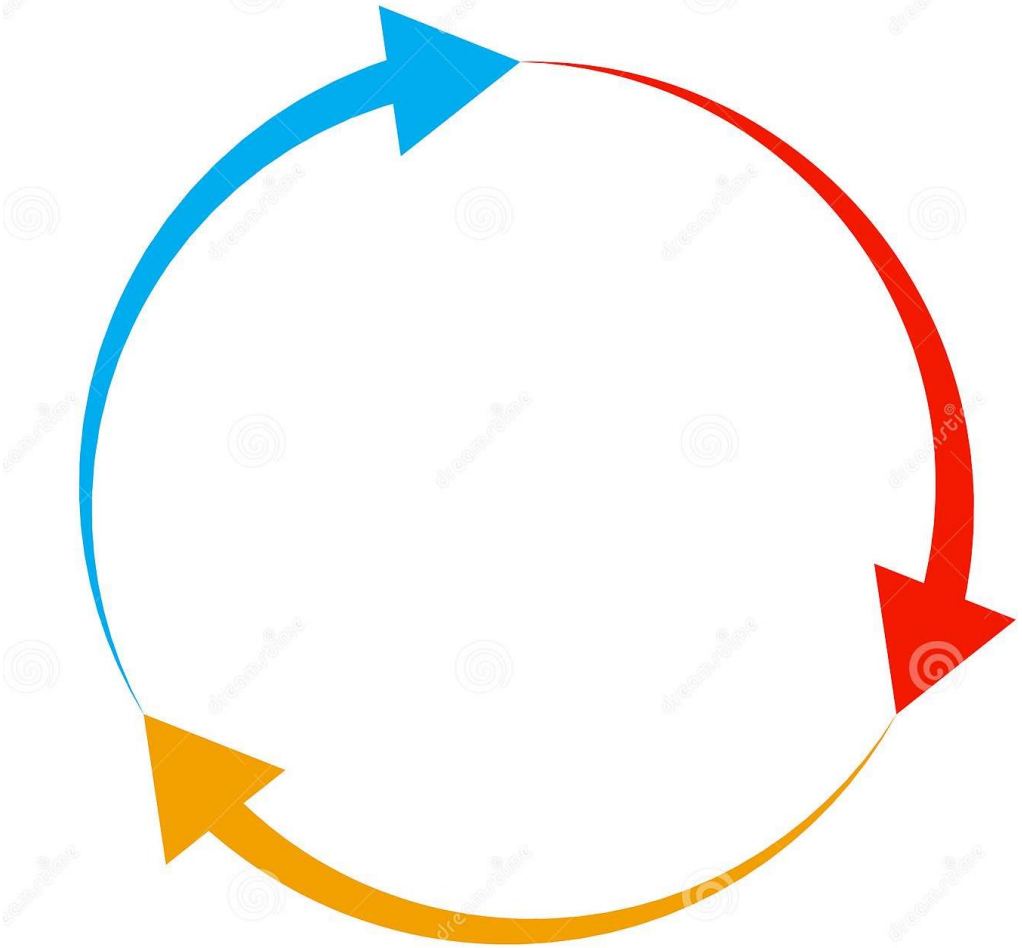
OBJECTIVES

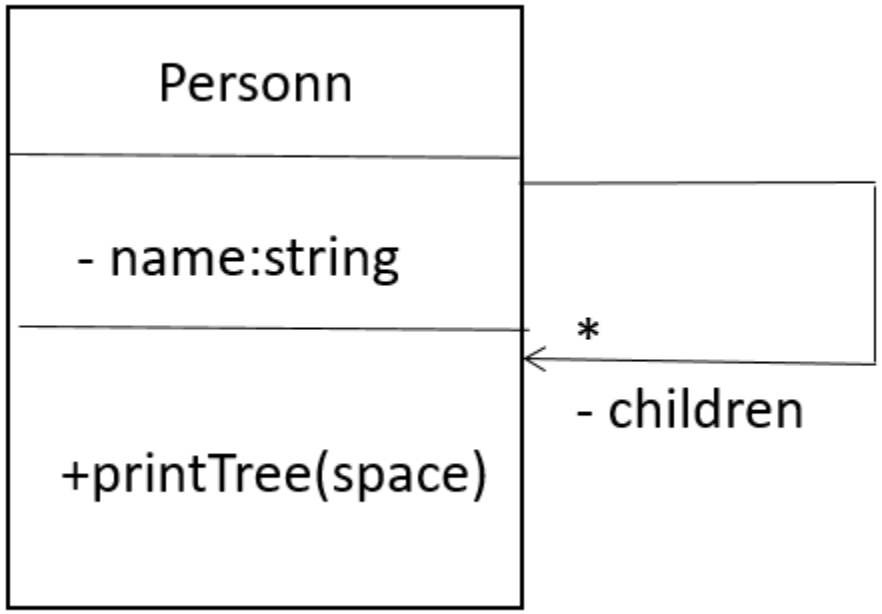
- ✓ Be able to write a **recursive** method
- ✓ Know what is a **circular reference** and how to fix it
- ✓ Be able to write a method to check if **2 object are equal**
