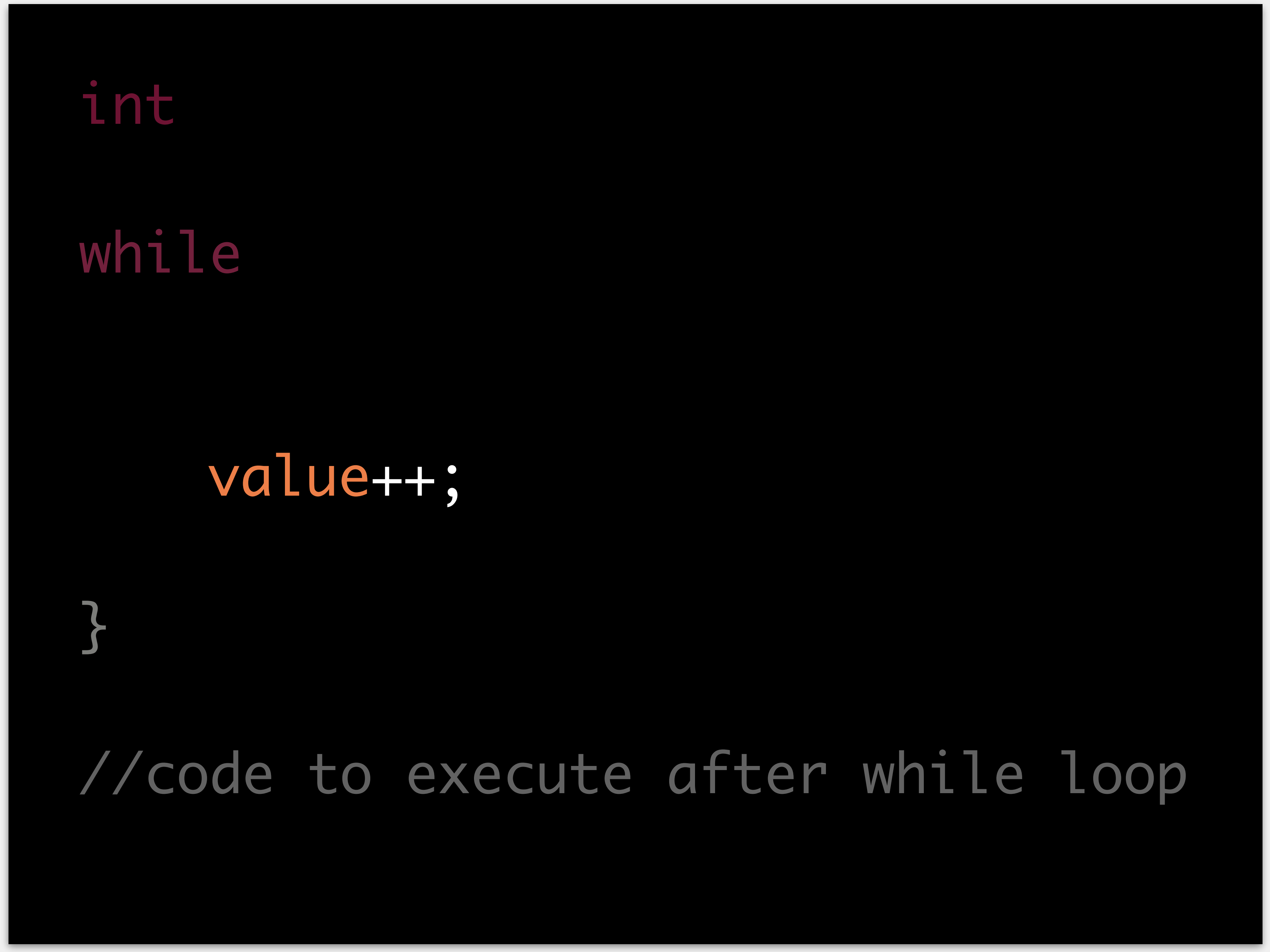
Parts of a Loop

```
int

while

    System.out.println(value);


}

//code to execute after while loop
```

Every loop has four parts

- initialization
    - set up a variable that will control the loop
- condition
    - a boolean expression to control when the loop stops
- work
    - the code the loop will repeat

Parts of a Loop



```
int

while


    value++;

}

//code to execute after while loop
```

Every loop has four parts

- initialization
  - set up a variable that will control the loop
- condition
  - a boolean expression to control when the loop stops
- work
  - the code the loop will repeat
- progress
  - how the loop moves closer to termination

Finite vs Infinite Loops

Usually, we want loops to stop at some point, resume with code after loop

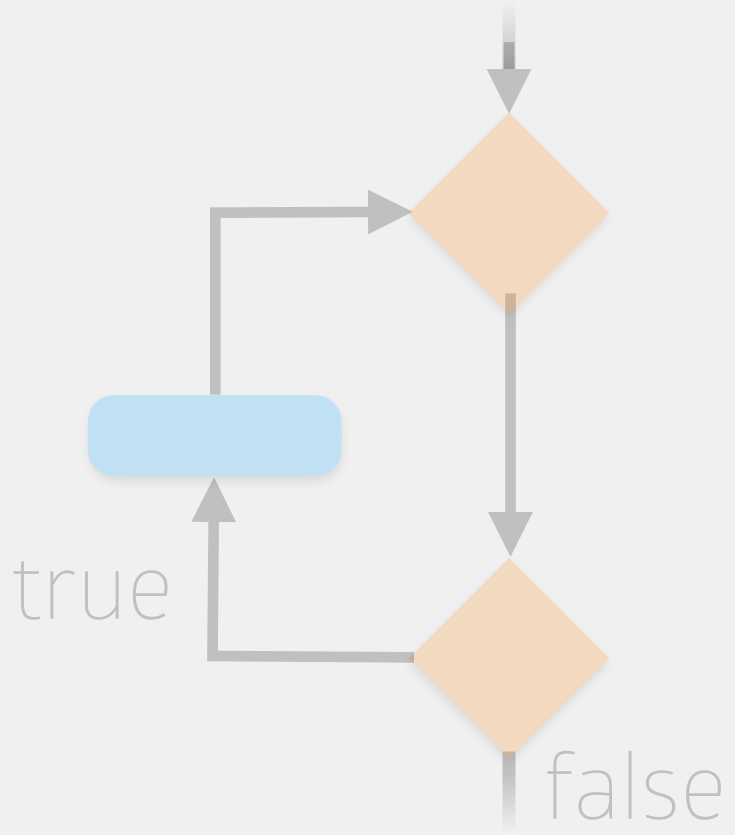
- isn't true for all applications
- will always be true for this class!

finite loops are those that stop

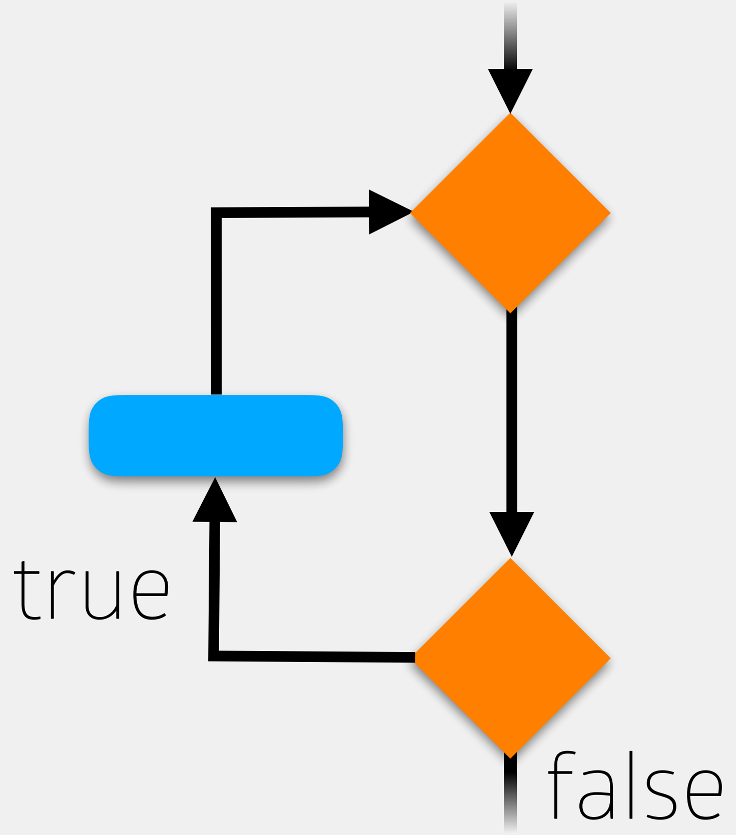
infinite loops are those that repeat forever

- will require you to manually terminate your program

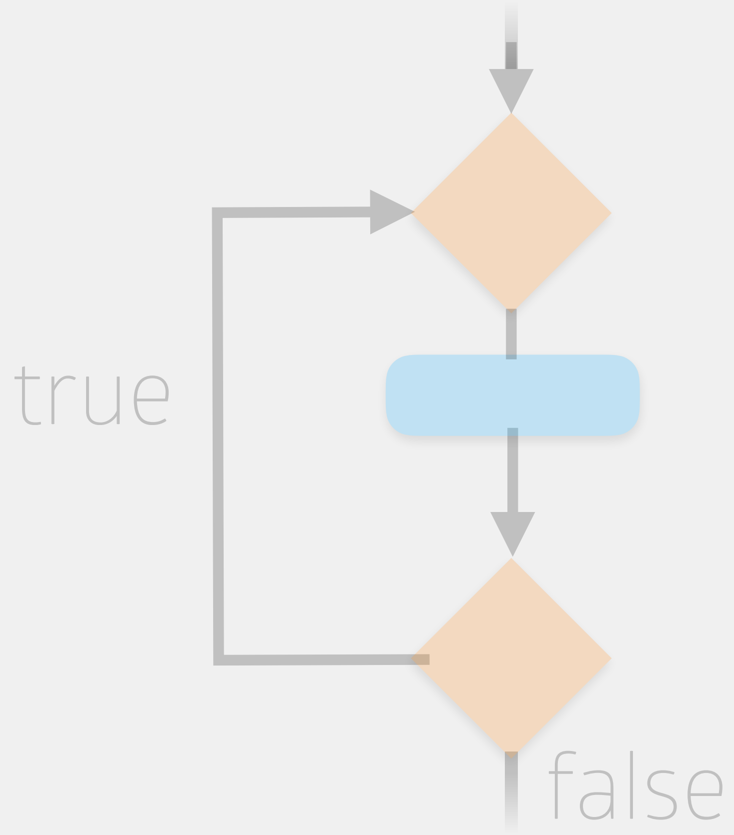
Types of Loops



`while`
loops



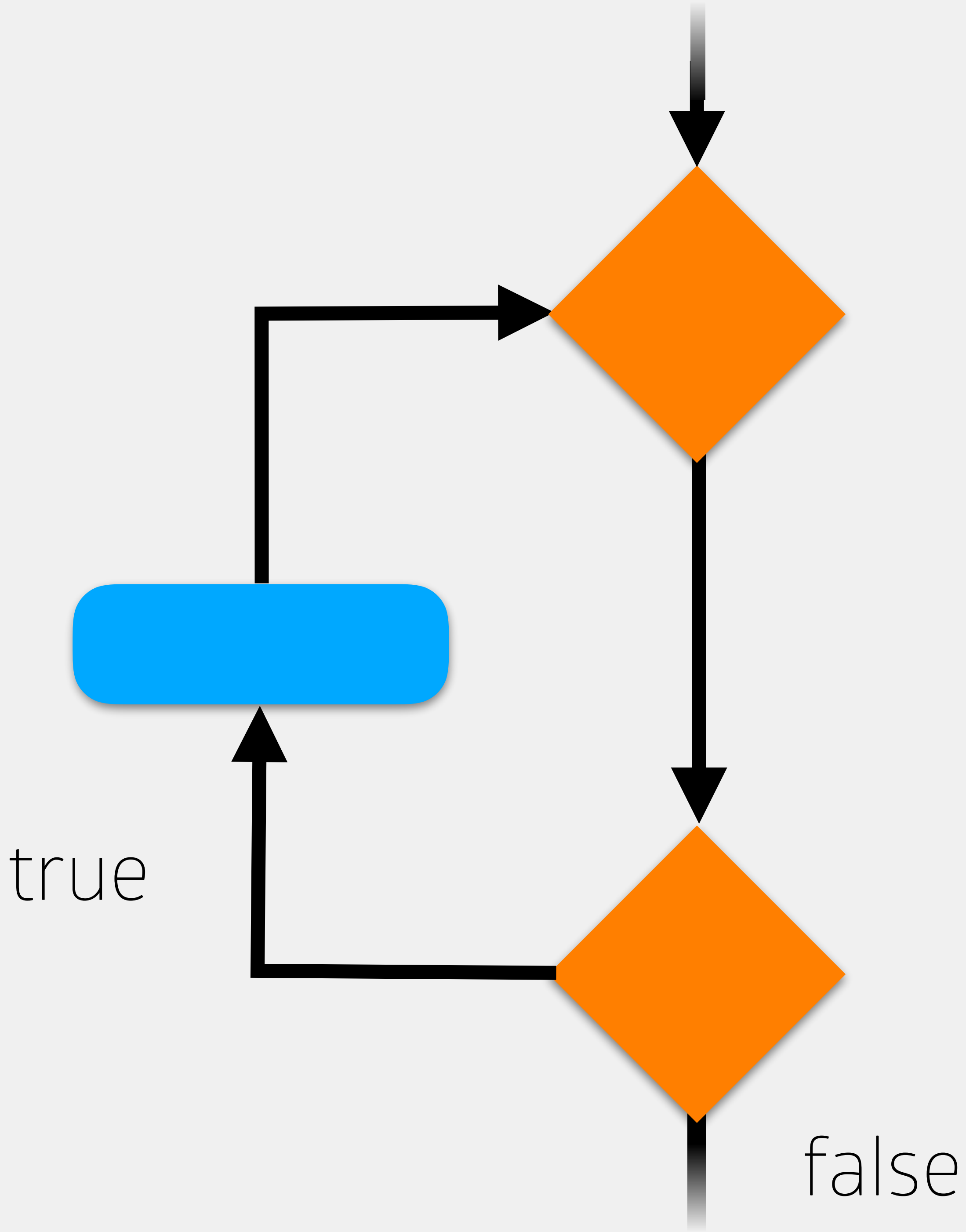
`for`
loops



`do-while`
loops

For Loops

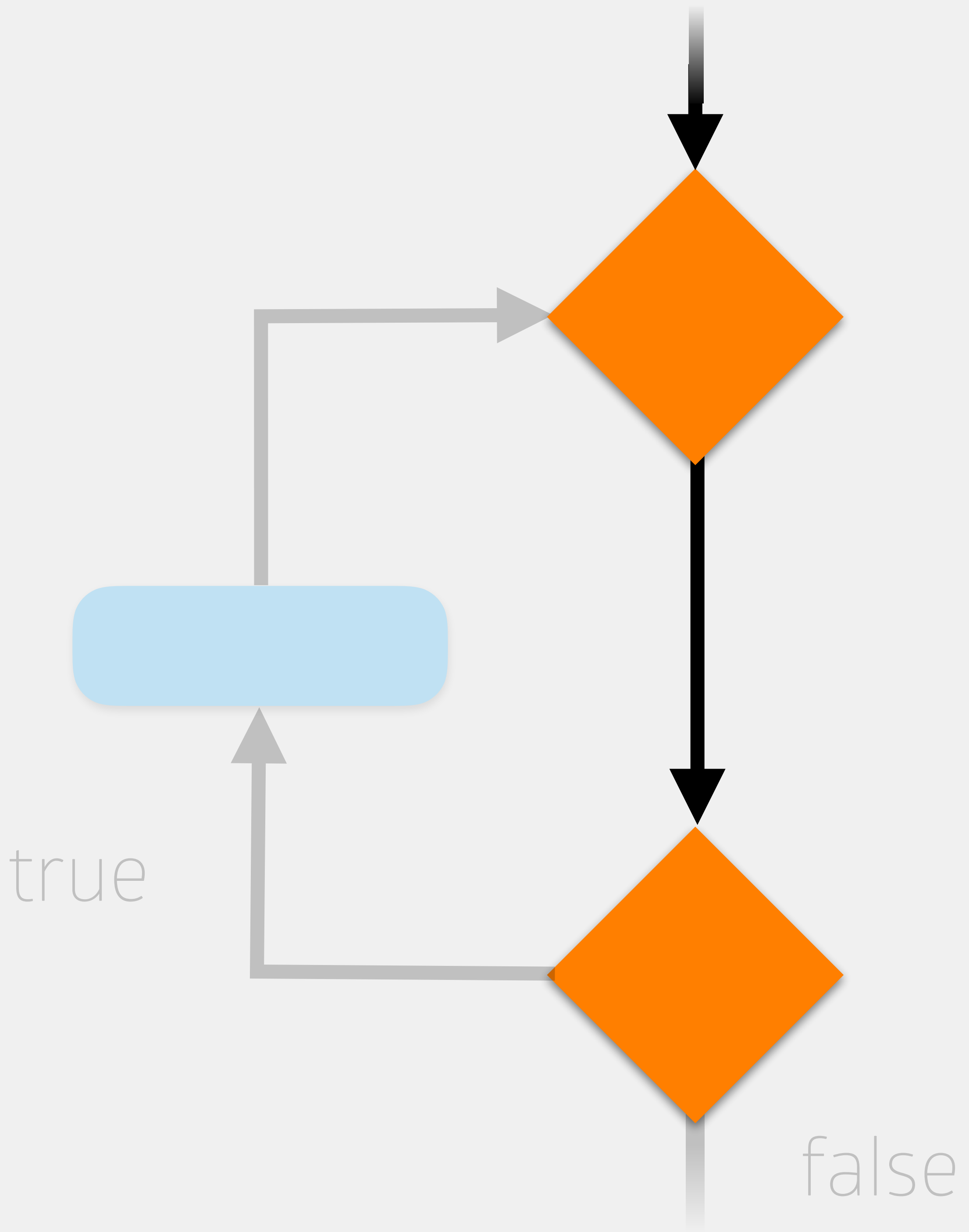
Explicitly recognizes the four parts of a loop in a single structure



```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

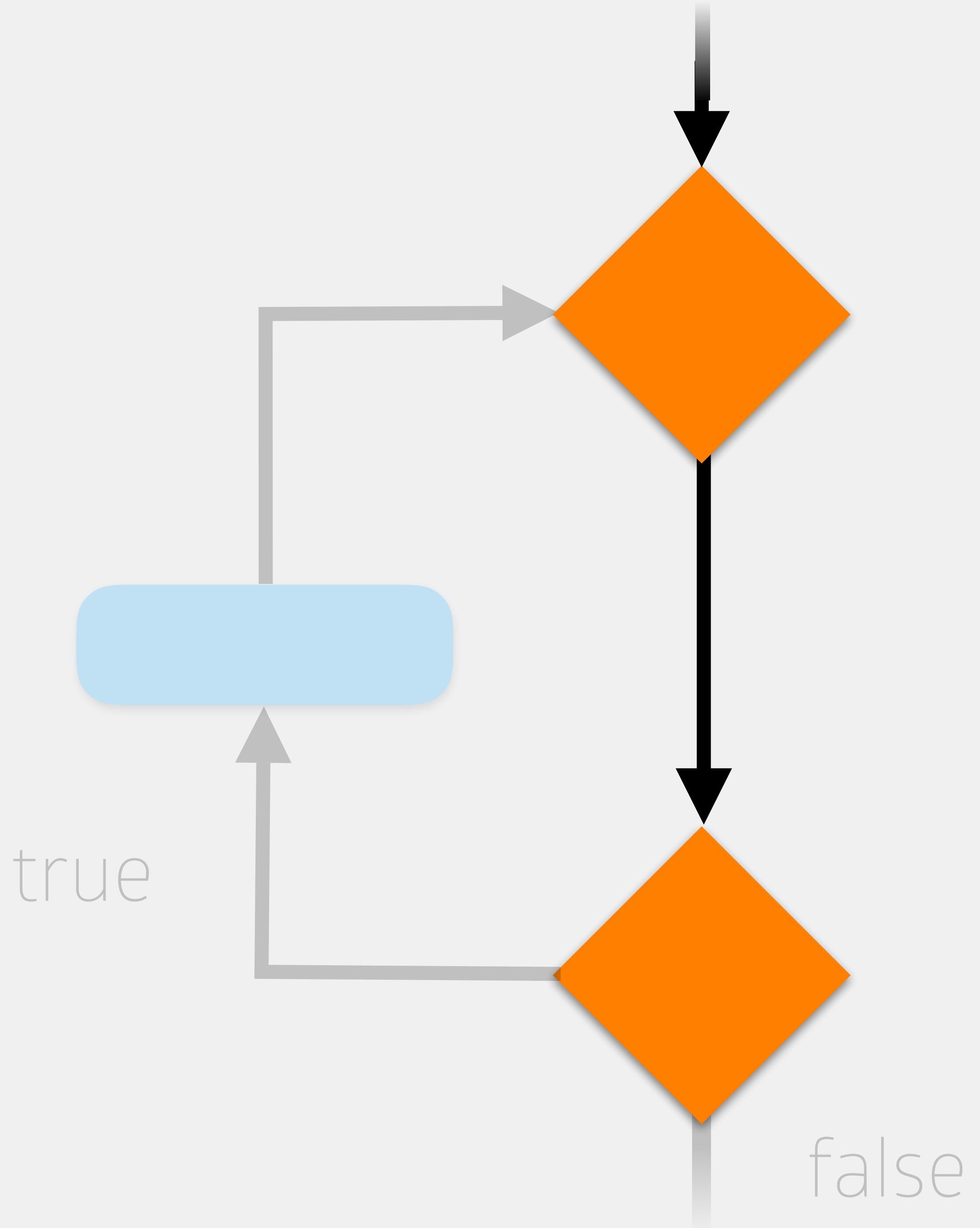


0 **only done once!**

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

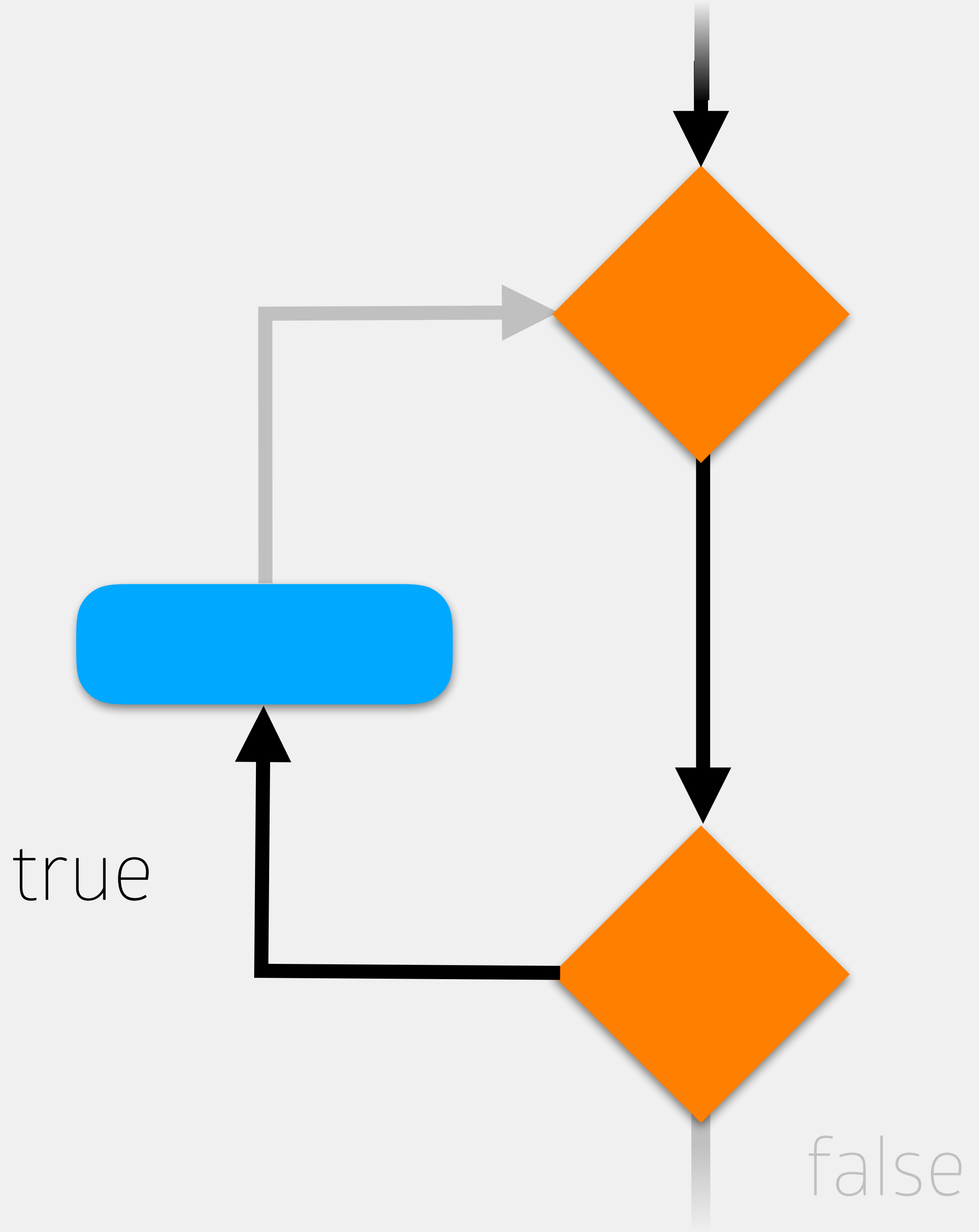


1 is this true?

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

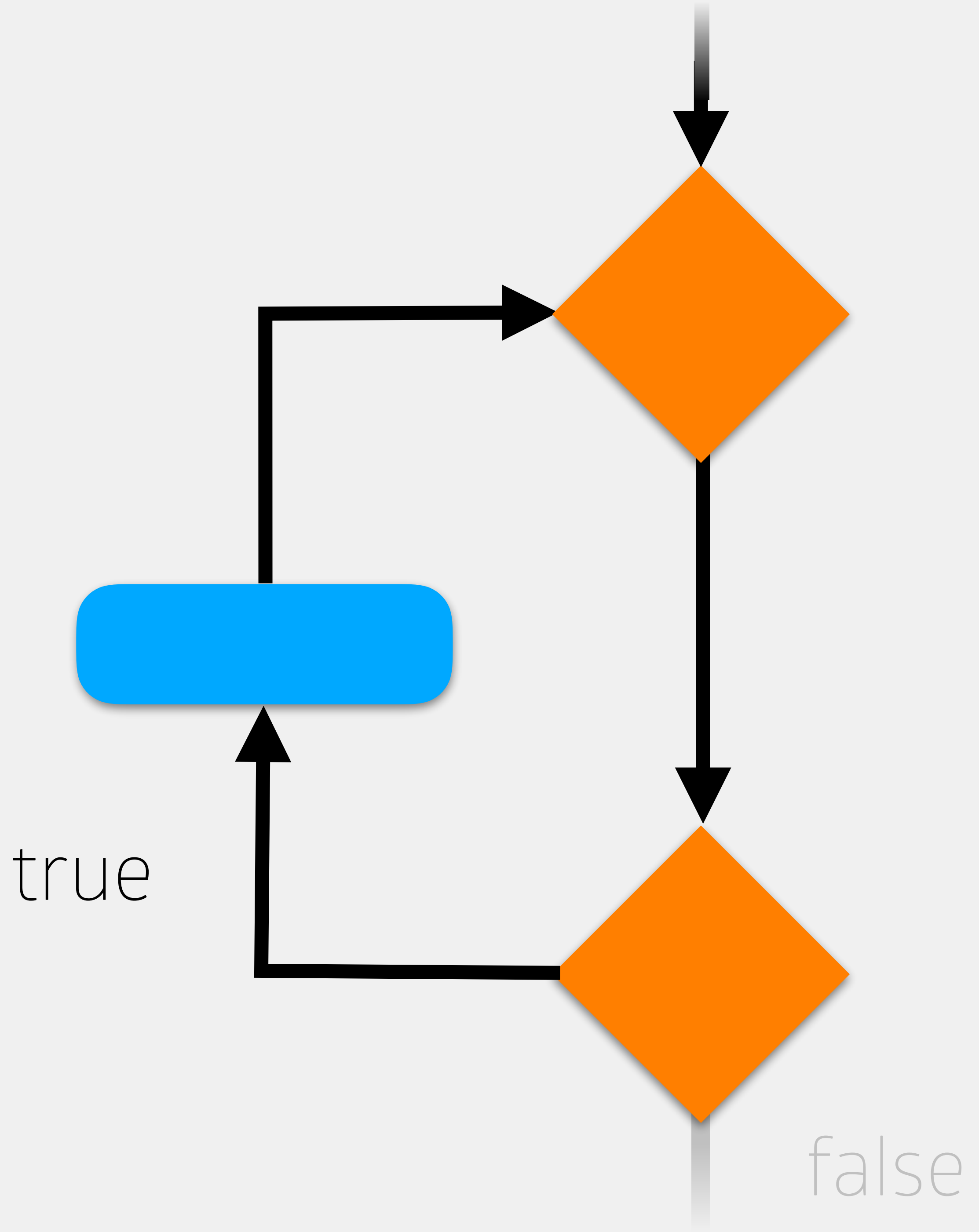
Explicitly recognizes the four parts of a loop in a single structure



