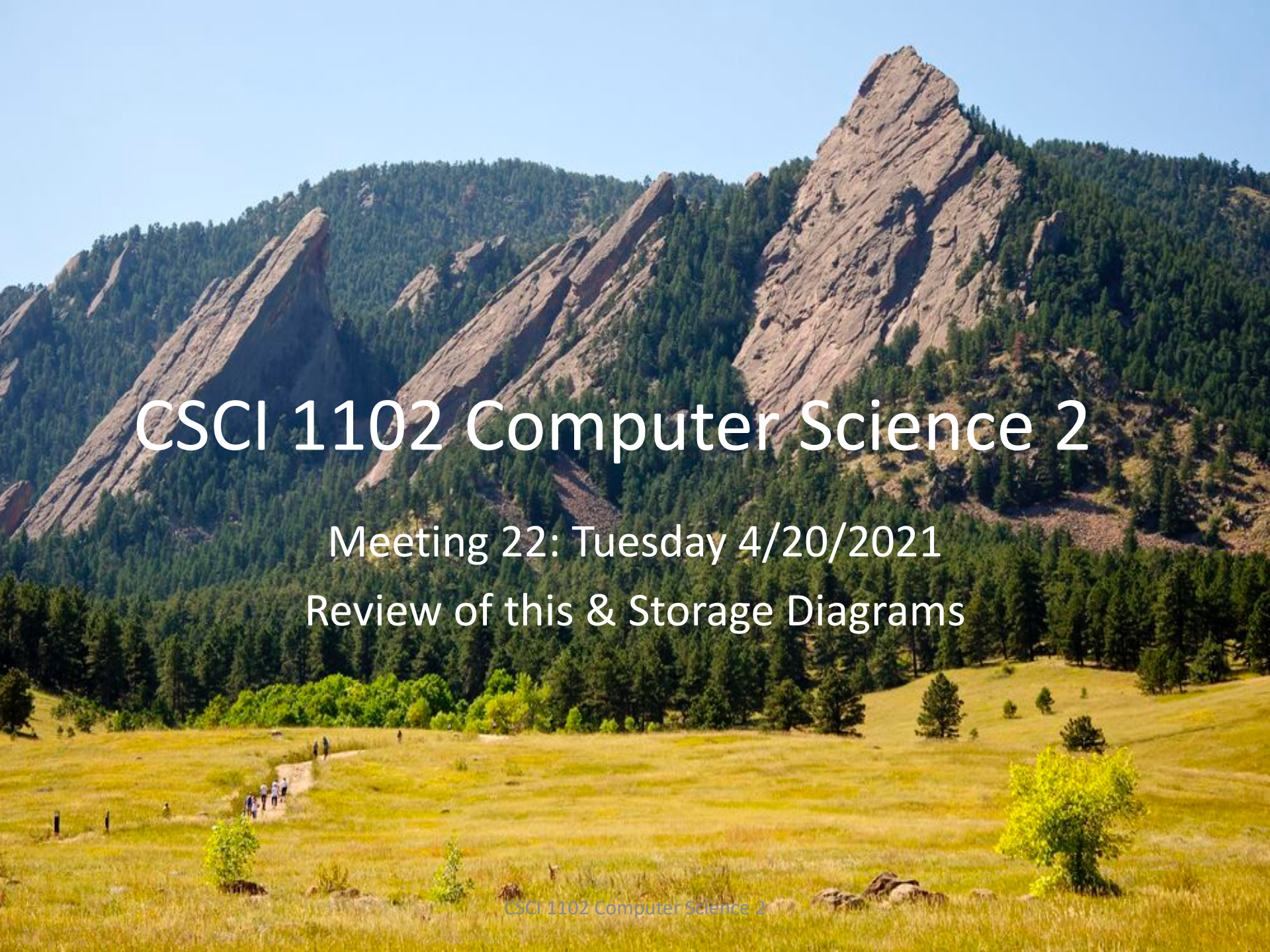


CSCI 1102 Computer Science 2

Meeting 22: Tuesday 4/20/2021

Review of this & Storage Diagrams

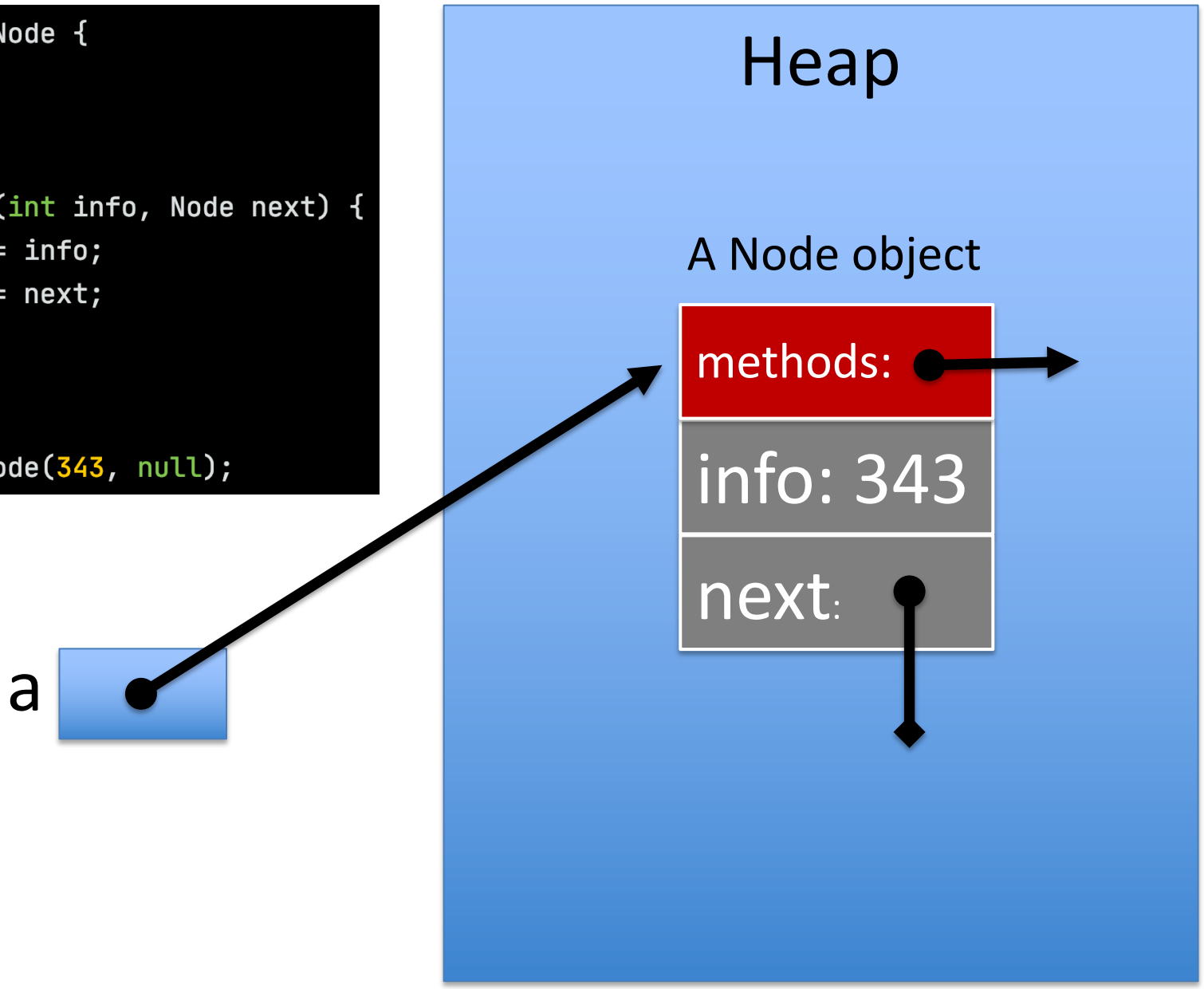

Barbara Liskov
Abstract Data Types
Turing Award 2008

Attribution & Creative Work

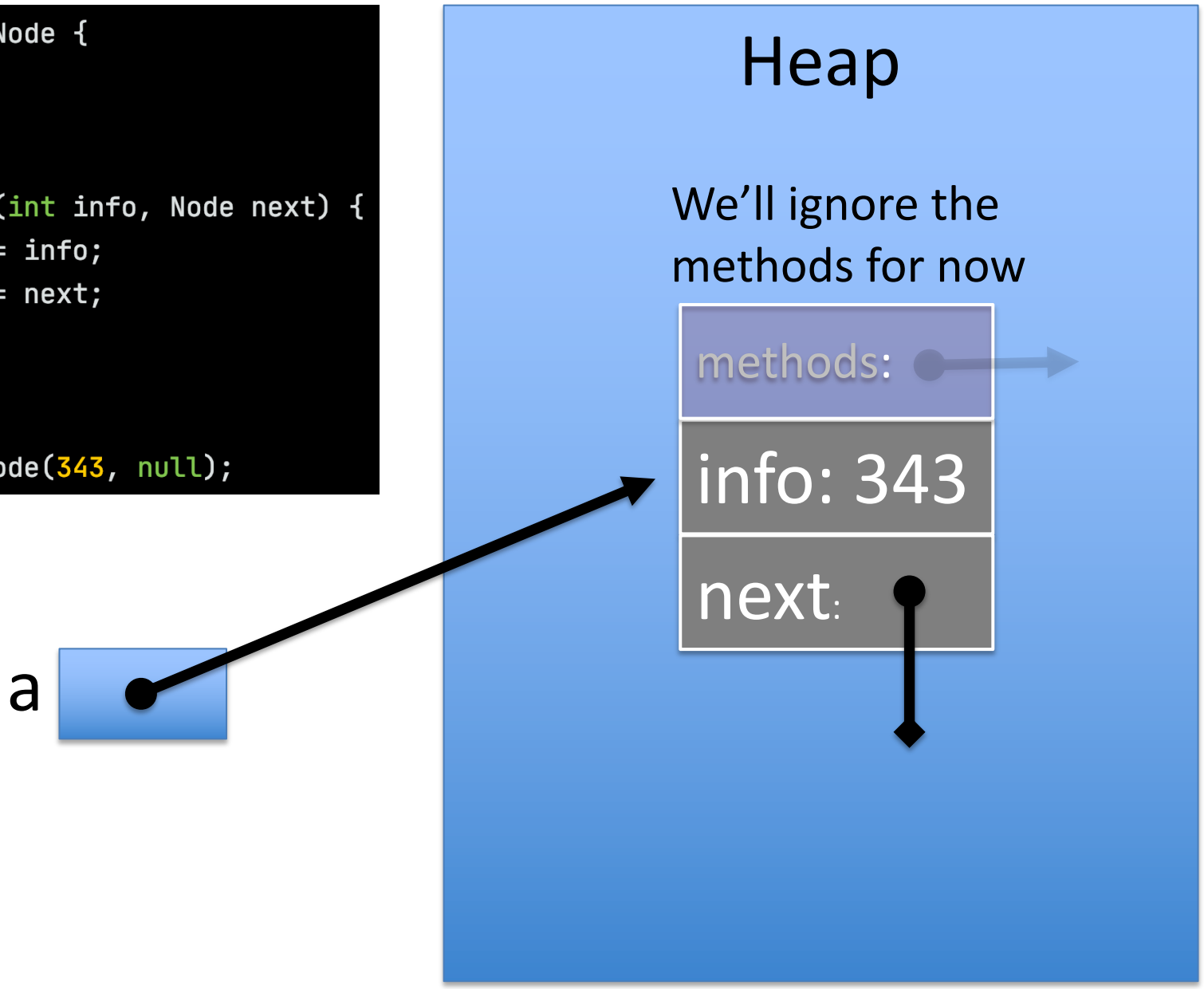
- /* Part 1 is cribbed from … source … */
- /* Algorithm A was partially inspired by … source … */
- /* I had several long talks with Martha about threaded trees …. */
- /* I found a related problem on geeksforgeeks.com and looked it over … */
- /* A related problem was posed in my MATH class, some of the ideas here are drawn from a solution to that problem. */

Review of this & Storage Diagrams

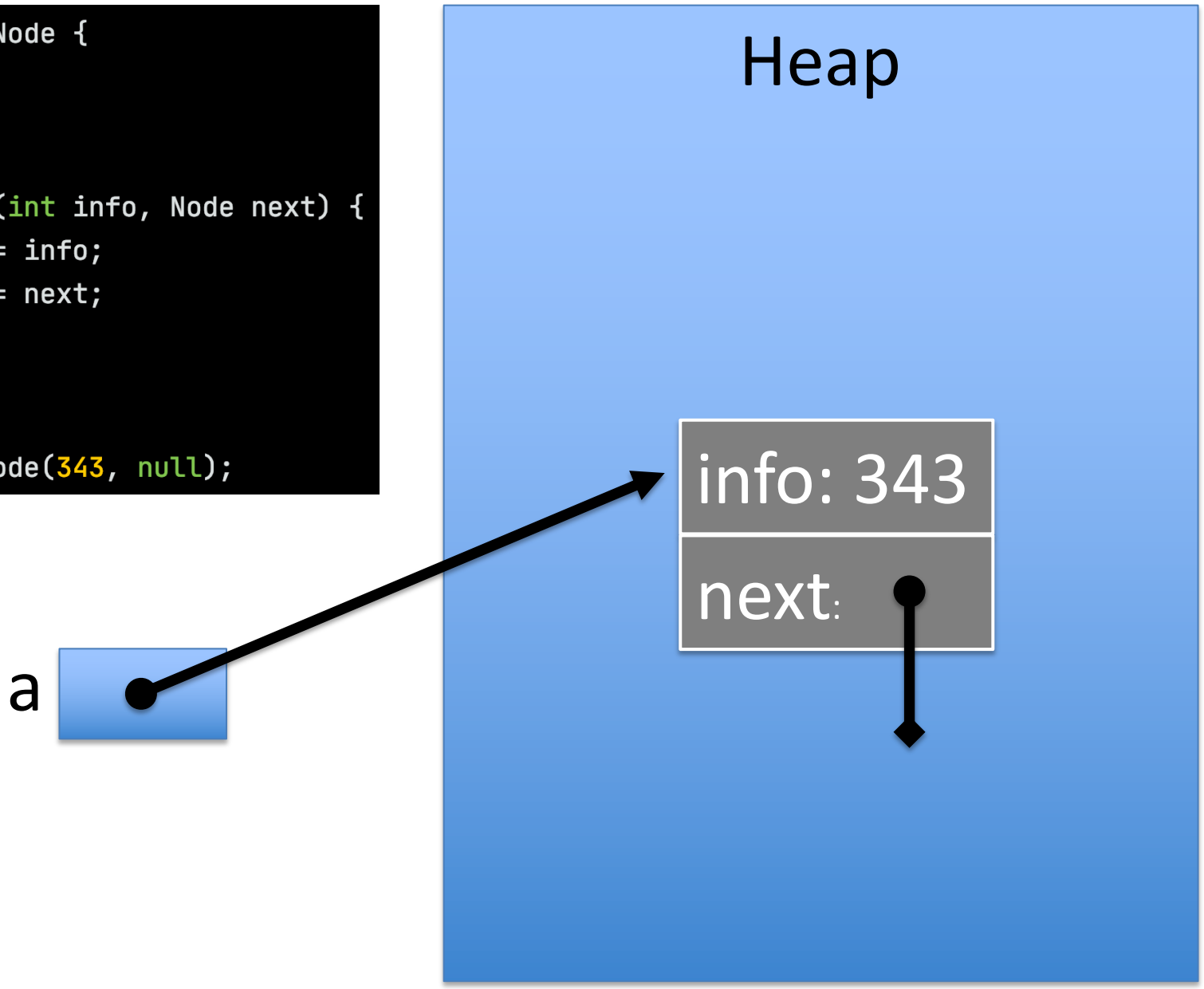
```java
private class Node {
  int info;
  Node next;

  private Node(int info, Node next) {
    this.info = info;
    this.next = next;
  }
}

Node a = new Node(343, null);
```



