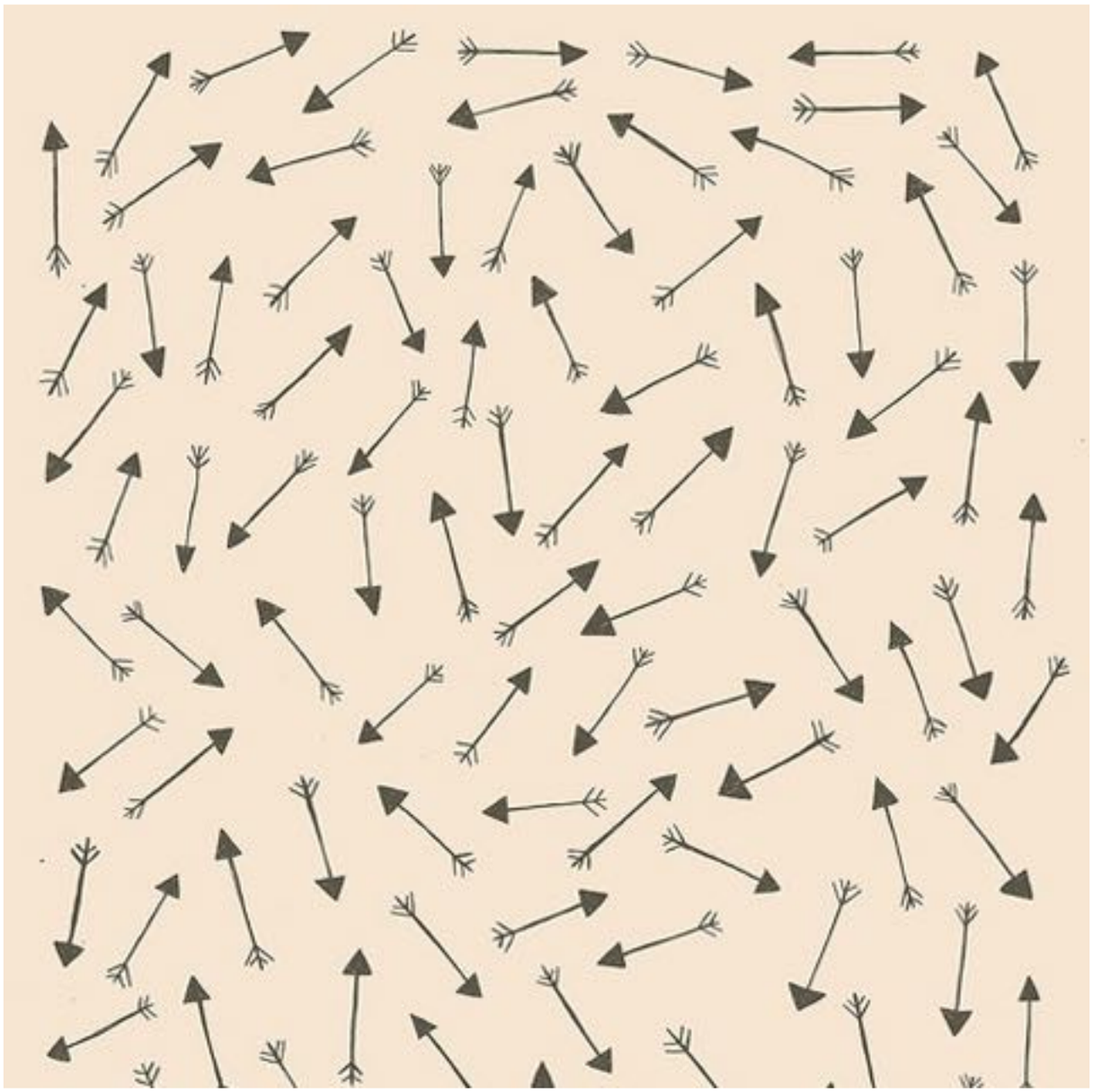


Java data types and passing by value

Primitive data types

A primitive data type is a data type that is built into a language and comes packaged with specific operations.

Java has 8 primitive data types:

`boolean`, `byte`, `char`, `short`, `int`, `long`, `float` and `double`

Primitive types directly contain values.

```
boolean tuesday
```

true

Reference types

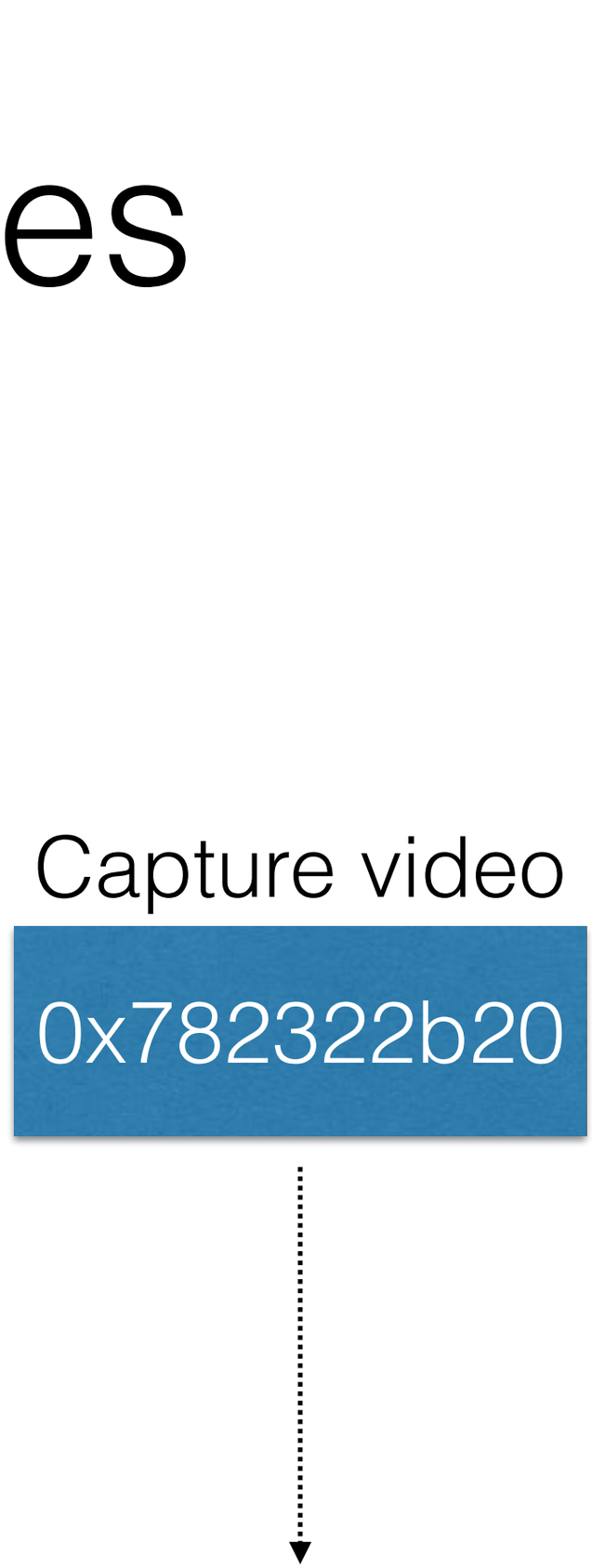
References types are created using constructors.

They include class types, interface types, and array types.

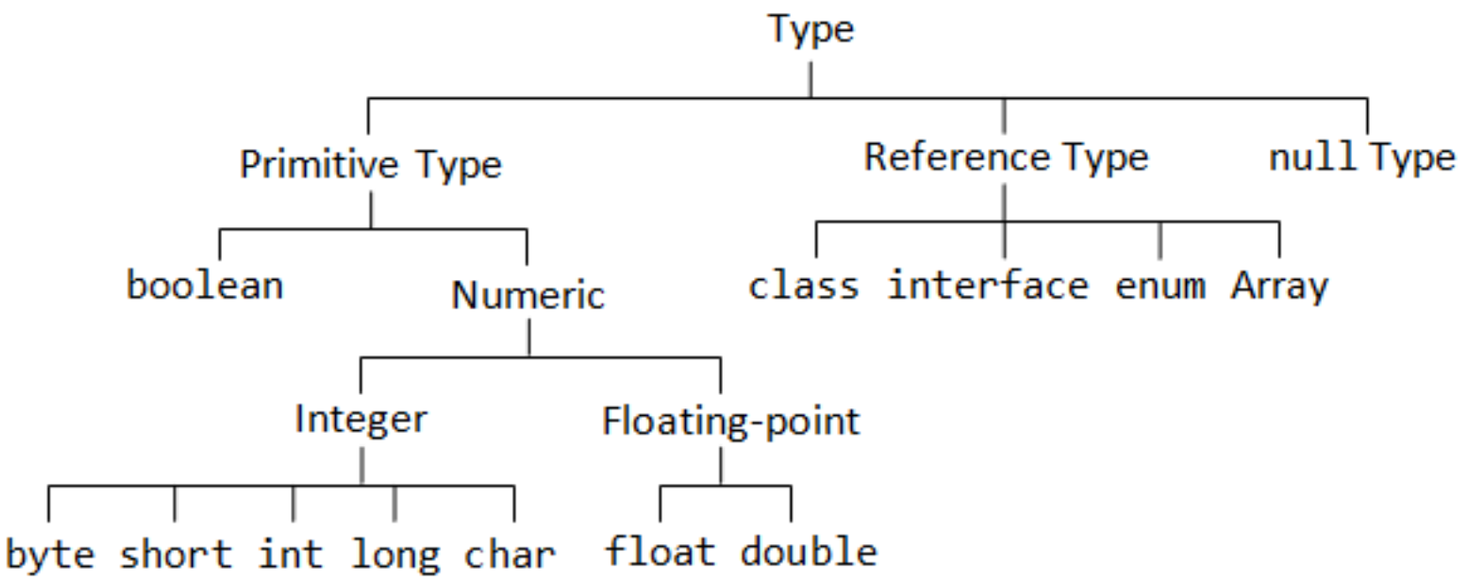
Reference types contain:

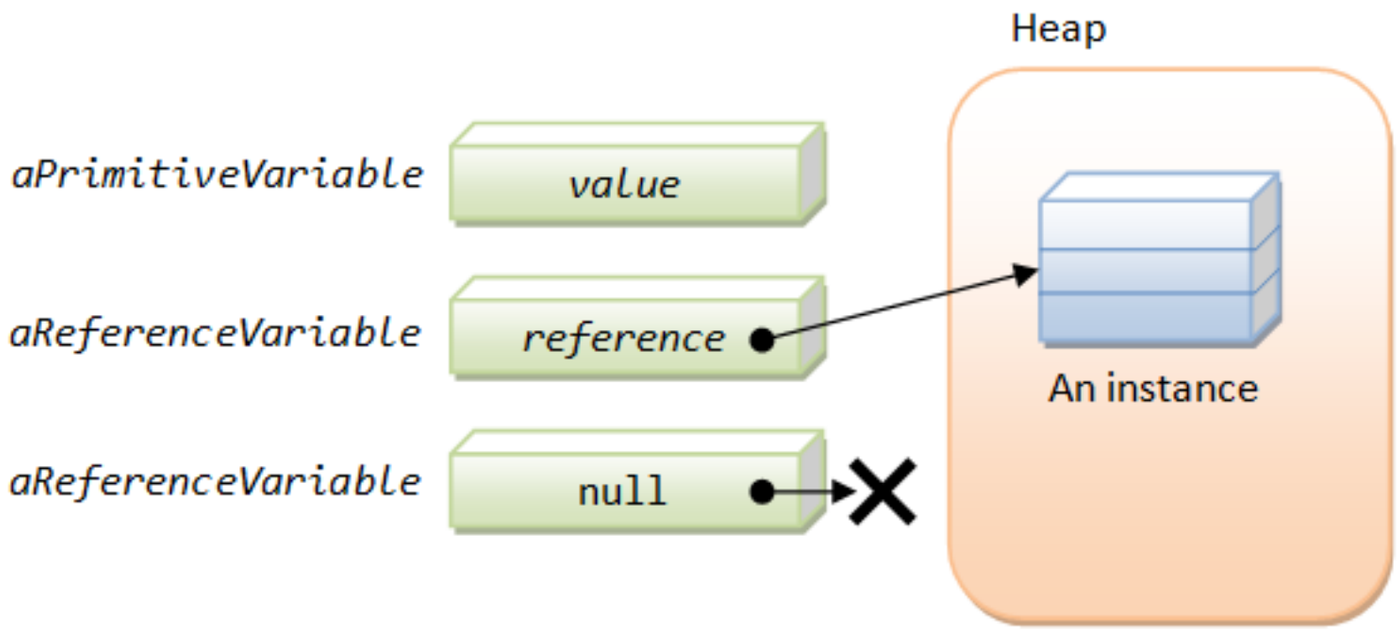
references to objects, i.e. point to addresses in memory.

references to null.



Types & what they point to





Method arguments: primitives

```
int age;

void setup() {
  age = 99;
}

void draw() {
  println("age before: "+age);
  changeAge(age);
  println("age after: "+age);
}

void changeAge(int a) {
  a++;
  println("a: "+a);
}
```

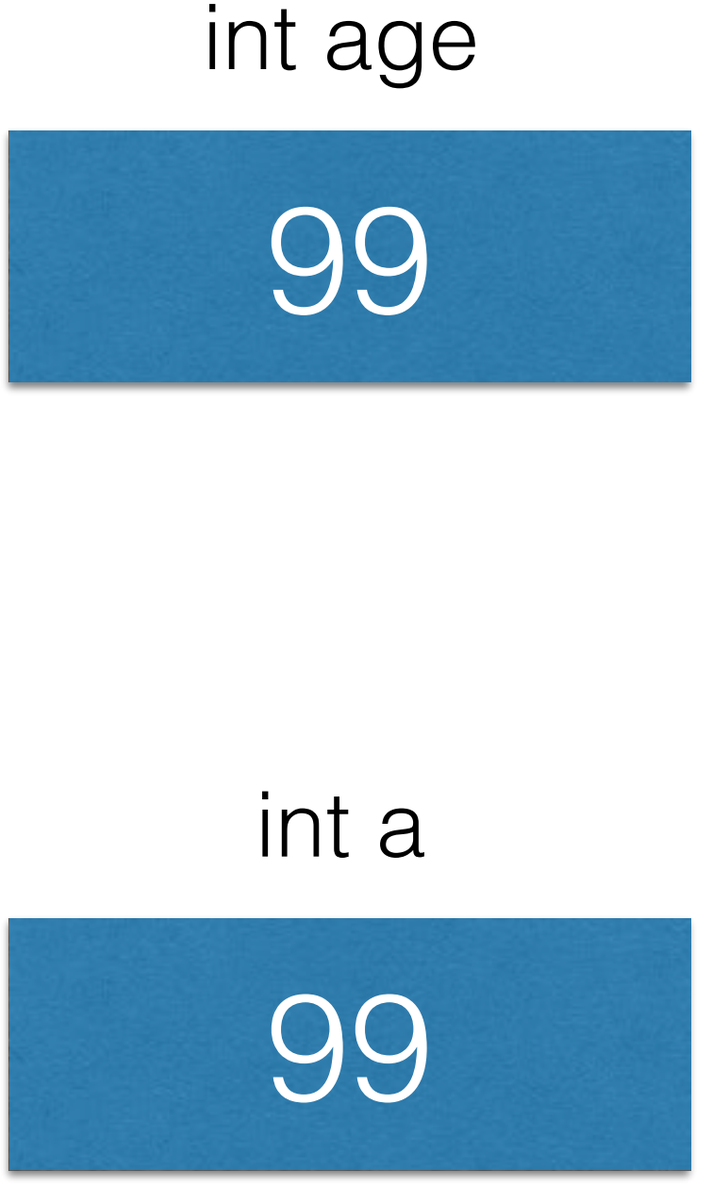
We can change the value of `a` from within changeAge, but it doesn't affect the value of `age`