```
for (<var init>; <boolean expr>; <progress>) {
2 //code to execute if boolean expression is true
}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

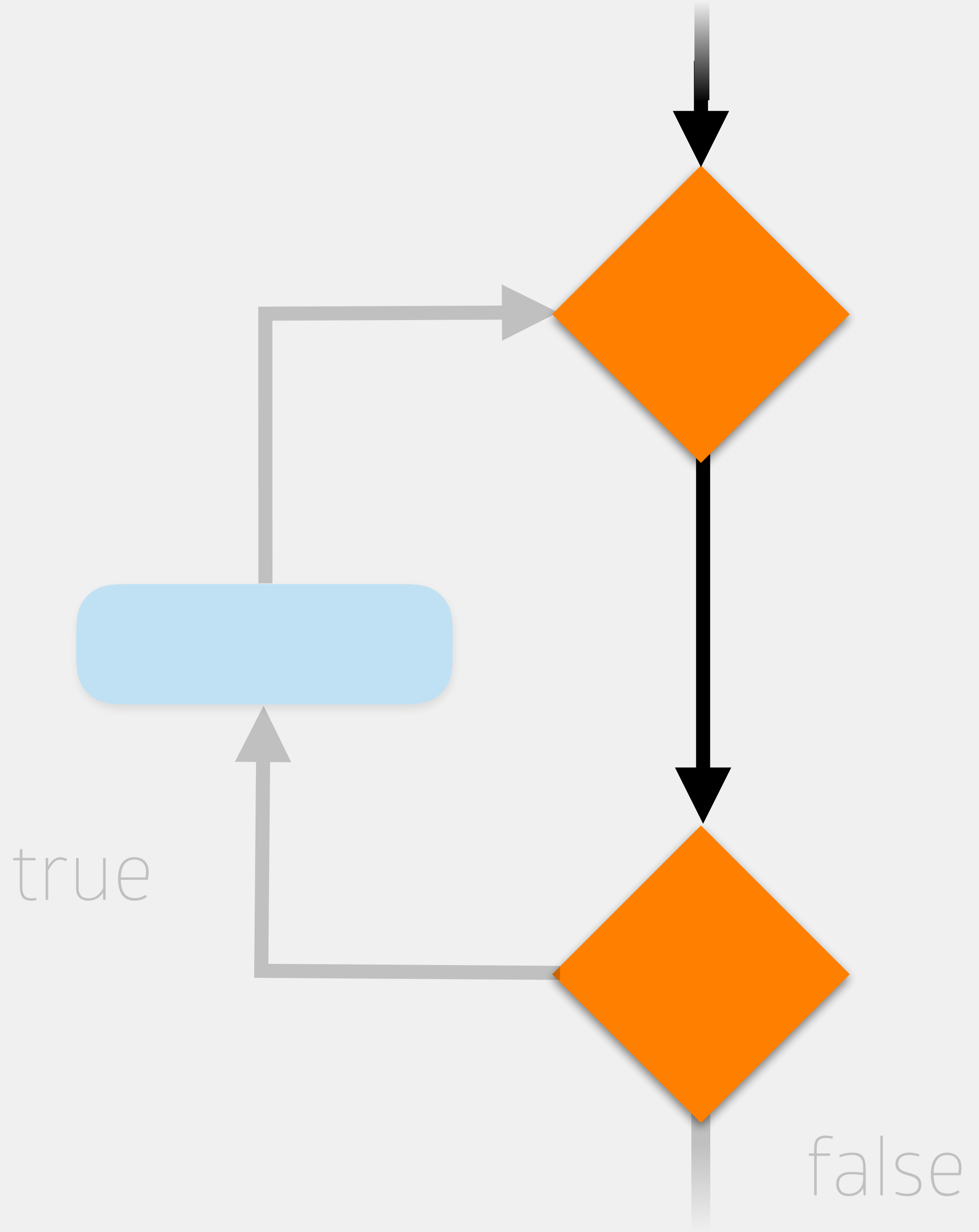


3

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

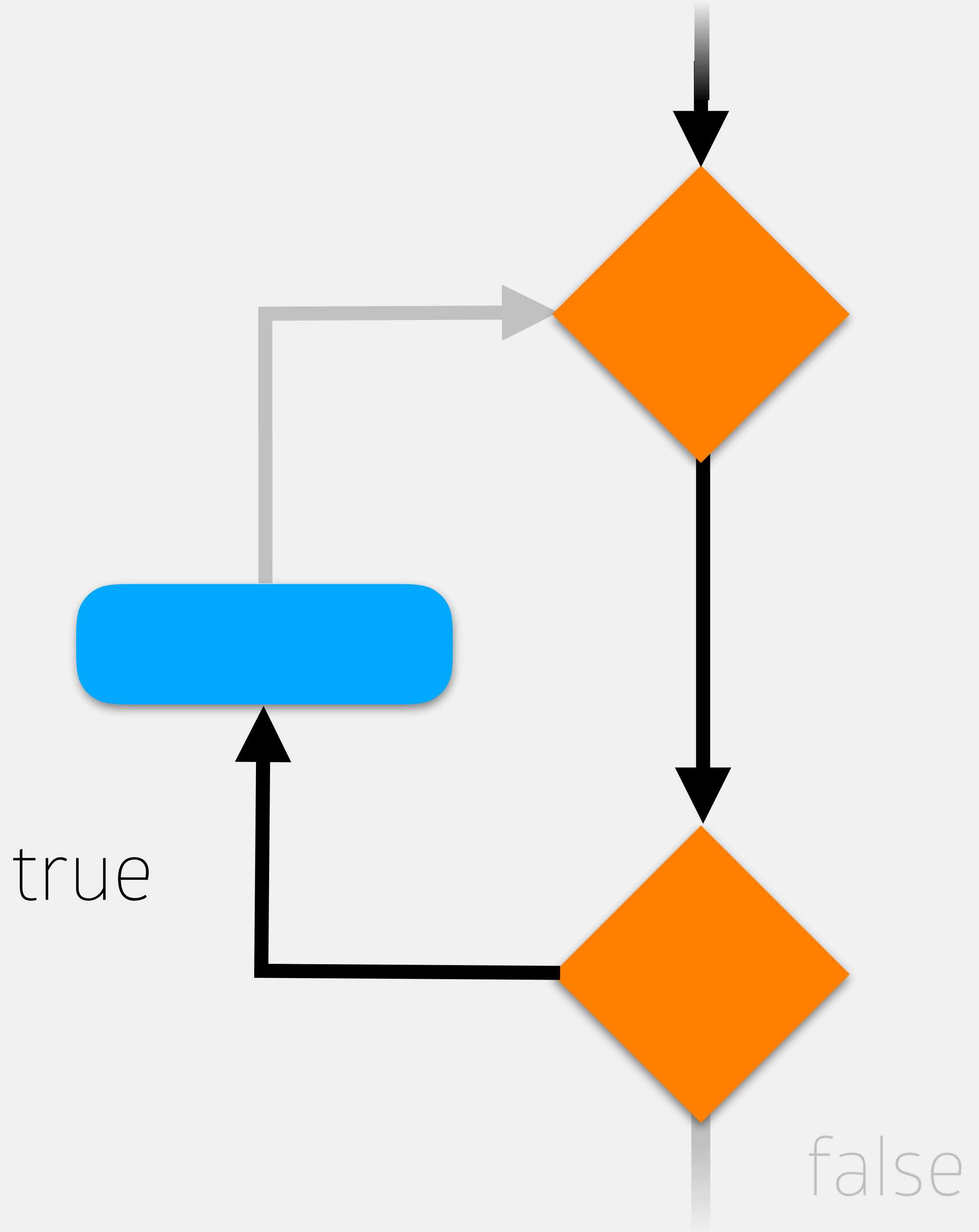


1 is this true?

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

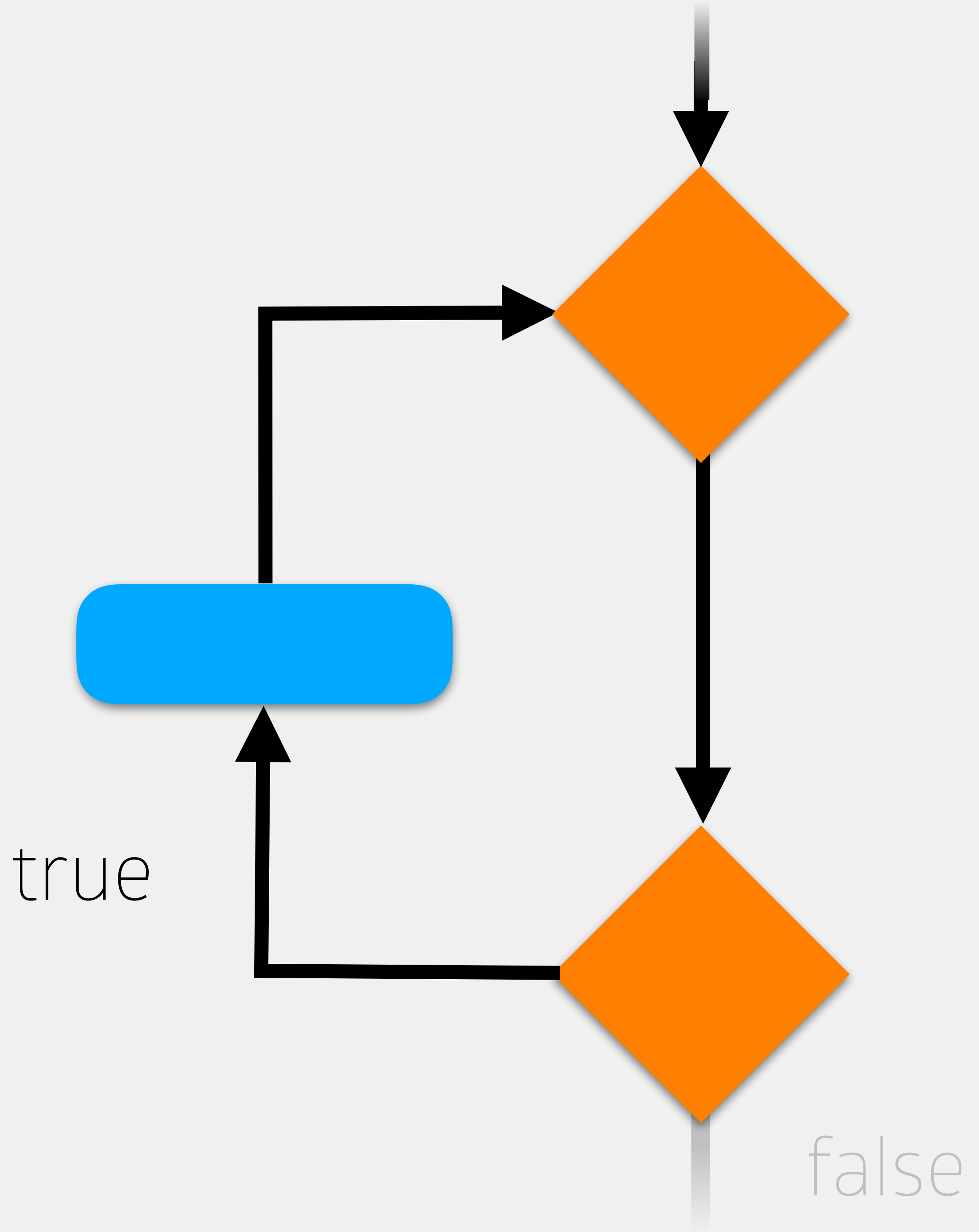
Explicitly recognizes the four parts of a loop in a single structure



```
for (<var init>; <boolean expr>; <progress>) {
2 //code to execute if boolean expression is true
}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

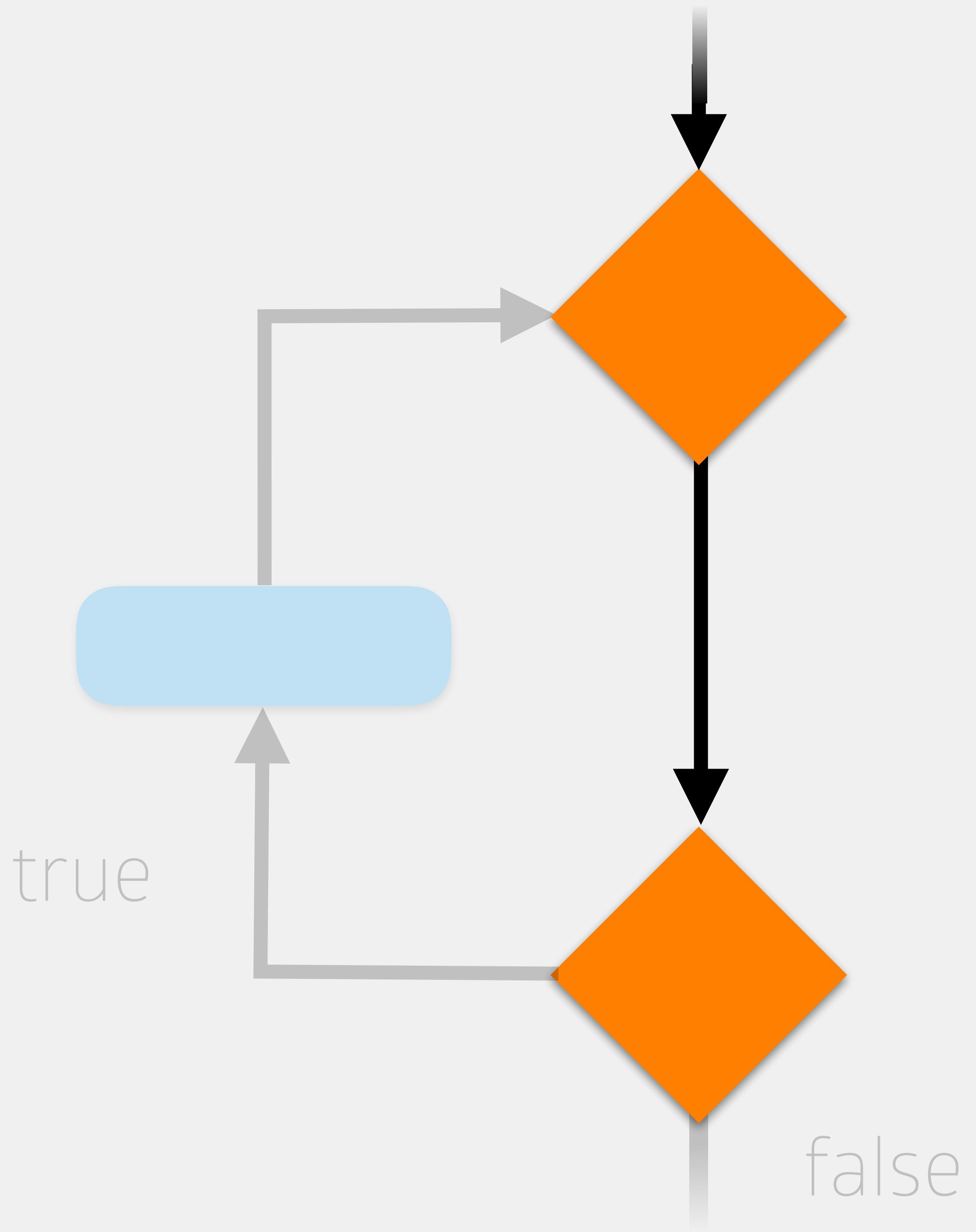


3

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

Explicitly recognizes the four parts of a loop in a single structure

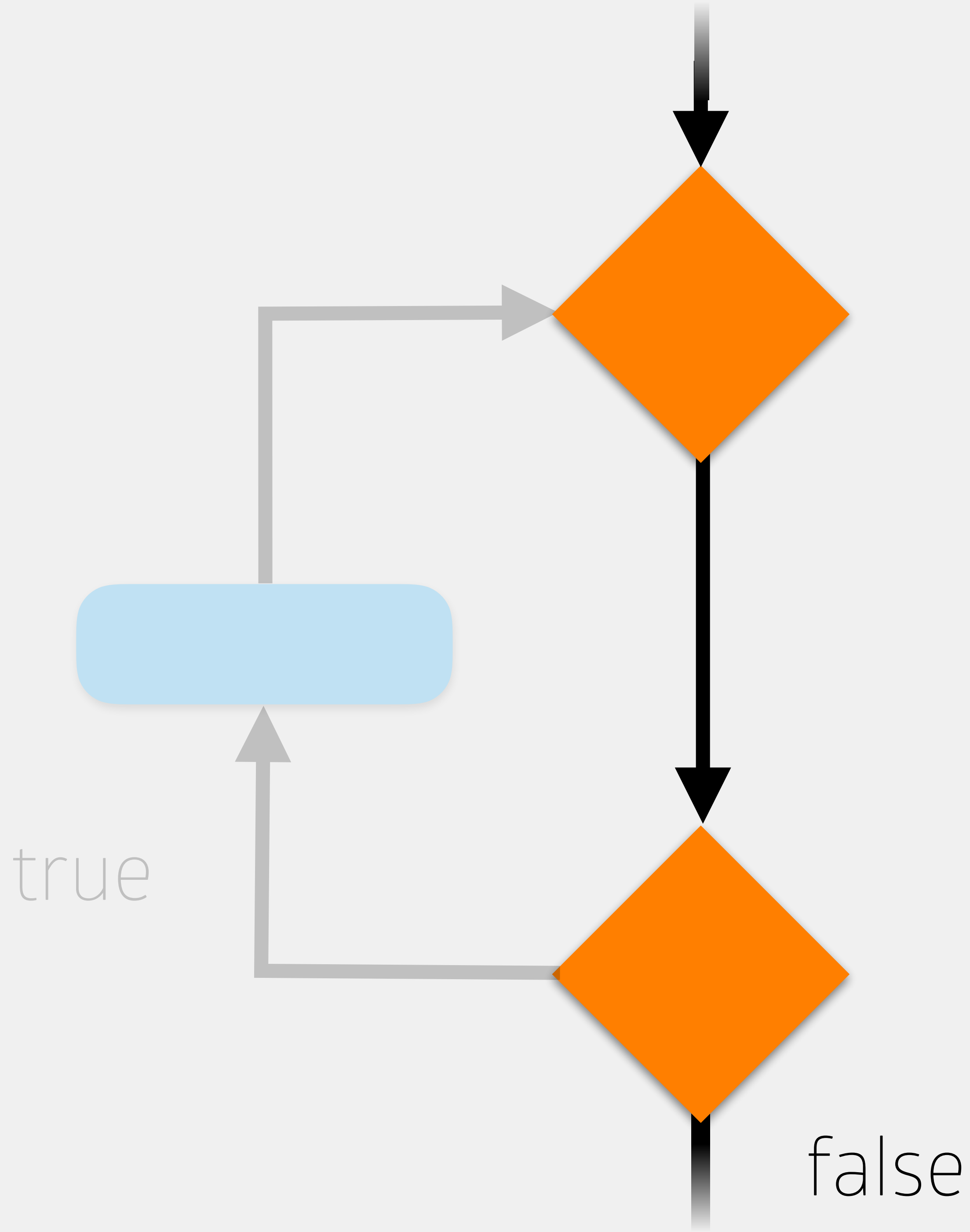


1 is this true?

```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

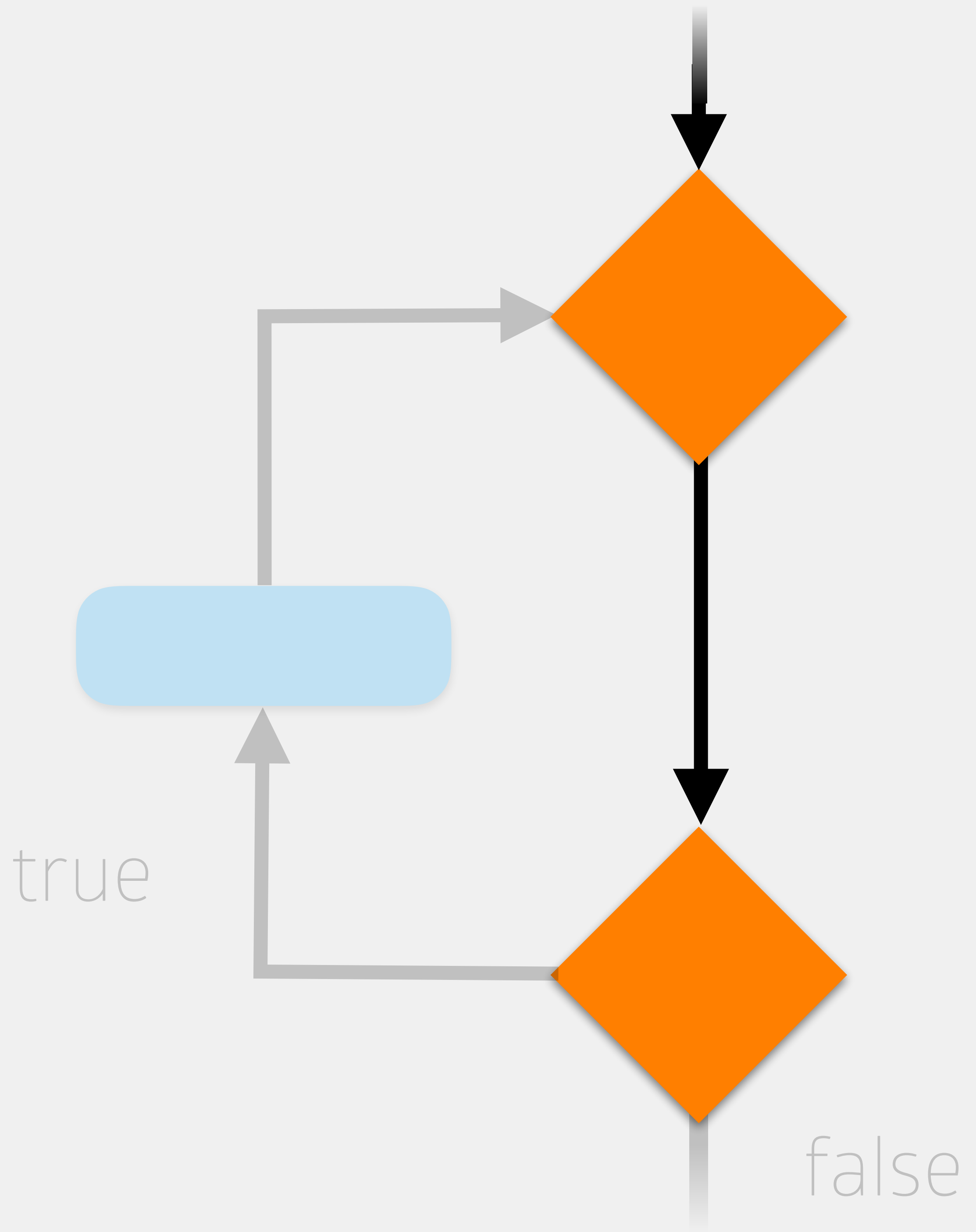
Explicitly recognizes the four parts of a loop in a single structure



```
for (<var init>; <boolean expr>; <progress>) {

    //code to execute if boolean expression is true

}

//code to execute after while loop
```

For Loops

“For i = 5, print the value of i while i is less than 8…”