- ✓ Understand the difference between **object with ID** and **without ID**

RECURSIVITY & CIRCULAR REFERNCE

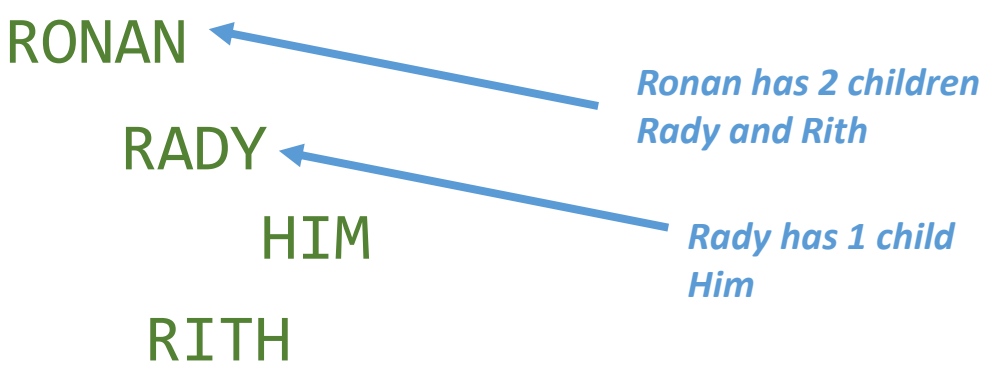


Understand a recursive call

A person can have **many children**:



We want to **print the tree** of persons:

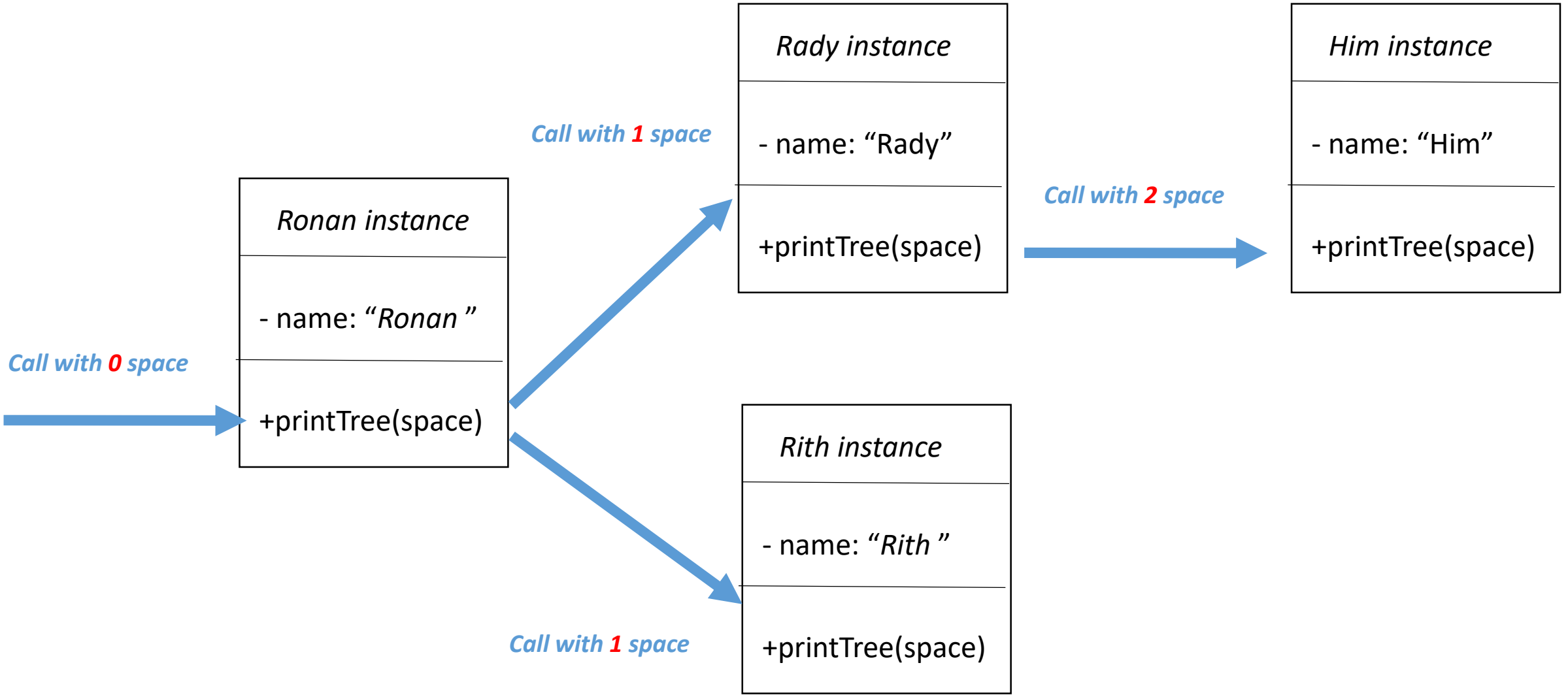


ACTIVITY 1

```
//  1 – Read the printTree method

//  2 – Understand how we increment the space
variable   to display the tree of personns


//  3 – Add a child to RITH and another child to
RADY
```

```
export class Person {
  private name: string;
  private children: Person[] = [];

  constructor(name: string) {
    this.name = name;
  }

  public addChild(child: Person) {
    this.children.push(child);
  }

  public printTree(space: string) {
    console.log(space + this.name);

    space += "\t";
    for (let child of this.children) {
      child.printTree(space);
    }
  }
}
```

We recursively call the same method on children



ACTIVITY 2

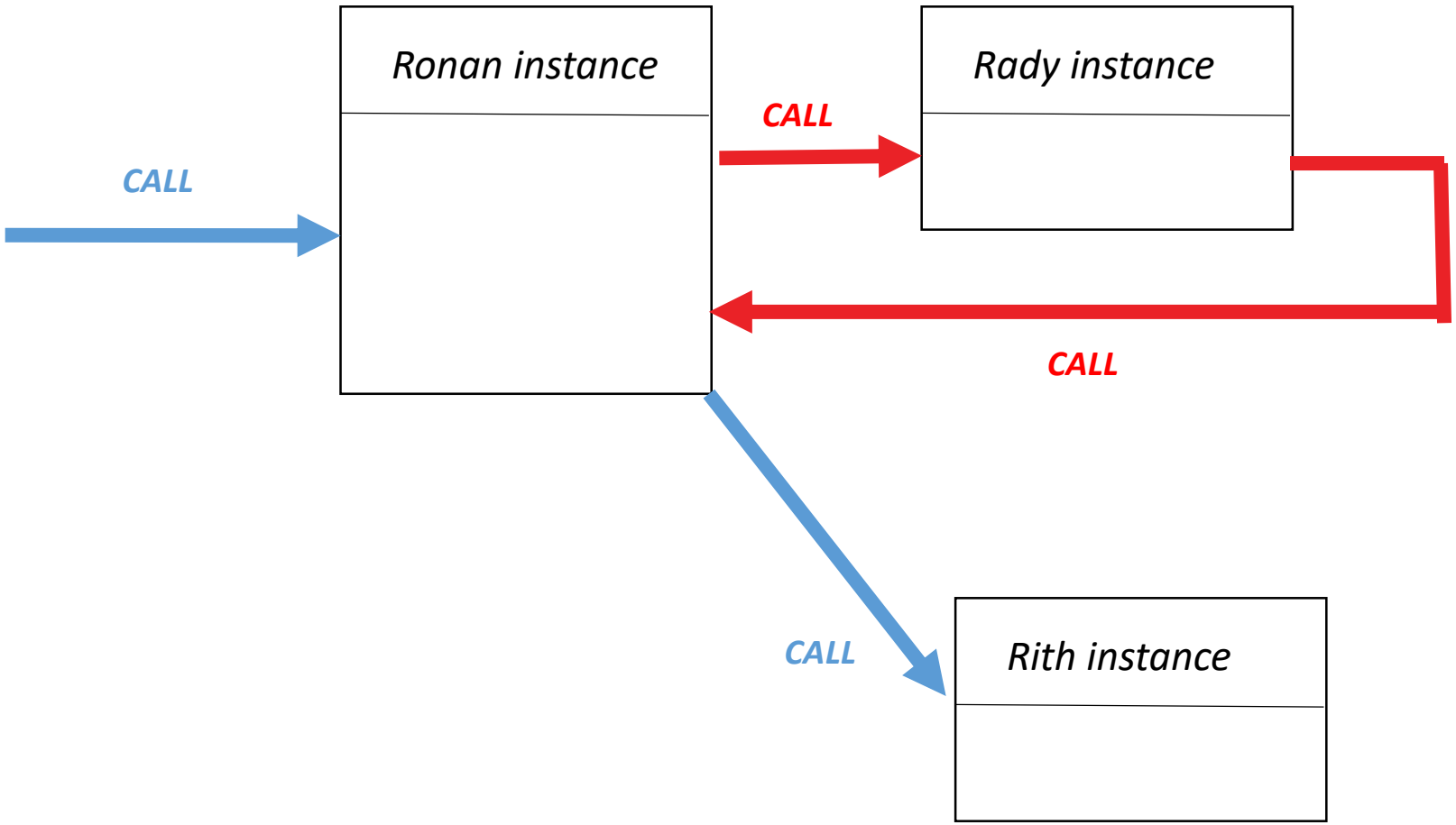
// 1 – Now Ronan is the child of Rady !!!!

// 2 – Run the code and understand see what happen !

Note : to break the code : **CTRL + Z**

```
export class Person {
  private name: string;
  private children: Person[] = [];

  constructor(name: string) {
    this.name = name;
  }

  public addChild(child: Person) {
    this.children.push(child);
  }

  public printTree(space: string) {
    console.log(space + this.name);

    space += "\t";
    for (let child of this.children) {
      child.printTree(space);
    }
  }
}
```

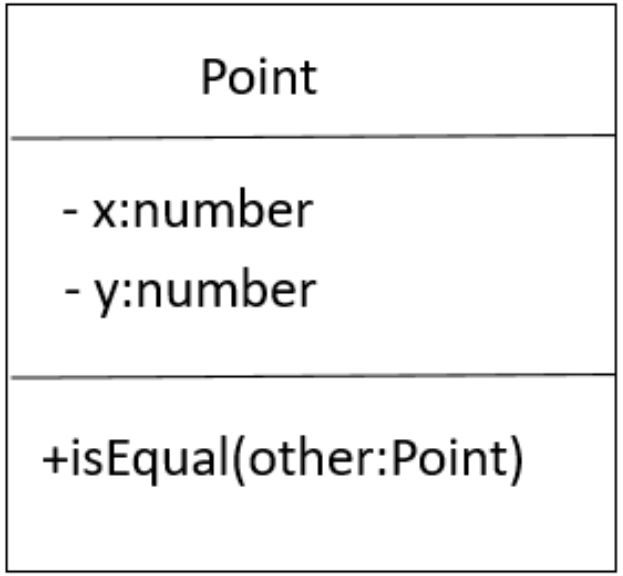
We loop because there is a circular reference



Equality of objects

Equality of objects

A point has 2 positions : X and Y



We want to **COMPARE** 2 points :

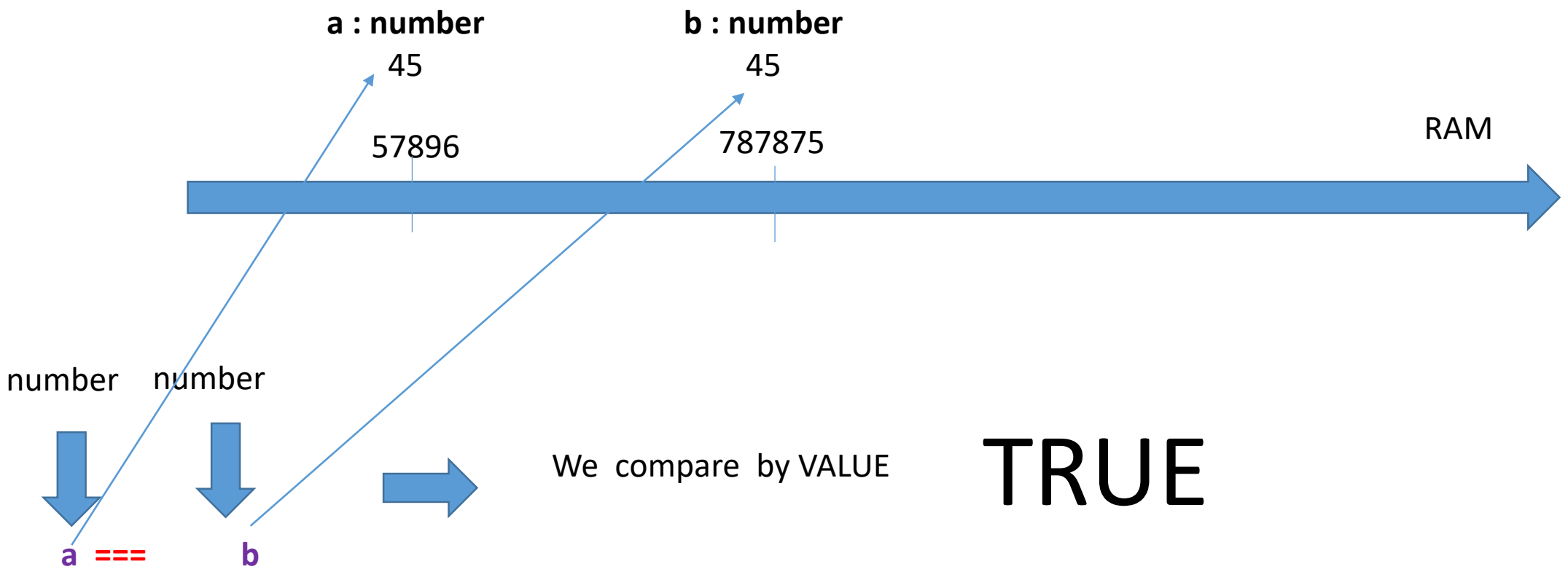
```
let p1 = new Point(10, 20);
let p2 = new Point(10, 20);
let p3 = new Point(88, 88);

console.log(p1.isEqual(p2)); // should be true

console.log(p1.isEqual(p3)); // should be false
```

ACTIVITY 3

```
//  1 - What p1 === p2 return false ?

//  2 - Implement the equal method on Point class


//  3 - Check that p1.equal(p2) return true

//  4 - Check that p1.equal(p3) return false
```

```
let p1 = new Point(10, 20);
let p2 = new Point(10, 20);
let p3 = new Point(88, 88);

console.log(p1 === p2); // should be false
console.log(p1.isEqual(p2)); // should be true

console.log(p1.isEqual(p3)); // should be false
```

FOR PRIMITIVE TYPES number boolean string

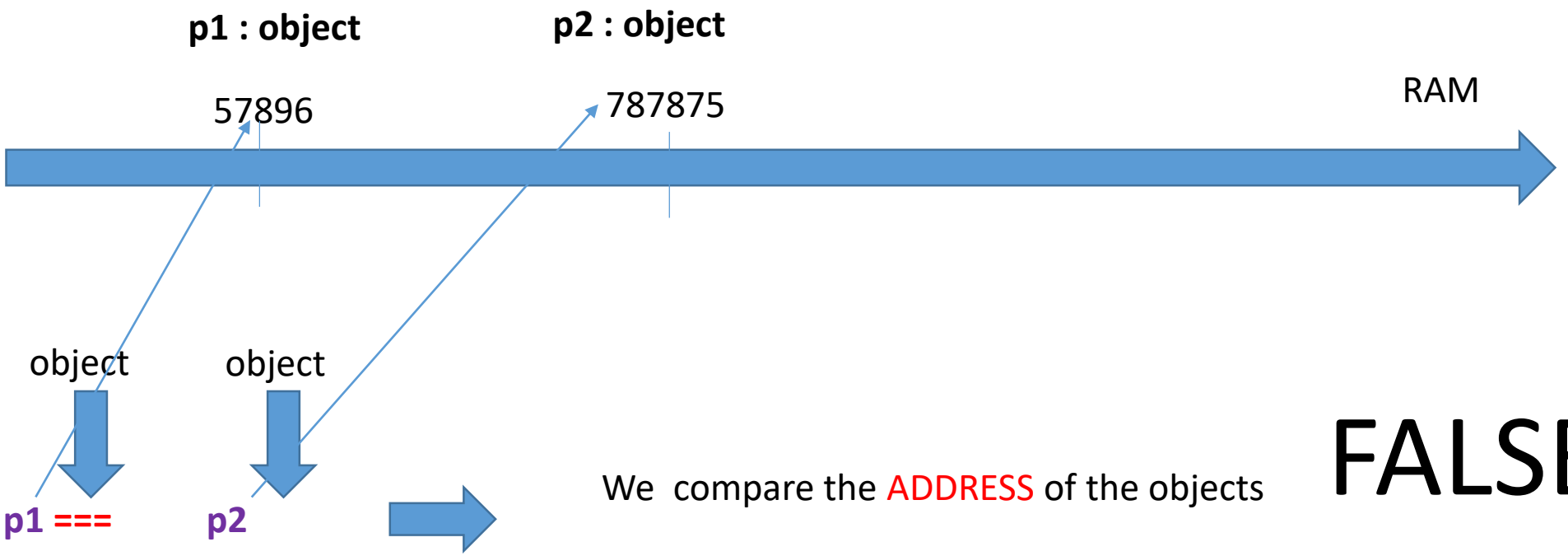


FOR OJECT/ARRAY TYPES

Point
x:number y:number

p1
X:45 Y:55

p2
X:45 Y:55



We compare the ADDRESS of the objects

FALSE

Equality of objects

```
let a = 45;
let b = 45;

let c = a === b
```

True because a and b are

primitive types

For primitives == is done is on the value

```
let a = new Person('x');
let b = new Person('x');

let c = a === b
```

False because a and b are

object types

For object == is done is on the @ in RAM

```
let a = new Person('x');
let b = new Person('x');

let c = a.equals(b)
```

True because now we really compare the 2 persons using their attributes