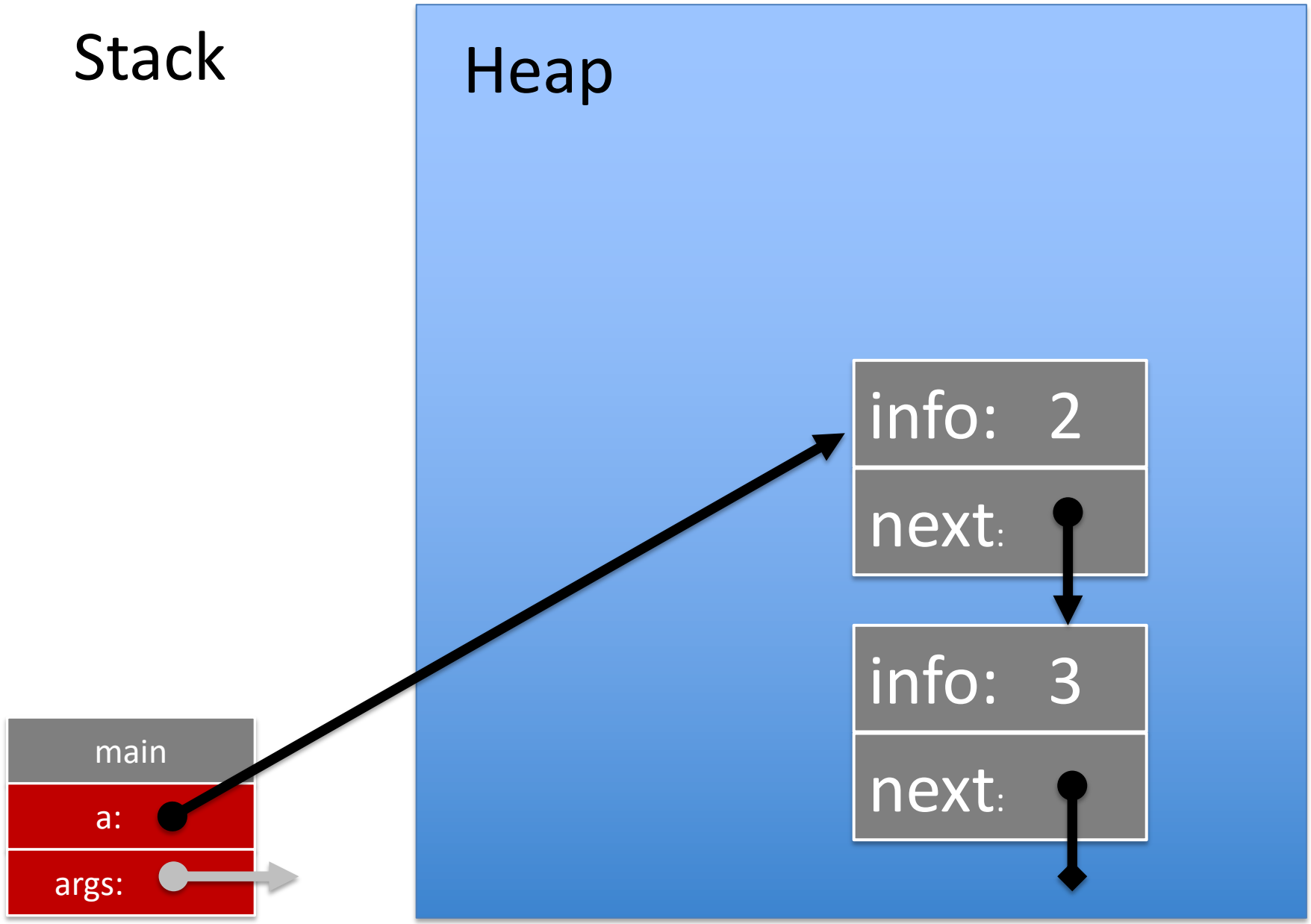
```
private class Node {
  int info;
  Node next;

  private Node(int info, Node next) {
    this.info = info;
    this.next = next;
  }
}

Node a = new Node(343, null);
```

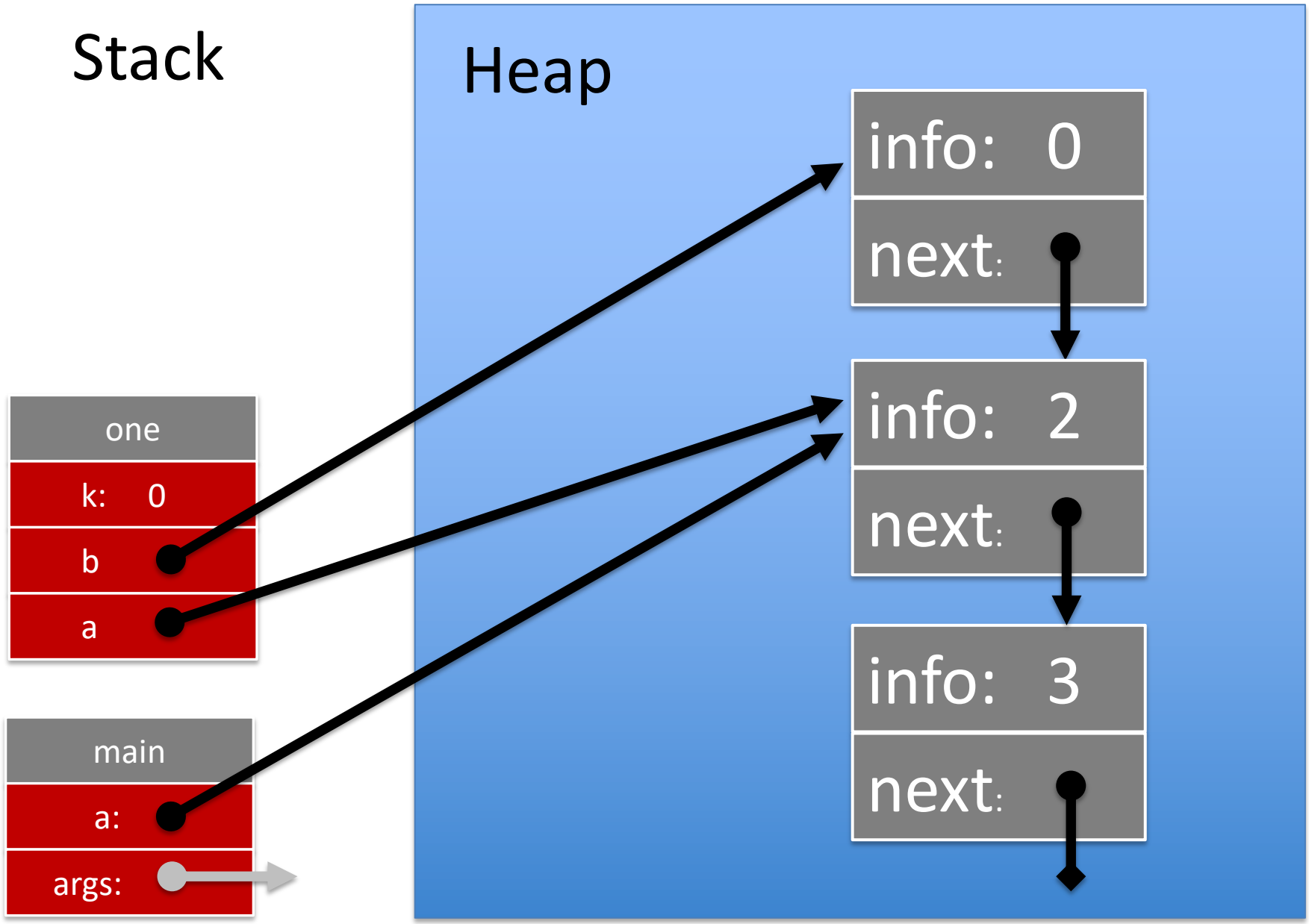


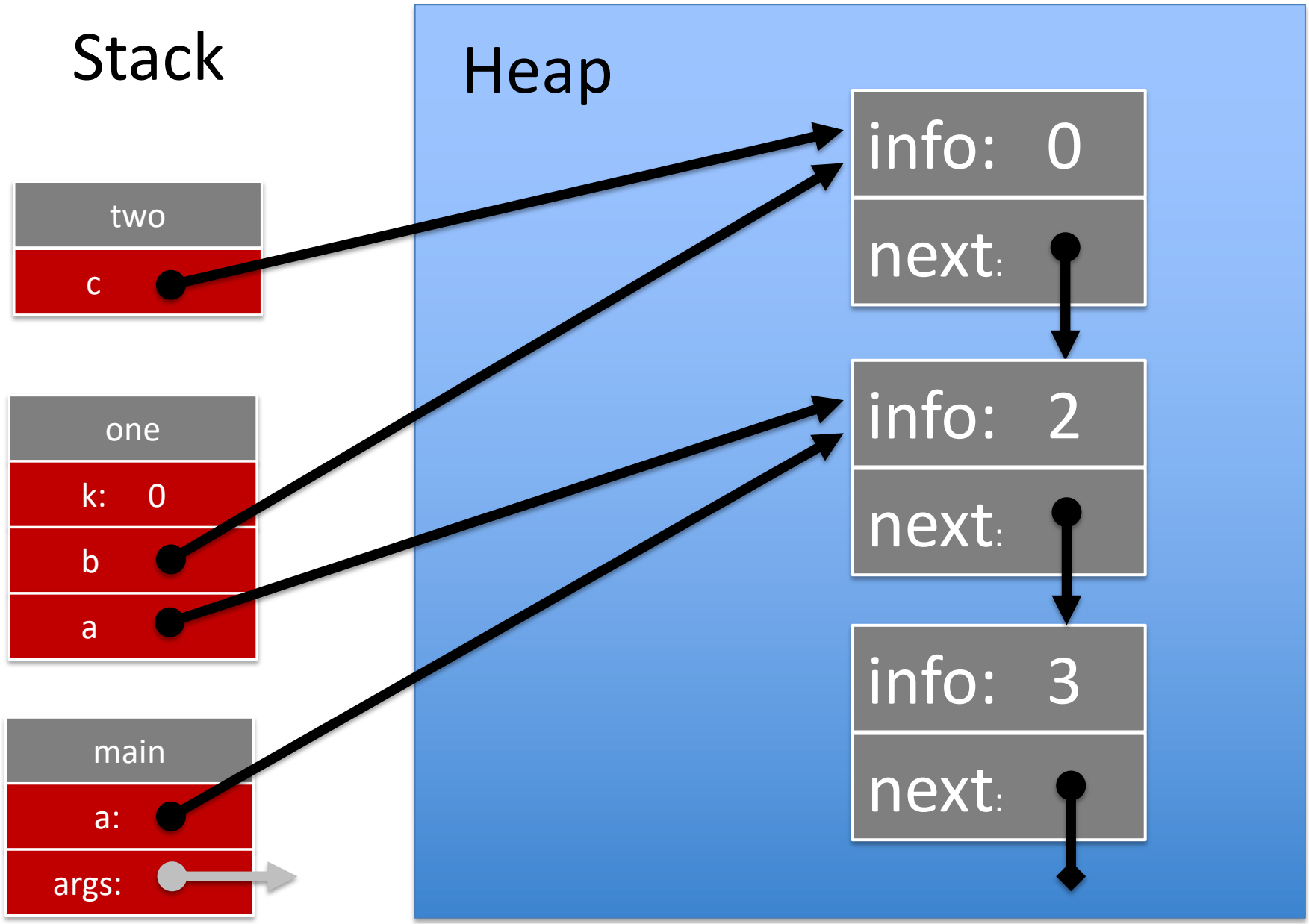
```java
private class Node {
  int info;
  Node next;

  private Node(int info, Node next) {
    this.info = info;
    this.next = next;
  }
}

Node a = new Node(343, null);
```