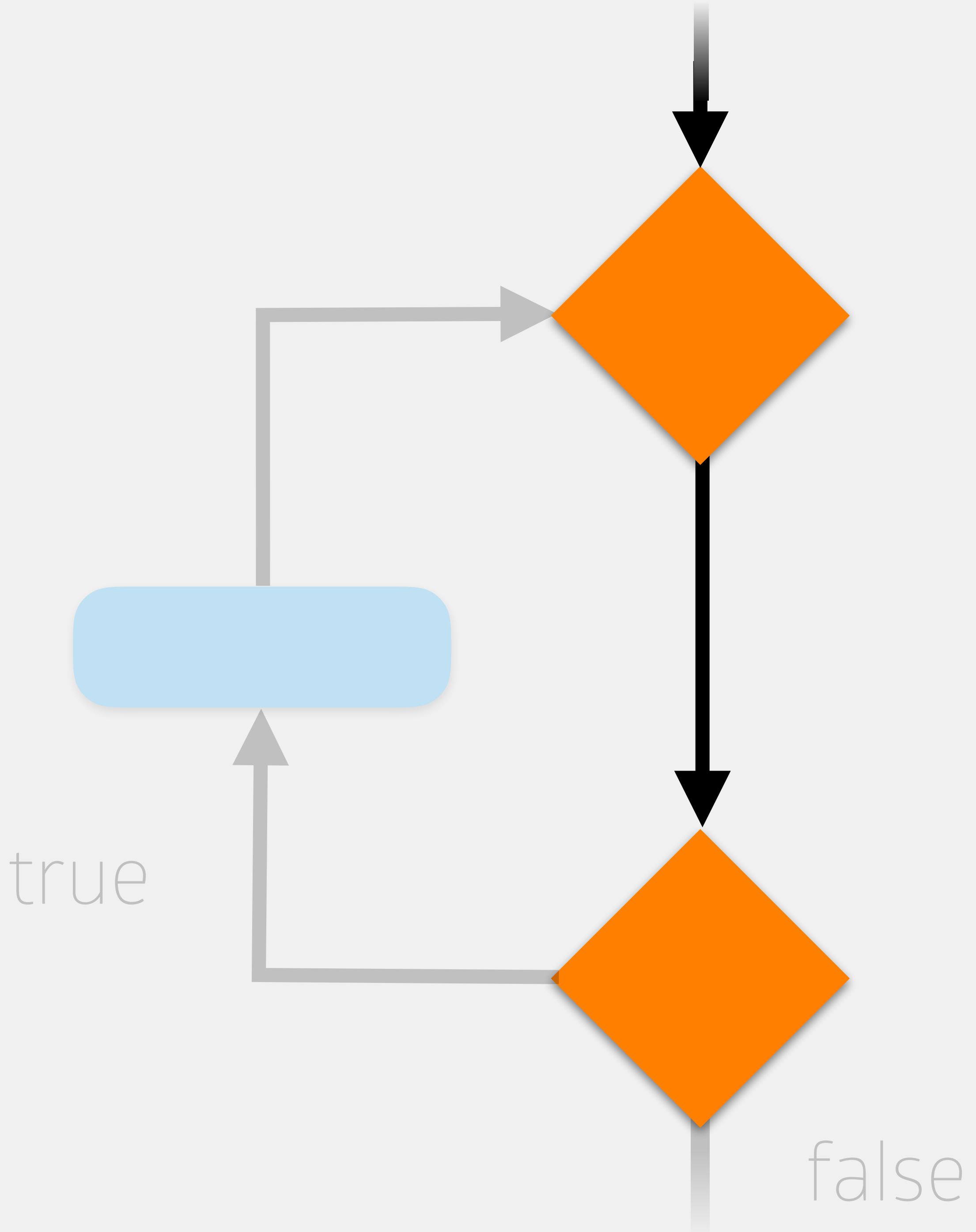
```
> for (int i = 5; i < 8; i++) {
      System.out.println(i);
  }

  //code to execute after while loop
```

memory

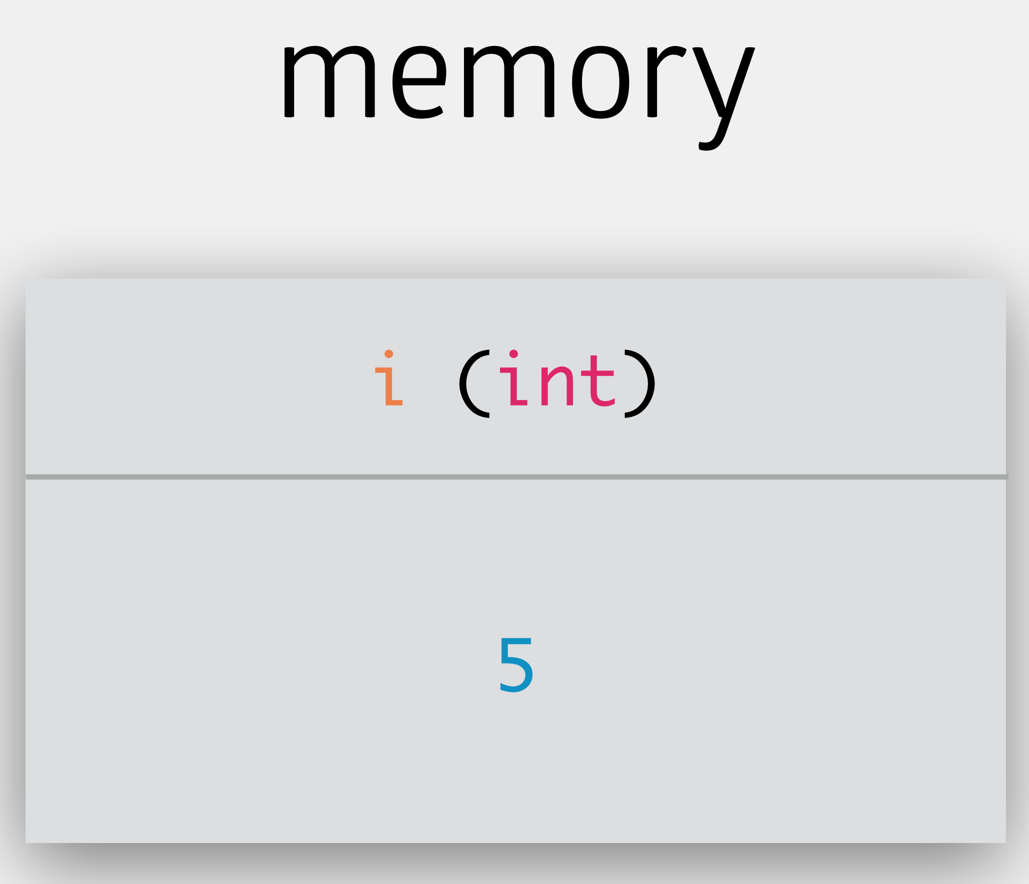
i (int)

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
> for   int i = 5


  }


  //code to execute after while loop
```

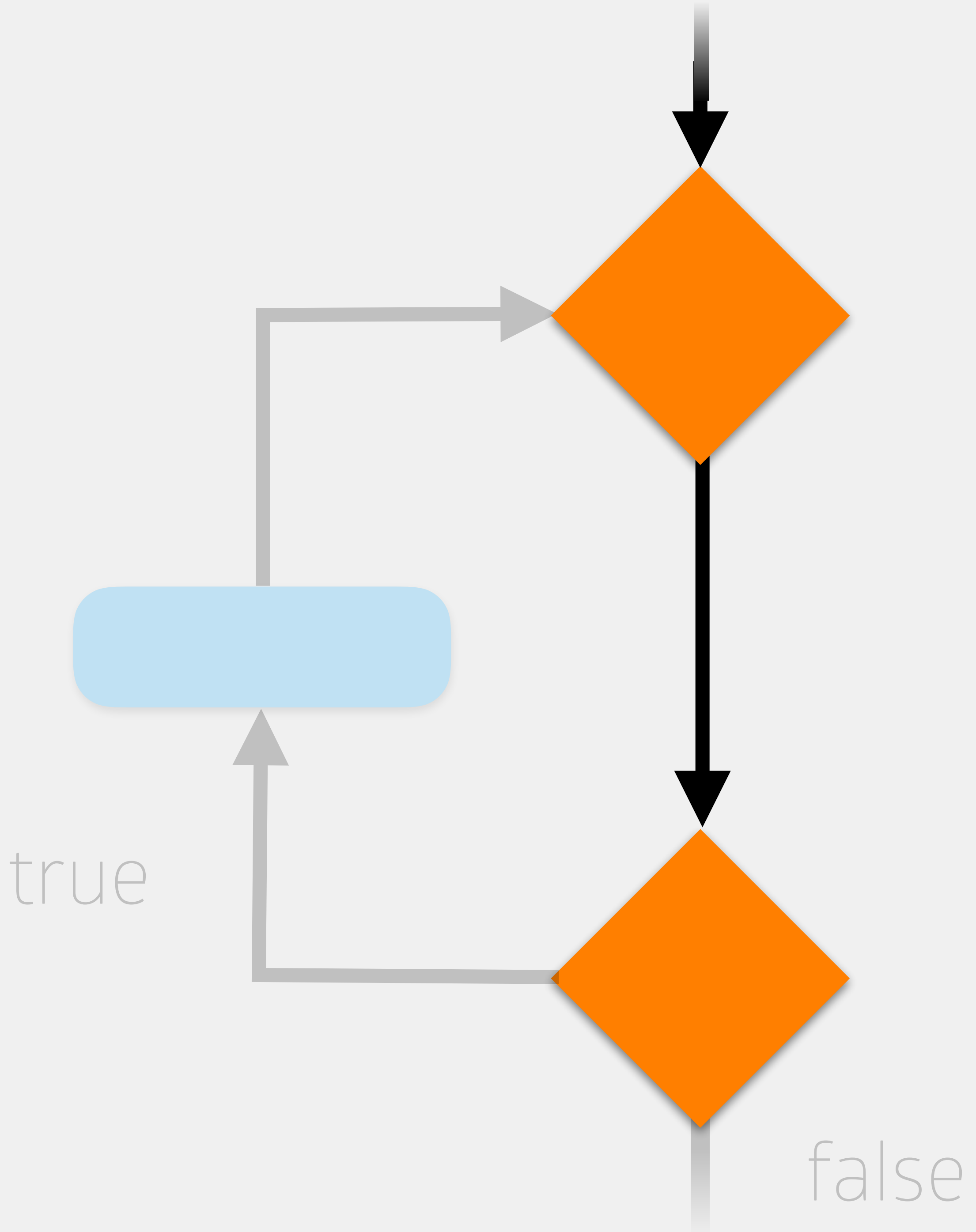

memory

i (int)

5

For Loops

“For i = 5, print the value of i while i is less than 8…”



is this true?

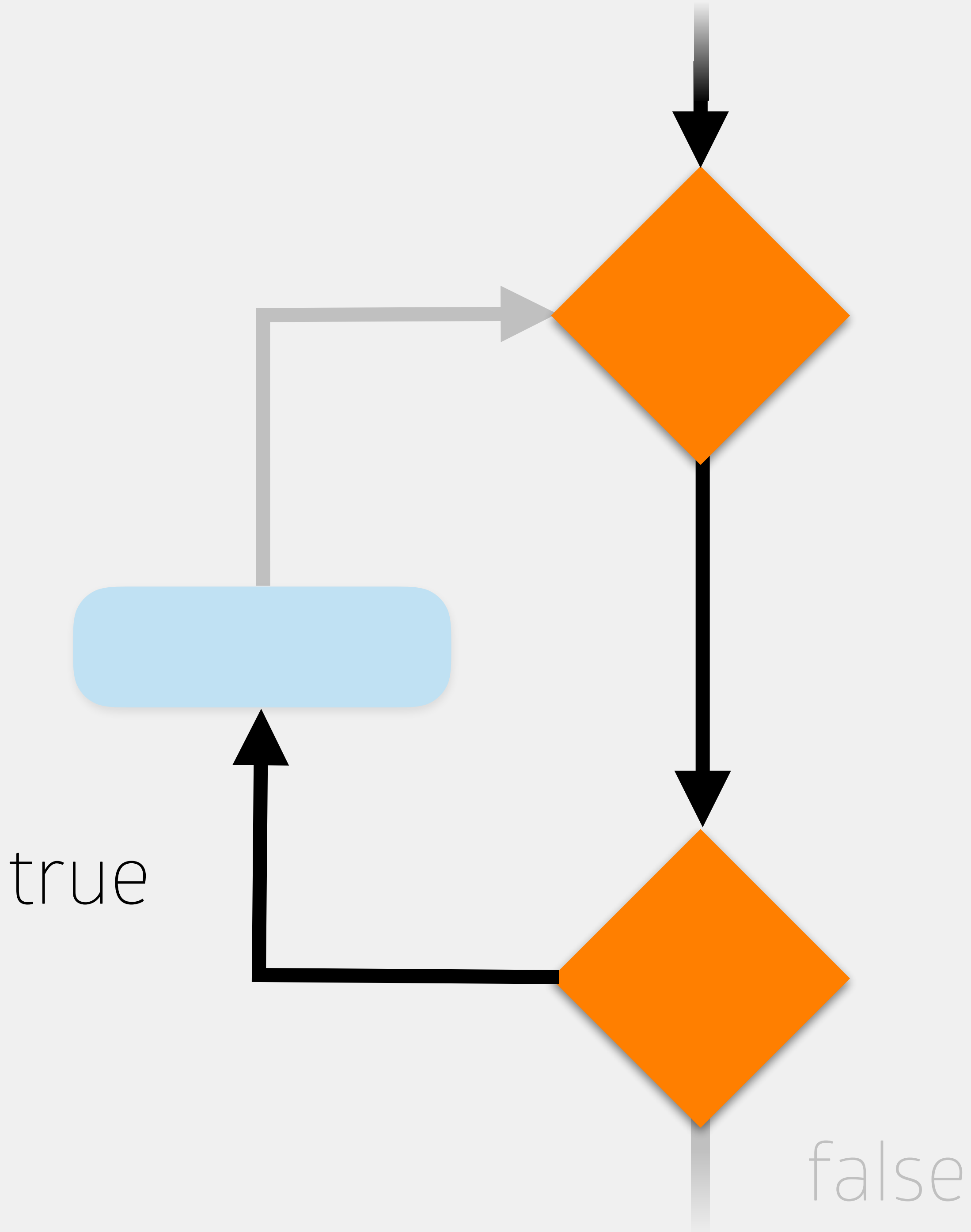
```
> for                  i < 8



  }



  //code to execute after while loop
```

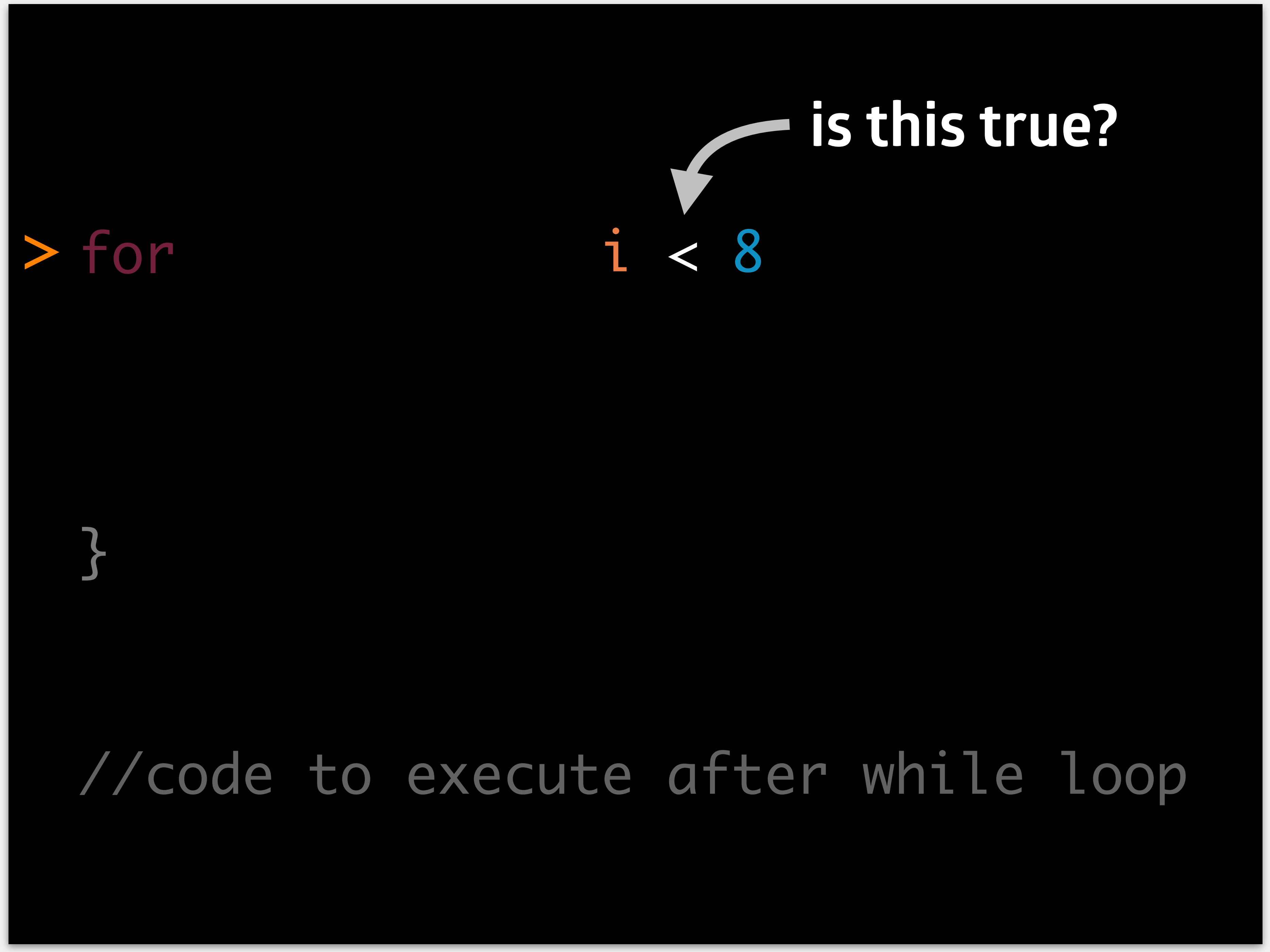
memory

i (int)
5

For Loops

“For i = 5, print the value of i while i is less than 8…”





is this true?

```
> for                  i < 8



  }



  //code to execute after while loop
```

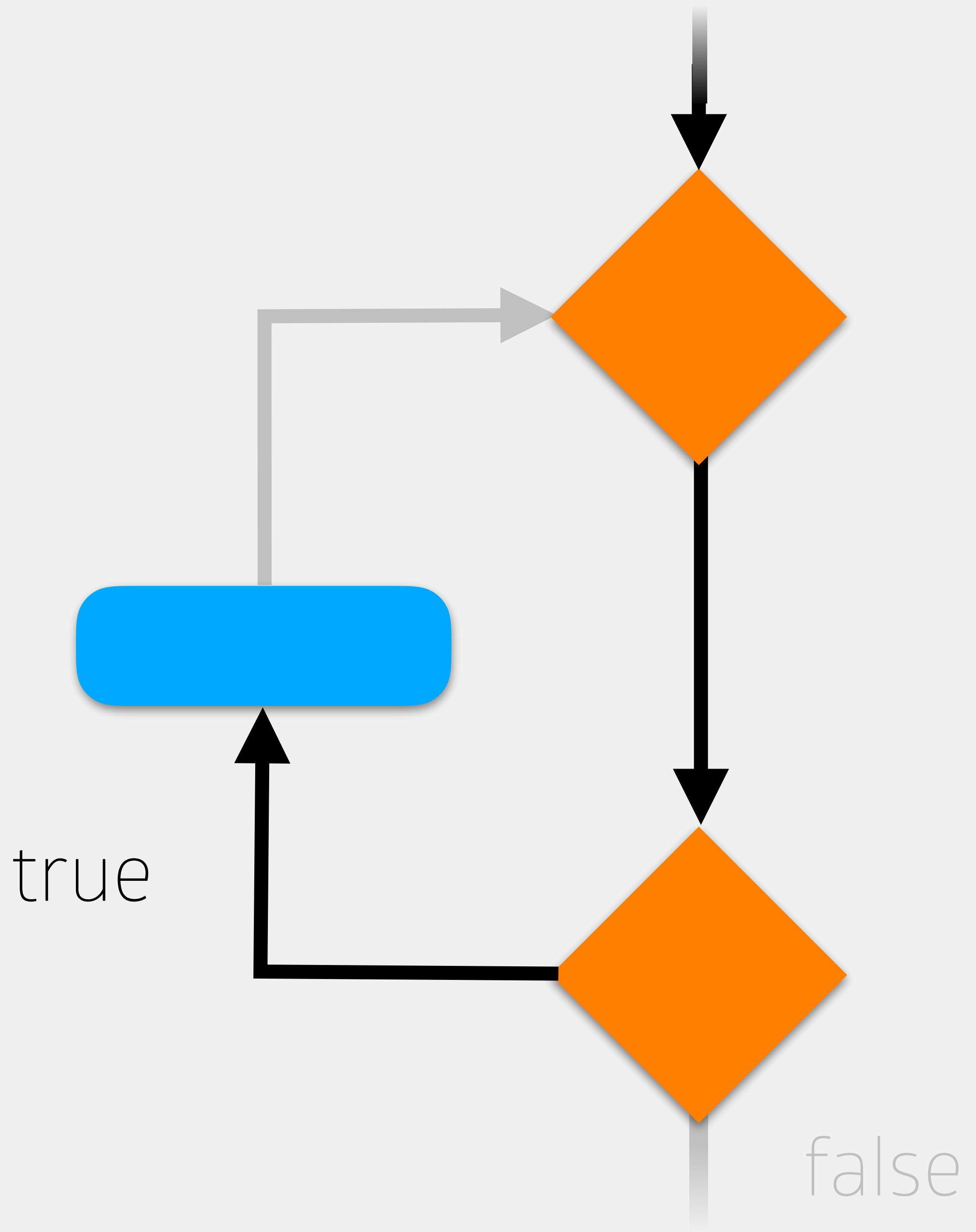
memory

i (int)

5

For Loops

“For i = 5, print the value of i while i is less than 8…”



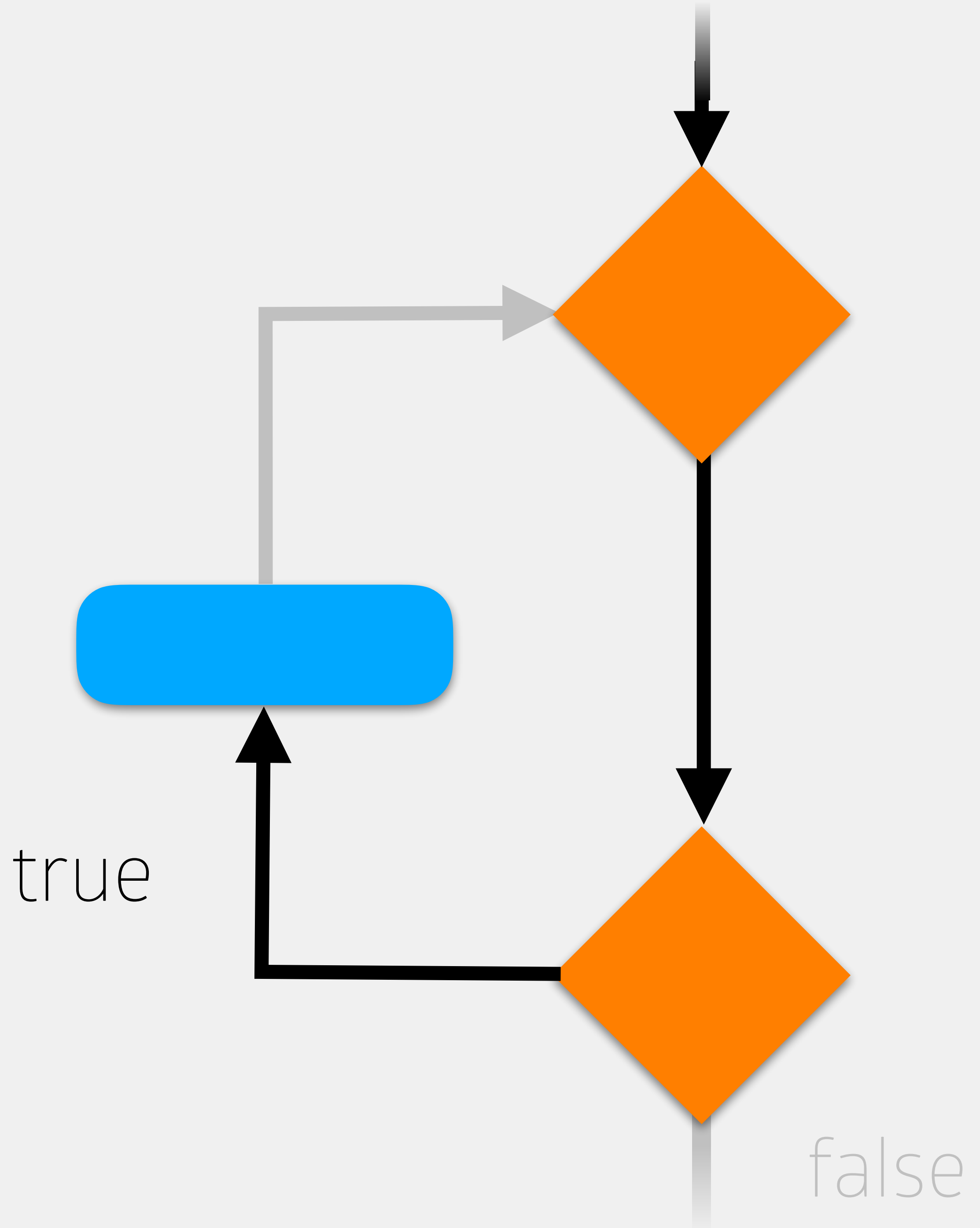
```
for (int i = 5; i < 8; i++) {
   > System.out.println(i);
}


//code to execute after while loop
```

memory

i (int)
5

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {

    System.out.println(i);
  >
}


//code to execute after while loop
```

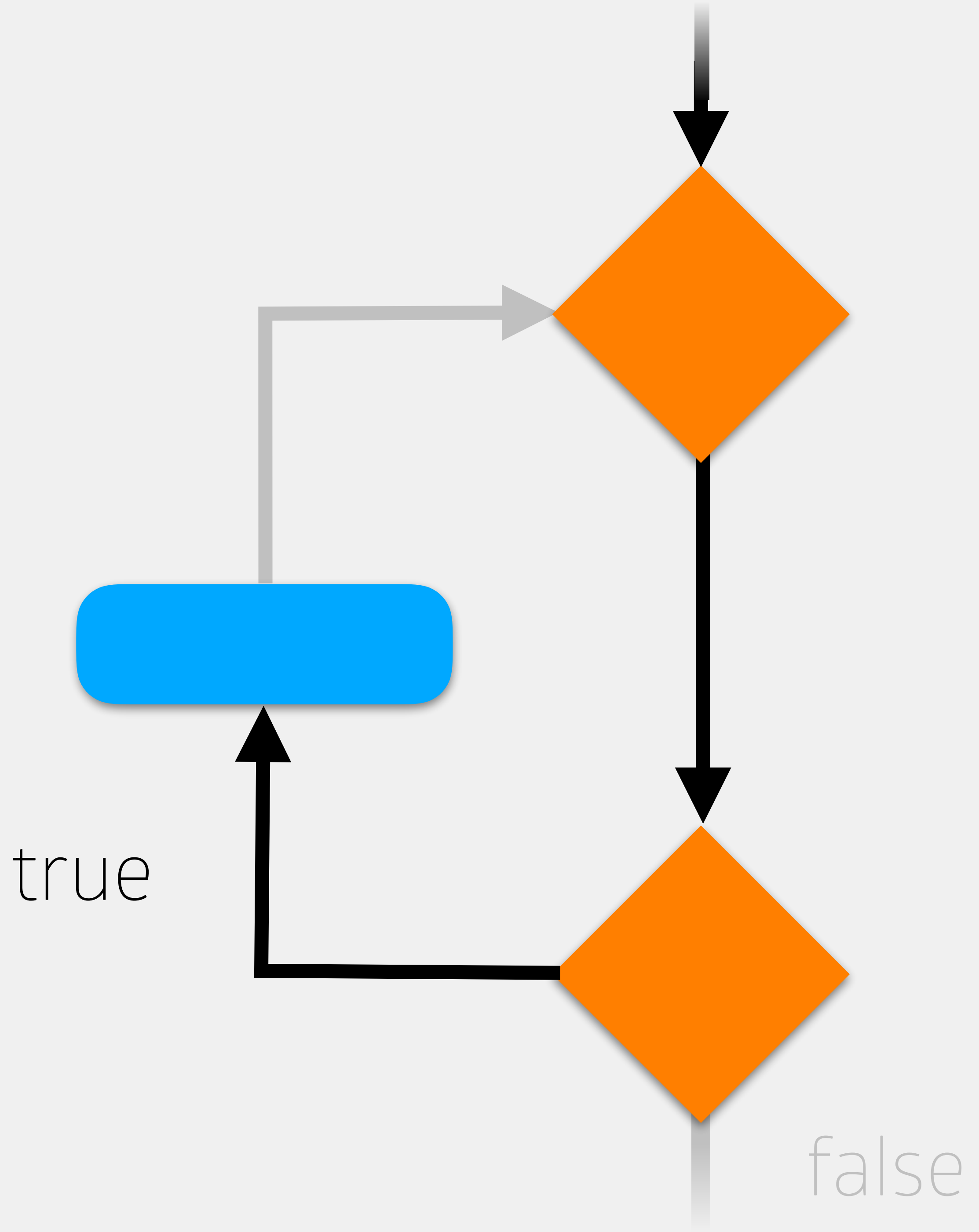
```
5
```

memory

i (int)
5

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
> for                         i++