This is because when we call `changeAge(a)`, what we pass through as `a` is a copy of the value we find pointed to by `age`.

Primitives are passed by value.

Method arguments: primitives

```
int age;

void setup() {
  age = 99;
}

void draw() {
  println("age before: "+age);
  changeAge(age);
  println("age after: "+age);
}

void changeAge(int a) {
  a++;
  println("a: "+a);
}
```



Method arguments: primitives

```
int age;

void setup() {
  age = 99;
}

void draw() {
  println("age before: "+age);
  changeAge(age);
  println("age after: "+age);
}

void changeAge(int a) {
  a++;
  println("a: "+a);
}
```

Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge:
"+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge:”+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```

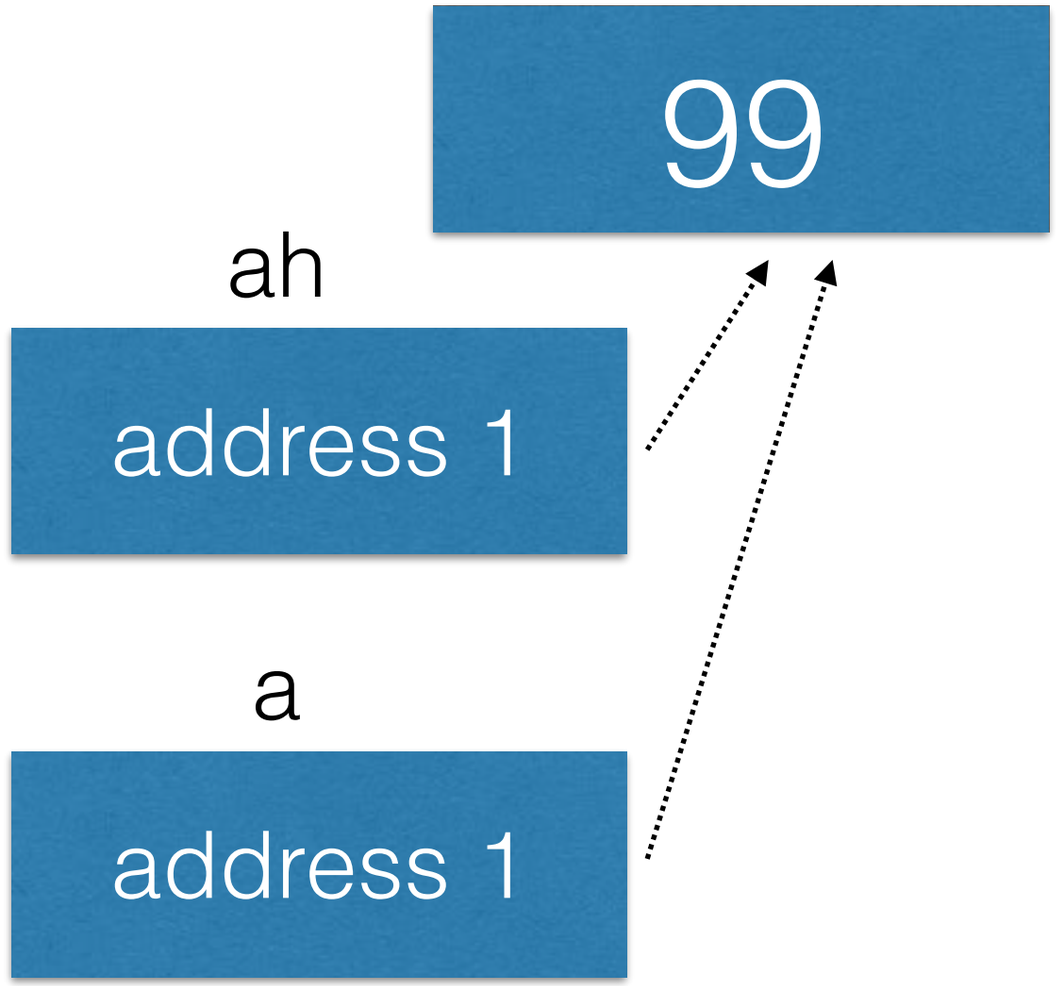
When we call changeAge(a) here, we are dealing with a copy of the address of `ah` (we create an alias).