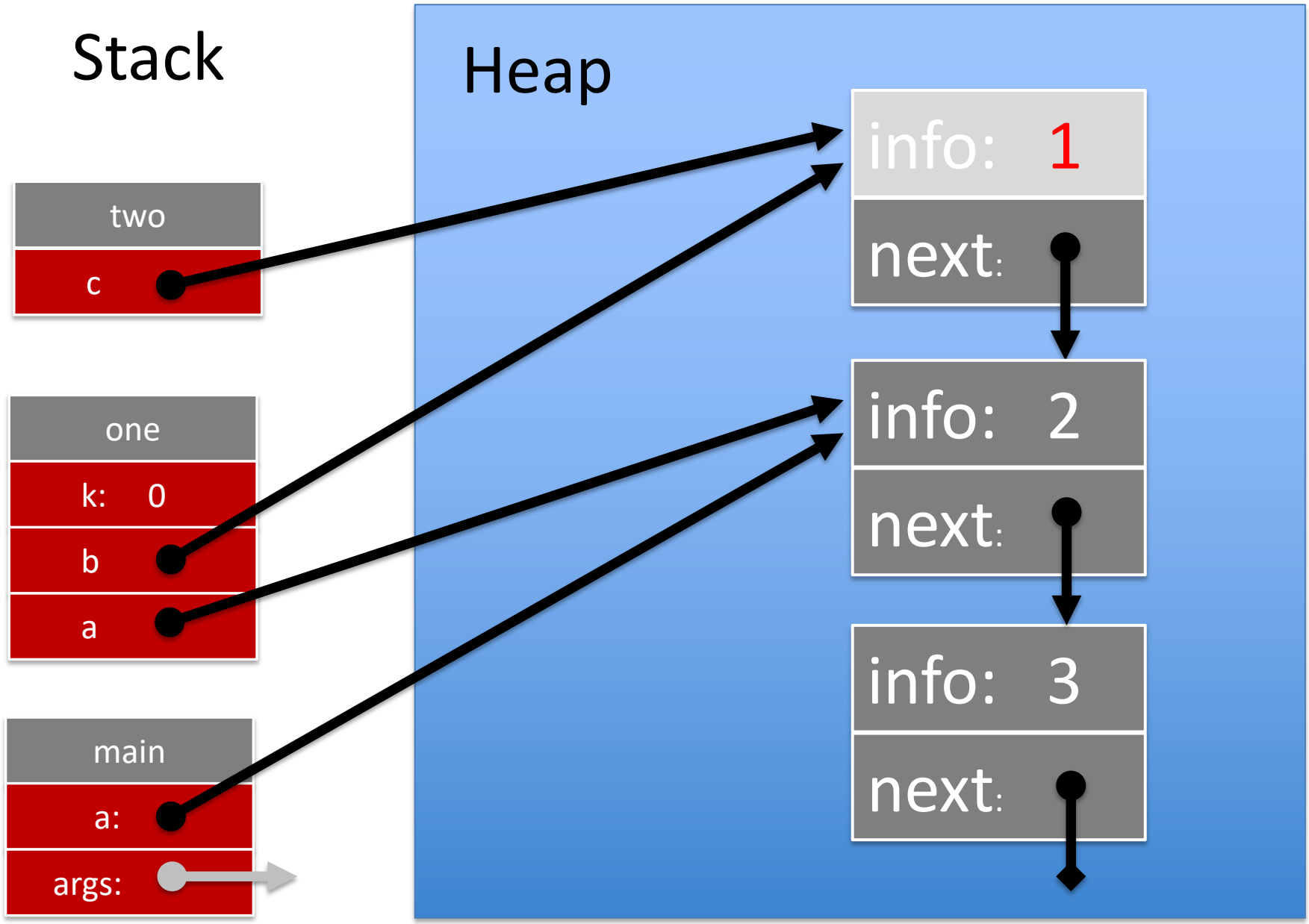


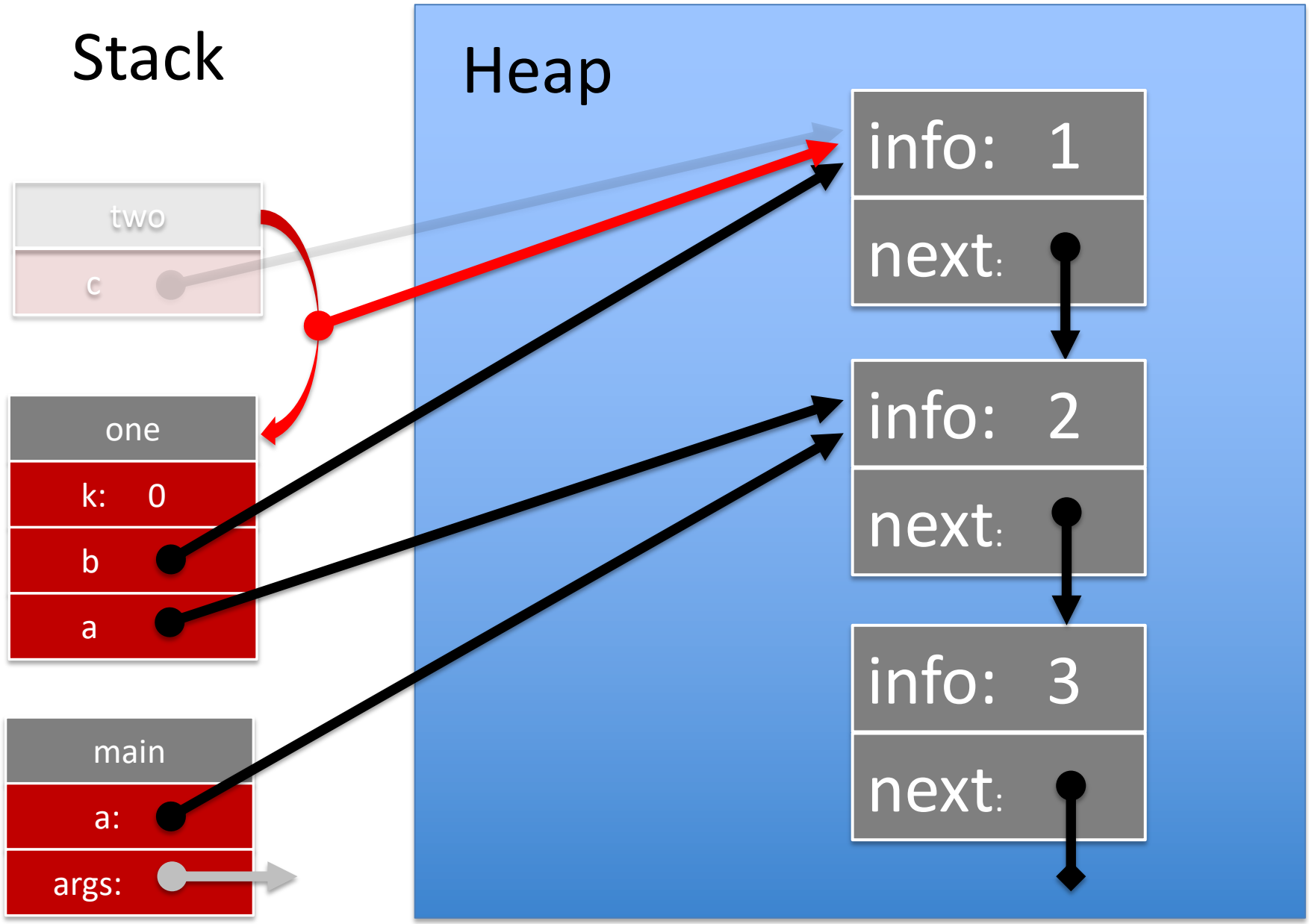
```
private static Node two(Node c) {
  c.info = c.info + 1;
  return c;
}

private static Node one(int k, Node a) {
  Node b = new Node(k, a);
  return two(b);
}

public static void main(String[] args) {
  Node a = new Node(2, new Node(3, null));
  a = one(0, a);
}
```

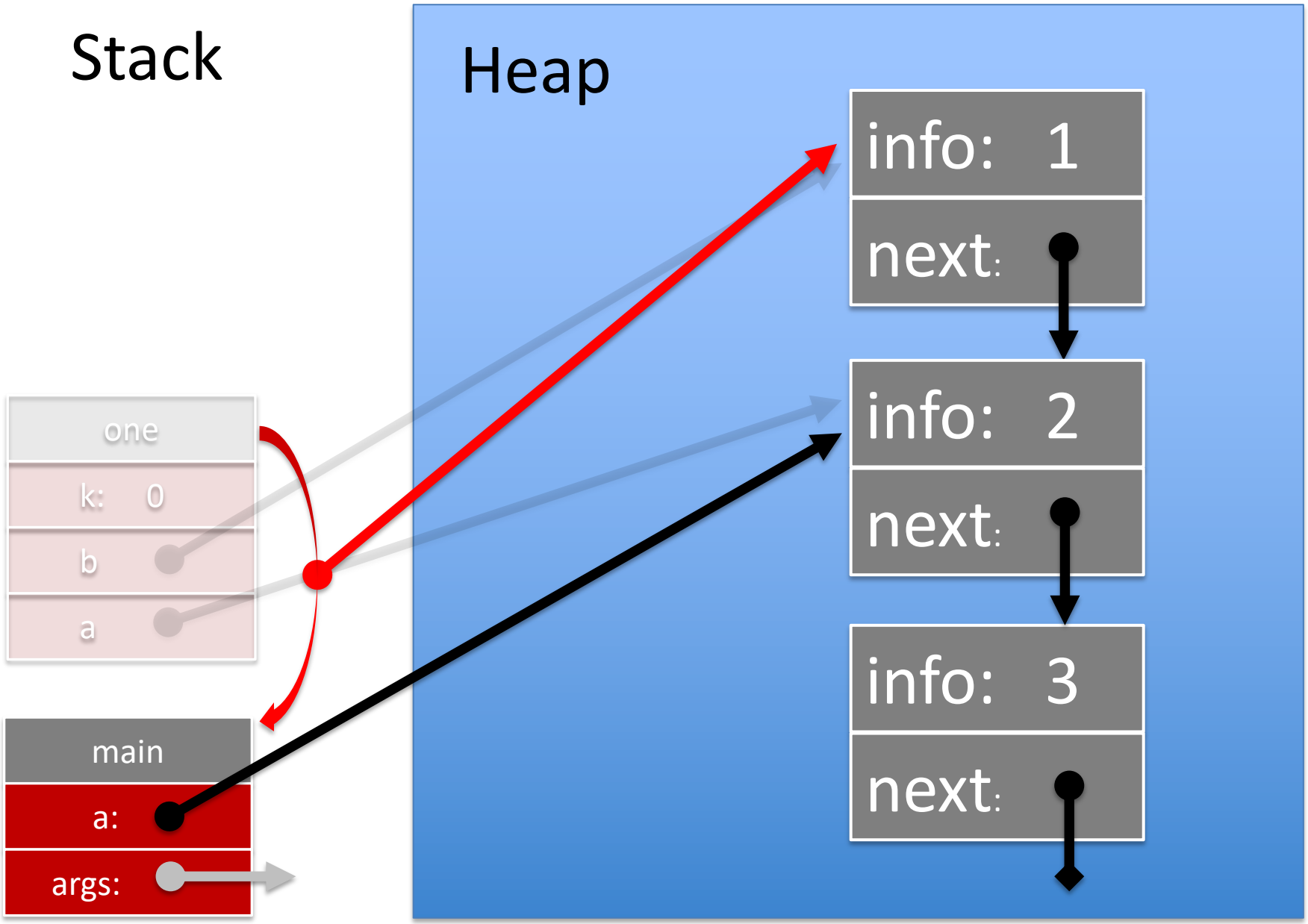

Stack
Heap
info: 2
next:
info: 3
next:
main
a:
args:

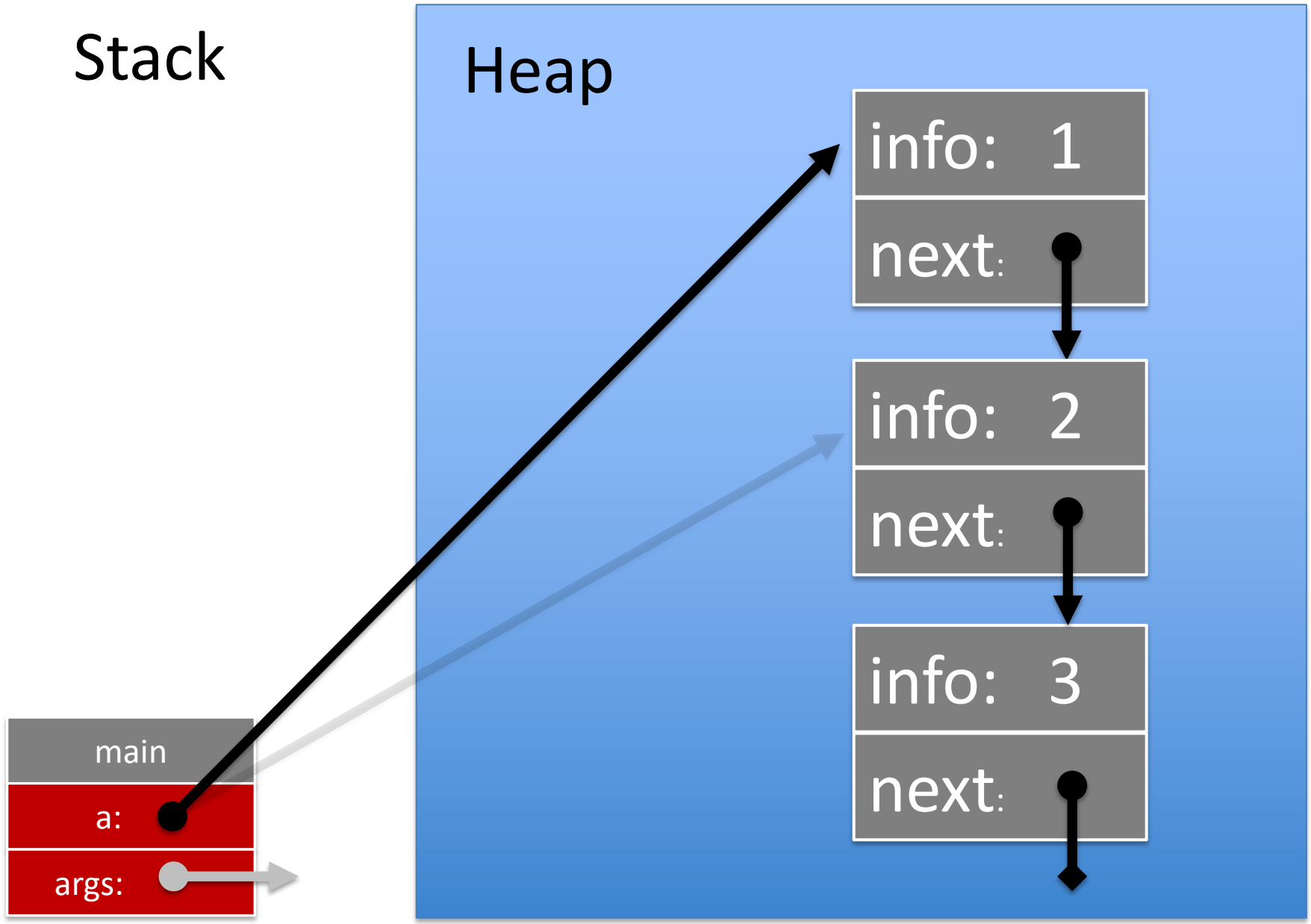

Stack
Heap
info: 0
next:
info: 2
next:
info: 3
next:
one
k: 0
b
a
main
a:
args:


Stack
Heap
two
c
one
k: 0
b
a
main
a:
args:
info: 0
next:
info: 2
next:
info: 3
next:


Stack
Heap
two
c
one
k: 0
b
a
main
a:
args:
info: 1
next:
info: 2
next:
info: 3
next:


Stack
Heap
two
c
one
k: 0
b
a
main
a:
args:
info: 1
next:
info: 2
next:
info: 3
next:

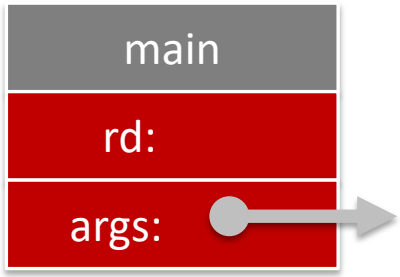

Stack
Heap
info: 1
next:
info: 2
next:
info: 3
next:
one
k: 0
b
a
main
a:
args:

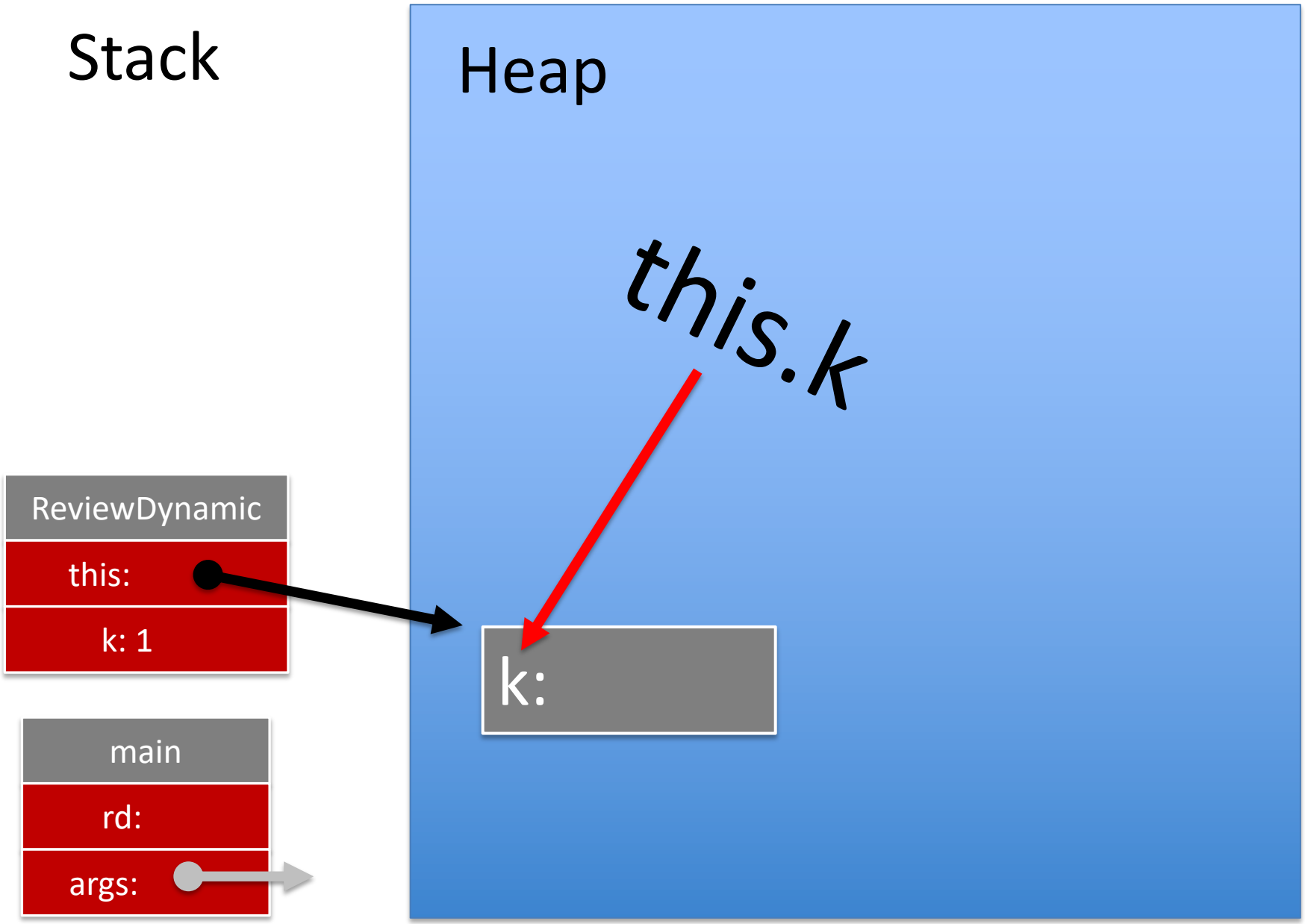

Stack
Heap
info: 1
next:
info: 2
next:
info: 3
next:
main
a:
args:

```java
private int k;

public ReviewDynamic(int k) {
  this.k = k;
}

private Node two(Node b) {
  Node c = new Node(this.k, node);
  return c;
}

private void one() {
  Node a = new Node(2, new Node(3, null));
  a = this.two(a);
  return a;
}

public static void main(String[] args) {
  ReviewDynamic rd = new ReviewDynamic(1);
  rd.one();
}
```

Stack

Heap

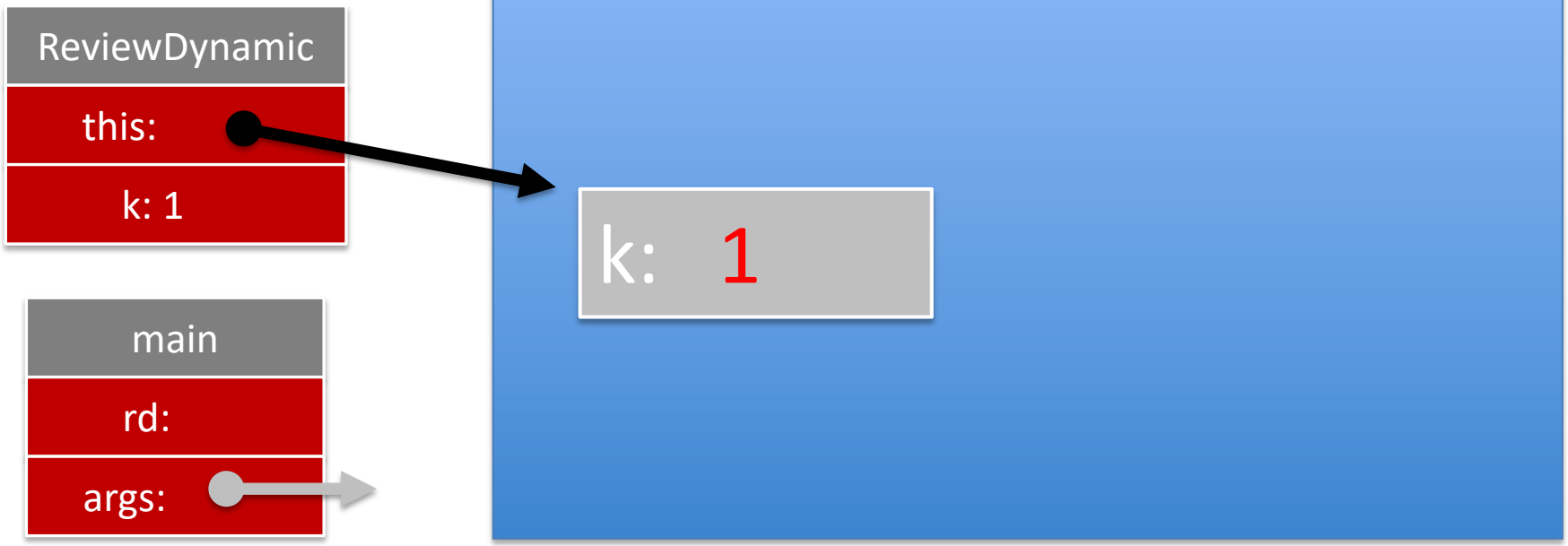


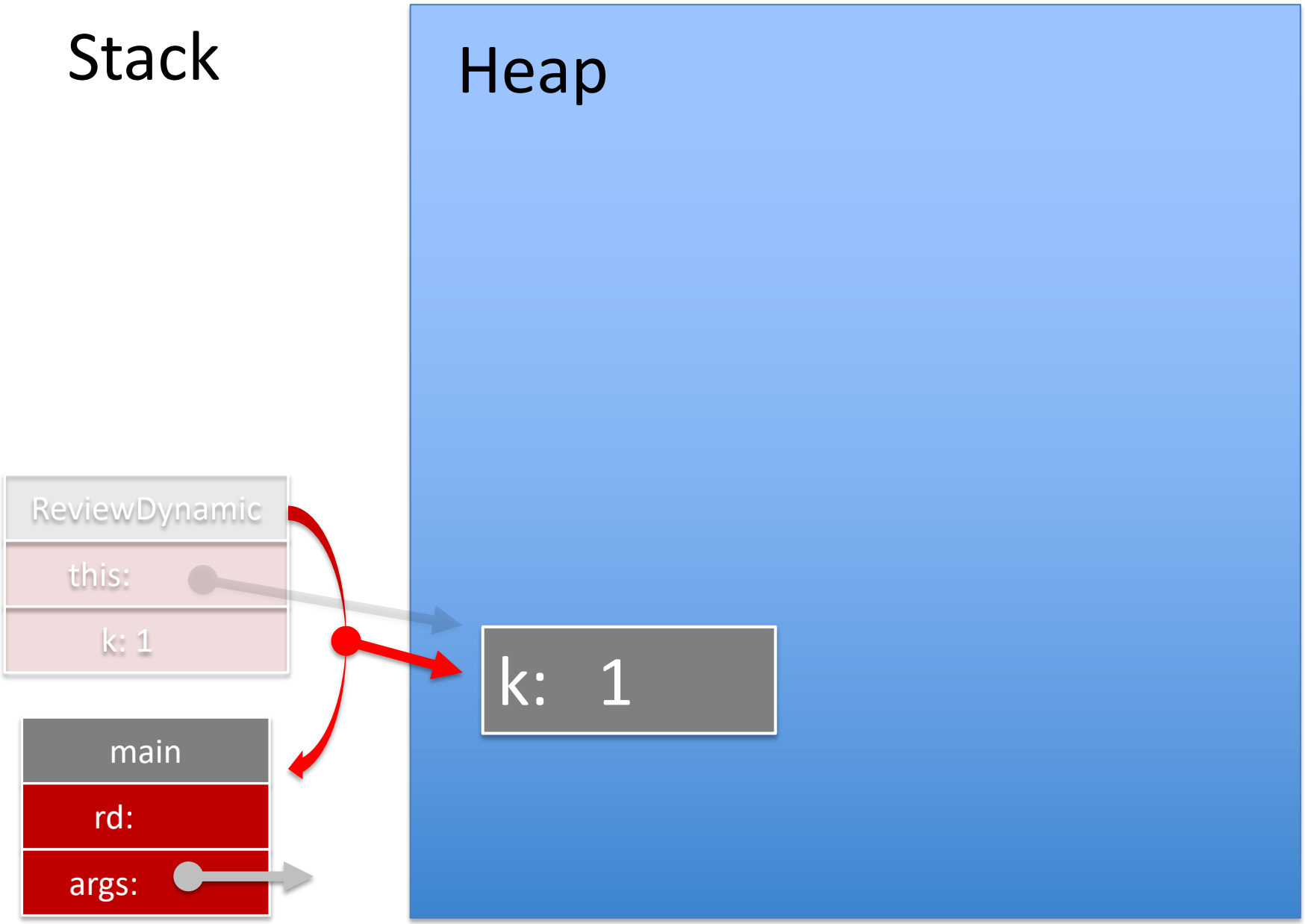

Stack
Heap
this.k
ReviewDynamic
this:
k: 1
k:
main
rd:
args:

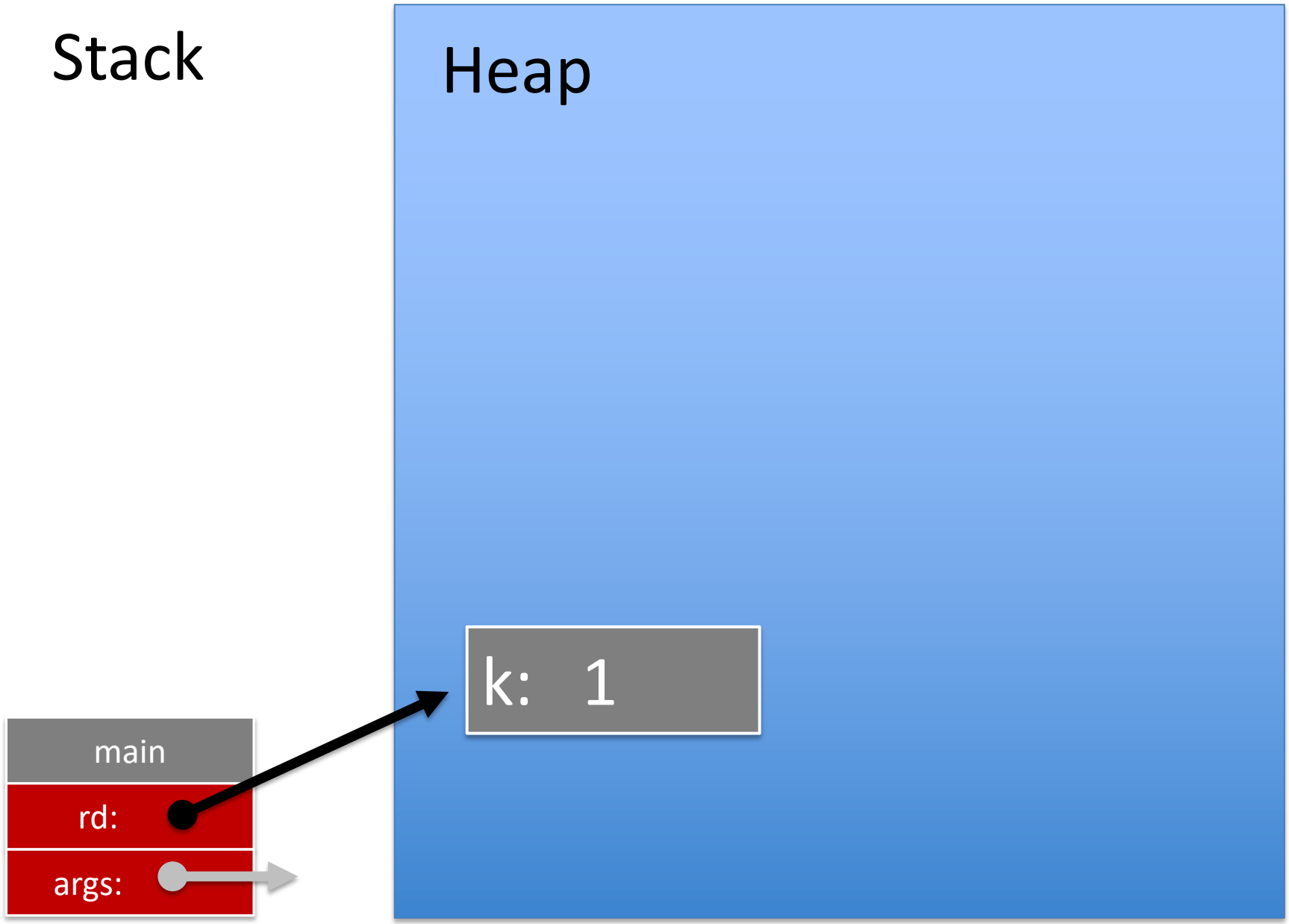
Stack

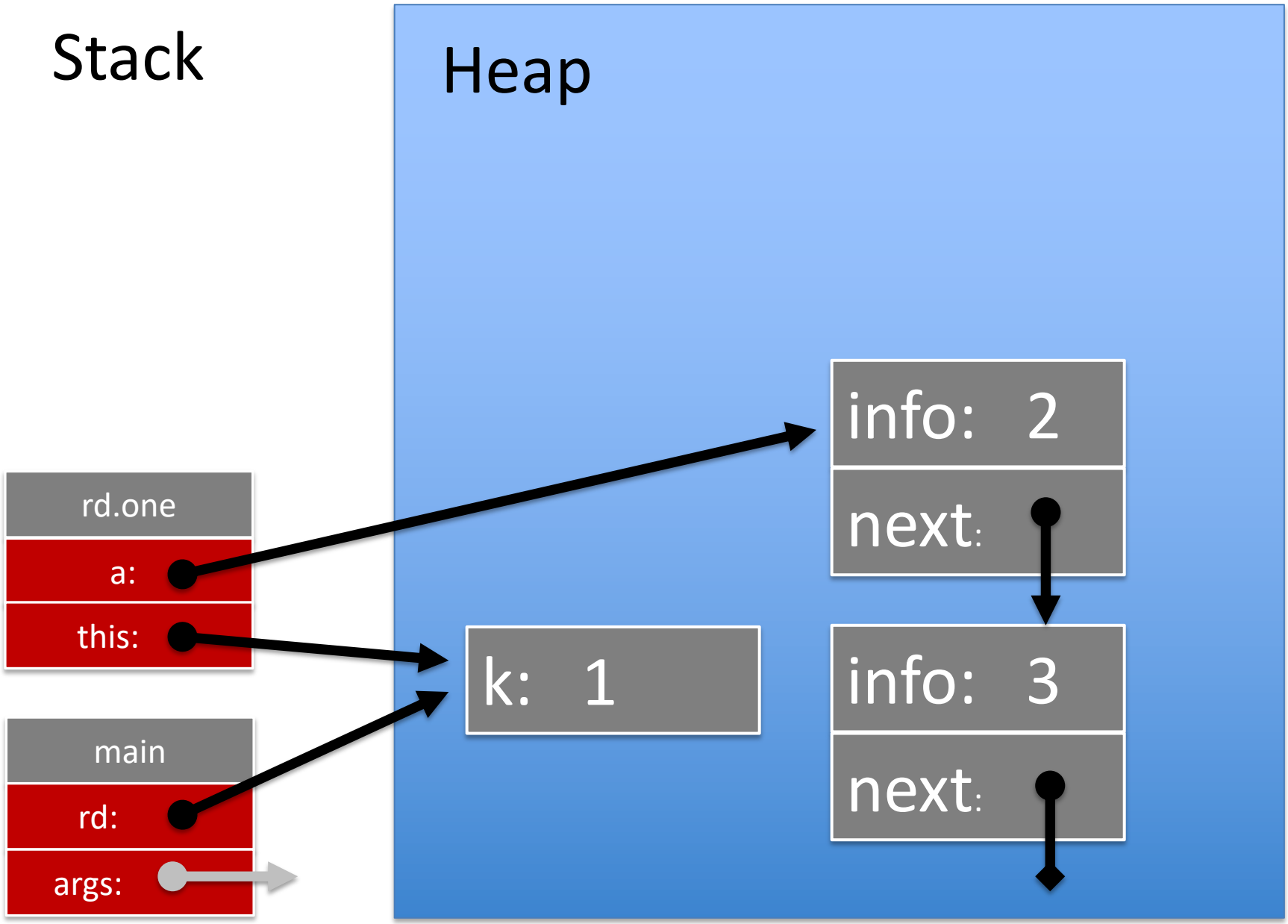
Heap

```
public ReviewDynamic(int k) {
    this.k = k;
}
```

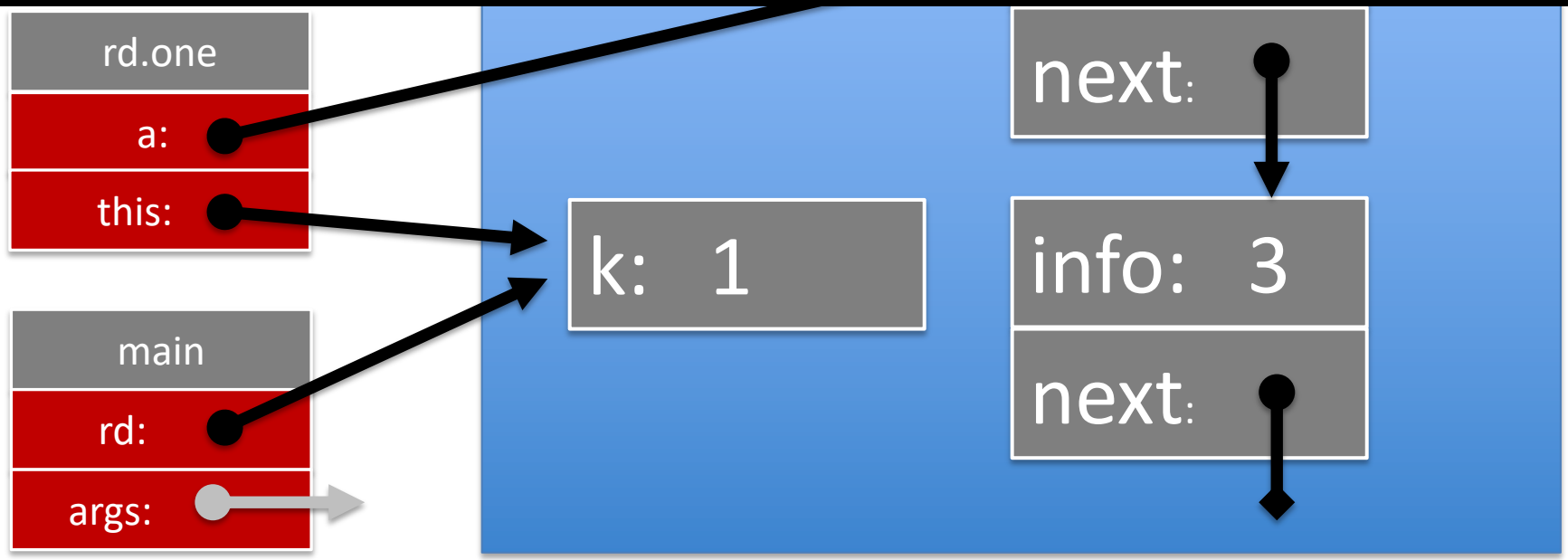



Stack
Heap
ReviewDynamic
this:
k: 1
k: 1
main
rd:
args:


Stack
Heap
k: 1
main
rd:
args:

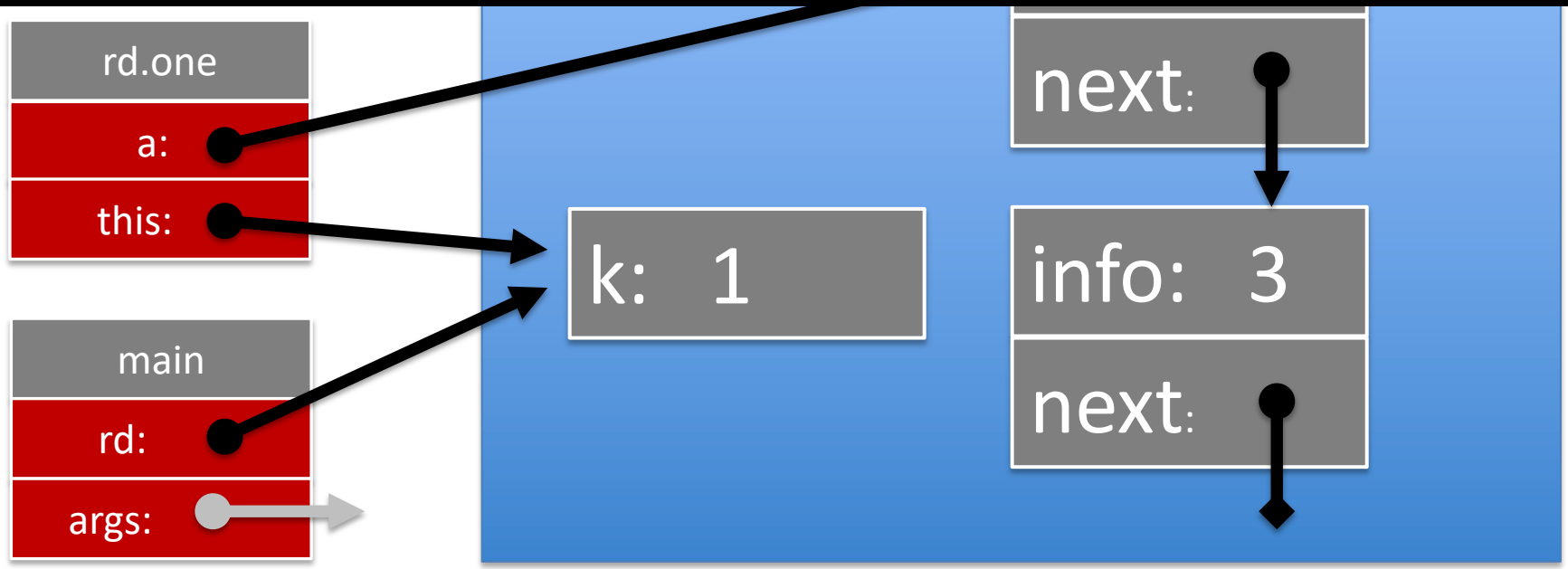

Stack
Heap
rd.one
a:
this:
main
rd:
args:
k: 1
info: 2
next:
info: 3
next:

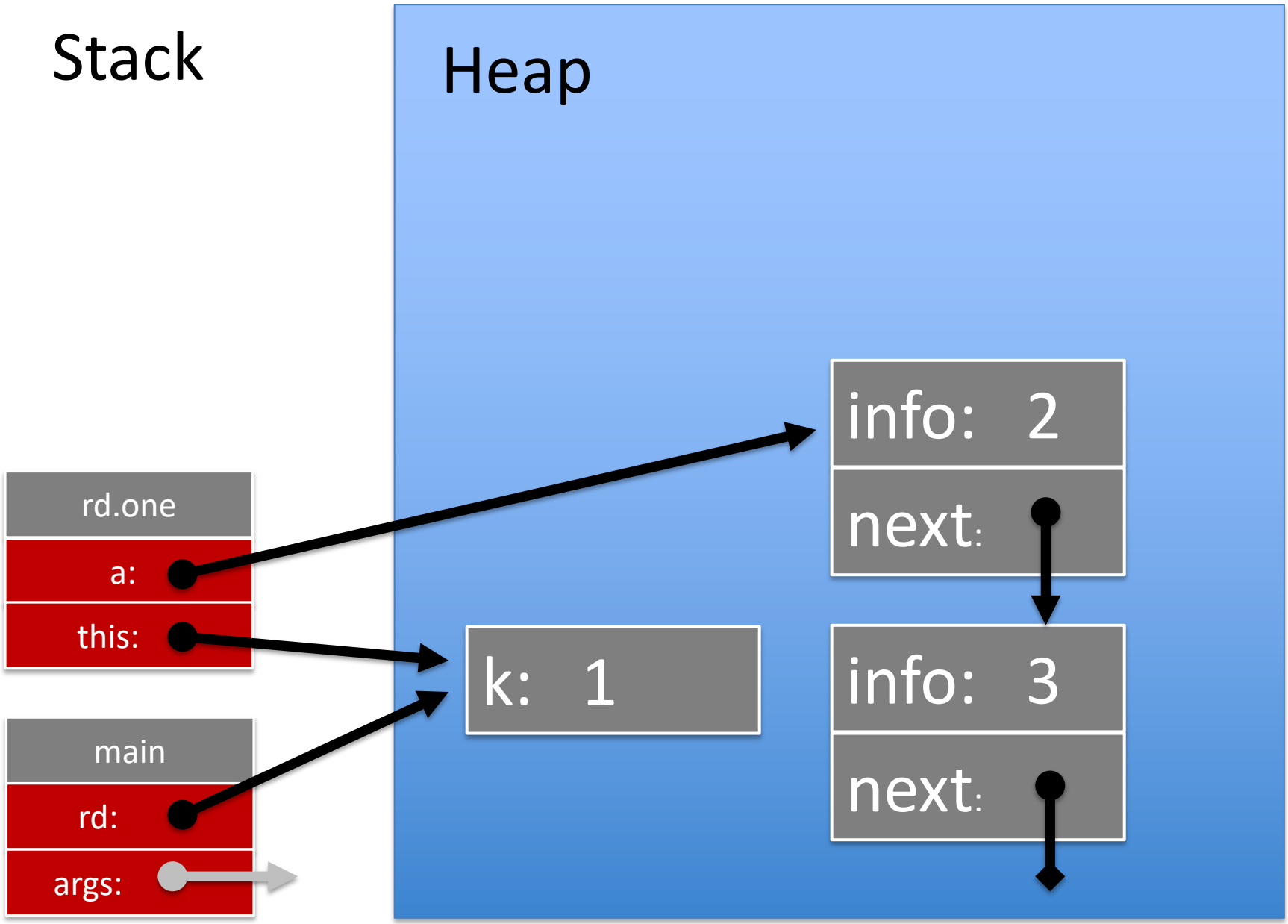
```java
private void one() {
  Node a = new Node(2, new Node(3, null));
  a = two(a);
  return a;
}

public static void main(String[] args) {
  ReviewDynamic rd = new ReviewDynamic(1);
  rd.one();
}
```

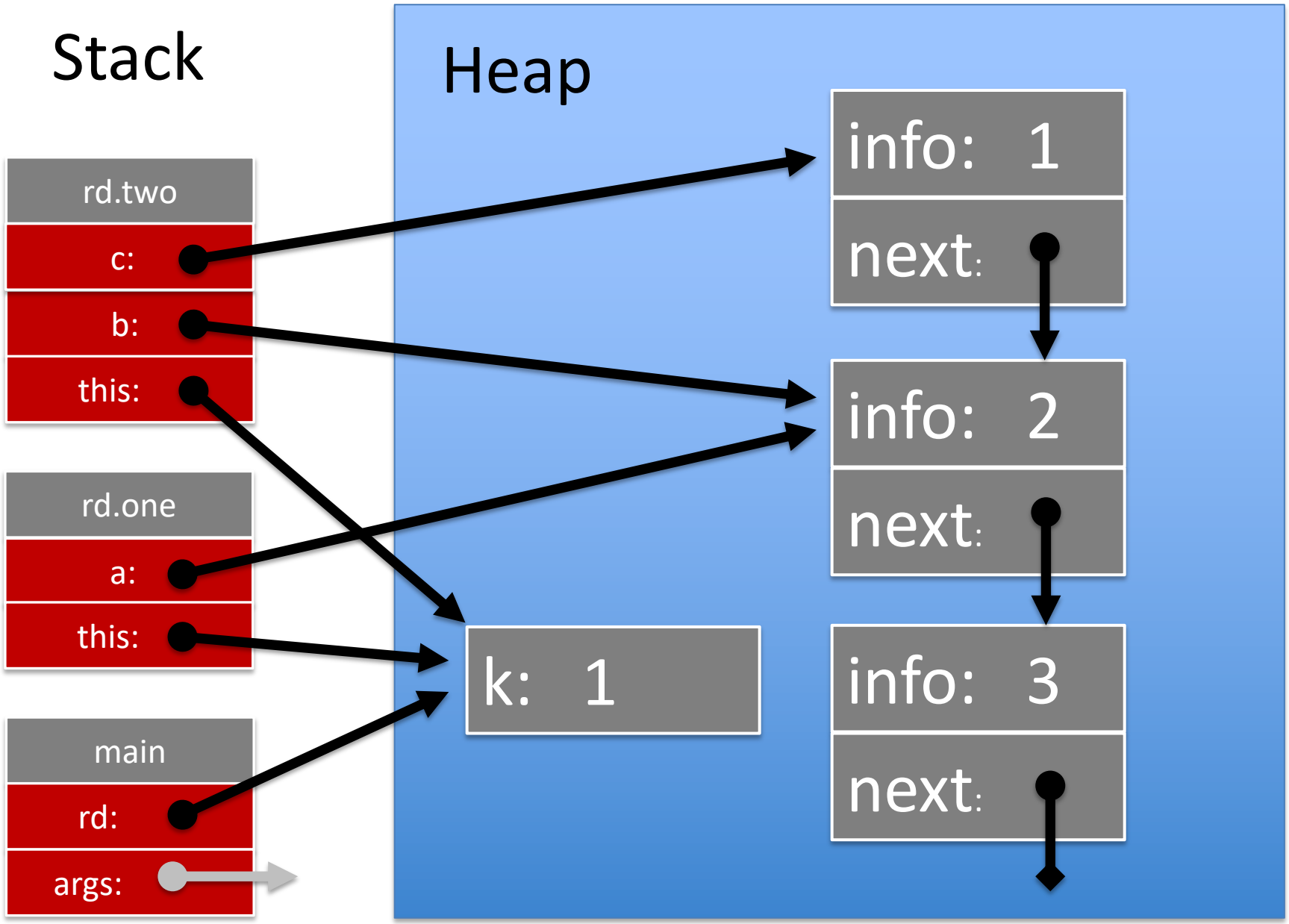


```
private void one(ReviewDynamic this) {
  Node a = new Node(2, new Node(3, null));
  a = two(a);
  return a;
}

public static void main(String[] args) {
  ReviewDynamic rd = new ReviewDynamic(1);
  rd.one(rd);
}
```