  }


  //code to execute after while loop
```

```
5
```

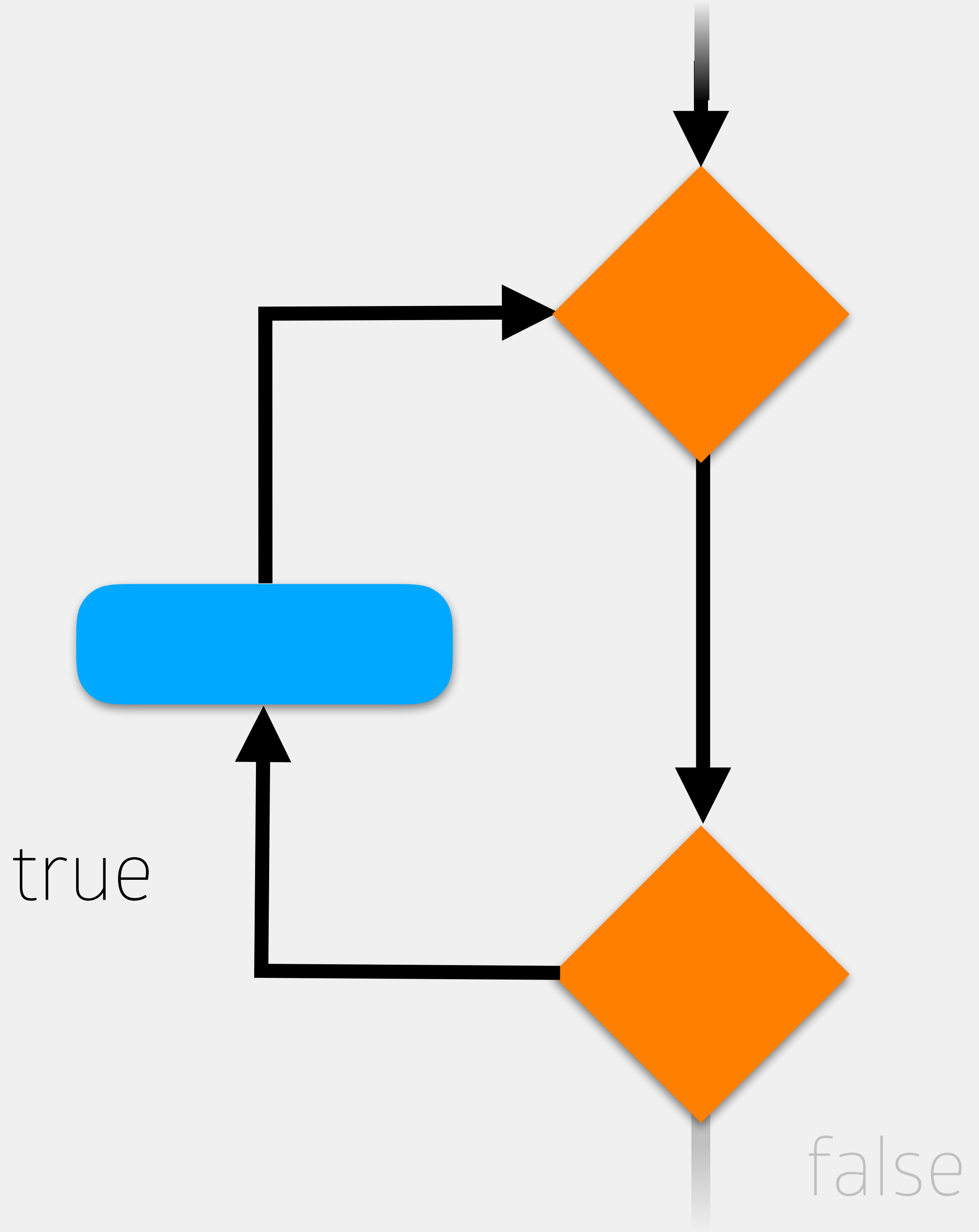
memory

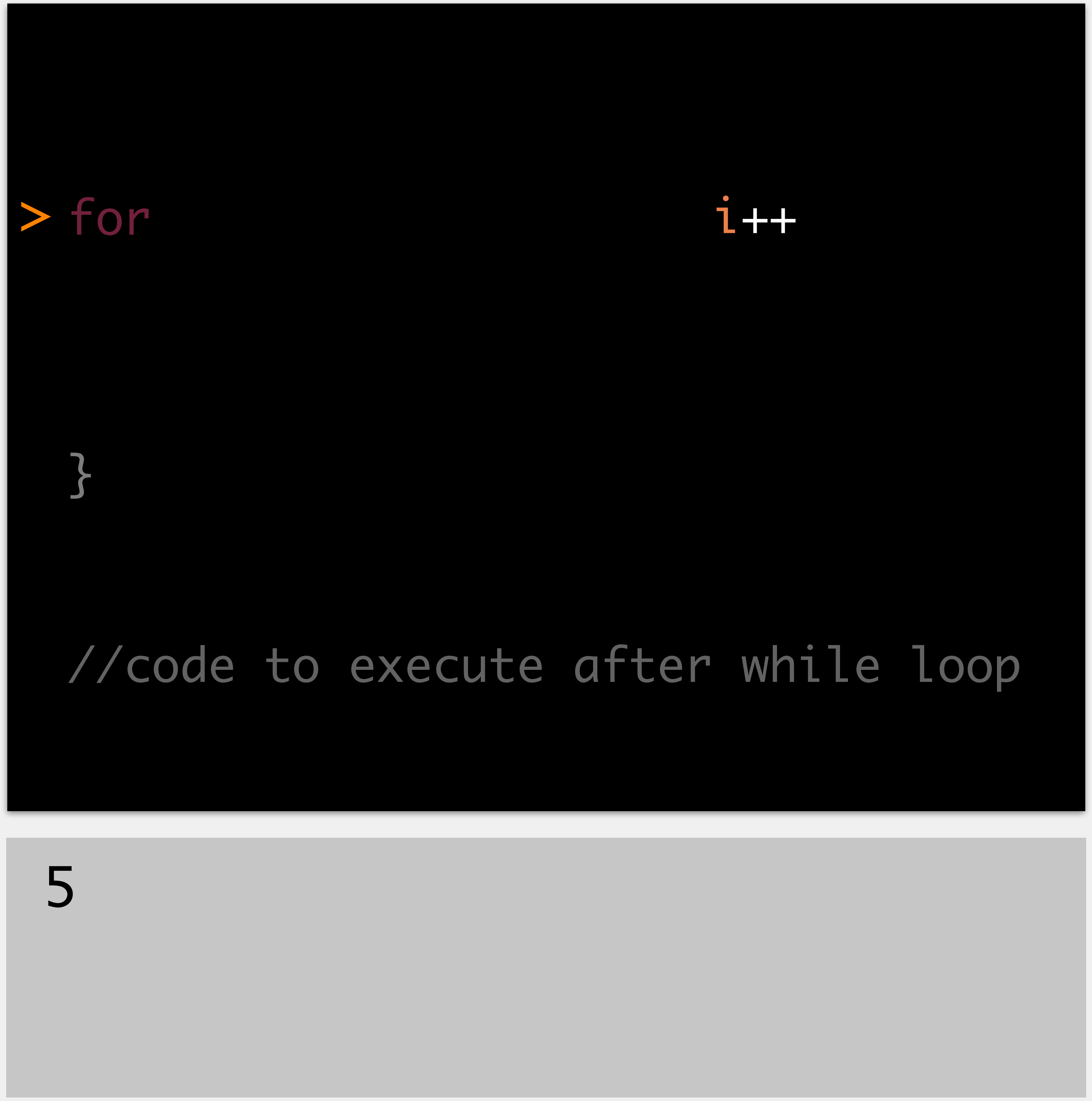
i (int)

5

For Loops

“For i = 5, print the value of i while i is less than 8…”





```
> for                         i++

  }

  //code to execute after while loop
```

```
5
```

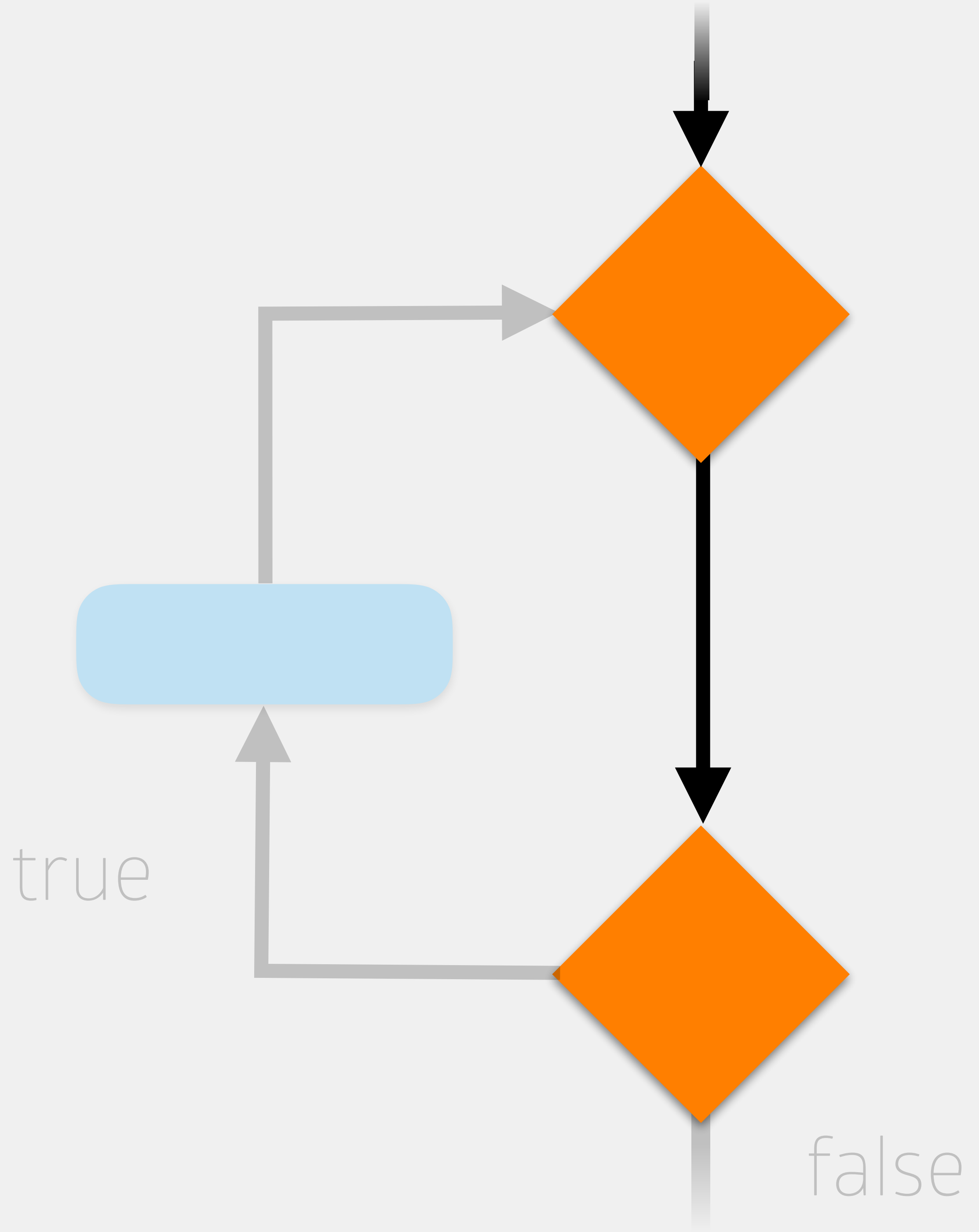
memory

i (int)

6

For Loops

“For i = 5, print the value of i while i is less than 8...”



is this true?

```
> for                  i < 8


  }


  //code to execute after while loop
```

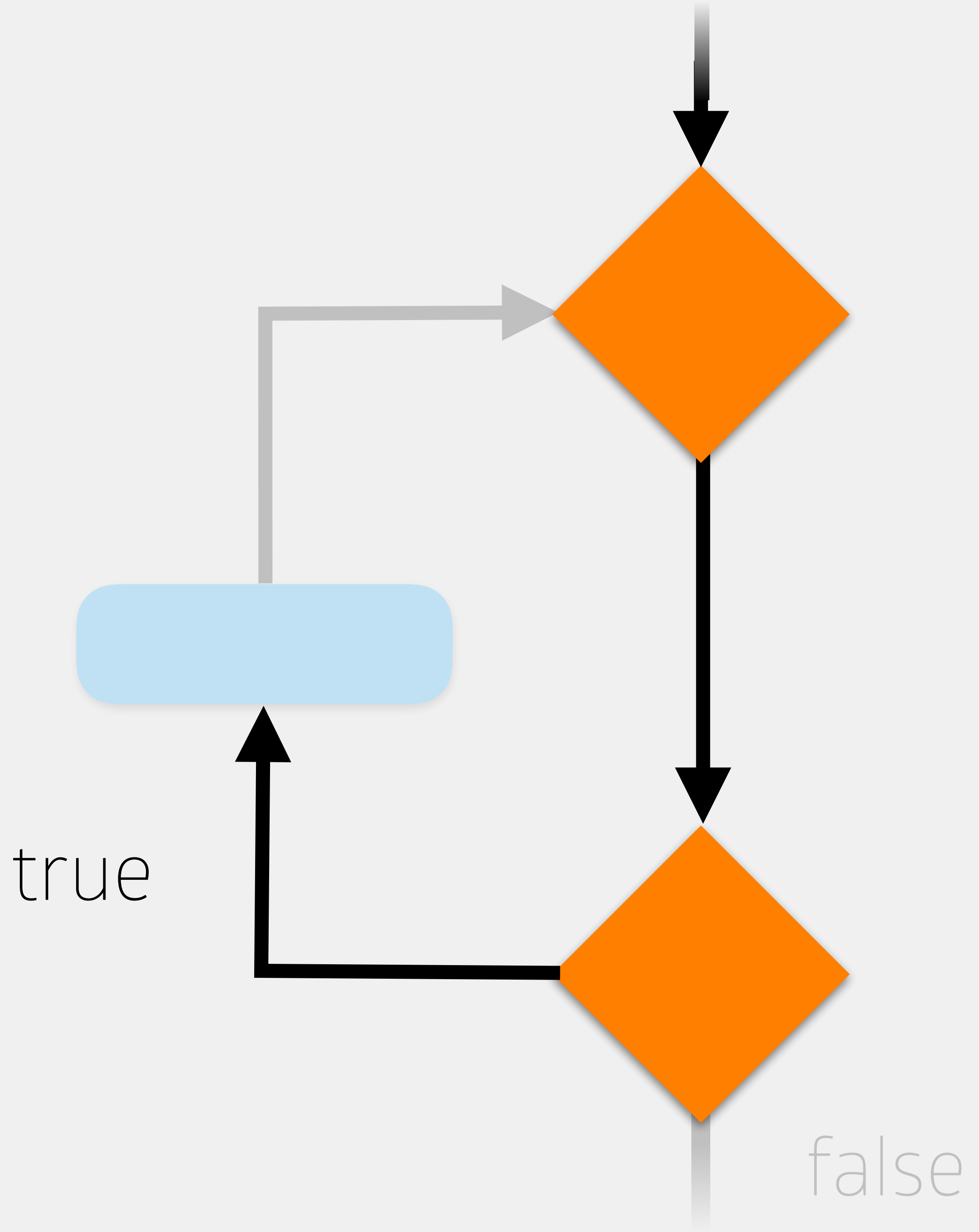
```
5
```

memory

i (int)
6

For Loops

“For i = 5, print the value of i while i is less than 8...”



is this true?

```
> for             i < 8

  }

  //code to execute after while loop
```

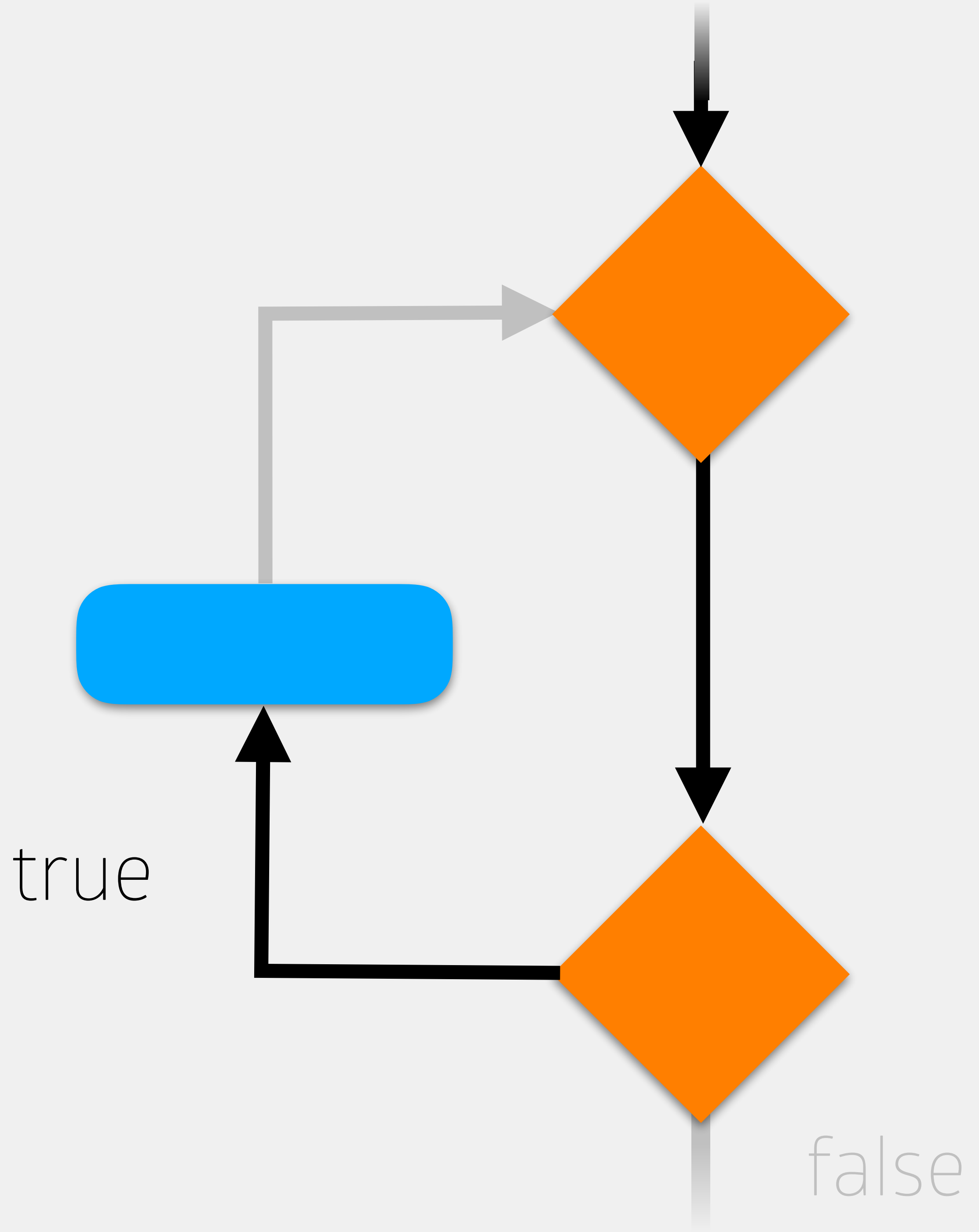
```
5
```

memory

i (int)
6

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {
   > System.out.println(i);
}

//code to execute after while loop
```

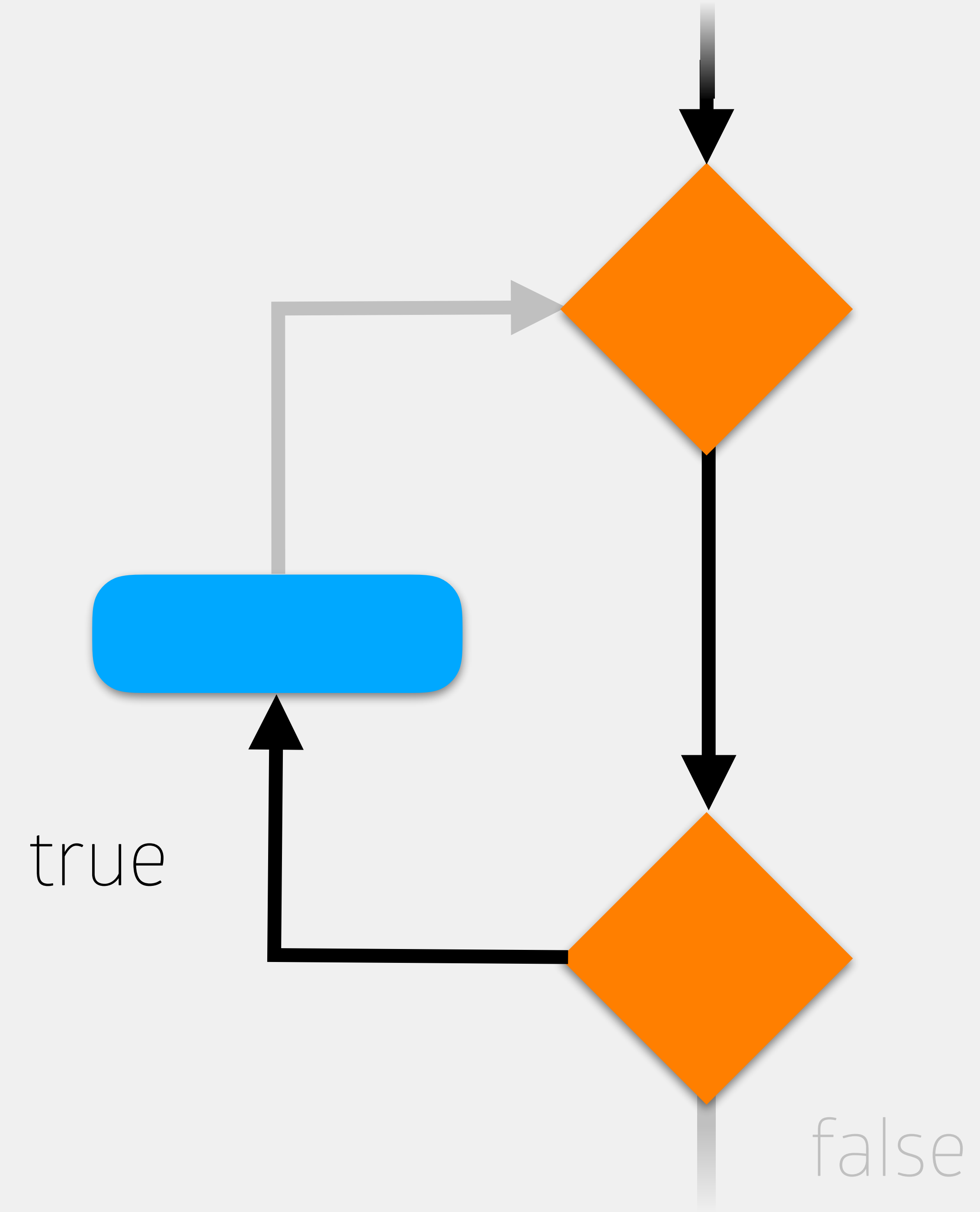
```
5
```

memory

i (int)
6

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {

    System.out.println(i);
  >
}


//code to execute after while loop
```

```
5
6
```

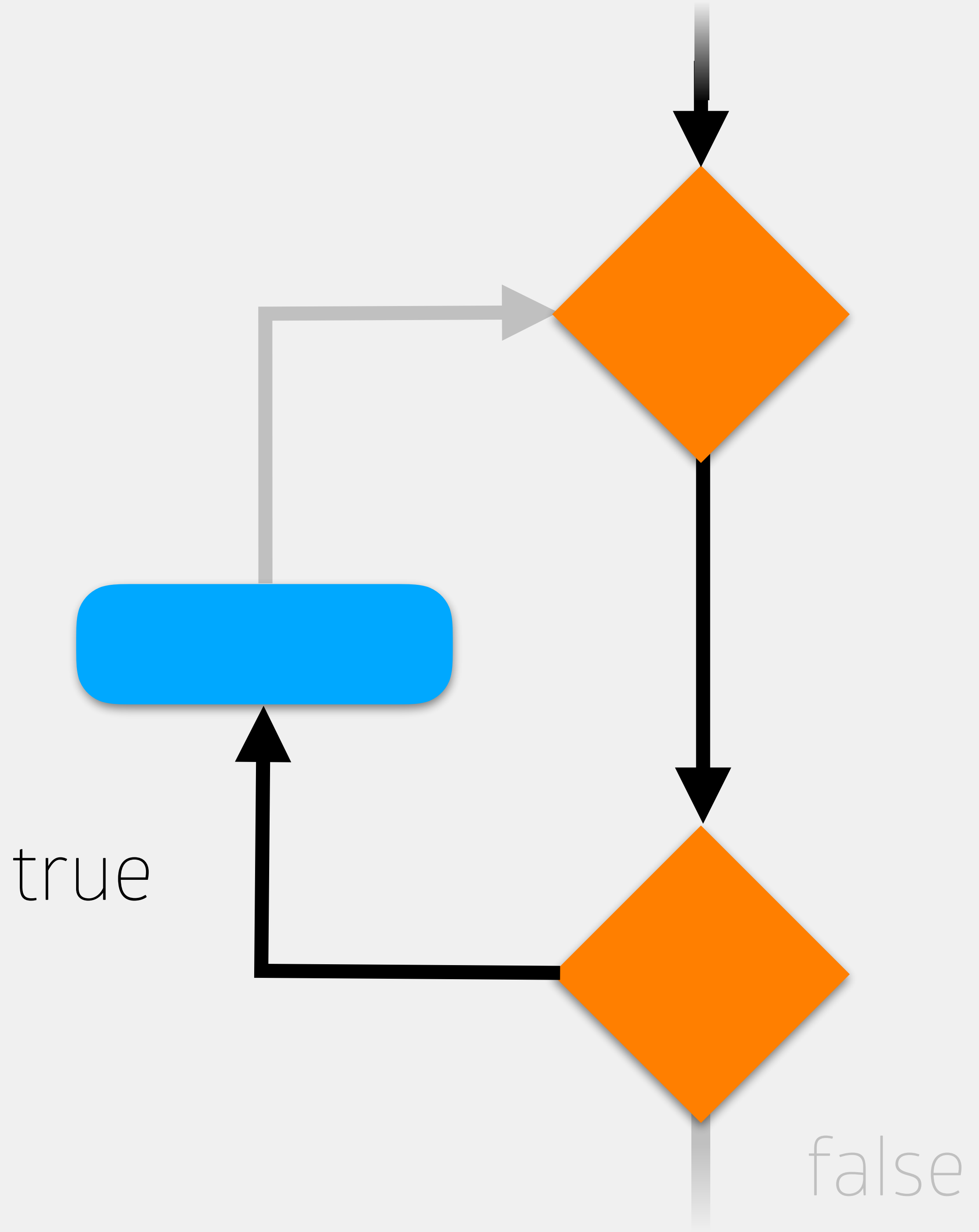
memory

i (int)

6

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
> for                         i++

  }

  //code to execute after while loop
```