When we reassign what `a` points to inside changeAge, it makes no difference to what `ah` points to.

Object references are passed by value.

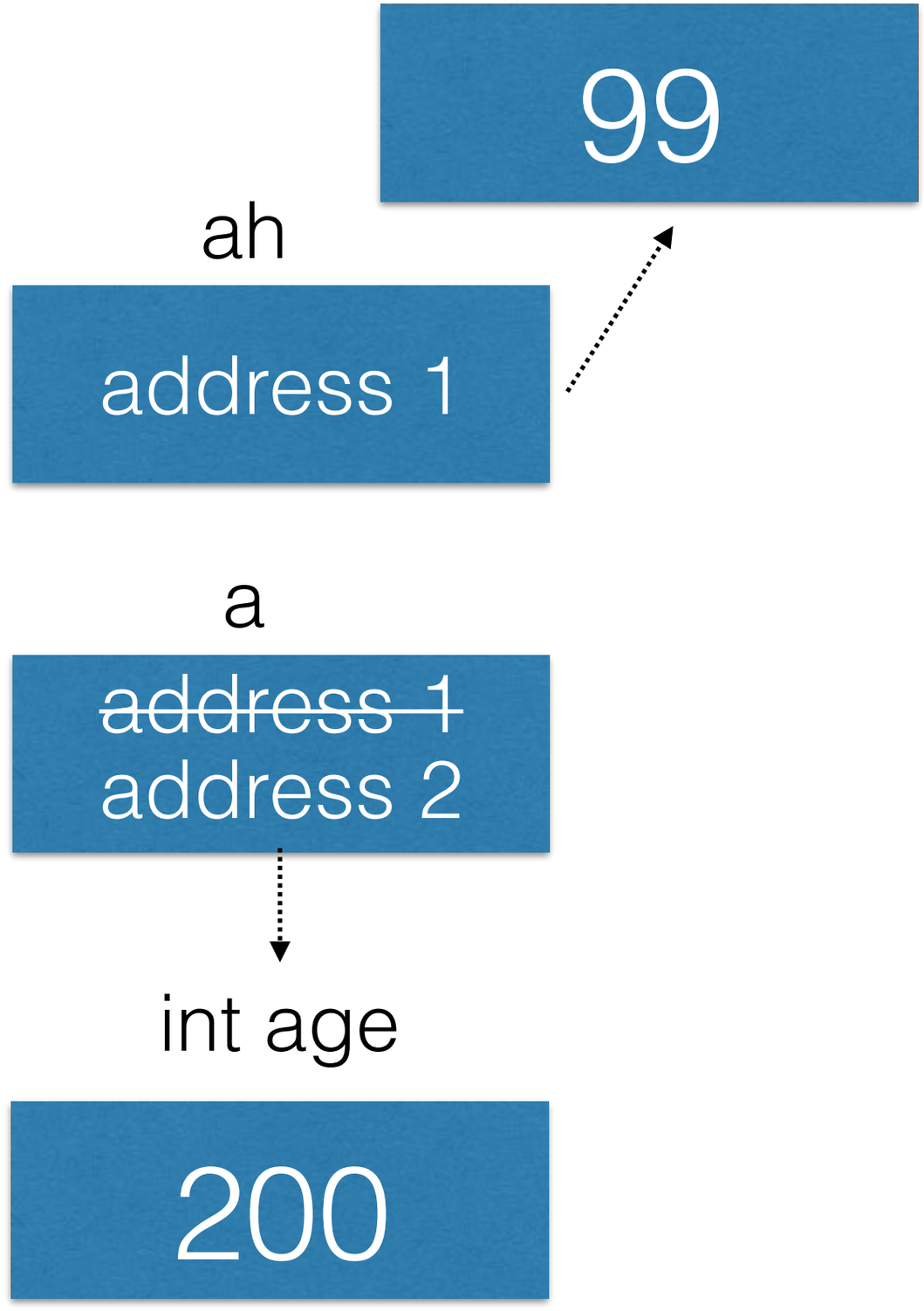
Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge:
"+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge:”+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



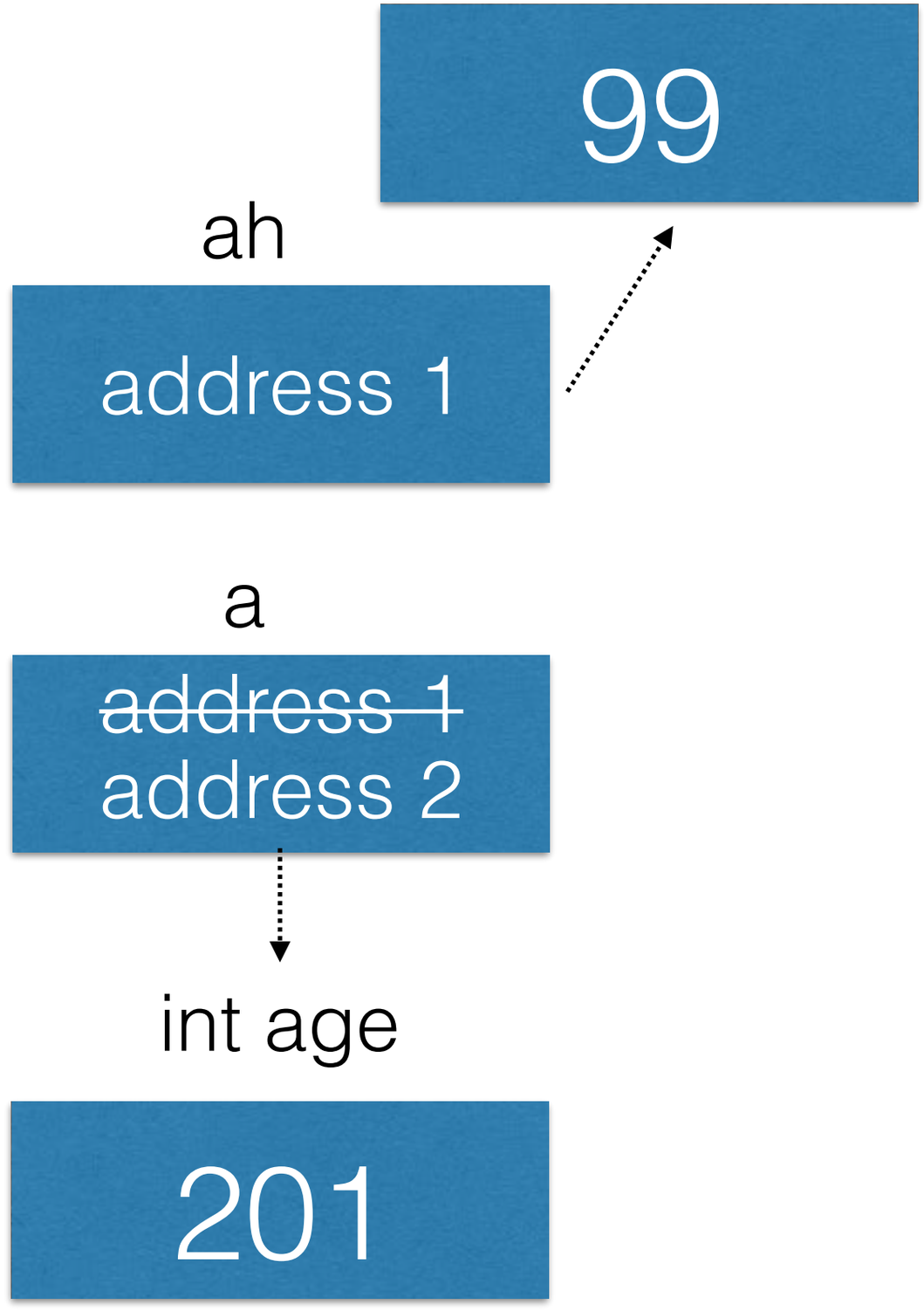
Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge:
"+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge:”+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge:
"+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge:”+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge: "+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a.age += 1;
  println("age inside changeAge first time:
"+a.age);

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge second time:
"+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```

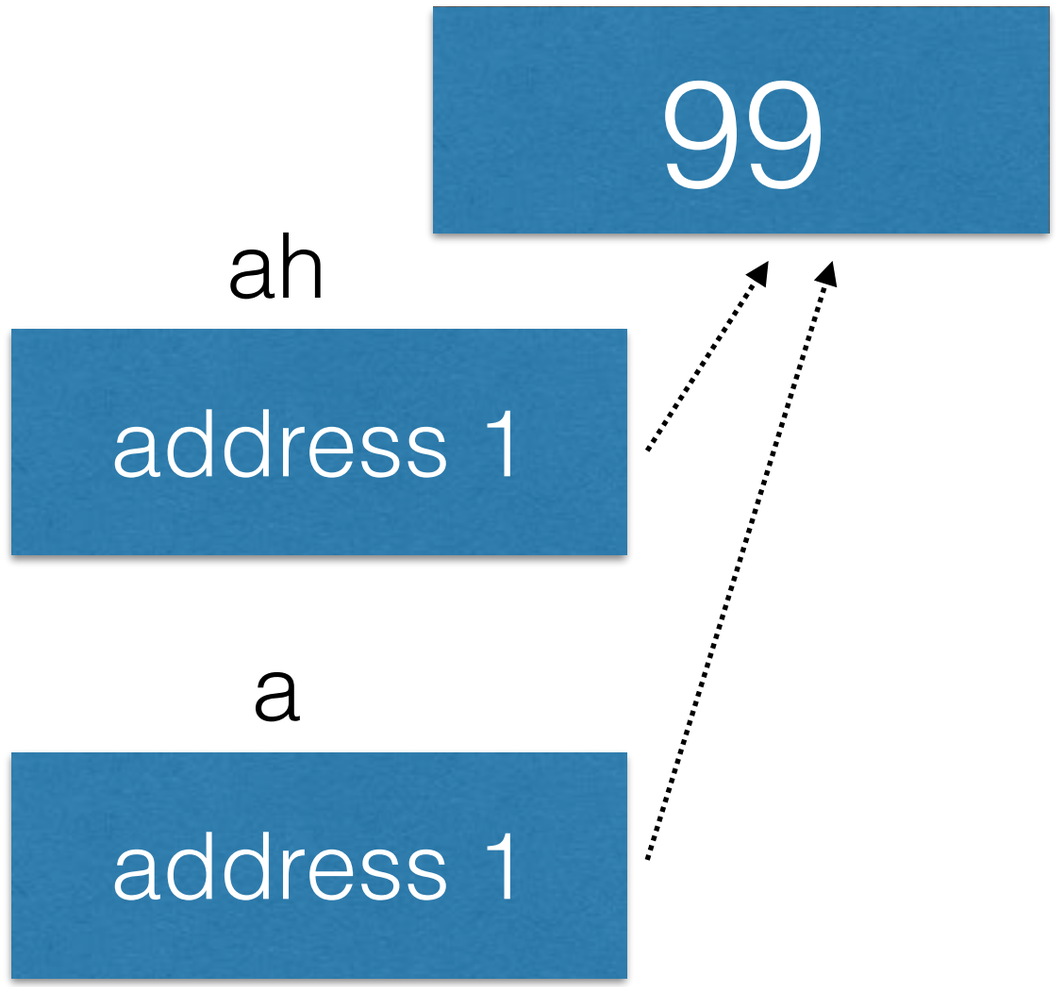
However: changing values of `a`'s variables can impact on `ah`'s variables.

In this case, it is because `ah` and `a` initially point to the same memory address.

Changing the `age` variable pointed to by a certain memory address means anything else pointing to that address will reflect a change in `age`.

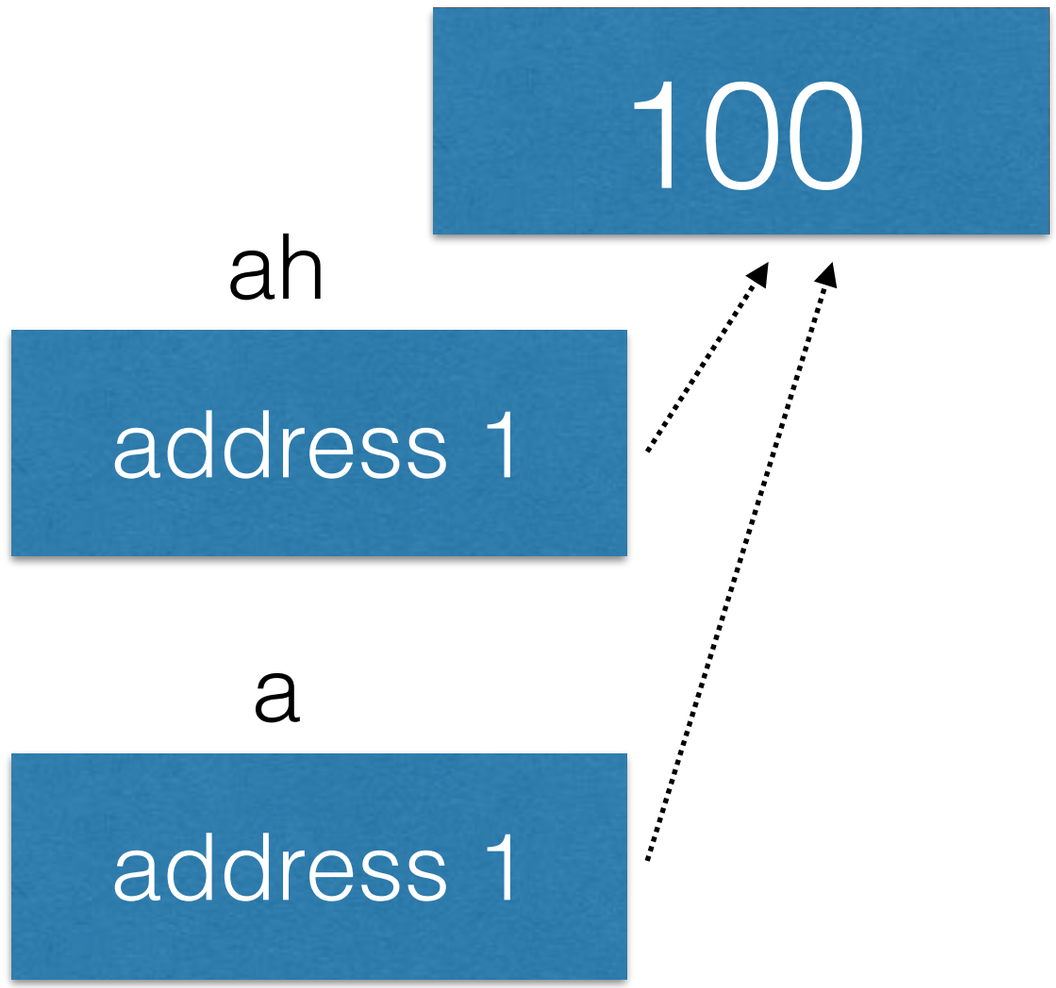
Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge: "+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a.age += 1;
  println("age inside changeAge first time:
"+a.age);

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge second time:
"+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



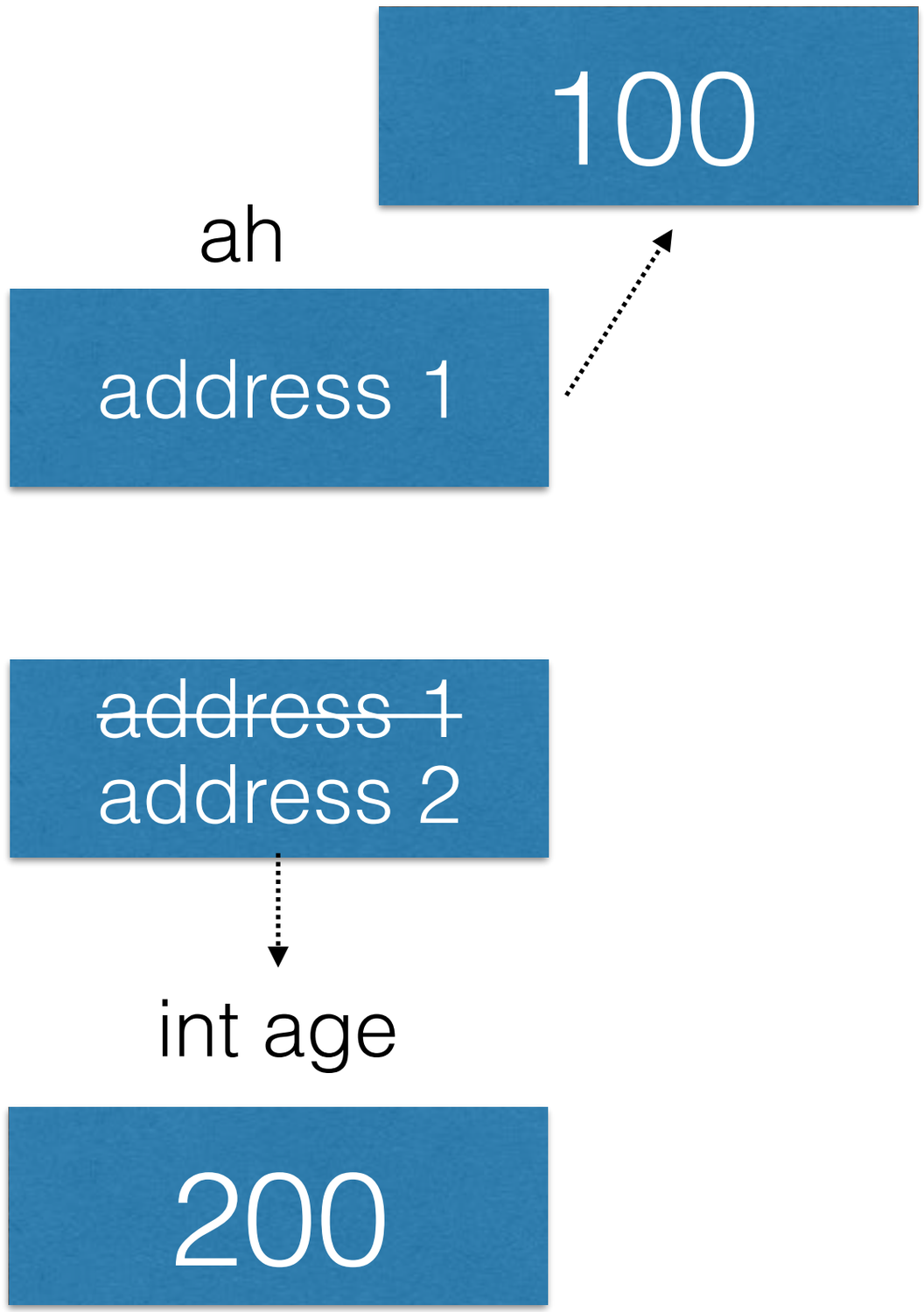
Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge: "+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a.age += 1;
  println("age inside changeAge first time:
"+a.age);

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge second time:
"+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge: "+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a.age += 1;
  println("age inside changeAge first time:
"+a.age);

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge second time:
"+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```



Method arguments: references

```
AgeHolder ah;

void setup() {
  ah = new AgeHolder(99);
}

void draw() {
  println("age before changeAge: "+ah.age);
  changeAge(ah);
  println("age after changeAge: "+ah.age);
}

void changeAge(AgeHolder a) {

  a.age += 1;
  println("age inside changeAge first time:
"+a.age);

  a = new AgeHolder(200);
  a.age += 1;
  println("age inside changeAge second time:
"+a.age);
}

class AgeHolder {
  int age;

  AgeHolder(int a) {
   age = a;
  }
}
```