Compiler adds this parameter and extra argument

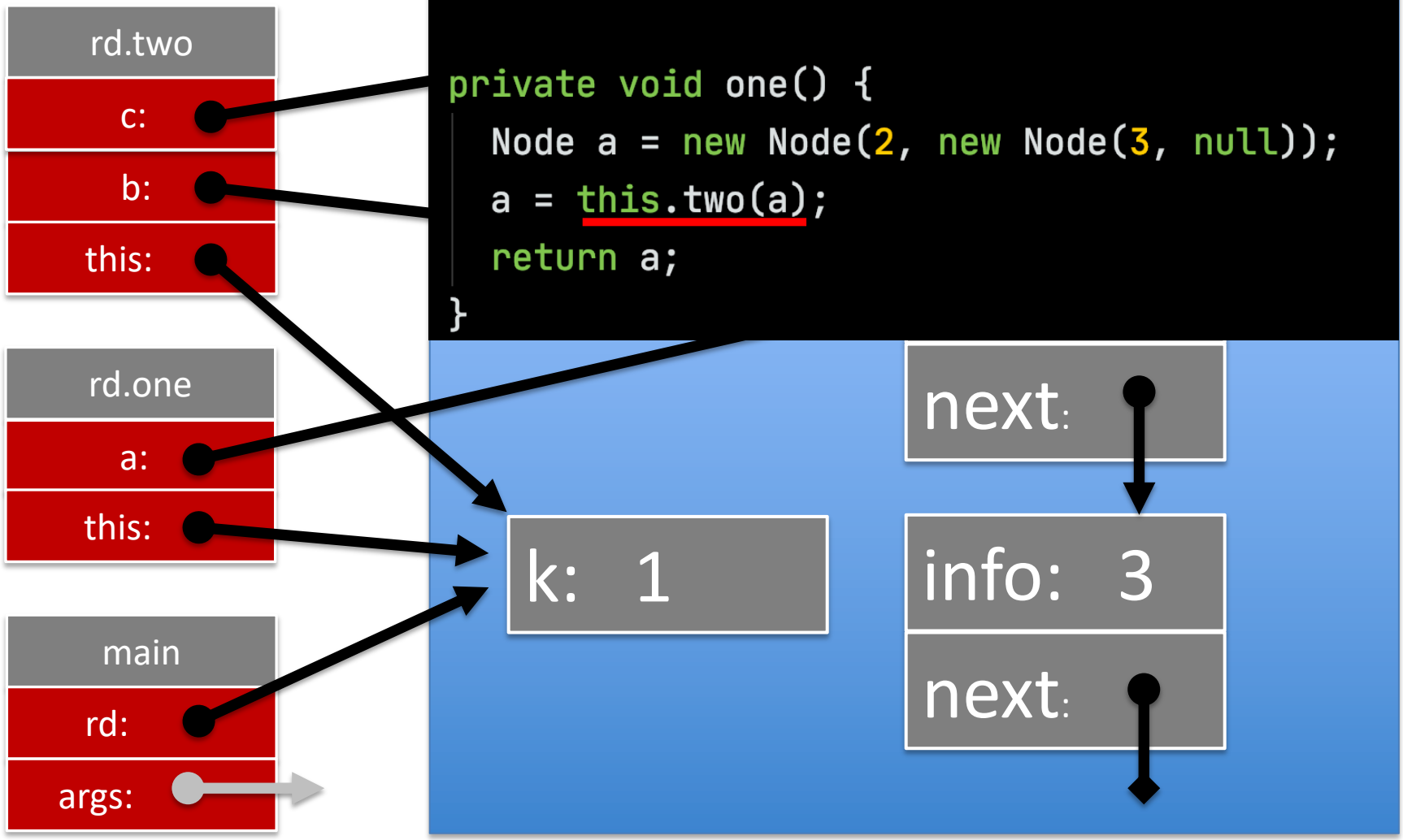



Stack
Heap
rd.one
a:
this:
main
rd:
args:
k: 1
info: 2
next:
info: 3
next:


Stack
Heap
rd.two
c:
b:
this:
rd.one
a:
this:
main
rd:
args:
k: 1
info: 1
next:
info: 2
next:
info: 3
next:

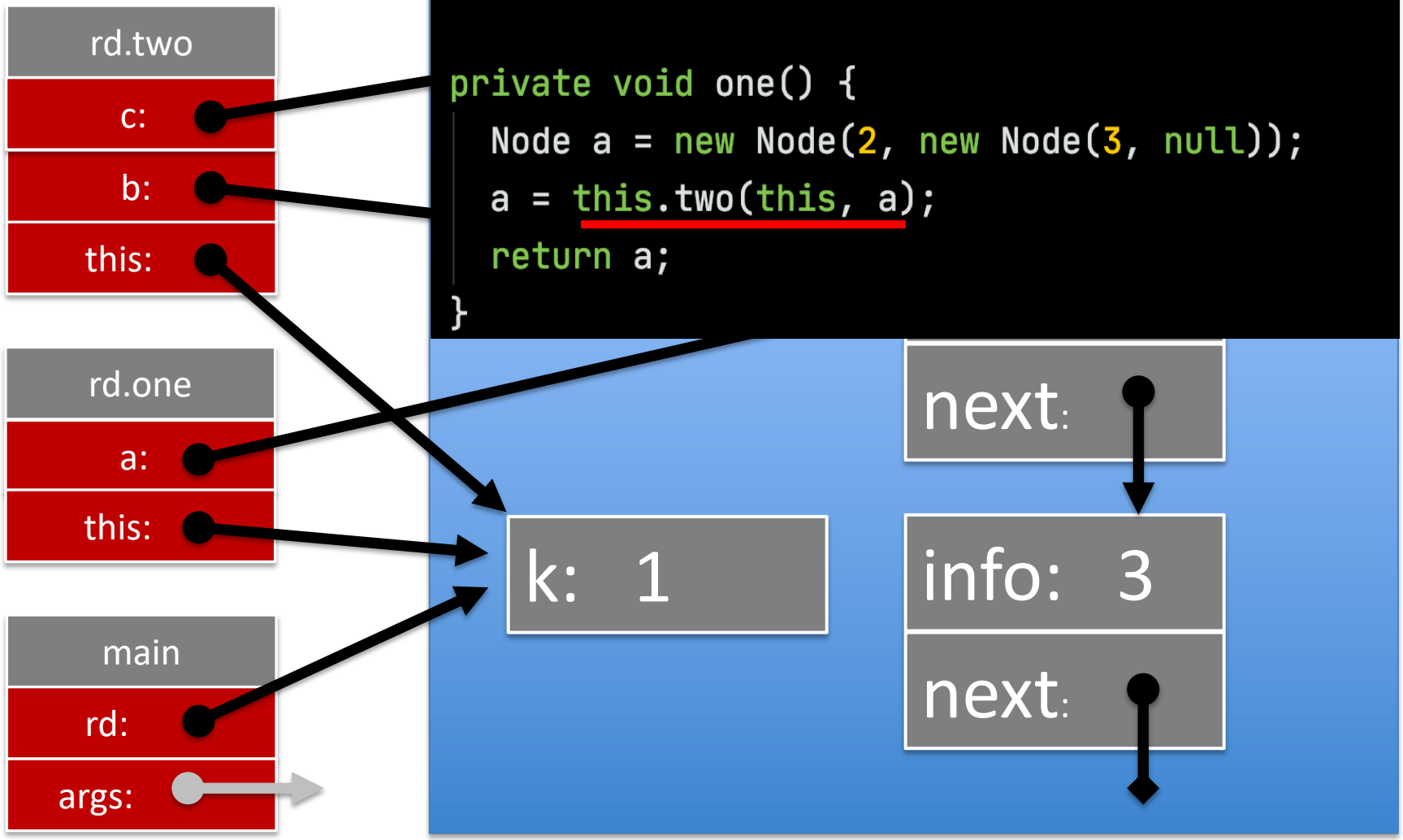
Stack

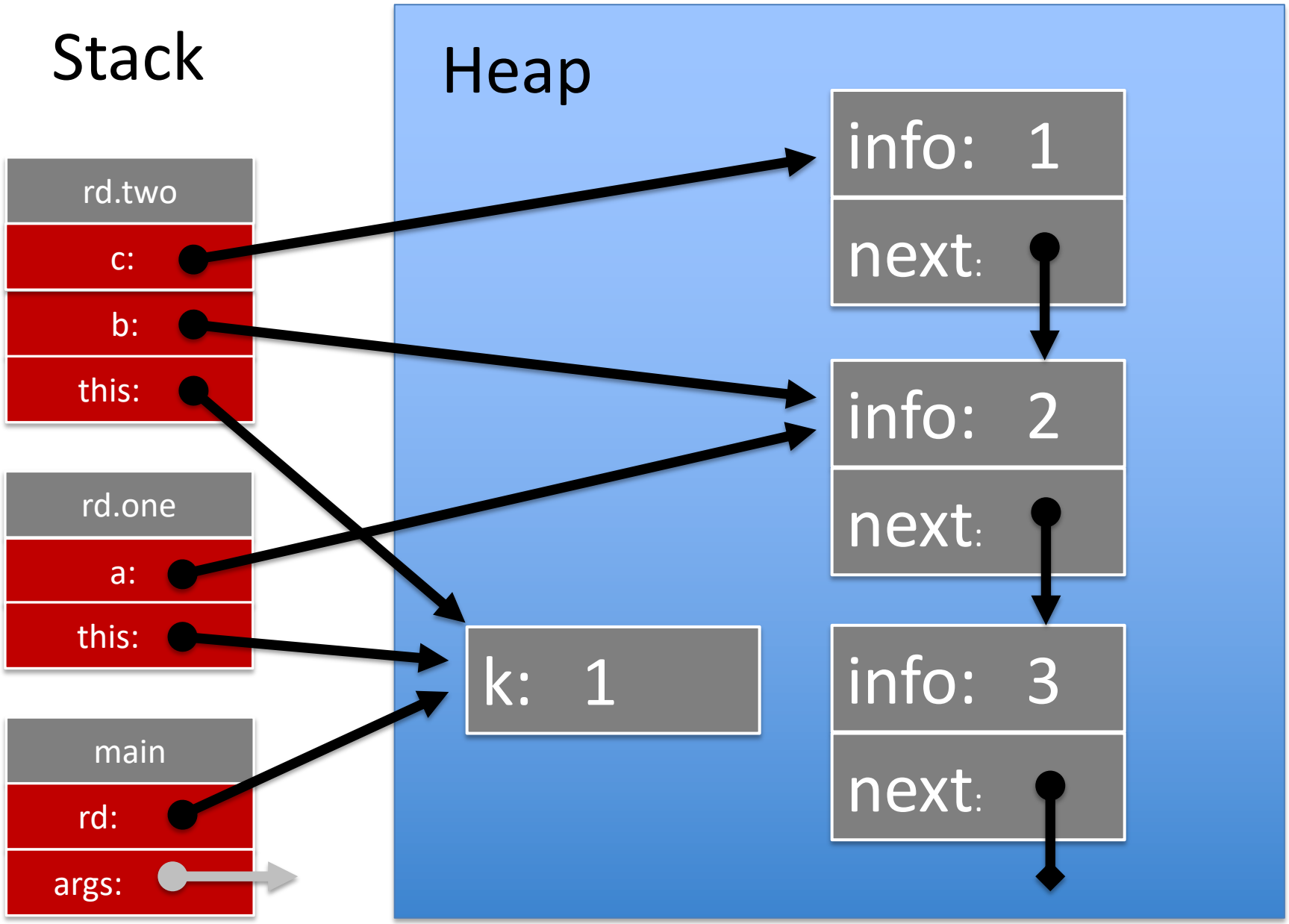
```
private Node two(Node b) {
  Node c = new Node(this.k, node);
  return c;
}

private void one() {
  Node a = new Node(2, new Node(3, null));
  a = this.two(a);
  return a;
}
```