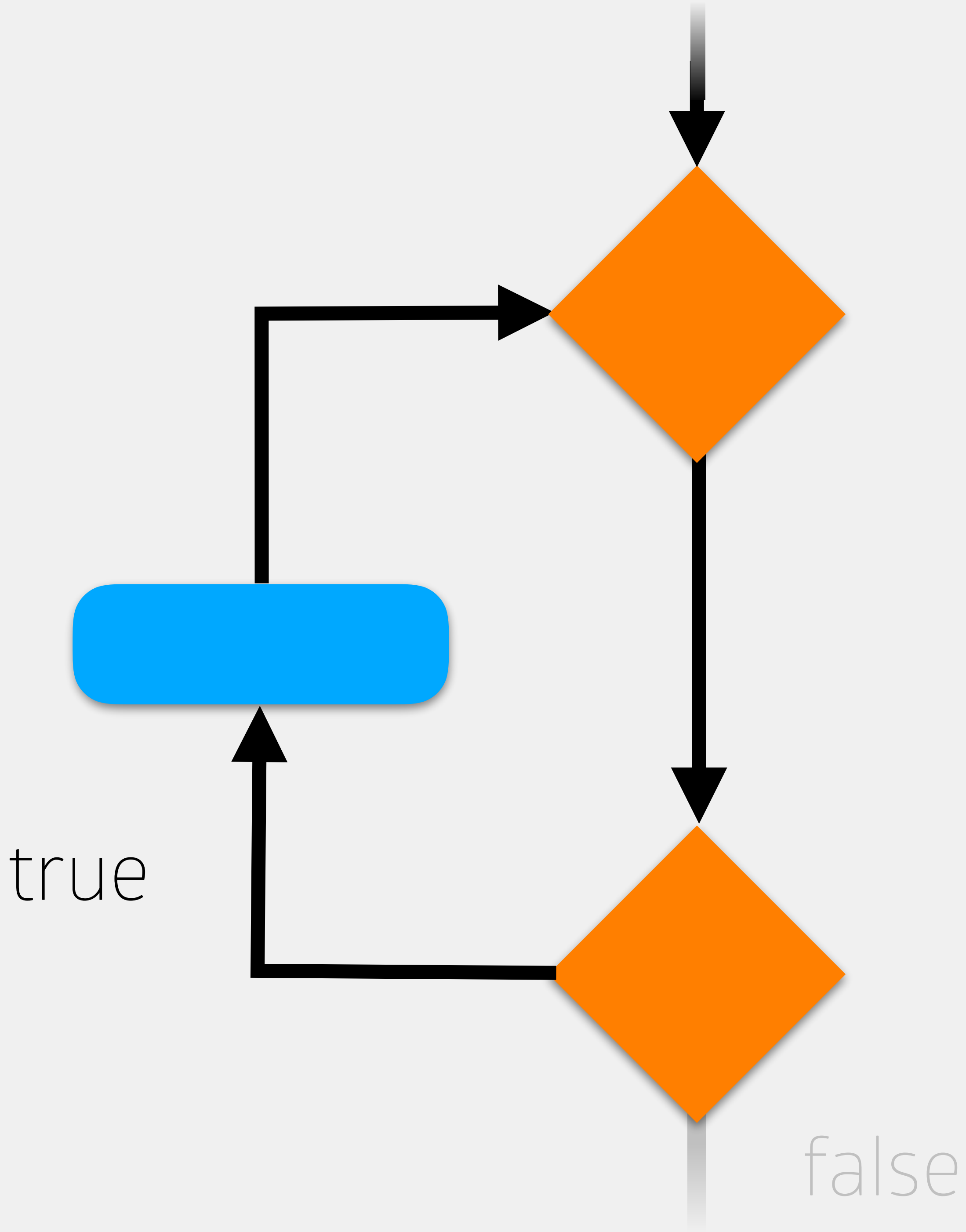
```
5
6
```

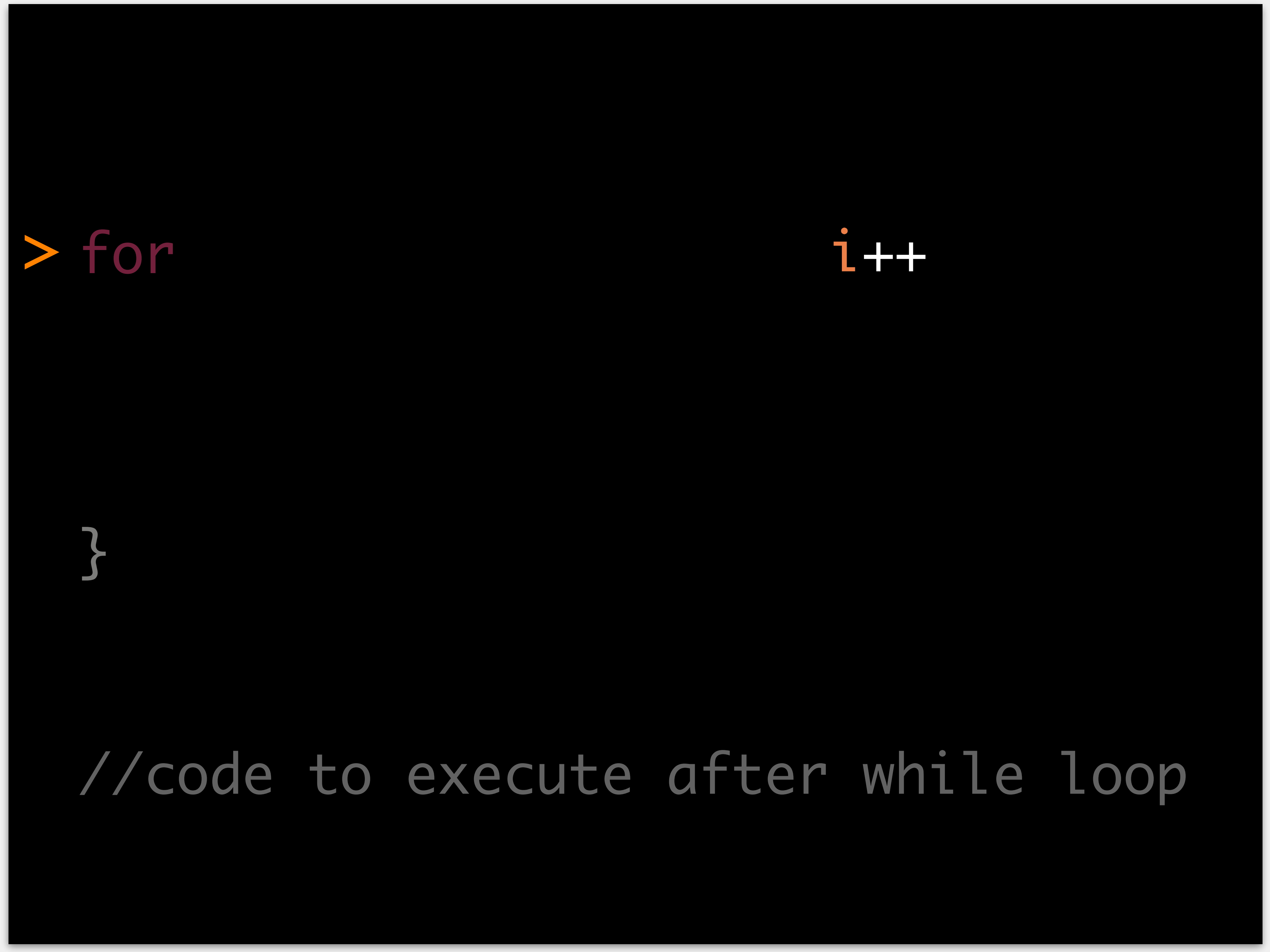
memory

i (int)
6

For Loops

“For i = 5, print the value of i while i is less than 8…”





```
> for                             i++

  }

  //code to execute after while loop
```

```
5
6
```

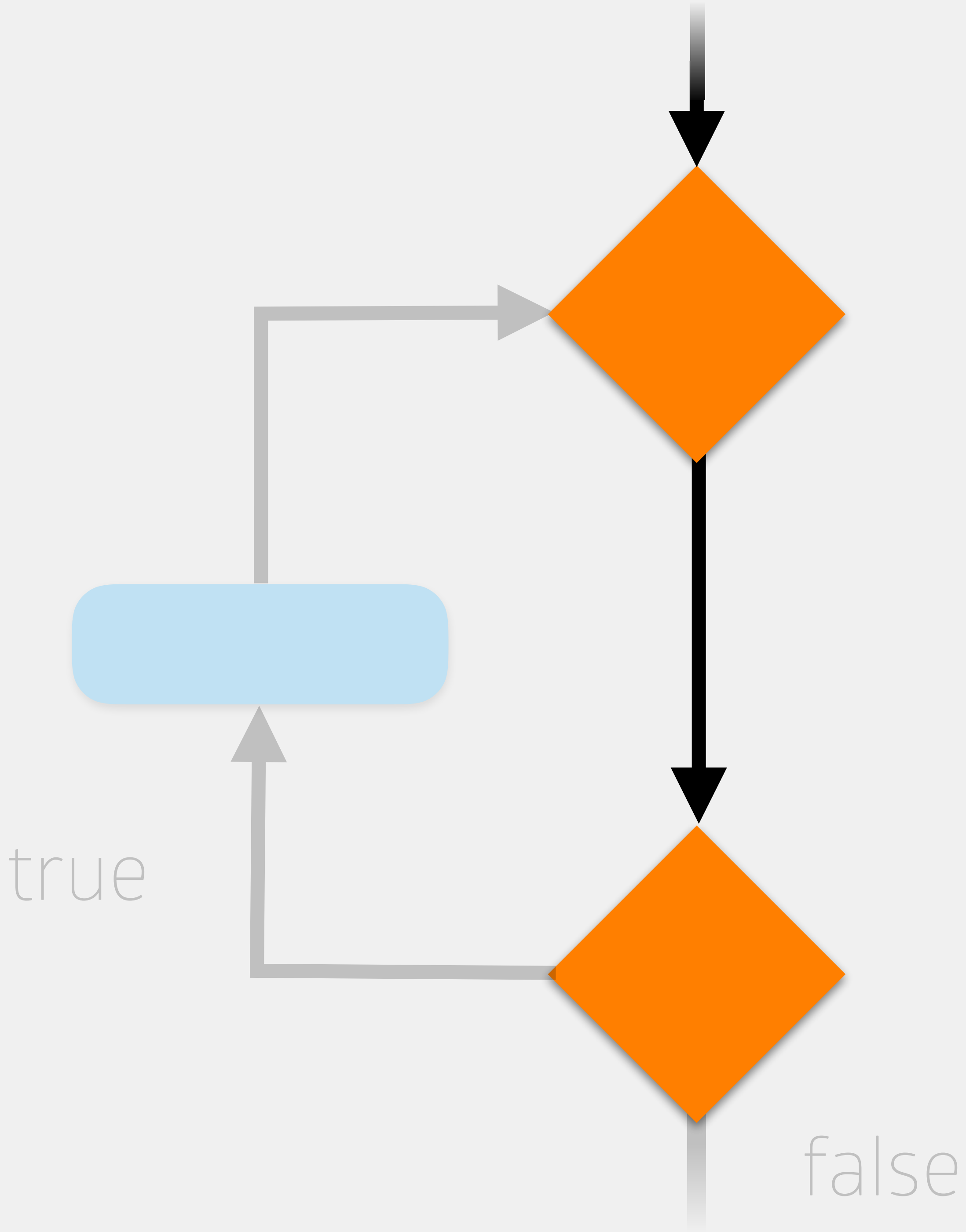
memory

i (int)

7

For Loops

“For i = 5, print the value of i while i is less than 8…”



is this true?

```
> for                    i < 8



 }



 //code to execute after while loop
```

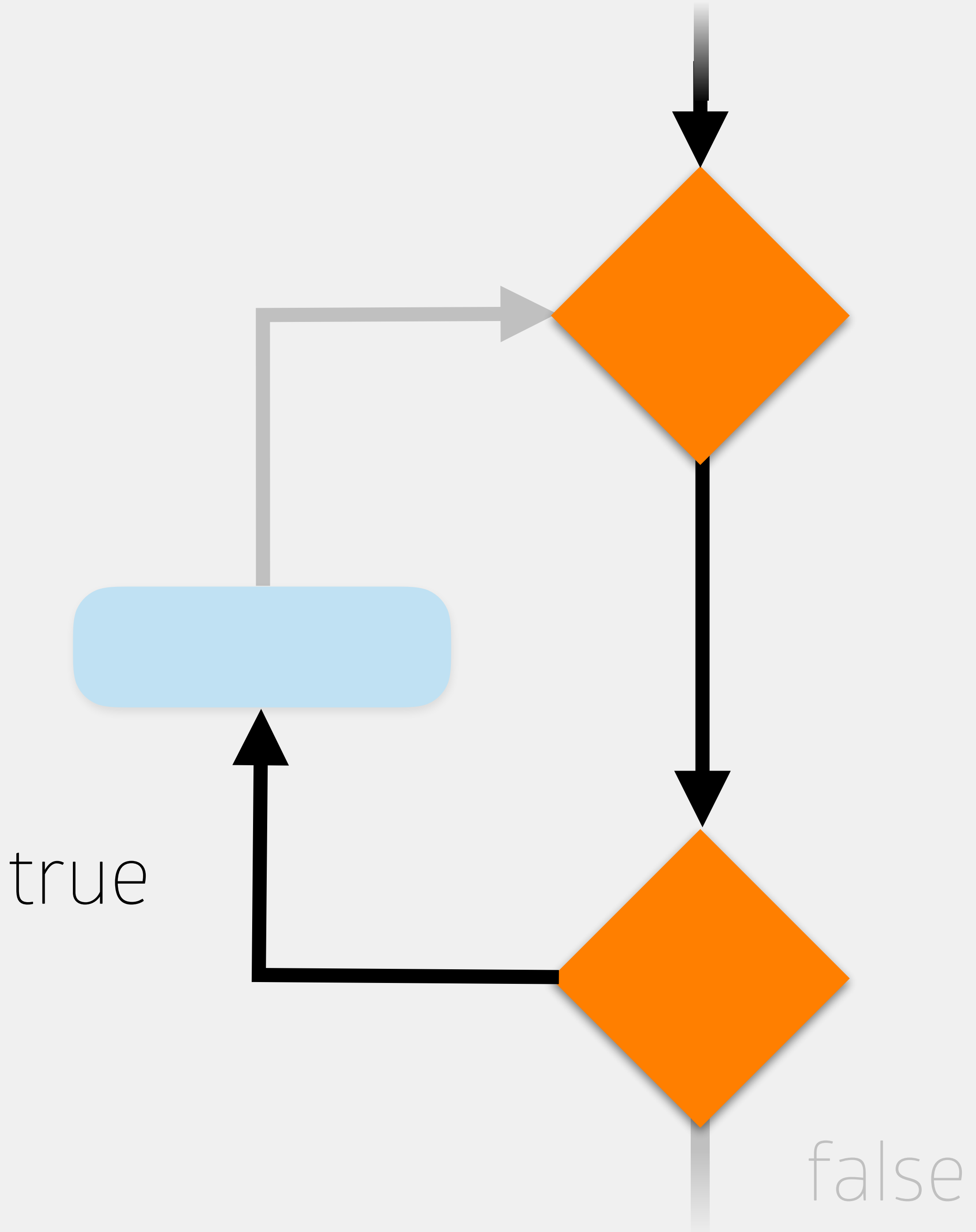
memory

i (int)
7

```
5
6
```

For Loops

“For i = 5, print the value of i while i is less than 8…”



is this true?

```
> for                    i < 8



  }



  //code to execute after while loop
```

```
5
6
```

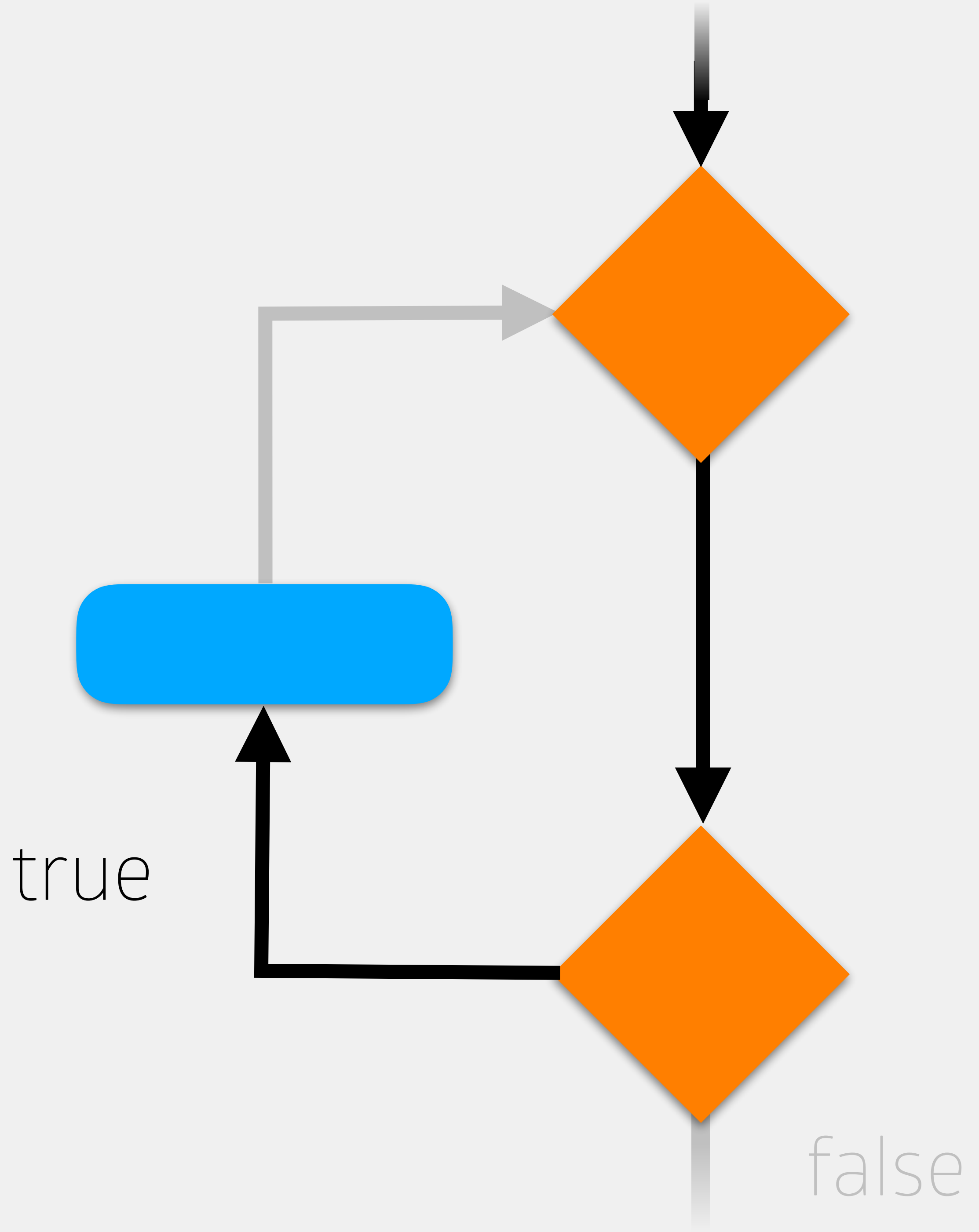
memory

i (int)

7

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {
  > System.out.println(i);
}


//code to execute after while loop
```

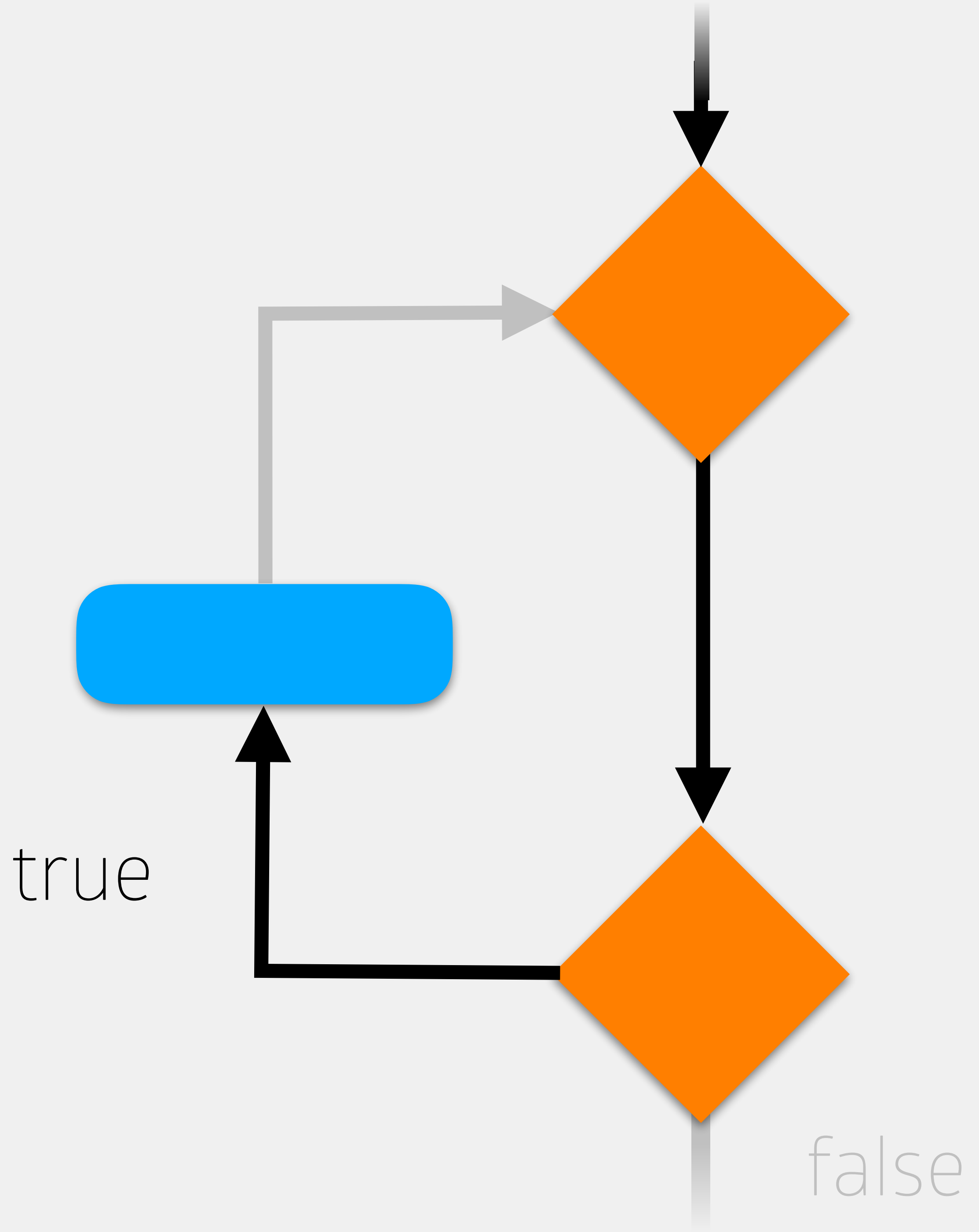
```
5
6
```

memory

i (int)
7

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {

    System.out.println(i);
  >
}


//code to execute after while loop
```

```
5
6
7
```

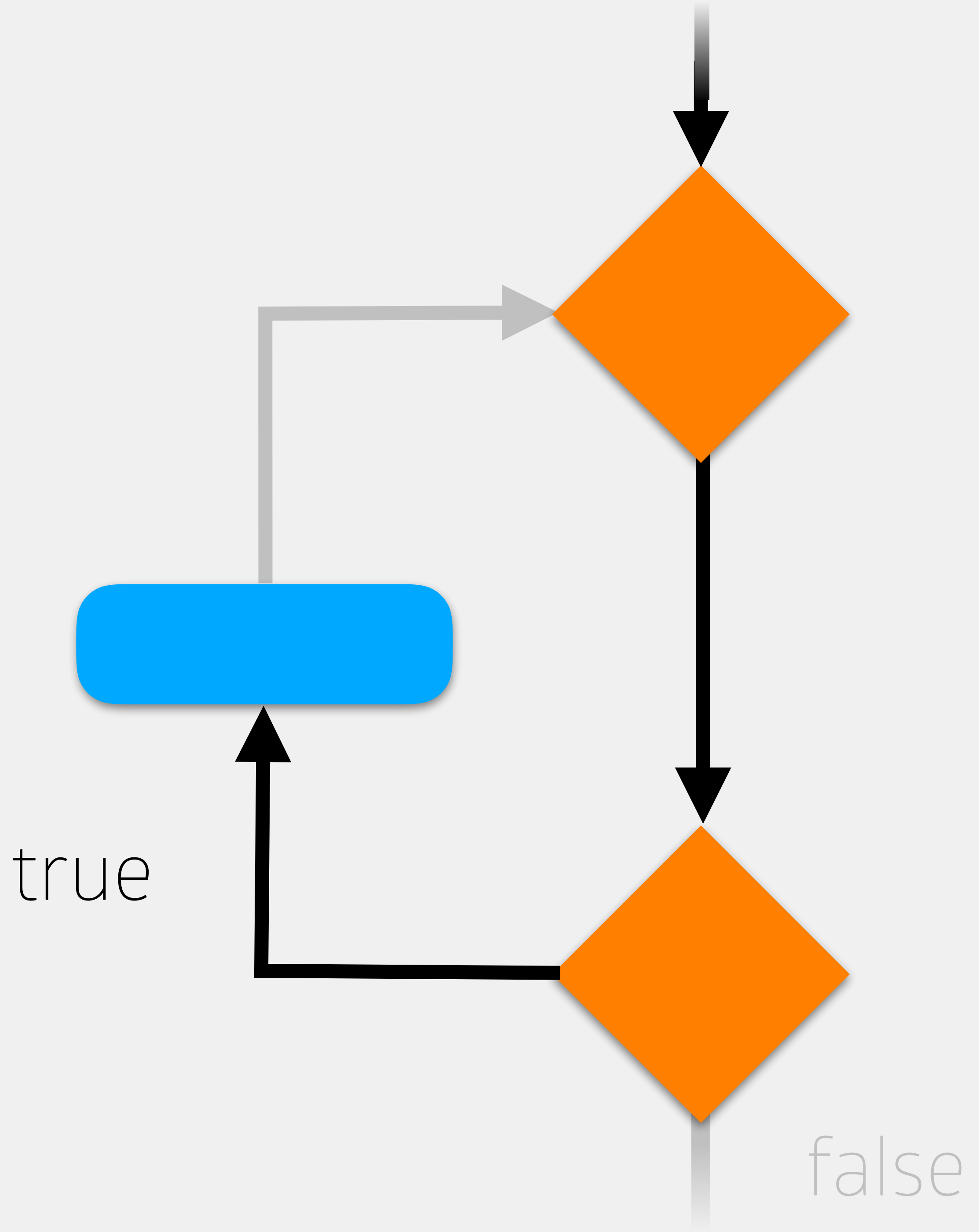
memory

i (int)

7

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
> for                    i++


   }


   //code to execute after while loop
```

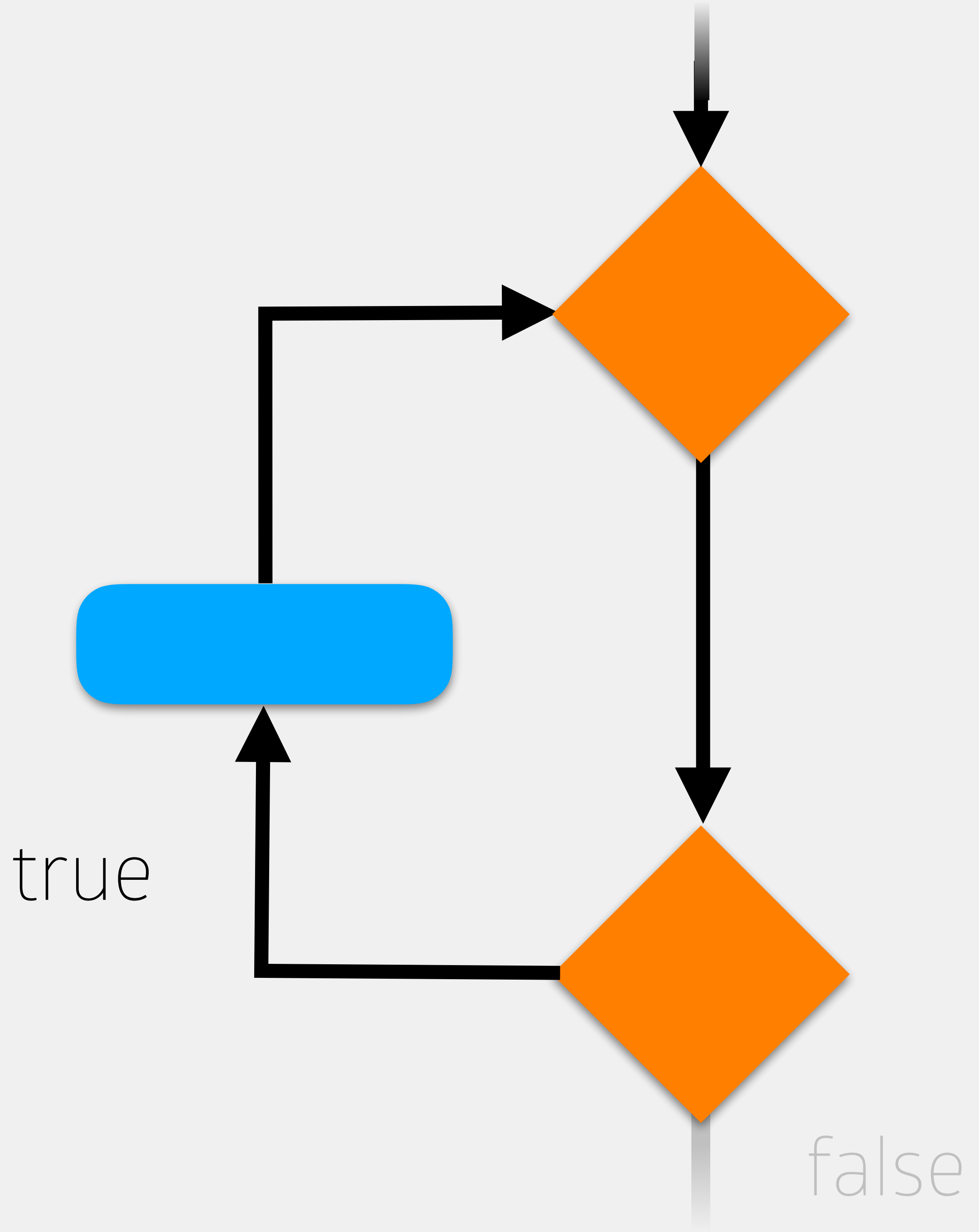
```
5
6
7
```

memory

i (int)
7

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
> for                        i++

  }

  //code to execute after while loop
```

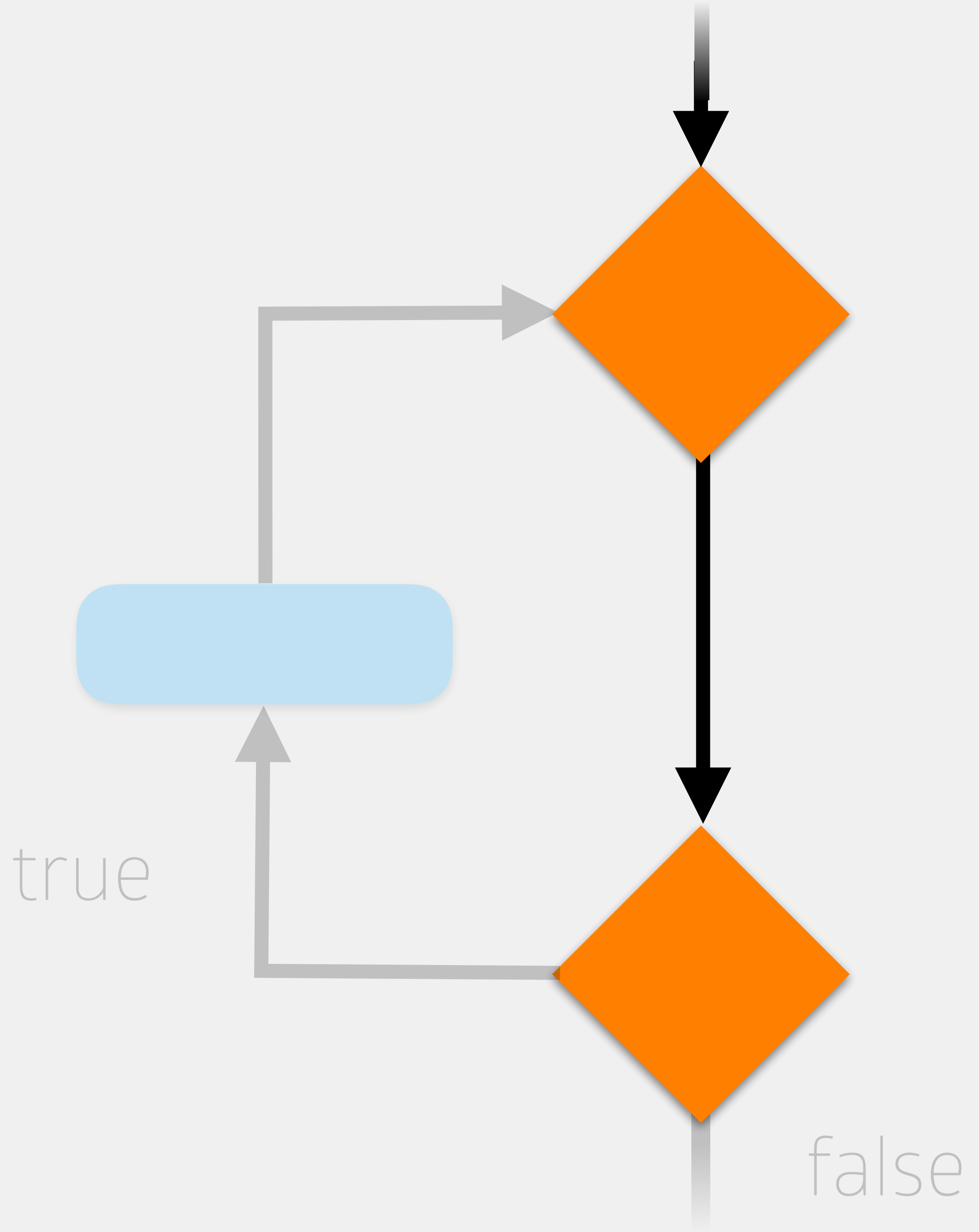
```
5
6
7
```

memory

i (int)
8

For Loops

“For i = 5, print the value of i while i is less than 8…”



is this true?

```
> for                  i < 8

  }


  //code to execute after while loop
```

```
5
6
7
```

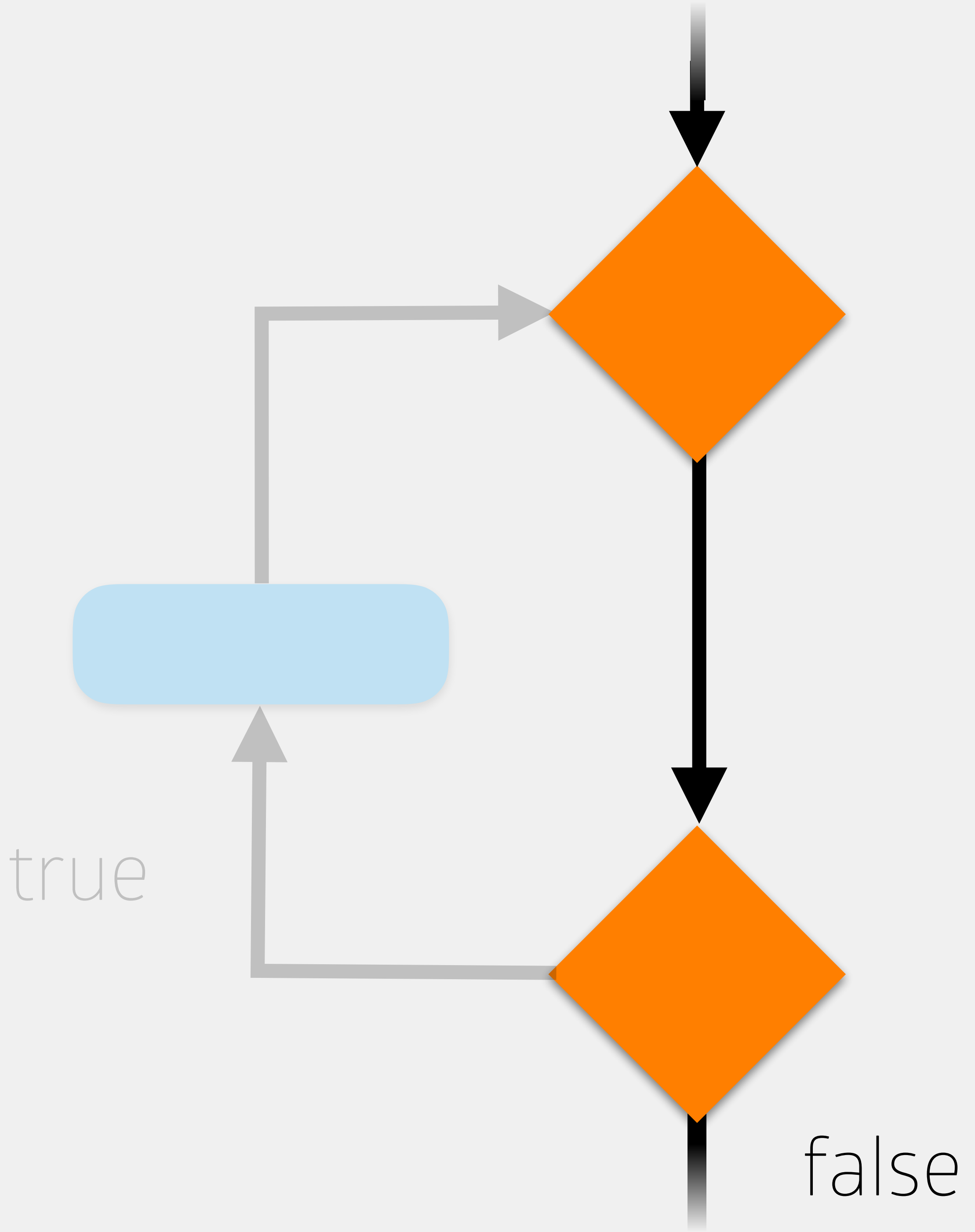
memory

i (int)

8

For Loops

“For i = 5, print the value of i while i is less than 8…”



```
for (int i = 5; i < 8; i++) {
    System.out.println(i);
}

> //code to execute after while loop
```

```
5
6
7
```

memory

i (int)
8

For Loop Notes

For loop variables are one of the few places where you can get away with single letter variable names

- but, if you can come up with a sensible name, use it!

Why use the for loop?

- for loops ensure you have all four parts of the loop there
- easier to miss one or more with the while loop

For Loop vs While Loop

For loops are used when we know how many times we want the loop to execute

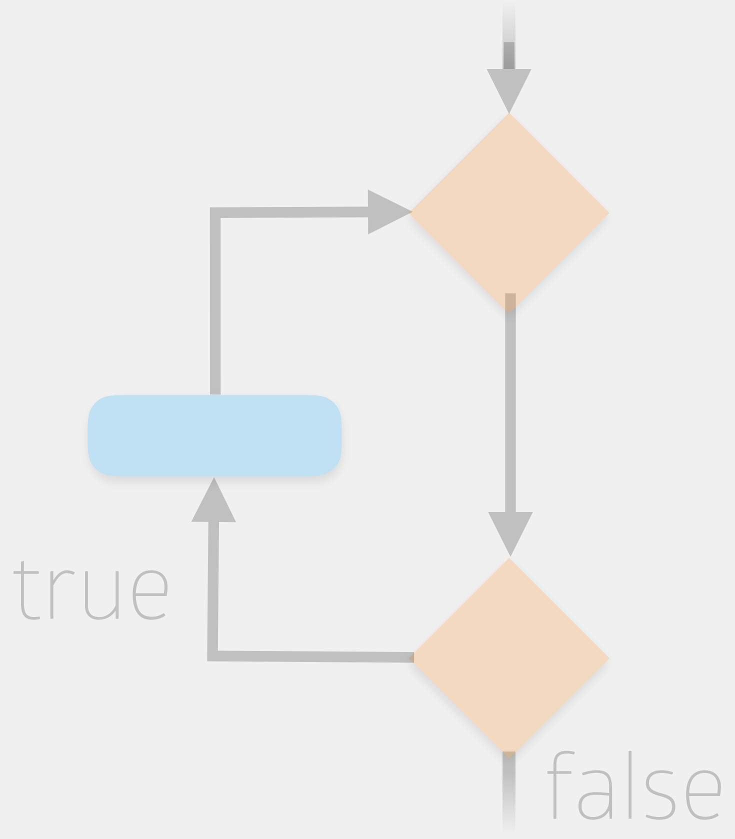
While loops are used when we aren’t sure how many times we want the loop to execute

In reality, can use for or while loops interchangeably

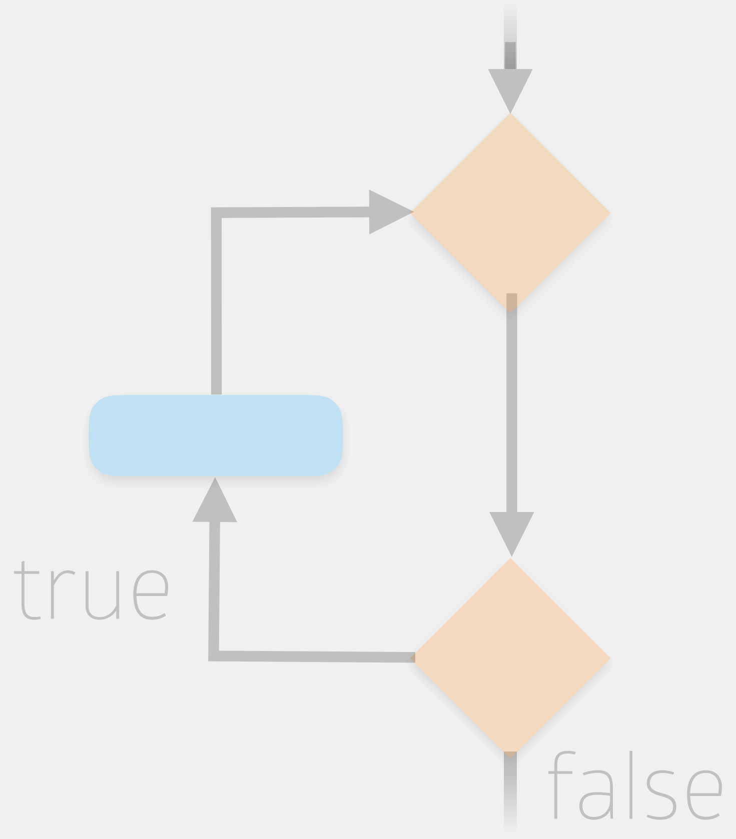
  …although it is often more natural to use one over the other in most cases

  should be able to understand how both work!

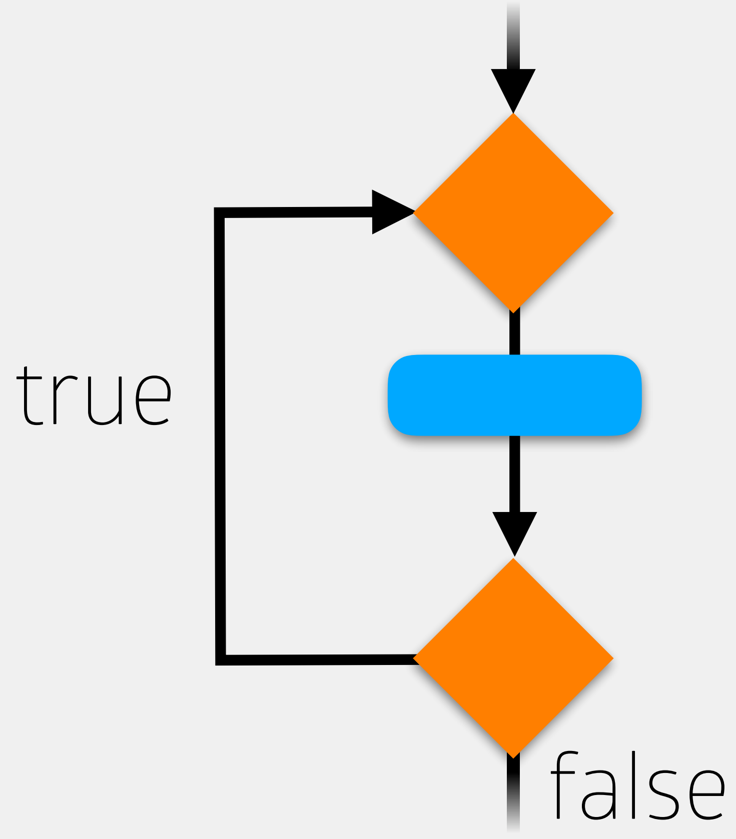
Types of Loops



while loops



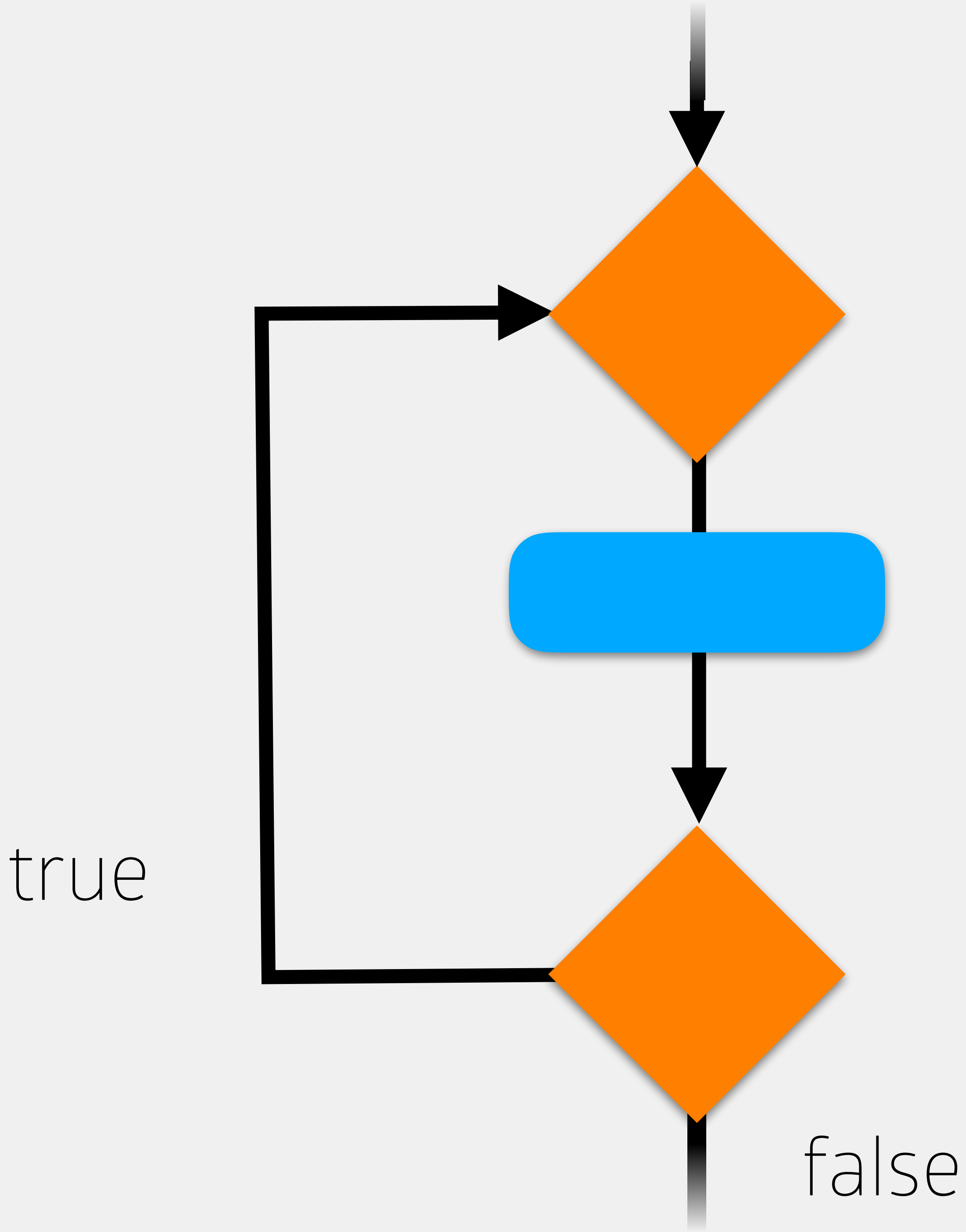
for loops



do while loops

Do While Loops

Similar to a while loop, but checks the condition last

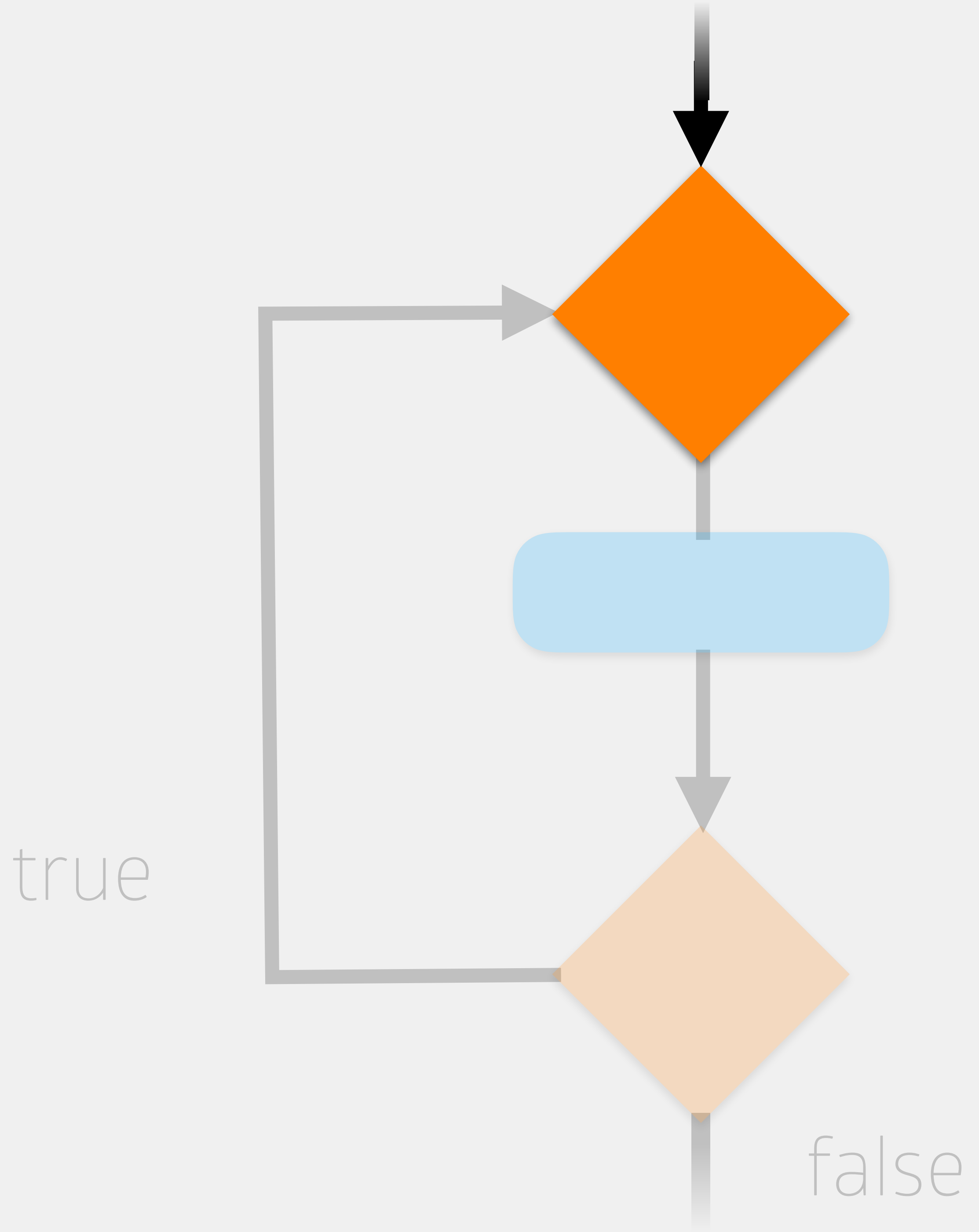


```
do {

    //code to execute if boolean expression is true

} while (<boolean expr>);

//code to execute after while loop
```

remember me!

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

> do {

      System.out.println(value);
      value++;

  } while (value < 8);

  //code to execute after while loop
```

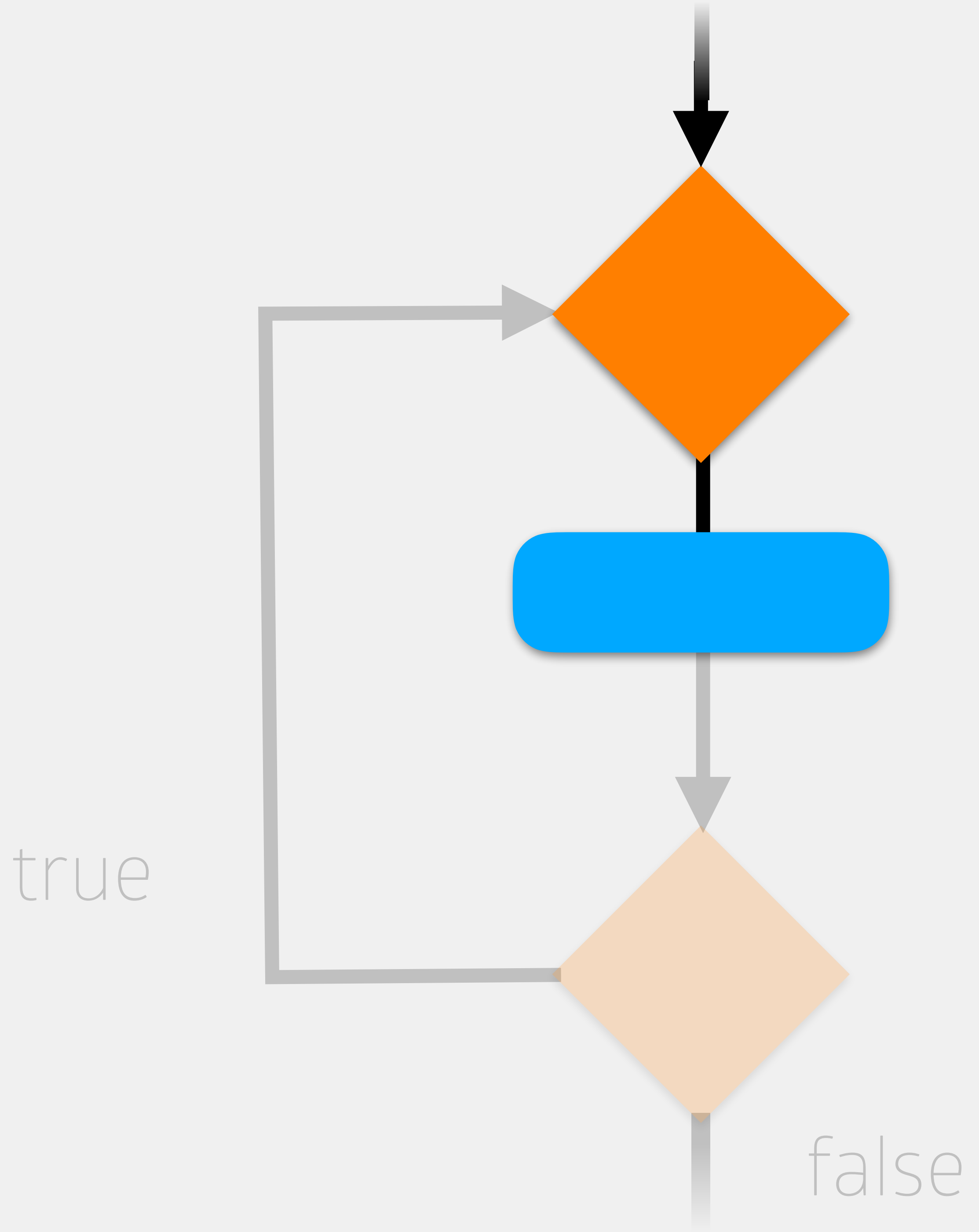
memory

value (int)

5

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

  > System.out.println(value);
    value++;

} while (value < 8);

//code to execute after while loop
```

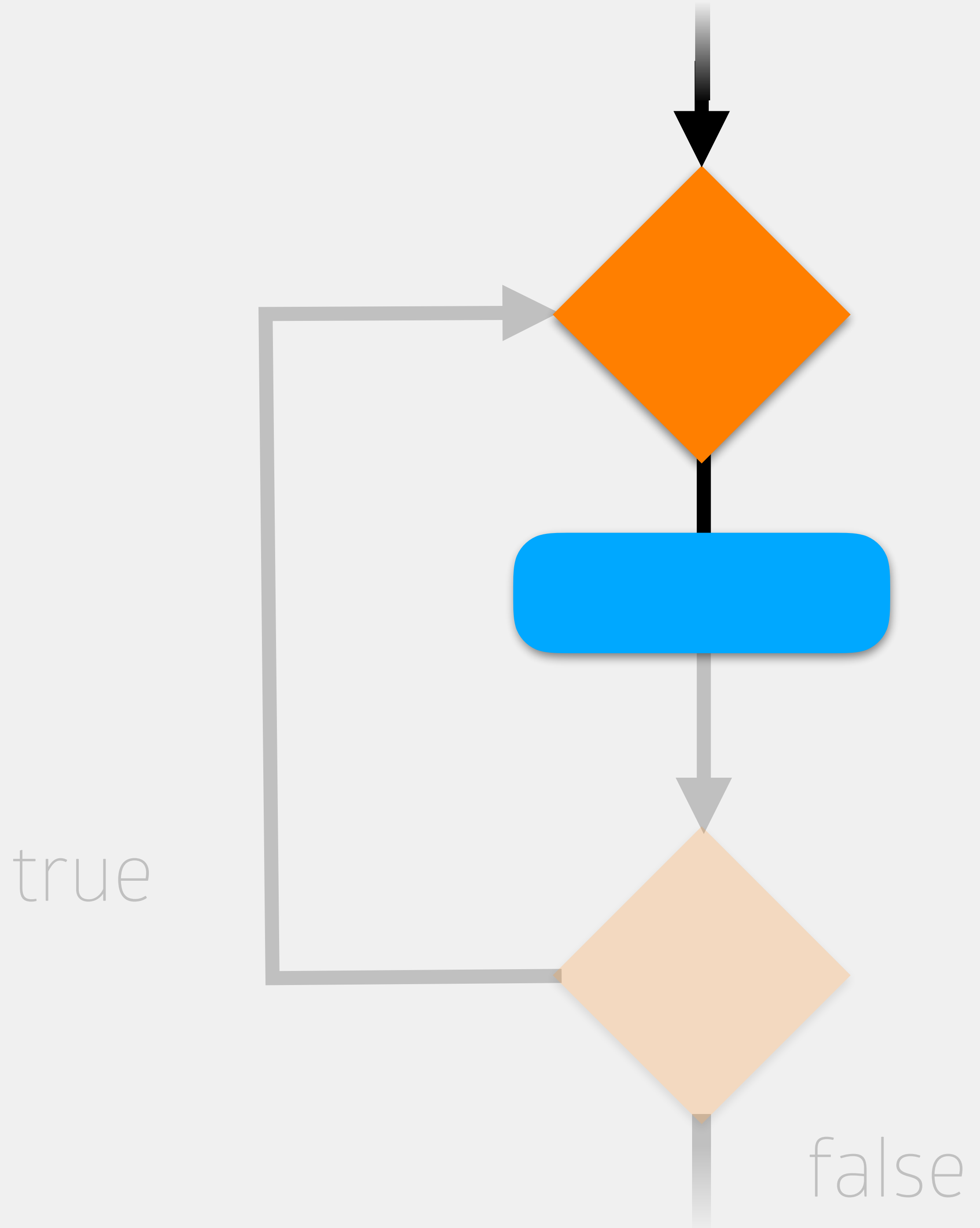
memory

value (int)

5

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

    System.out.println(value);
  > value++;

} while (value < 8);

//code to execute after while loop
```

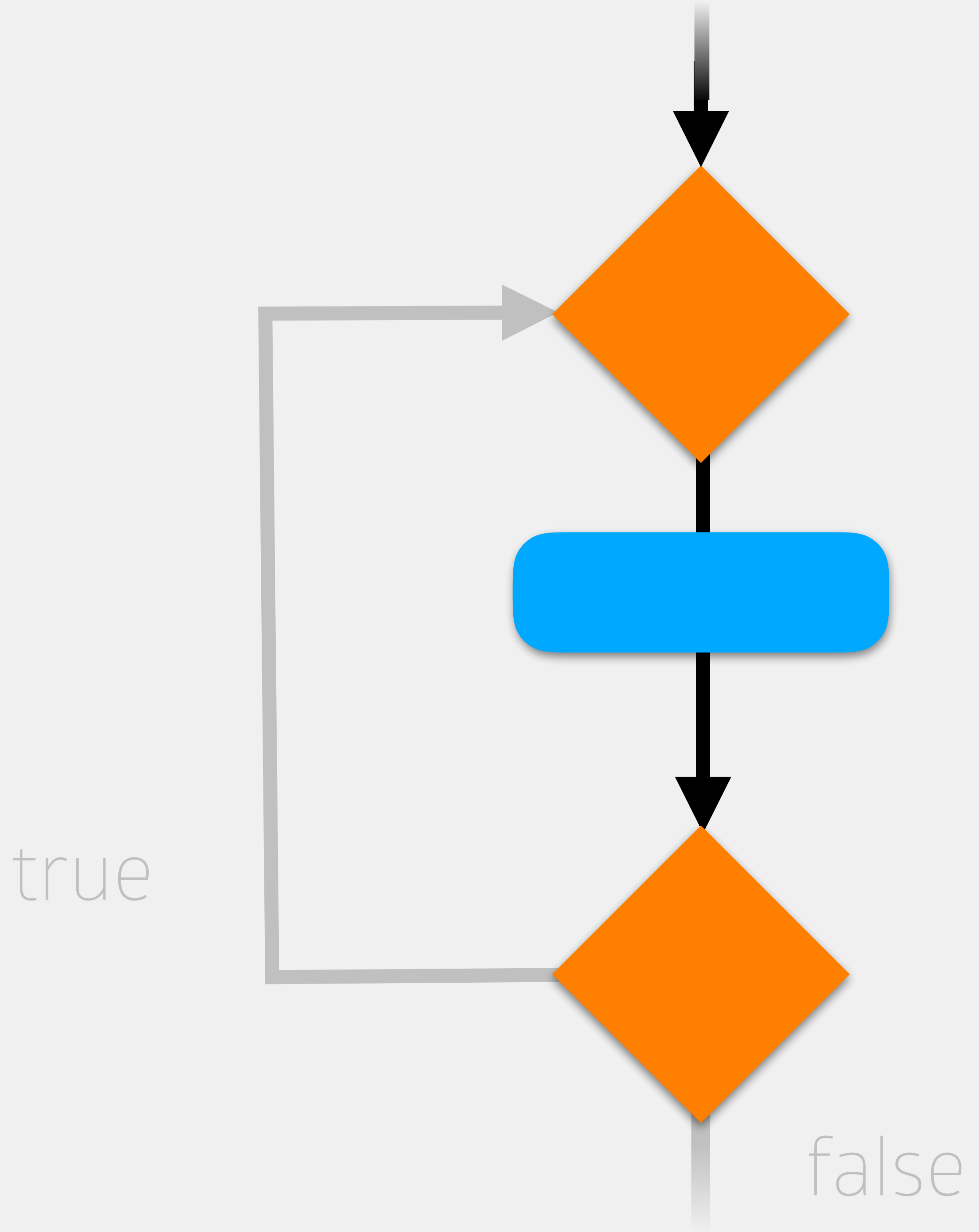
```
5
```

memory

value (int)
5

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

> } while (value < 8);

  //code to execute after while loop
```

is this true?

```
5
```

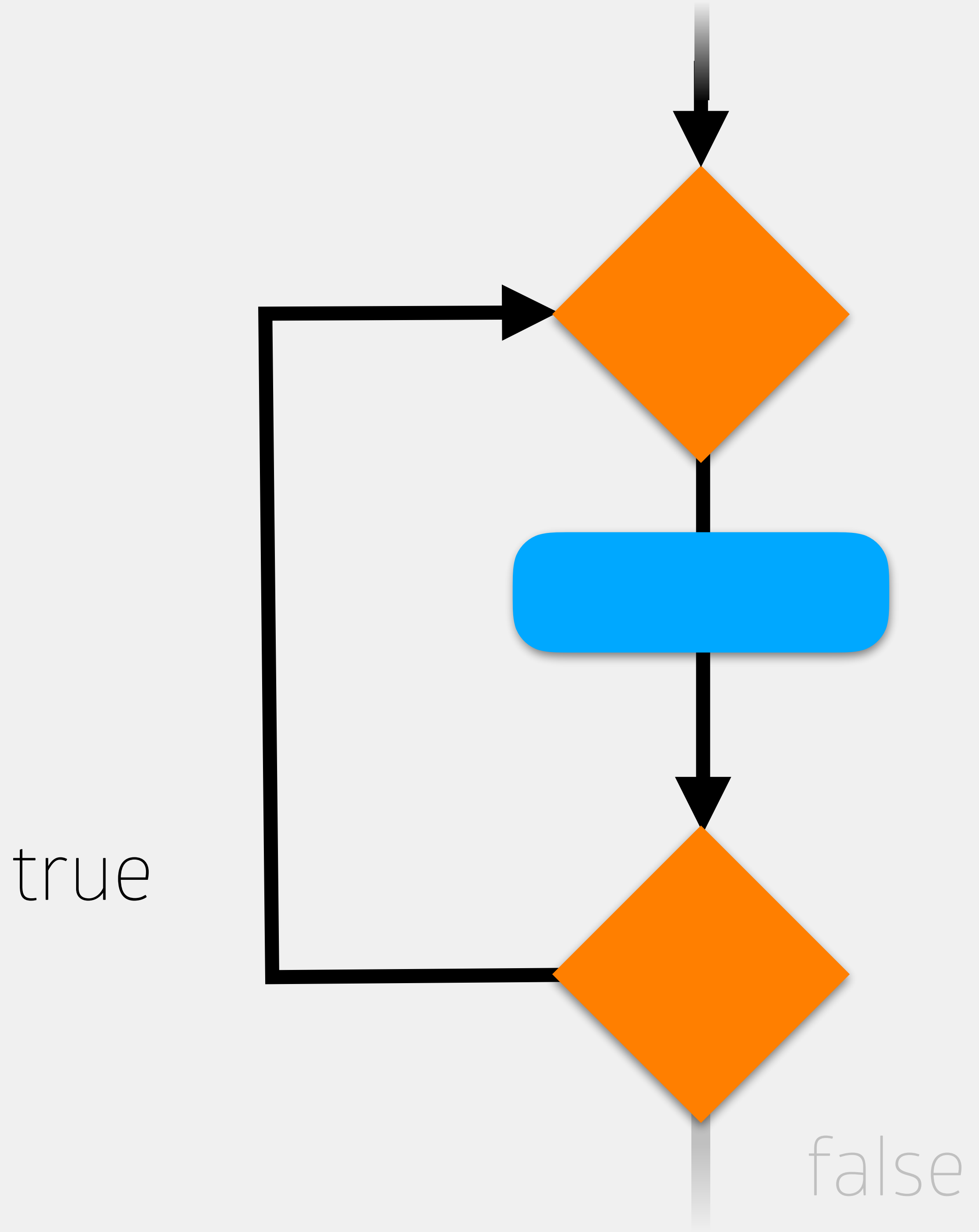
memory

value (int)

6

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

> } while (value < 8);

  //code to execute after while loop
```

```
5
```

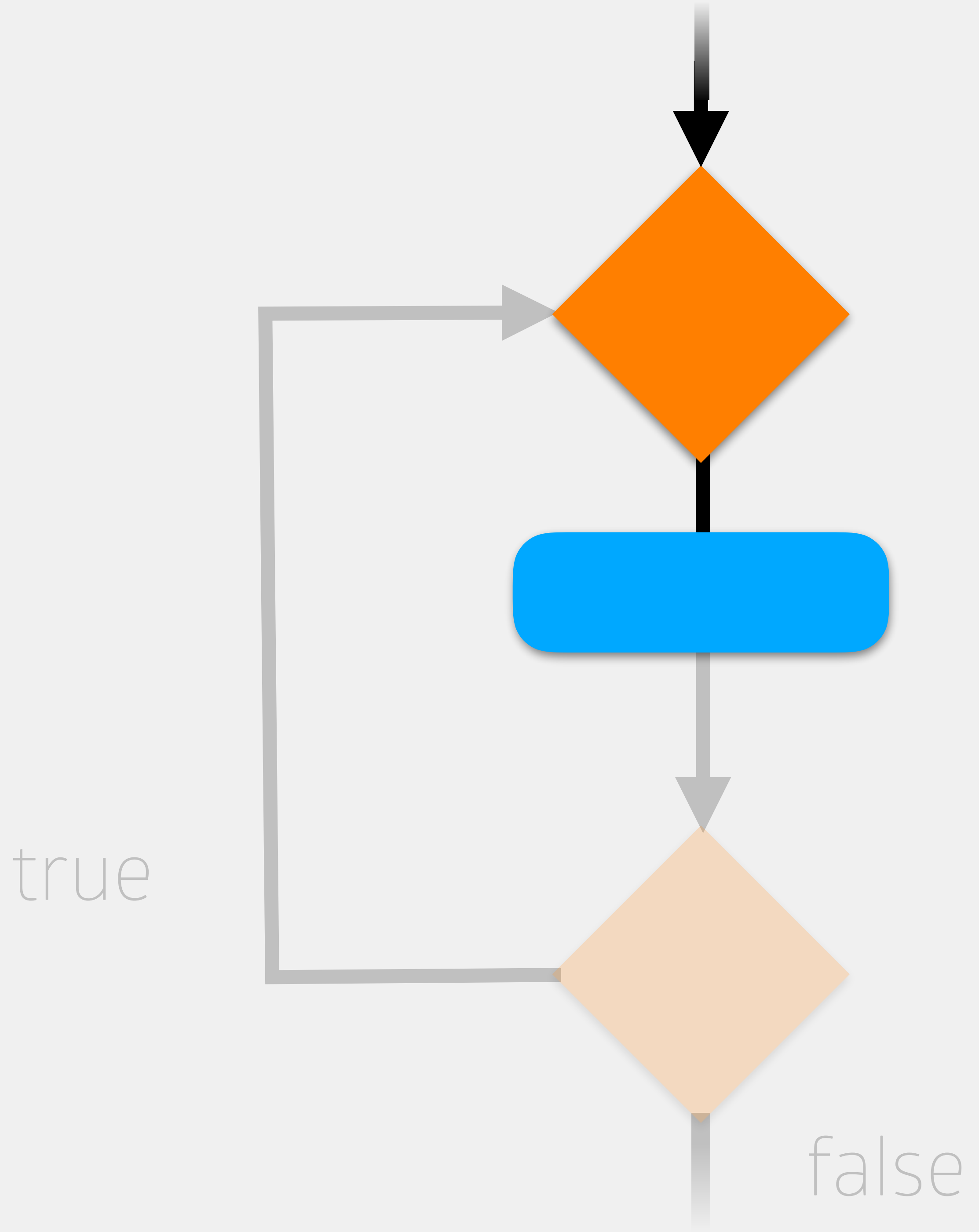
memory

value (int)

6

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

> do {

      System.out.println(value);
      value++;

  } while (value < 8);

  //code to execute after while loop
```

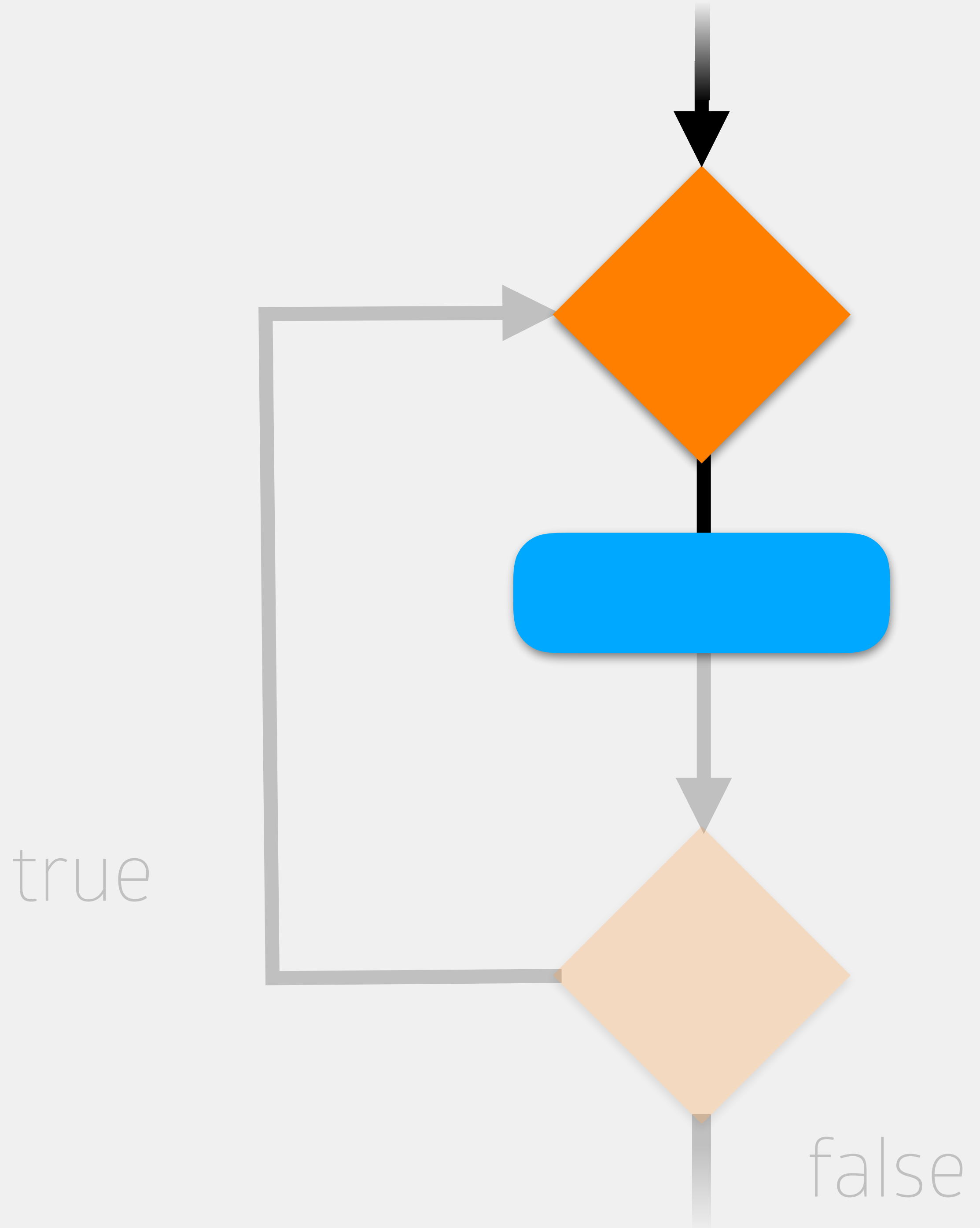
```
5
```

memory

value (int)
6

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

  > System.out.println(value);
    value++;

} while (value < 8);

//code to execute after while loop
```

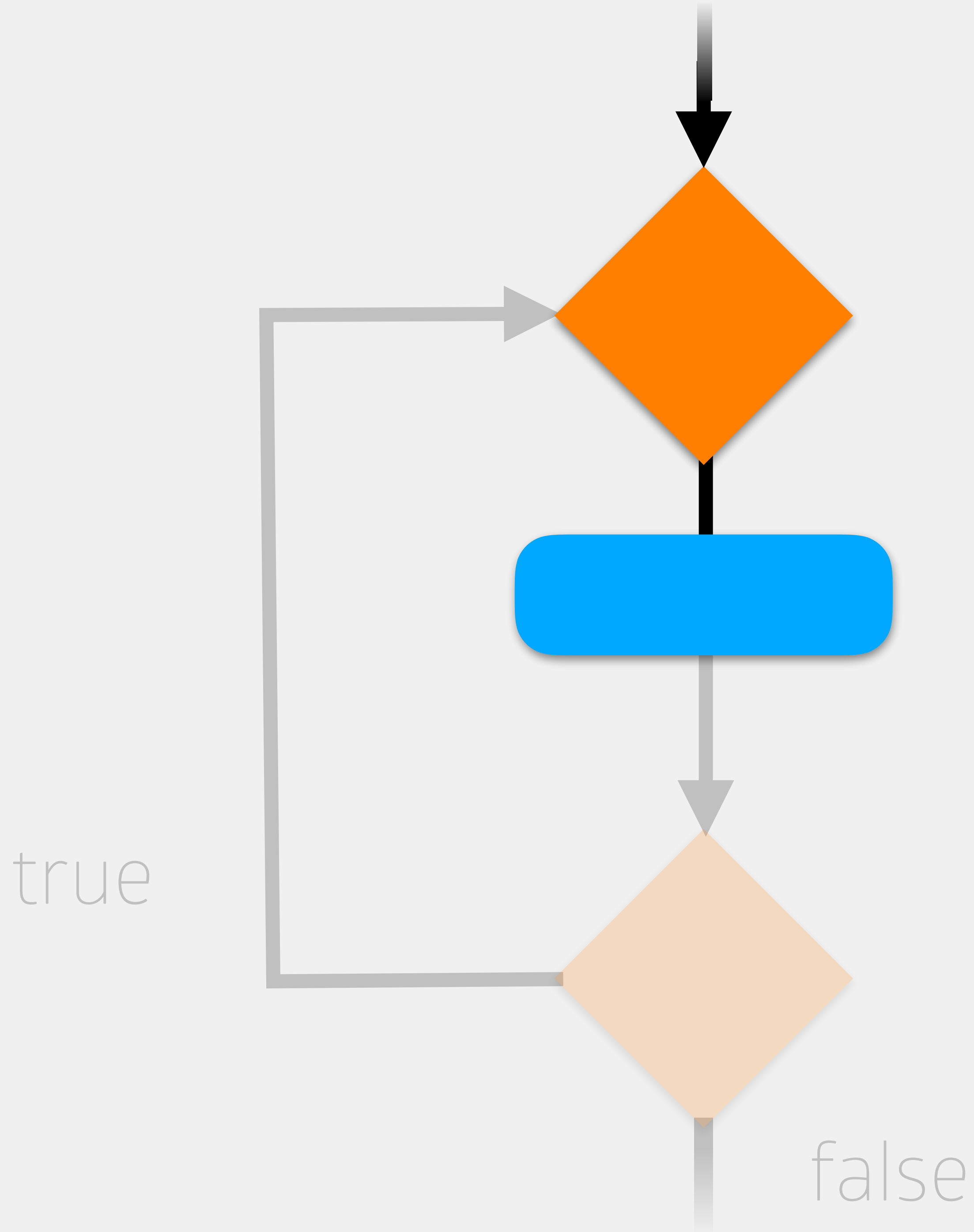
```
5
```

memory

value (int)
6

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

    System.out.println(value);
  > value++;

} while (value < 8);

//code to execute after while loop
```

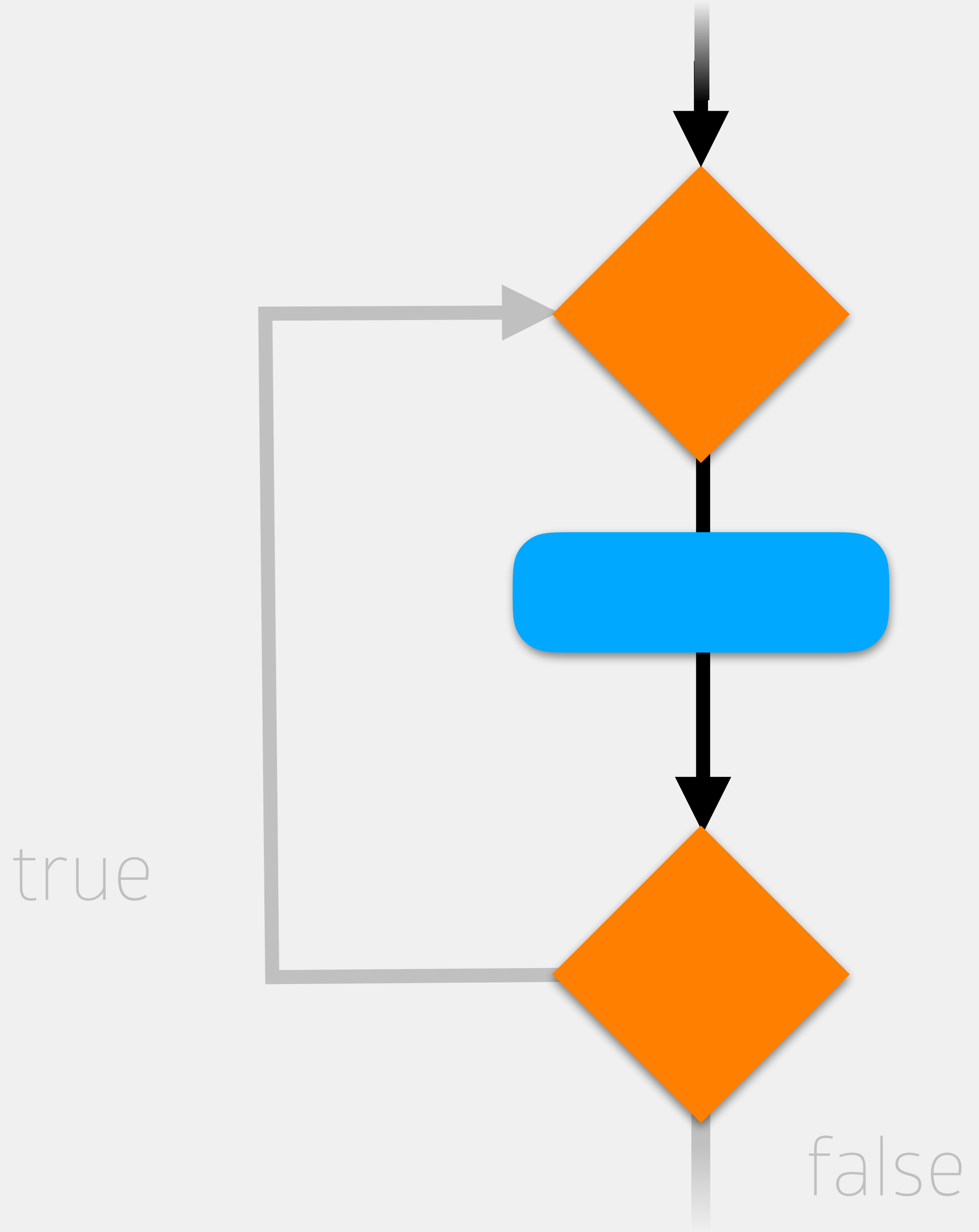
```
5
6
```

memory

value (int)
6

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

> } while (value < 8);

  //code to execute after while loop
```

is this true?

```
5
6
```

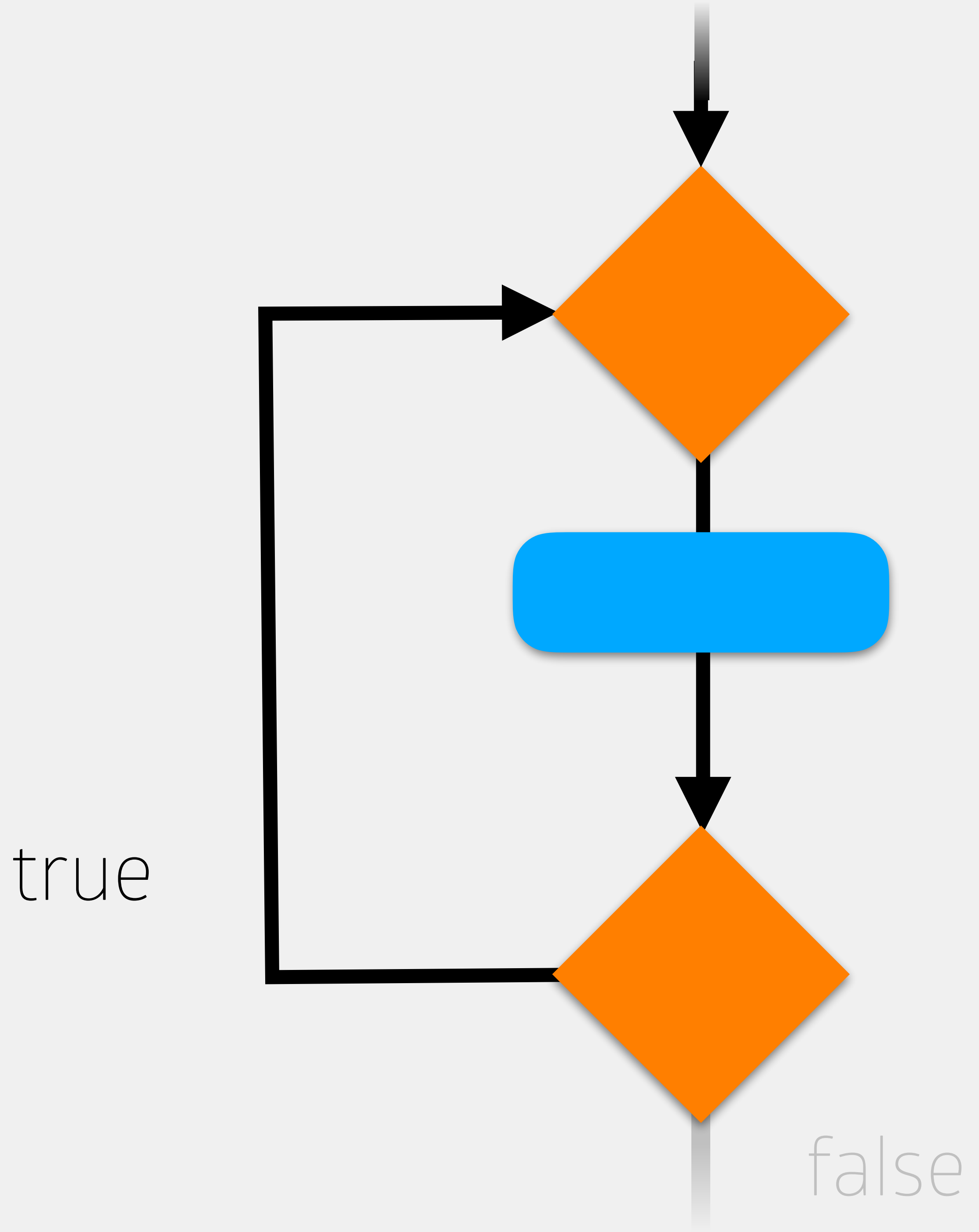
memory

value (int)

7

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

> } while (value < 8);

  //code to execute after while loop
```

```
5
6
```

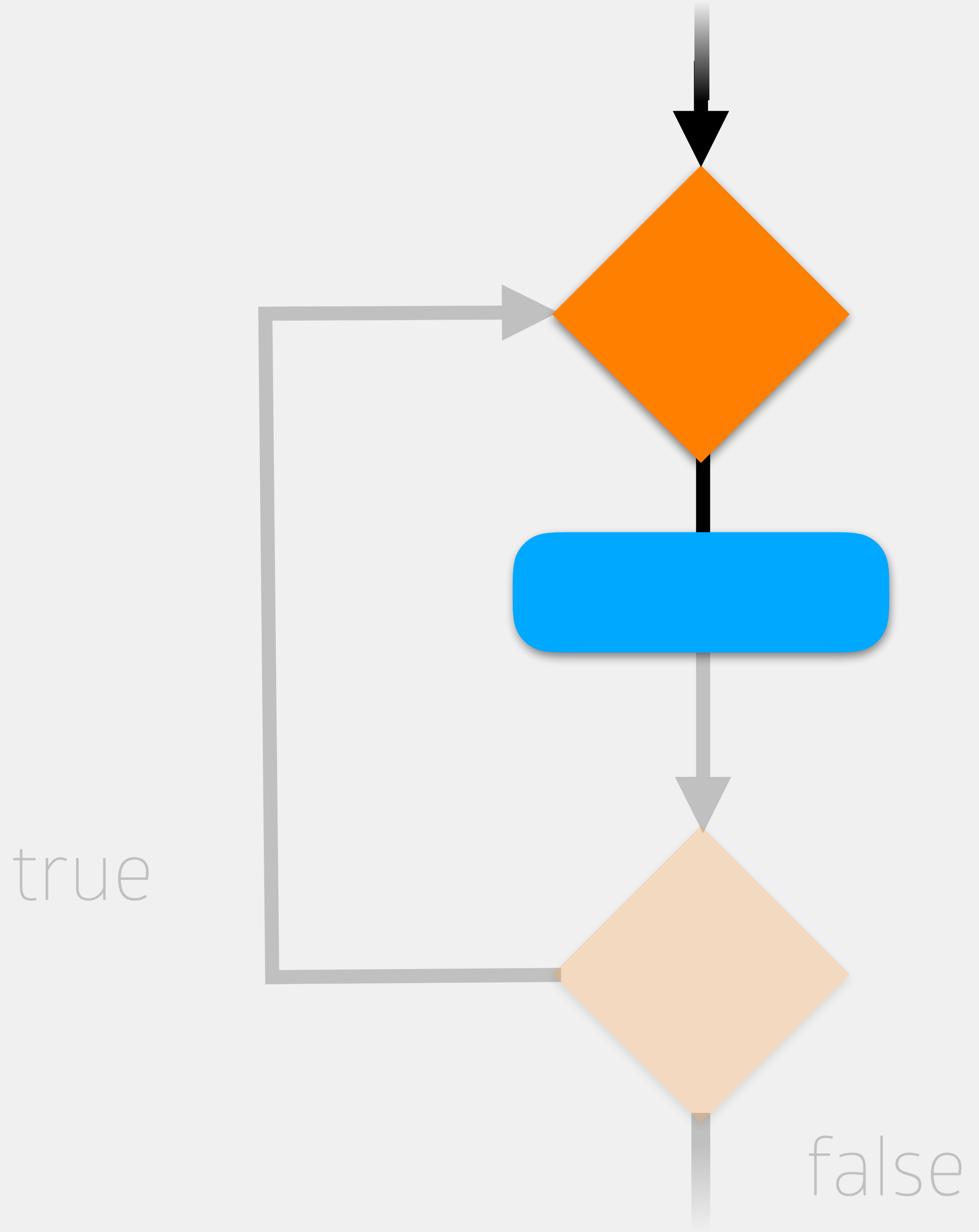
memory

value (int)

7

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

> do {

      System.out.println(value);
      value++;

  } while (value < 8);

  //code to execute after while loop
```

```
5
6
```

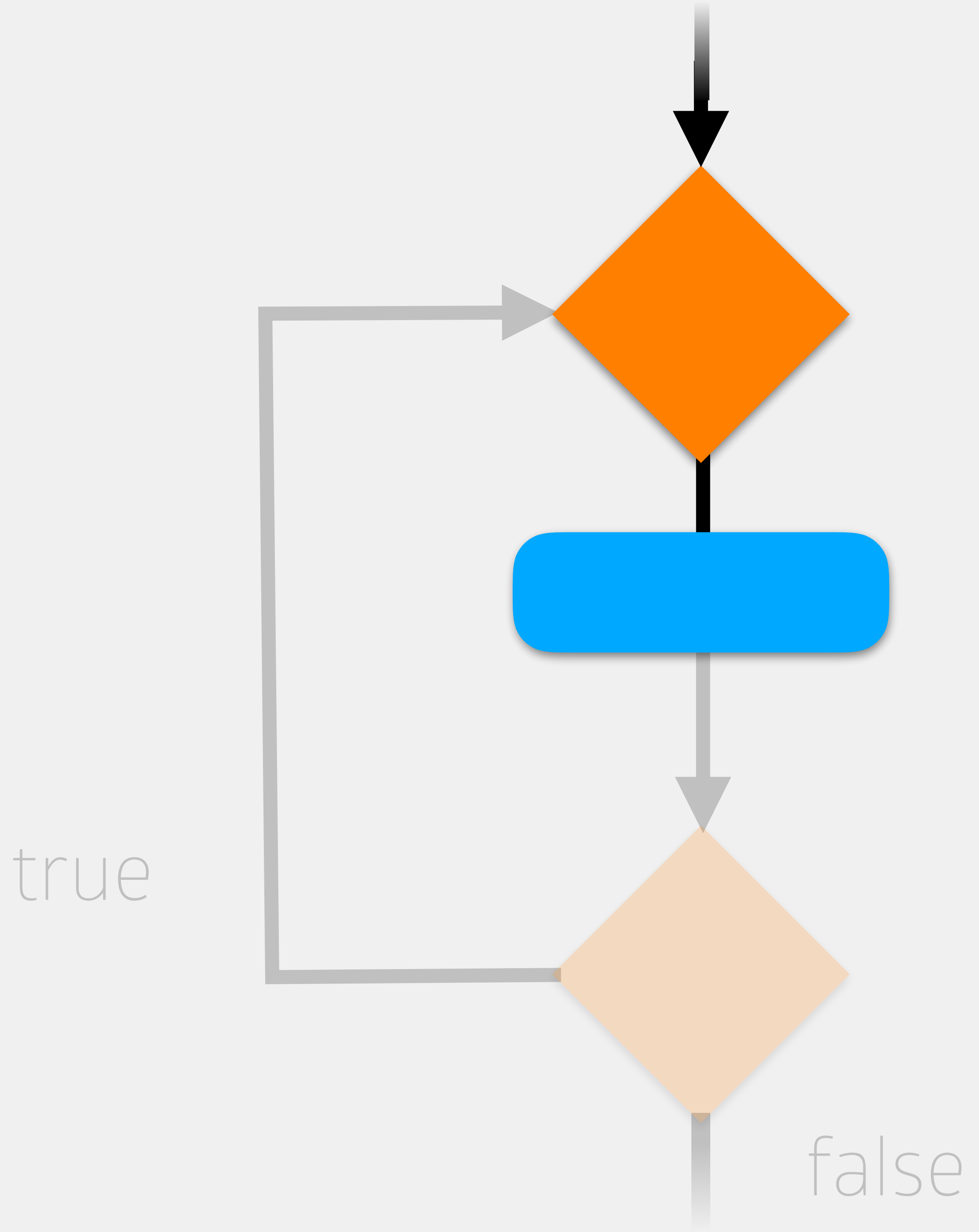
memory

value (int)

7

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

  > System.out.println(value);
    value++;

} while (value < 8);

//code to execute after while loop
```

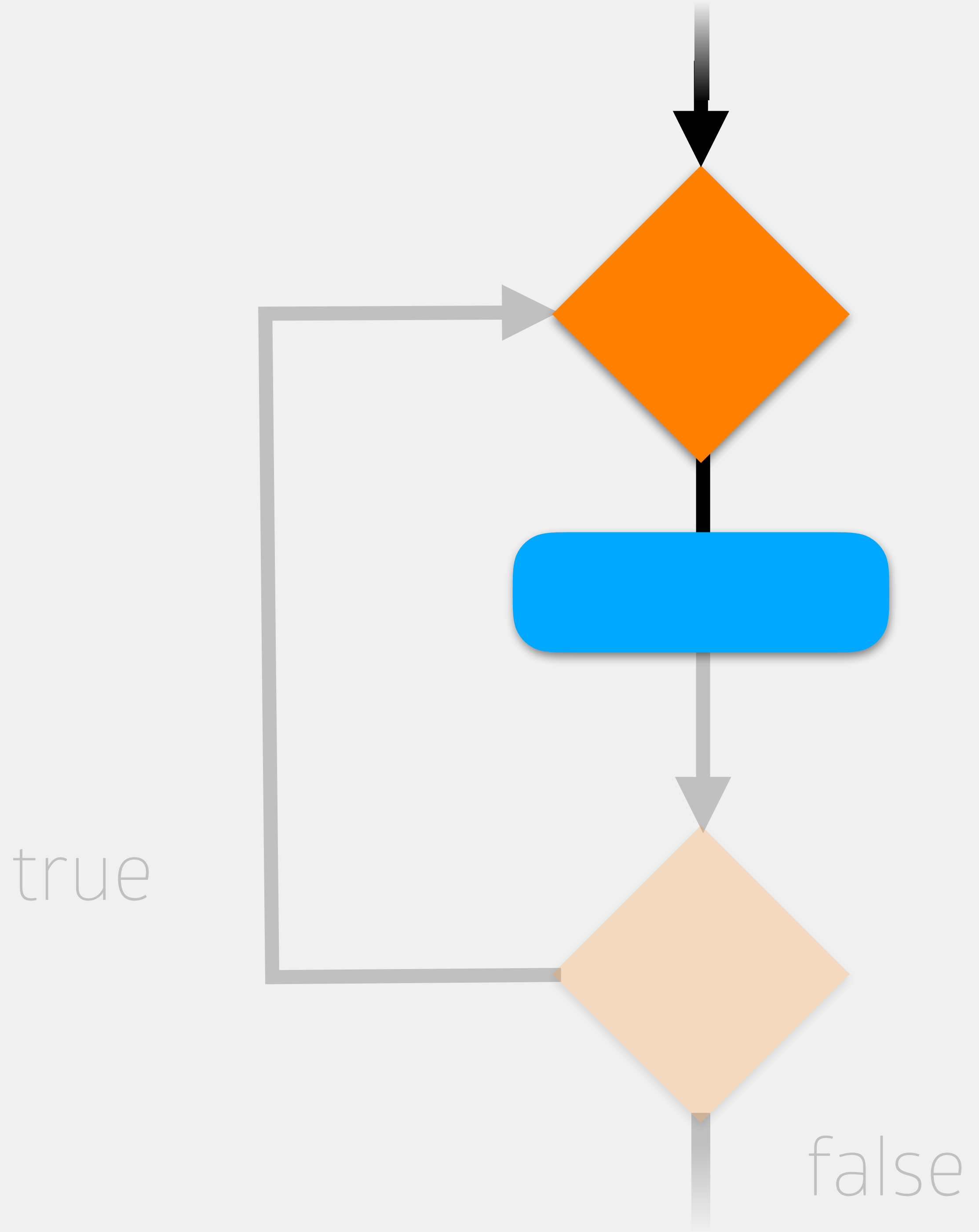
```
5
6
```

memory

value (int)
7

Do While Loops

Similar to a while loop, but checks the condition last



```
int value = 5;

do {

    System.out.println(value);
  > value++;

} while (value < 8);

//code to execute after while loop
```

```
5
6
7
```

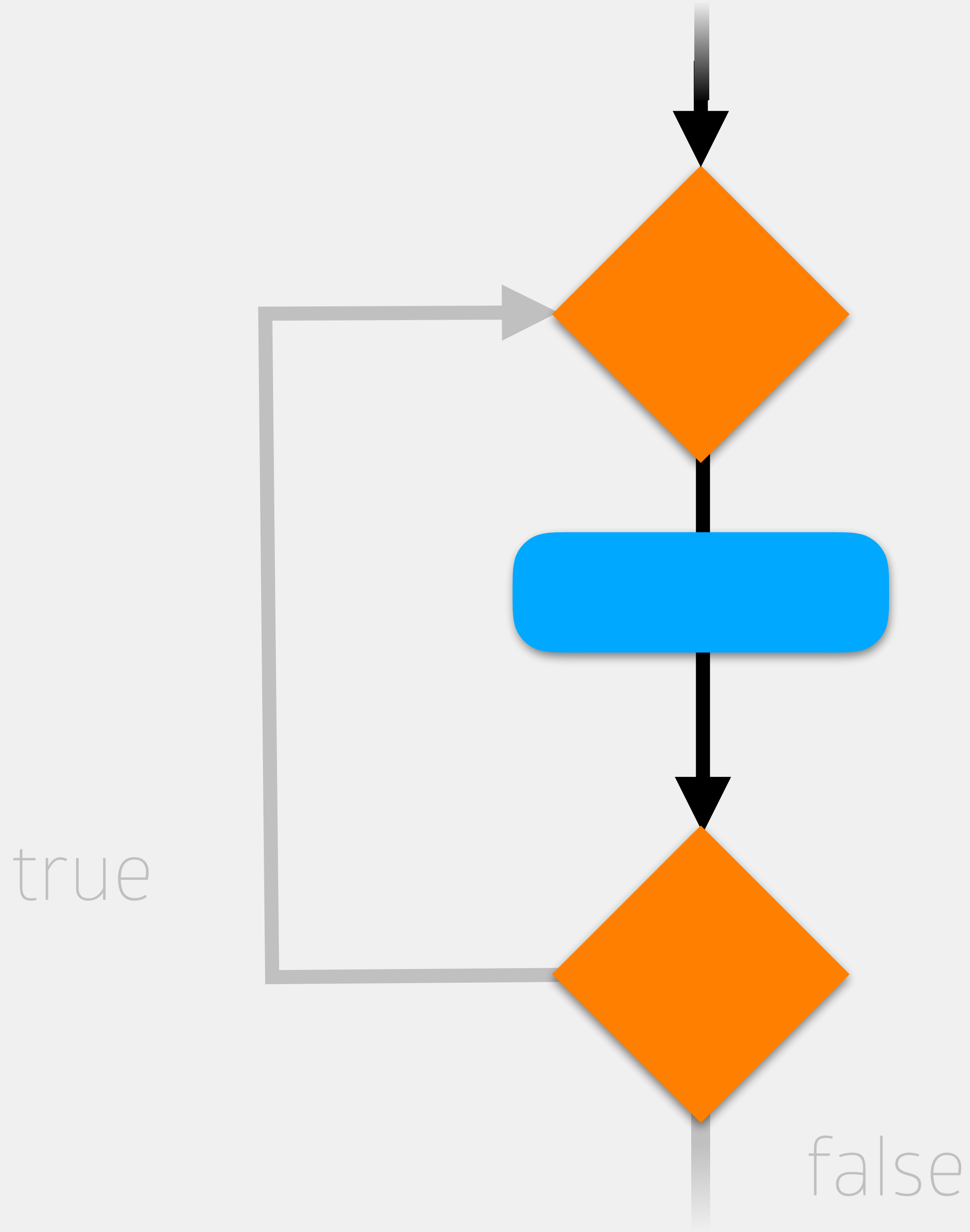
memory

value (int)

7

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

> } while (value < 8);

  //code to execute after while loop
```

is this true?

memory

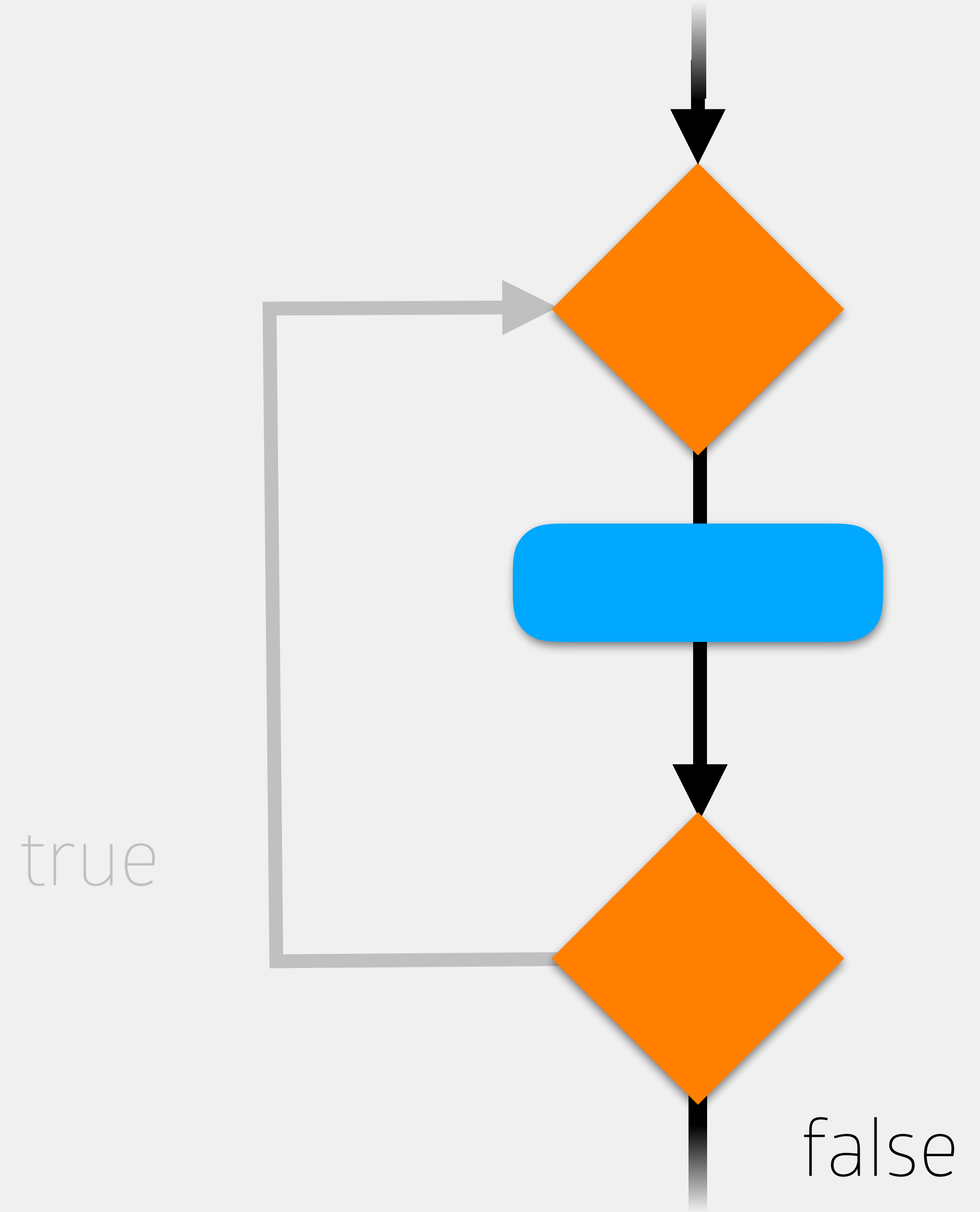
value (int)

8

```
5
6
7
```

Do While Loops

Similar to a while loop, but checks the condition last



```
  int value = 5;

  do {

      System.out.println(value);
      value++;

  } while (value < 8);

> //code to execute after while loop
```

```
5
6
7
```

memory

value (int)

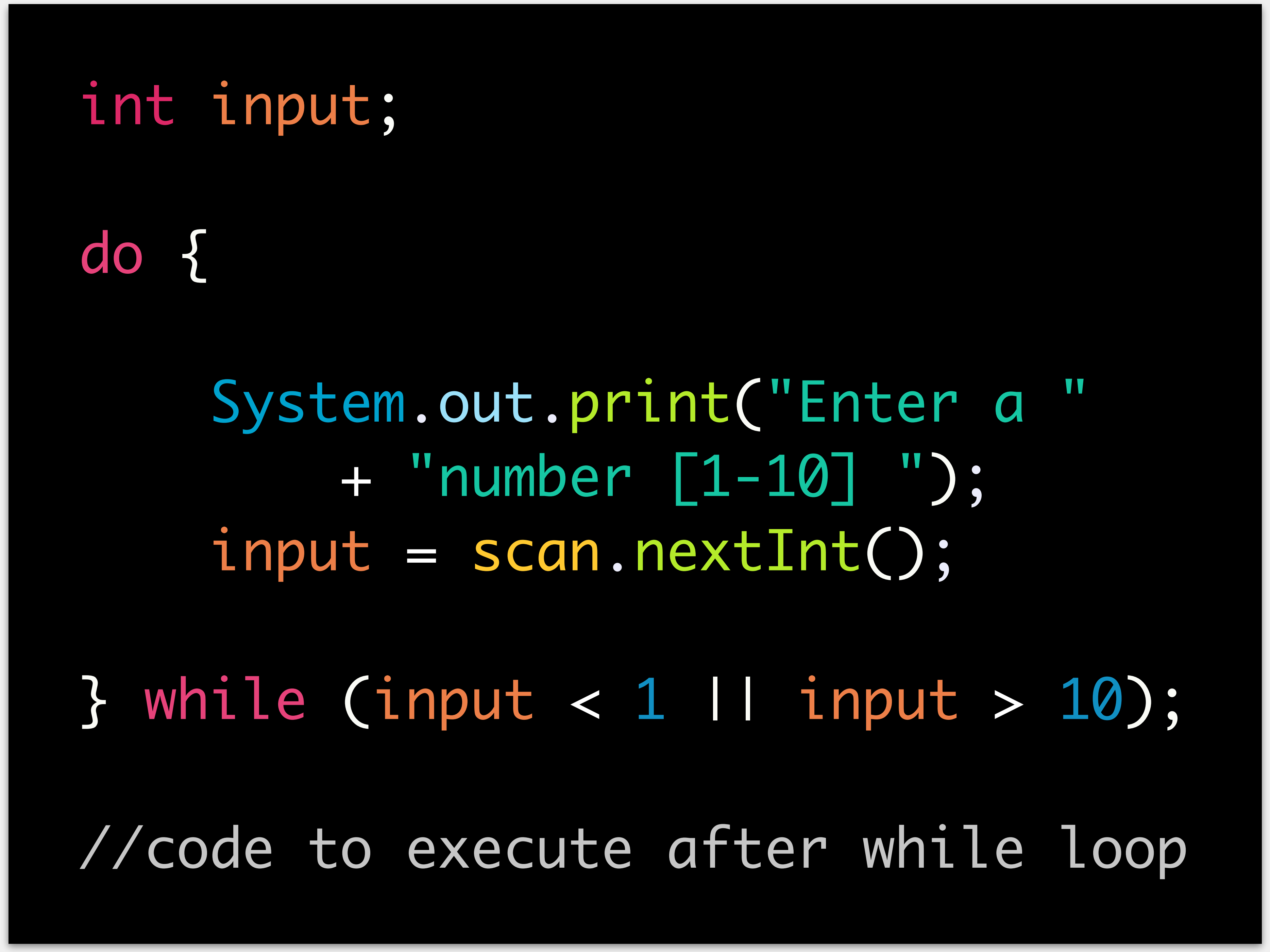
8

Why Do While Loops?

When we want to guarantee that our work is performed at least once

Primary use is for checking validity of user input

- we always want to ask for input once
- only if the input is invalid do we want to ask again



```
int input;

do {

    System.out.print("Enter a "
        + "number [1-10] ");
    input = scan.nextInt();

} while (input < 1 || input > 10);

//code to execute after while loop
```