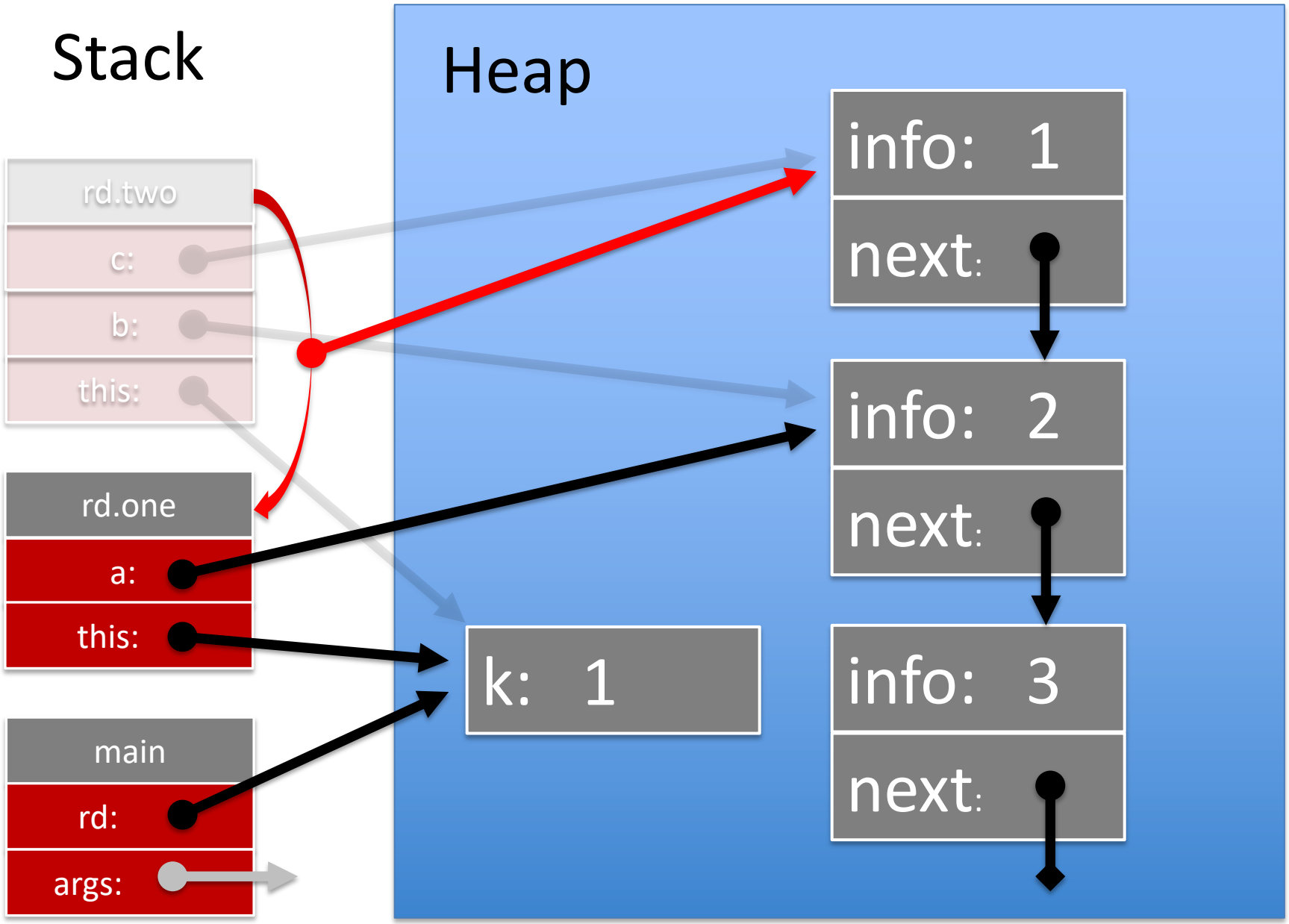


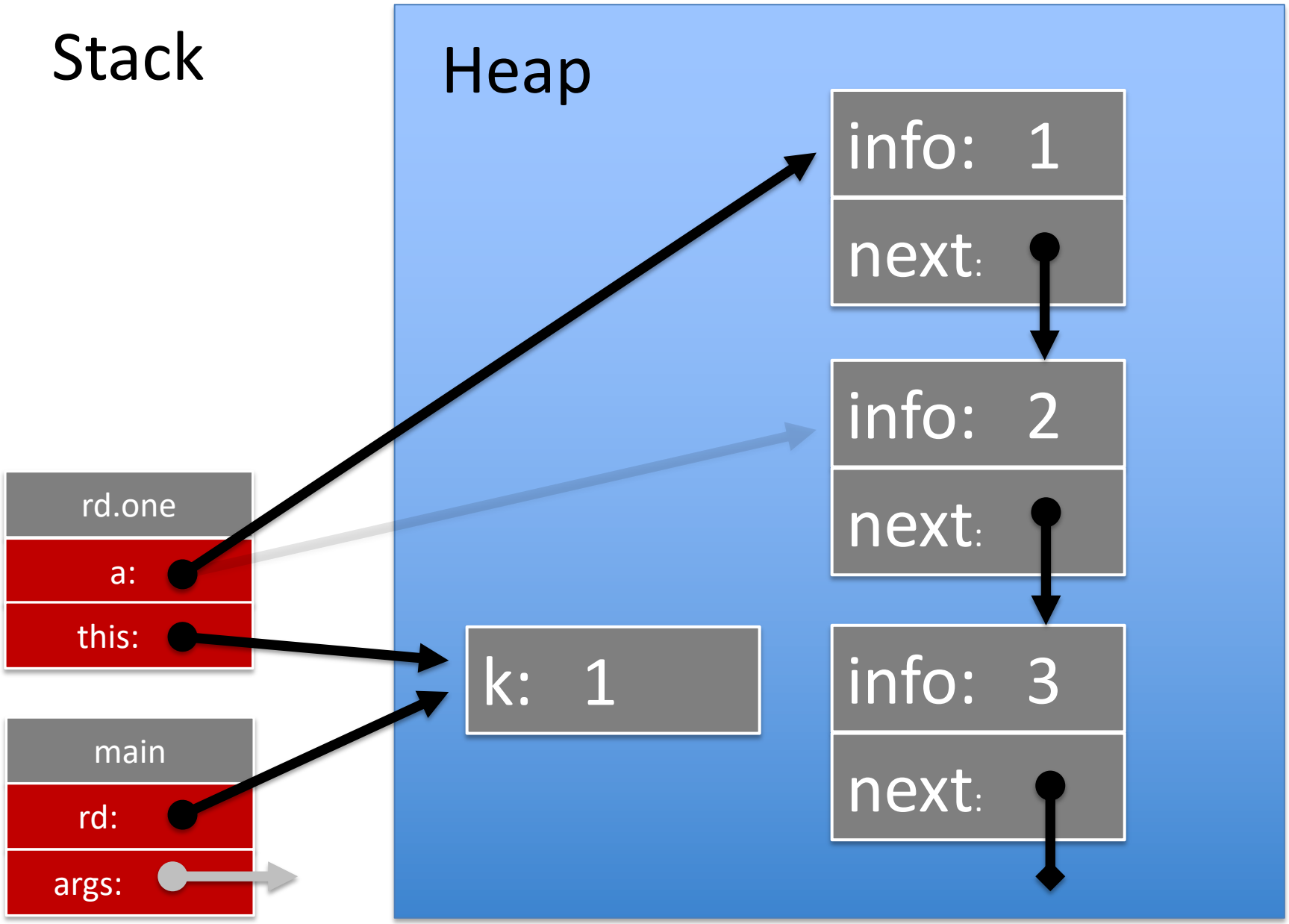
Stack

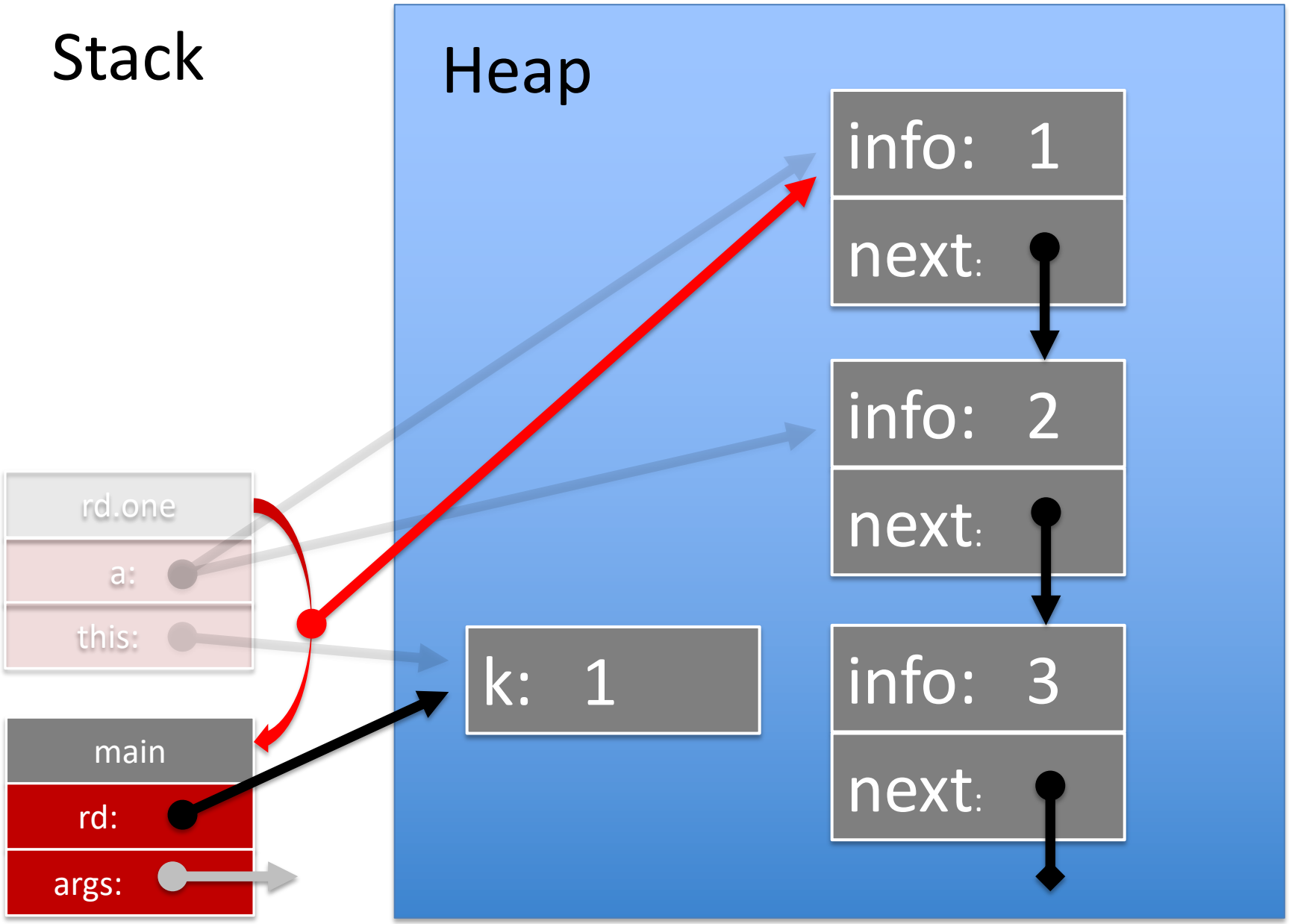
```
private Node two(ReviewDynamic this, Node b) {
  Node c = new Node(this.k, node);
  return c;
}

private void one() {
  Node a = new Node(2, new Node(3, null));
  a = this.two(this, a);
  return a;
}
```

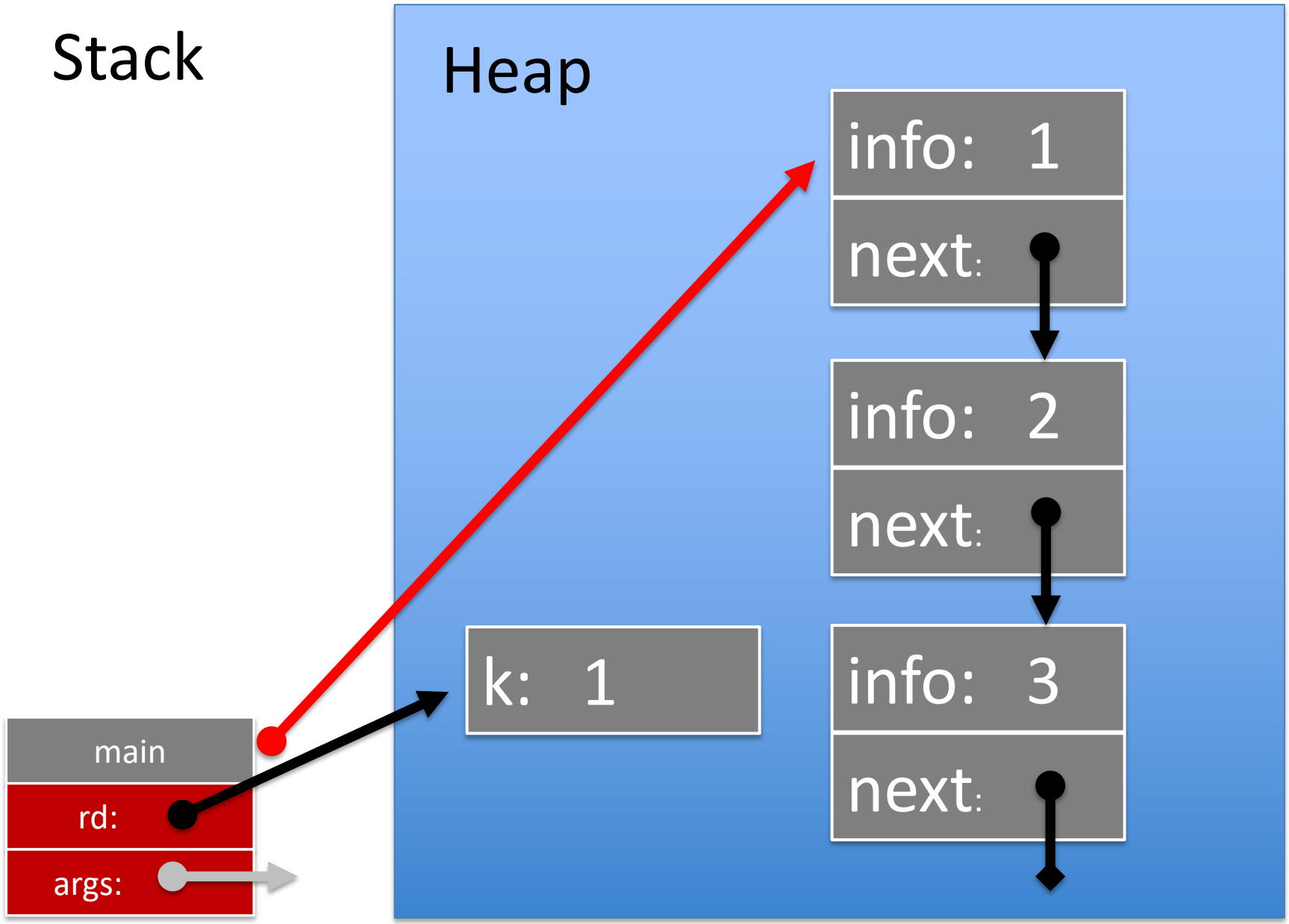



Stack
Heap
rd.two
c:
b:
this:
rd.one
a:
this:
main
rd:
args:
k: 1
info: 1
next:
info: 2
next:
info: 3
next:


Stack
Heap
rd.two
c:
b:
this:
rd.one
a:
this:
main
rd:
args:
k: 1
info: 1
next:
info: 2
next:
info: 3
next:


Stack
Heap
info: 1
next:
info: 2
next:
info: 3
next:
rd.one
a:
this:
k: 1
main
rd:
args:

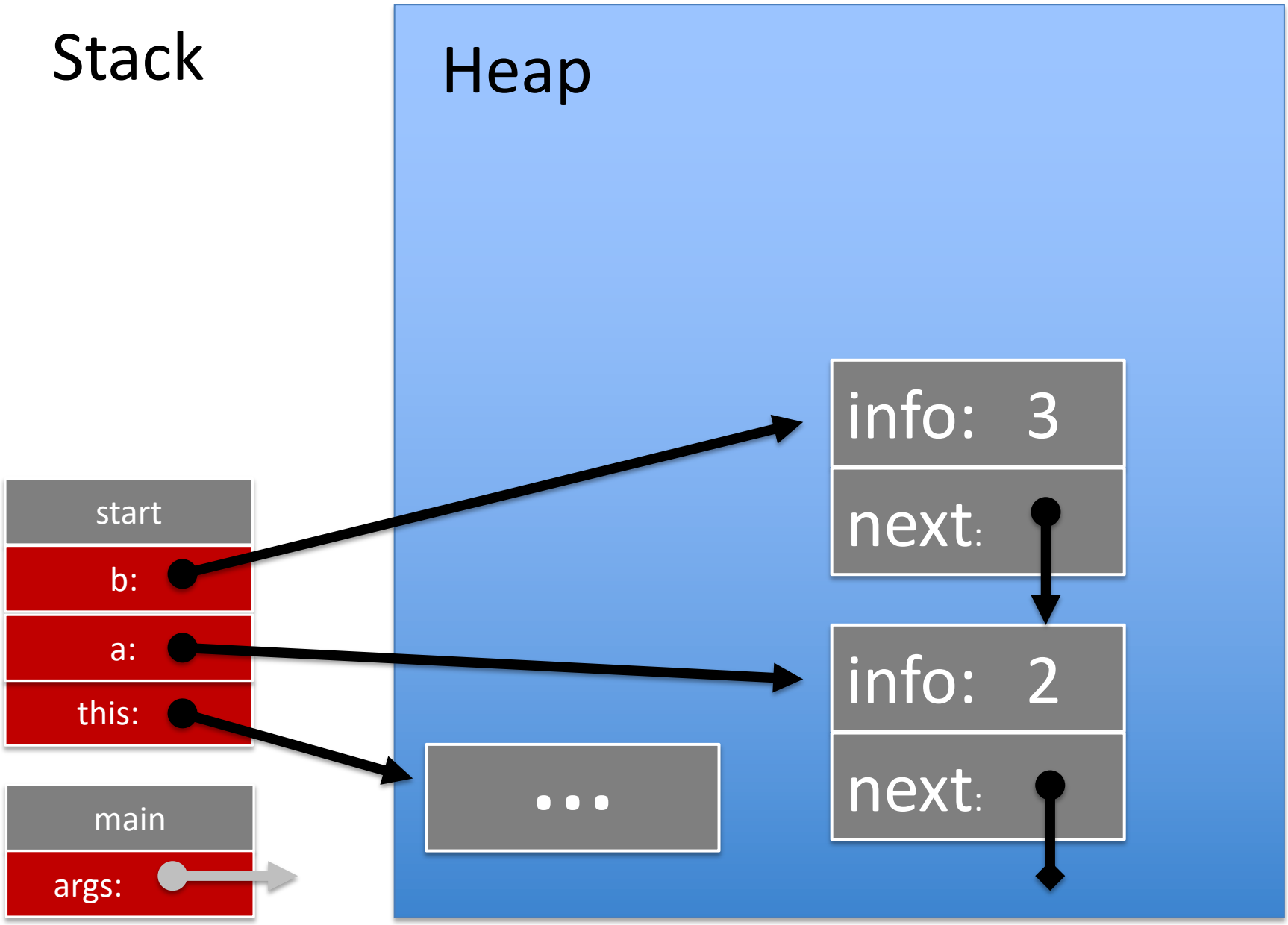

Stack
Heap
info: 1
next:
info: 2
next:
info: 3
next:
rd.one
a:
this:
k: 1
main
rd:
args:

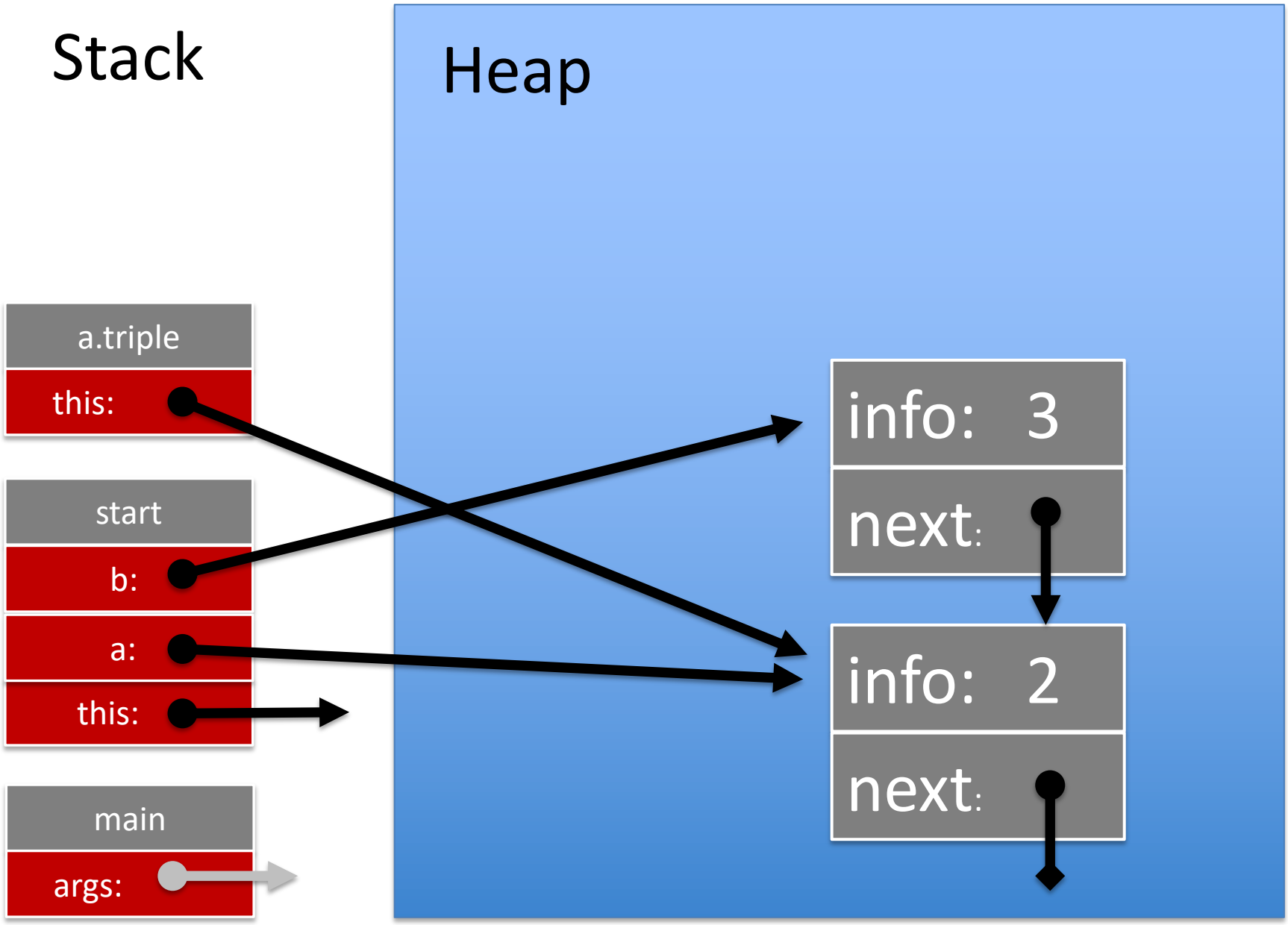

Stack
Heap
info: 1
next:
info: 2
next:
k: 1
info: 3
next:
main
rd:
args:

```java
public class ReturnThis {

  private class Node {
    int info;
    Node next;

    private Node(int info, Node next) {
      this.info = info;
      this.next = next;
    }
    private Node triple() {
      this.info = this.info * 3;
      return this;
    }
  }
  private void start() {
    Node a = new Node(2, null);
    Node b = new Node(3, a);
    a = a.triple();
  }
  public static void main(String[] args) {
    new ReturnThis().start();
  }
```

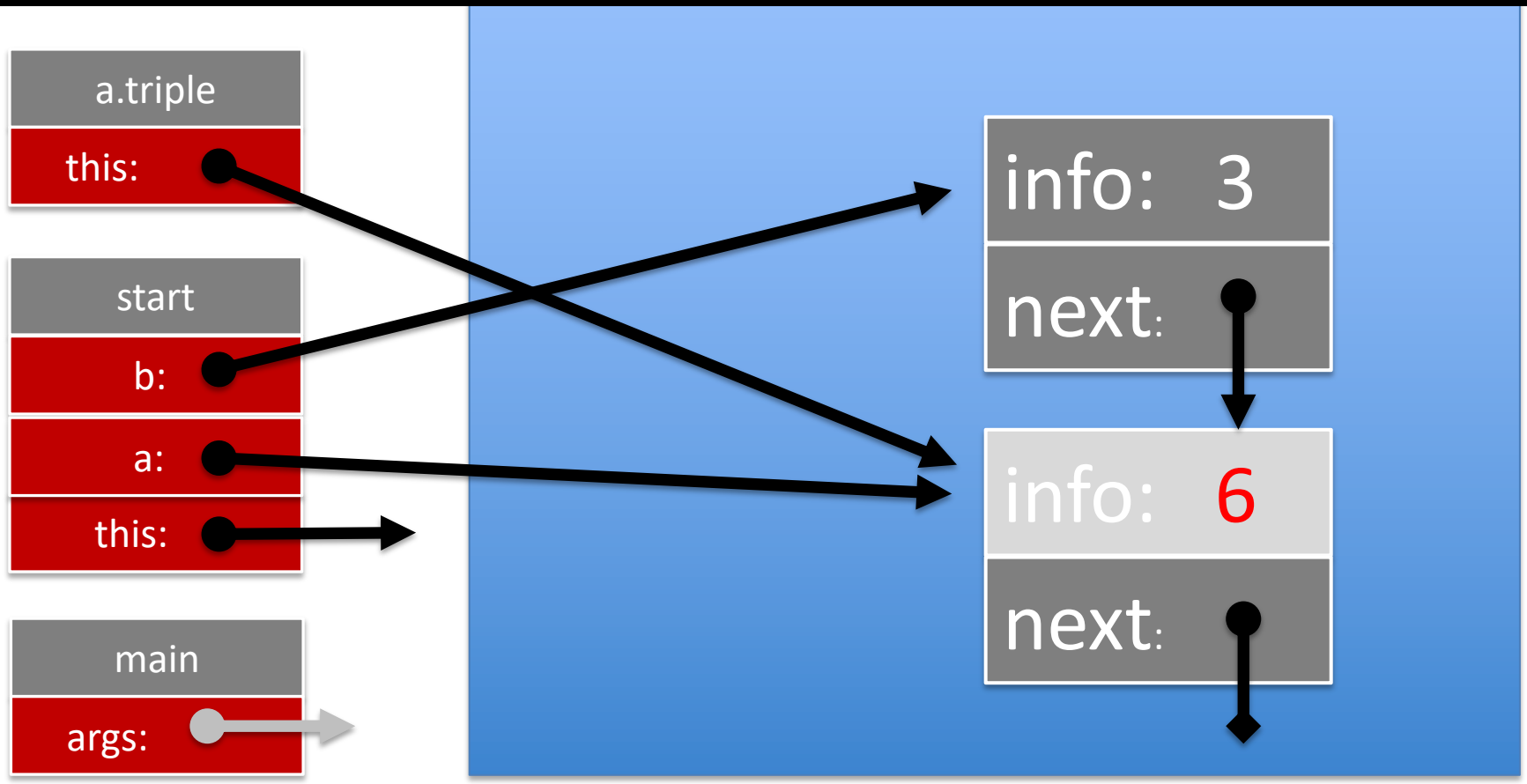
Stack

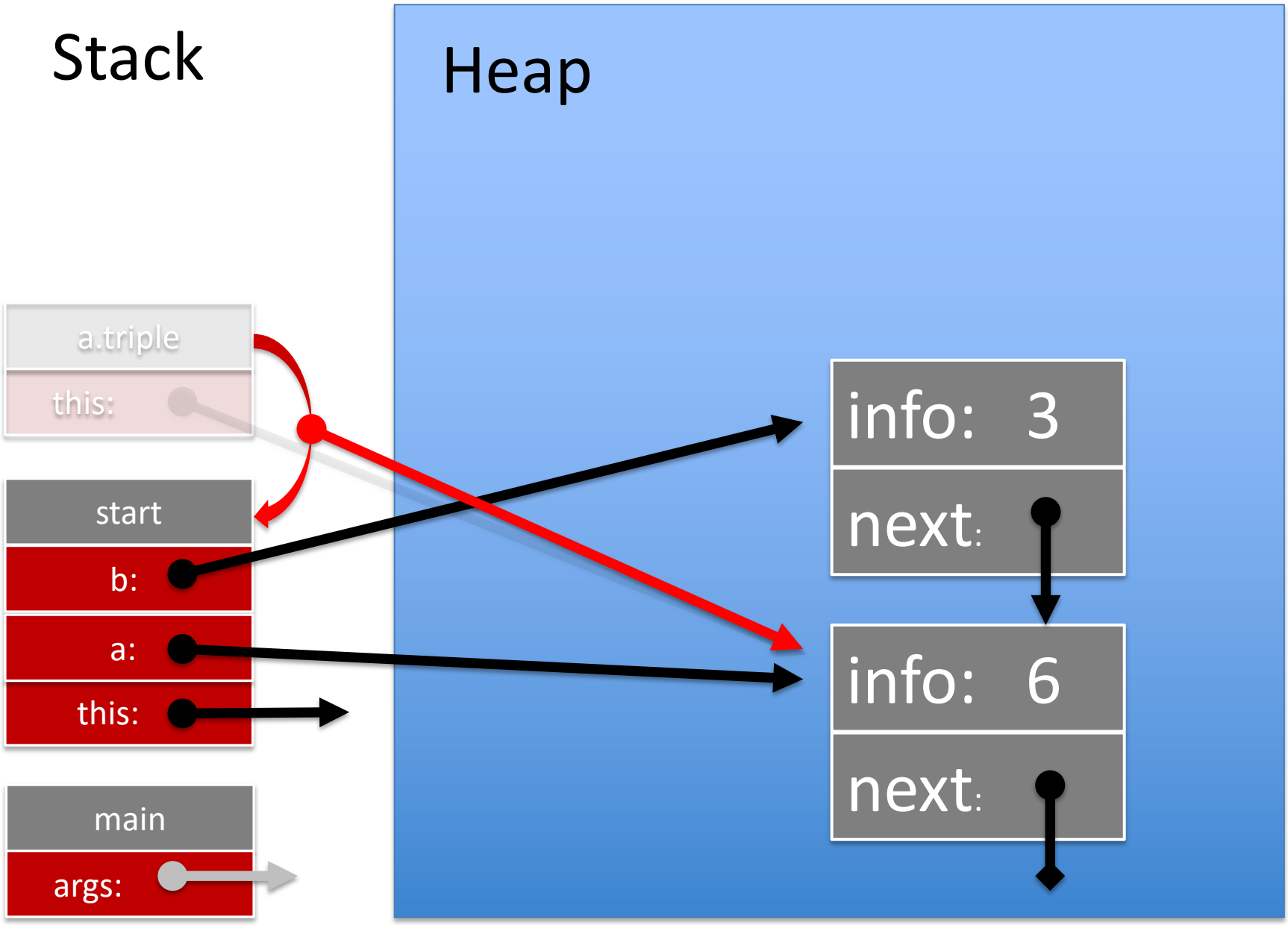
Heap


Stack
Heap
start
b:
a:
this:
main
args:
info: 3
next:
info: 2
next:
• • •


Stack
Heap
a.triple
this:
start
b:
a:
this:
main
args:
info: 3
next:
info: 2
next:

```
  private Node triple() {
    this.info = this.info * 3;
    return this;
  }
}
```




Stack
Heap
a.triple
this:
start
b:
a:
this:
main
args:
info: 3
next:
info: 6
next: