Curso Básico de

Programación Orientada a Objetos en **JavaScript**

👨‍💻 JuanDC

Requisitos

- Prework 💻
- Pensamiento Lógico 🧠
- Programación Básica 👩‍💻

Requisitos

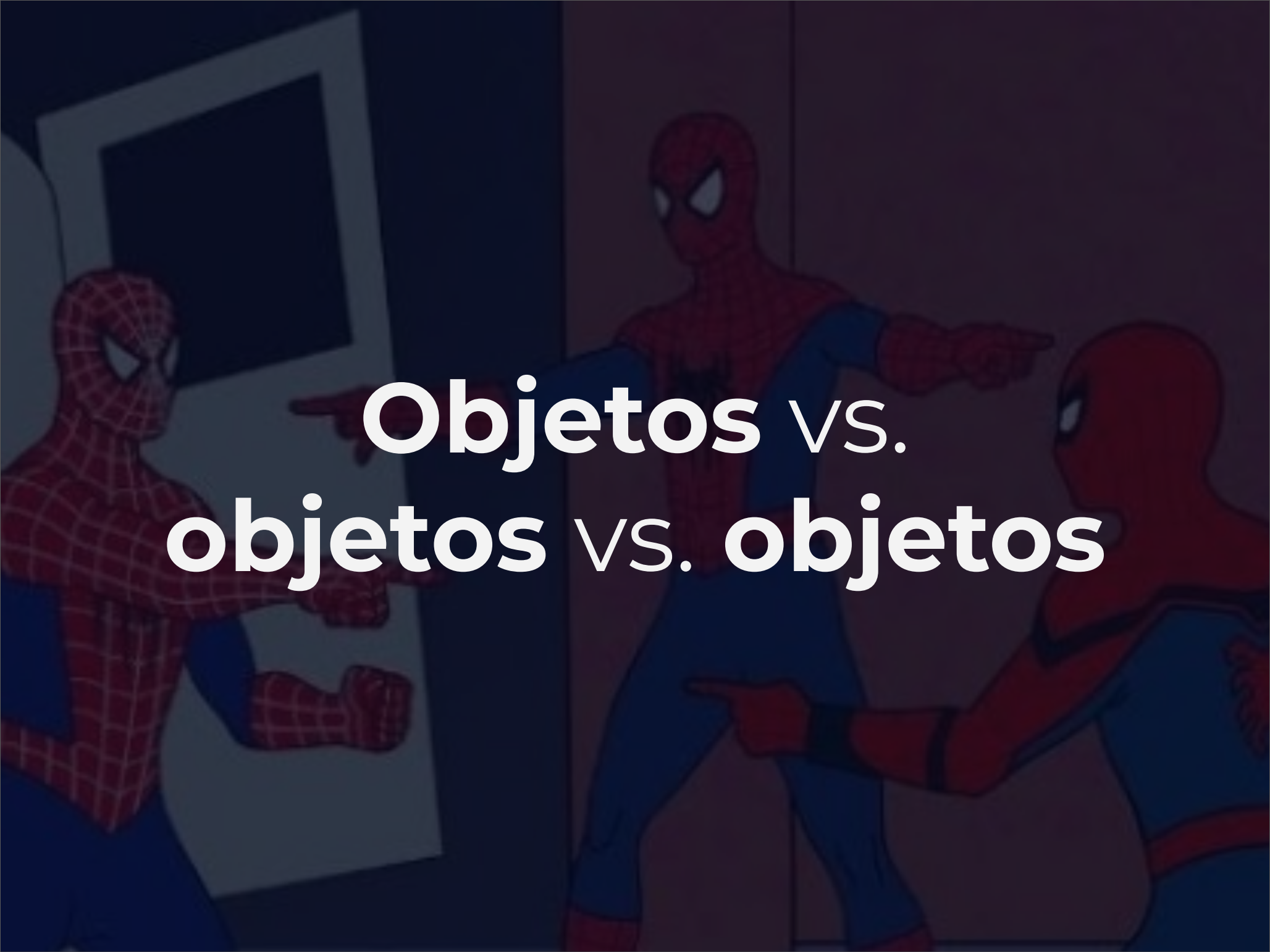
- HTML y JS Básico 
- Closures y Scope
- Fundamentos de POO 📦

Platzi

ESCUELA DE DESARROLLO WEB

www.platzi.com/web

JavaScript:
orientado a **objetos**,
basado en **prototipos**


Objetos vs.
objetos vs. objetos


Objetos

JavaScript

Temario

- ✅ 🍼 Pilares de POO
- 🔜 🤯 Objetos a profundidad
- 🔜 😈 SOLID

¿Qué es la **programación orientada a objetos**?


Paradigmas

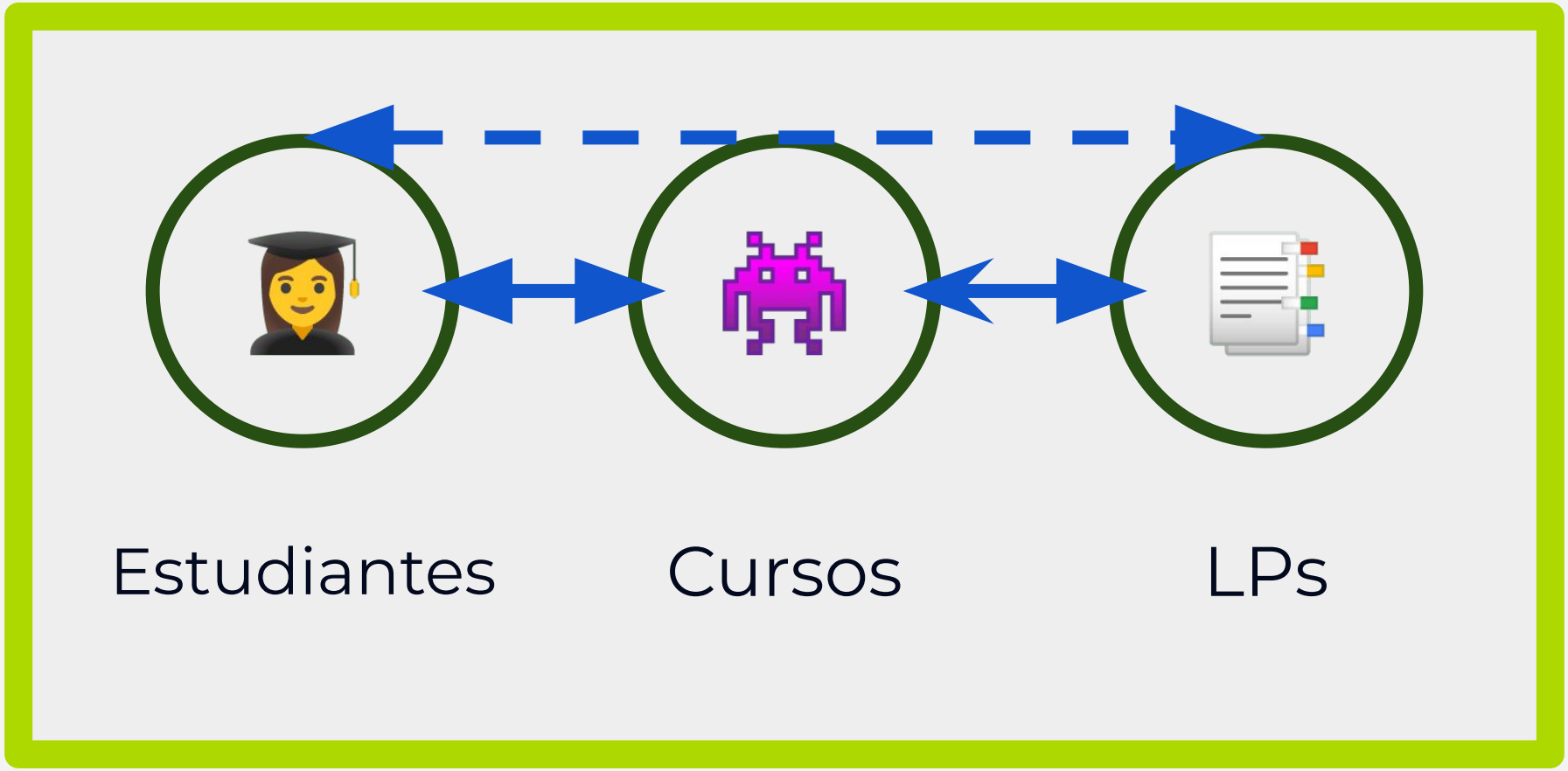

Paradigmas

Paradigmas

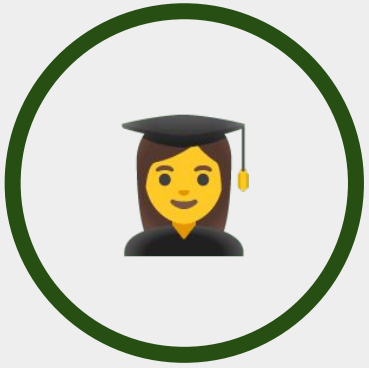
- Estructurado 
- Orientado a objetos 📦
- Funcional 🏋️‍♀️


Orden

Objetos

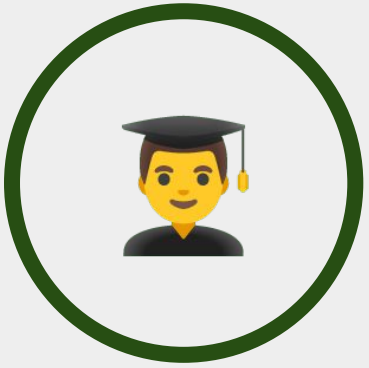


Estudiantes



Natalia

Miguel



JuanDC

Natalia

Nombre

Edad

PlatziRank

Reutilizar código


Moldes
de galletas

Clases

Estudiante


POO en JavaScript

Qué es un objeto en JavaScript

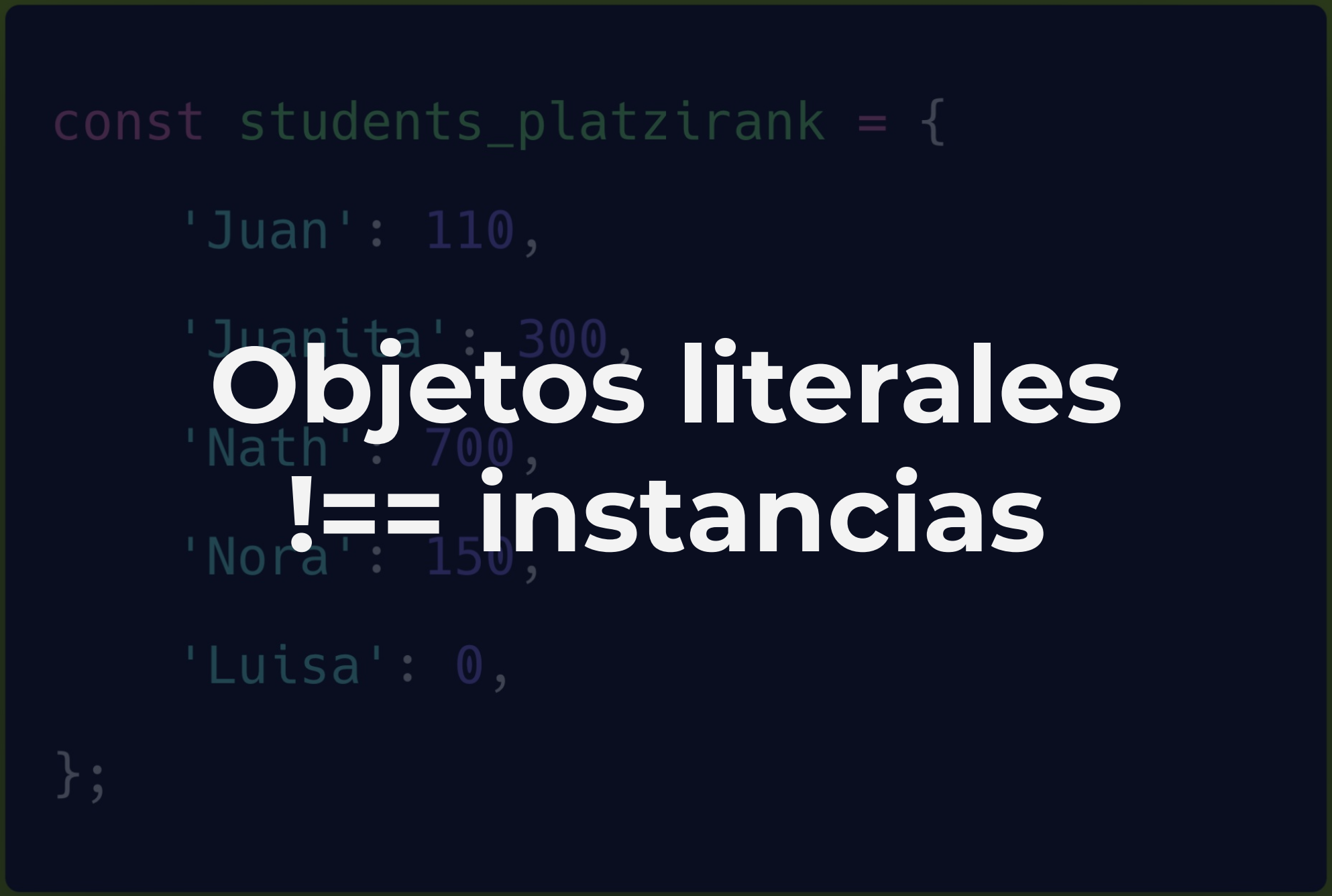
```
const students_platzirank = [
  'Juan',
  'Juanita',
  'Nath',
  'Nora',
  'Luisa',
];
```

```php
$students_platzirank = [
    "Juan",
    "Juanita",
    "Nath",
    "Nora",
    "Luisa",
];
```

```php
$students_platzirank = [
    "Juan" => 110,
    "Juanita" => 300,
    "Nath" => 700,
    "Nora" => 150,
    "Luisa" => 0,
];
```

```
students_platzirank = {
  'Juan': 110,
  'Juanita': 300,
  'Nath': 700,
  'Nora': 150,
  'Luisa': 0,
}
```

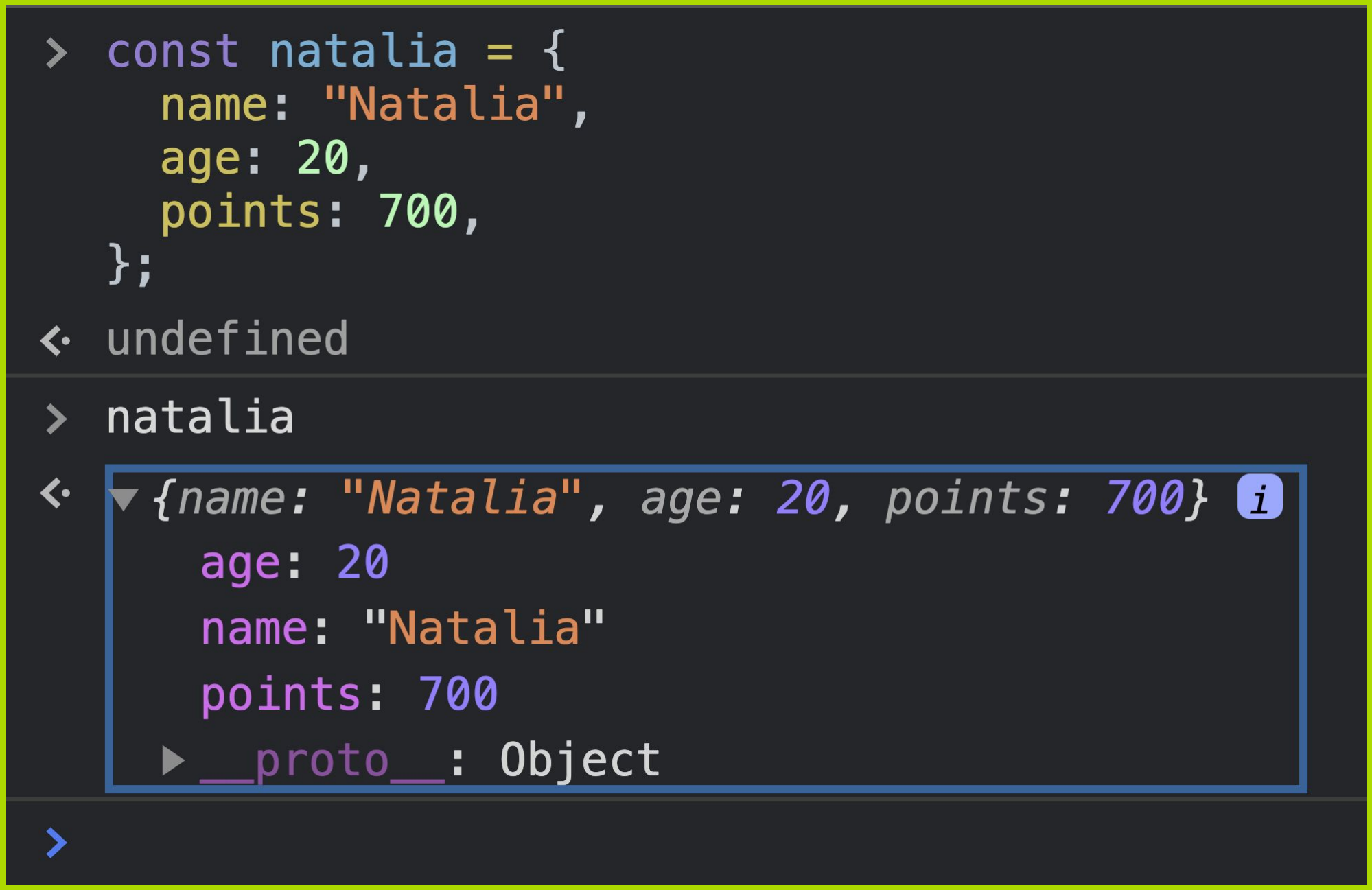
```
const students_platzirank = {
    'Juan': 110,
    'Juanita': 300,
    'Nath': 700,
    'Nora': 150,
    'Luisa': 0,
};
```

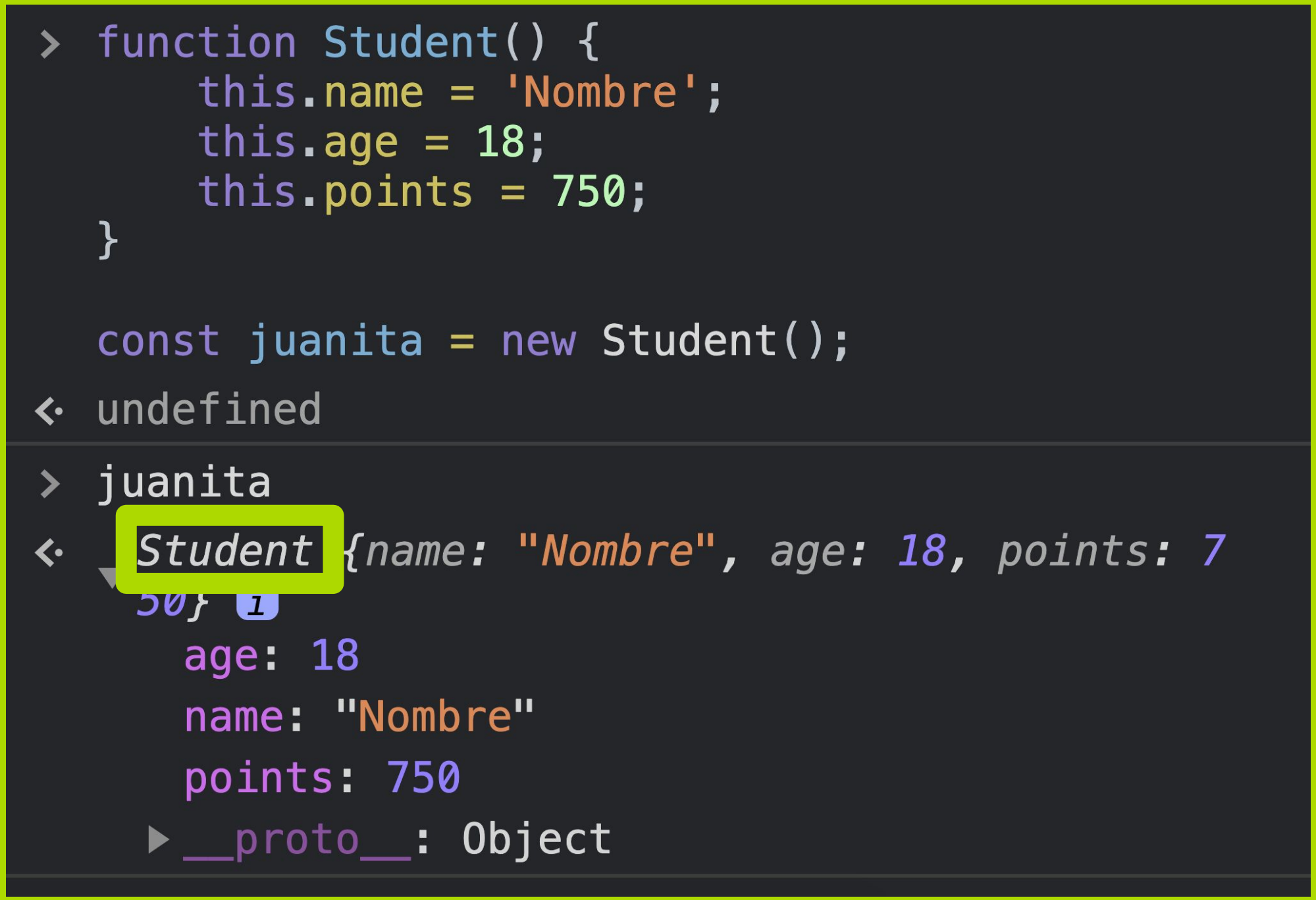

const students_platzirank = {
'Juan': 110,
'Juanita': 300,
'Nath': 700,
'Nora': 150,
'Luisa': 0,
};
Objetos literales
!== instancias

```php
class Student {
    public $name = 'Nombre';
    public $age = 18;
    public $points = 750;
}

$juanita = new Student;
```

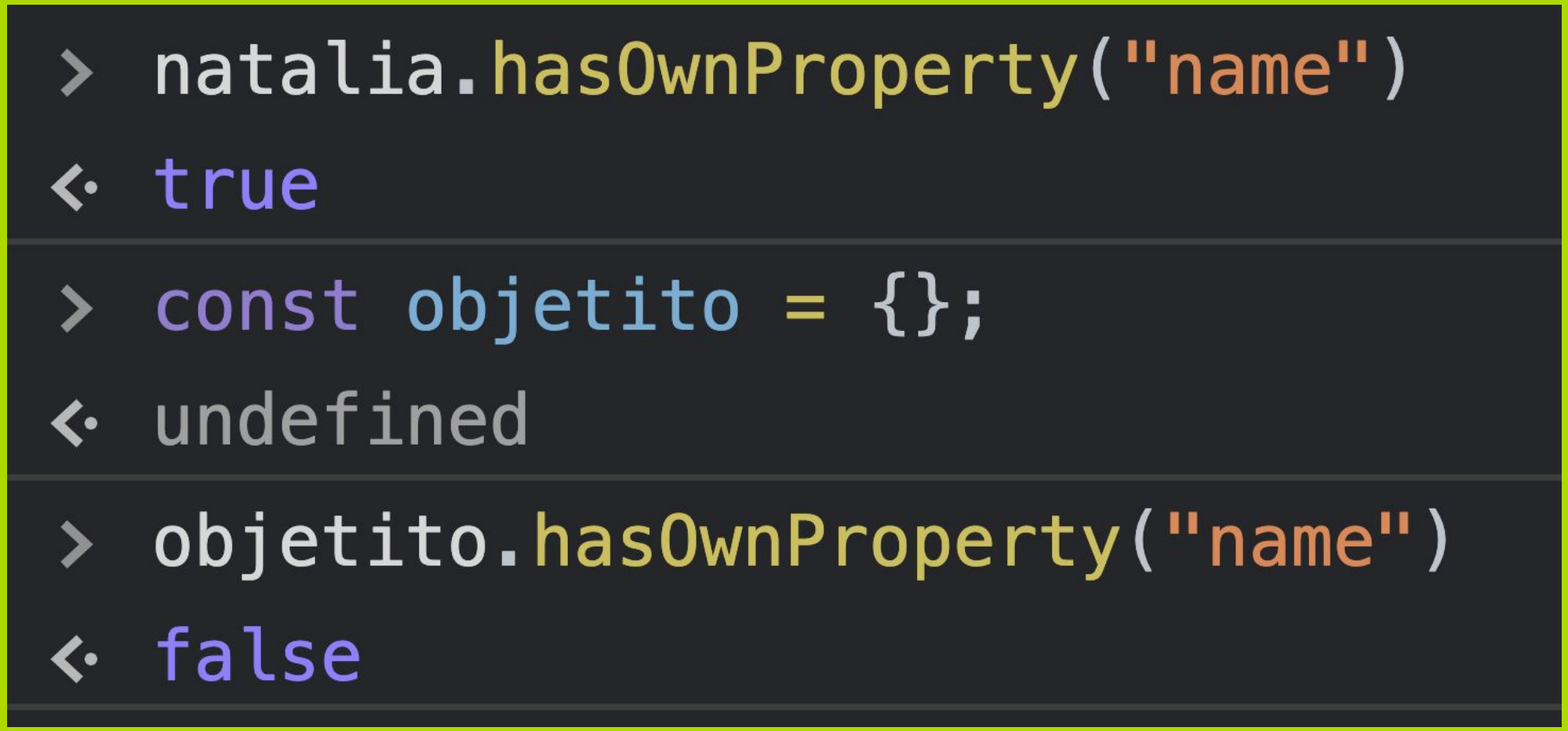
```
class Student:
    name = 'Nombre'
    age = 18
    points = 750



juanita = Student();
```

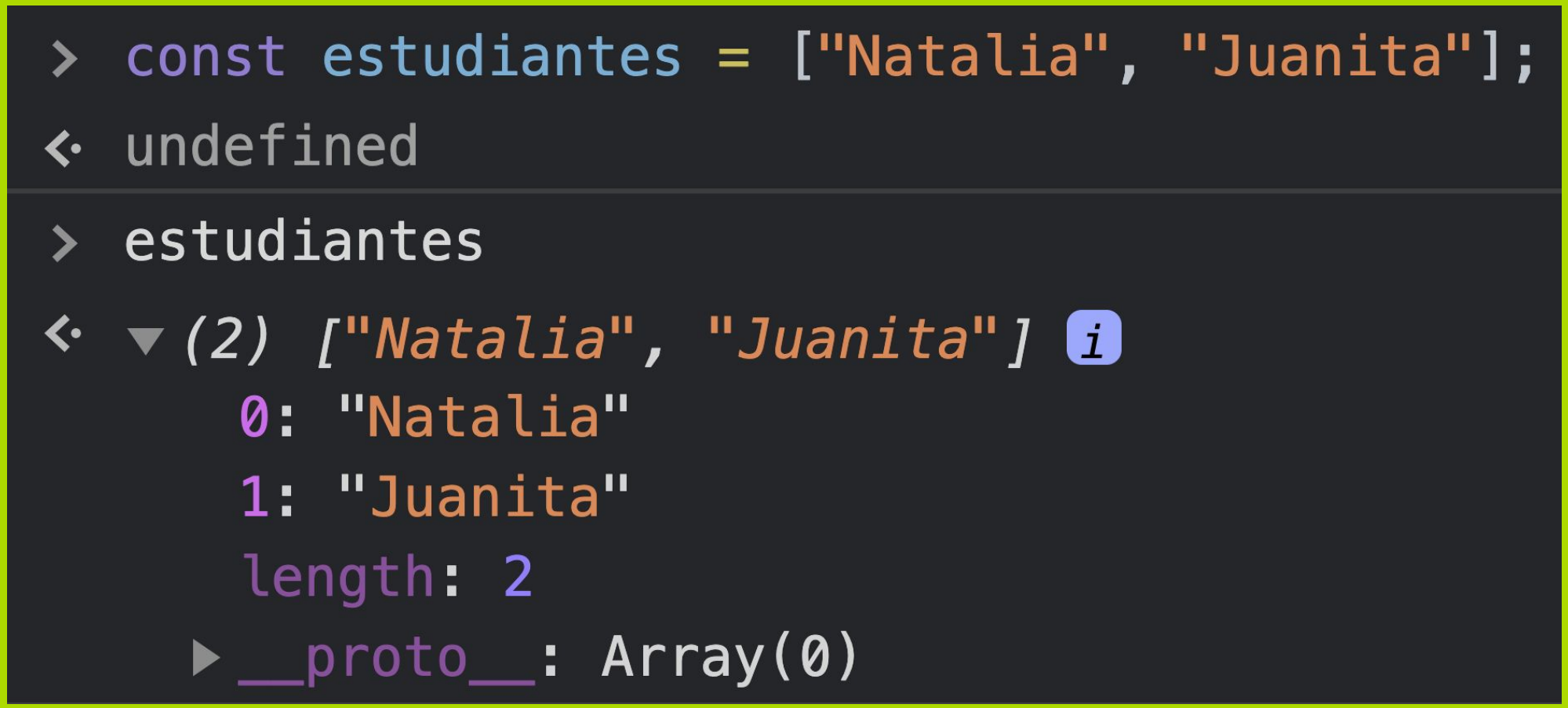
```
function Student() {
    this.name = 'Nombre';
    this.age = 18;
    this.points = 750;
}

const juanita = new Student();
```


const natalia = {
name: "Natalia",
age: 20,
points: 700,
};
undefined
natalia
{name: "Natalia", age: 20, points: 700}
age: 20
name: "Natalia"
points: 700
__proto__: Object


function Student() {
this.name = 'Nombre';
this.age = 18;
this.points = 750;
}
const juanita = new Student();
undefined
juanita
Student {name: "Nombre", age: 18, points: 750} i
age: 18
name: "Nombre"
points: 750
__proto__: Object

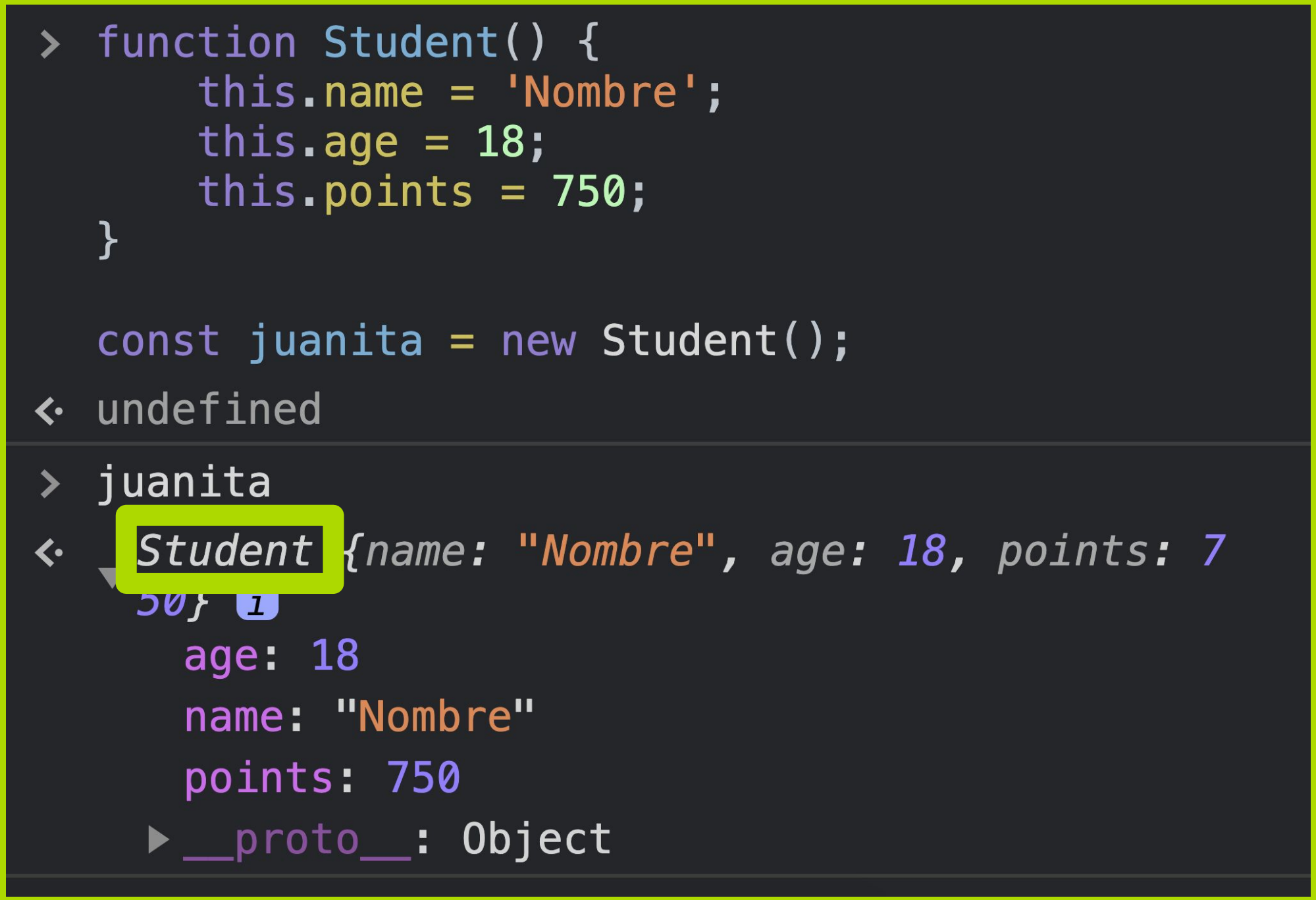

__proto__:
constructor: ƒ Object()
hasOwnProperty: ƒ hasOwnProperty()
isPrototypeOf: ƒ isPrototypeOf()
propertyIsEnumerable: ƒ propertyIsEnum…
toLocaleString: ƒ toLocaleString()
toString: ƒ toString()
valueOf: ƒ valueOf()
__defineGetter__: ƒ __defineGetter__()
__defineSetter__: ƒ __defineSetter__()
__lookupGetter__: ƒ __lookupGetter__()
__lookupSetter__: ƒ __lookupSetter__()
get __proto__: ƒ __proto__()
set __proto__: ƒ __proto__()

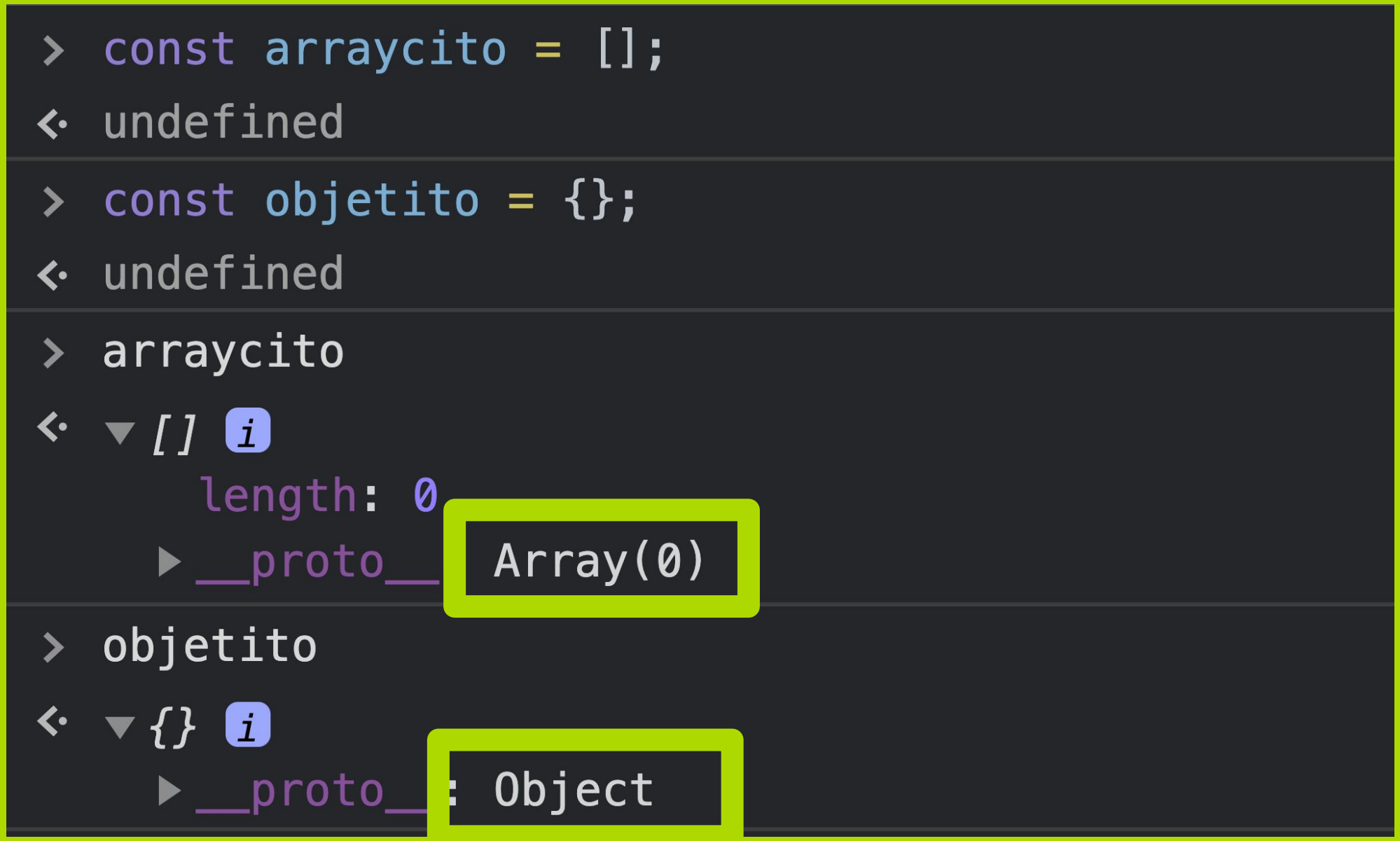

> natalia.hasOwnProperty("name")
true
> const objetito = {};
undefined
> objetito.hasOwnProperty("name")
false


> const estudiantes = ["Natalia", "Juanita"];
undefined
> estudiantes
(2) ["Natalia", "Juanita"] i
0: "Natalia"
1: "Juanita"
length: 2
__proto__: Array(0)

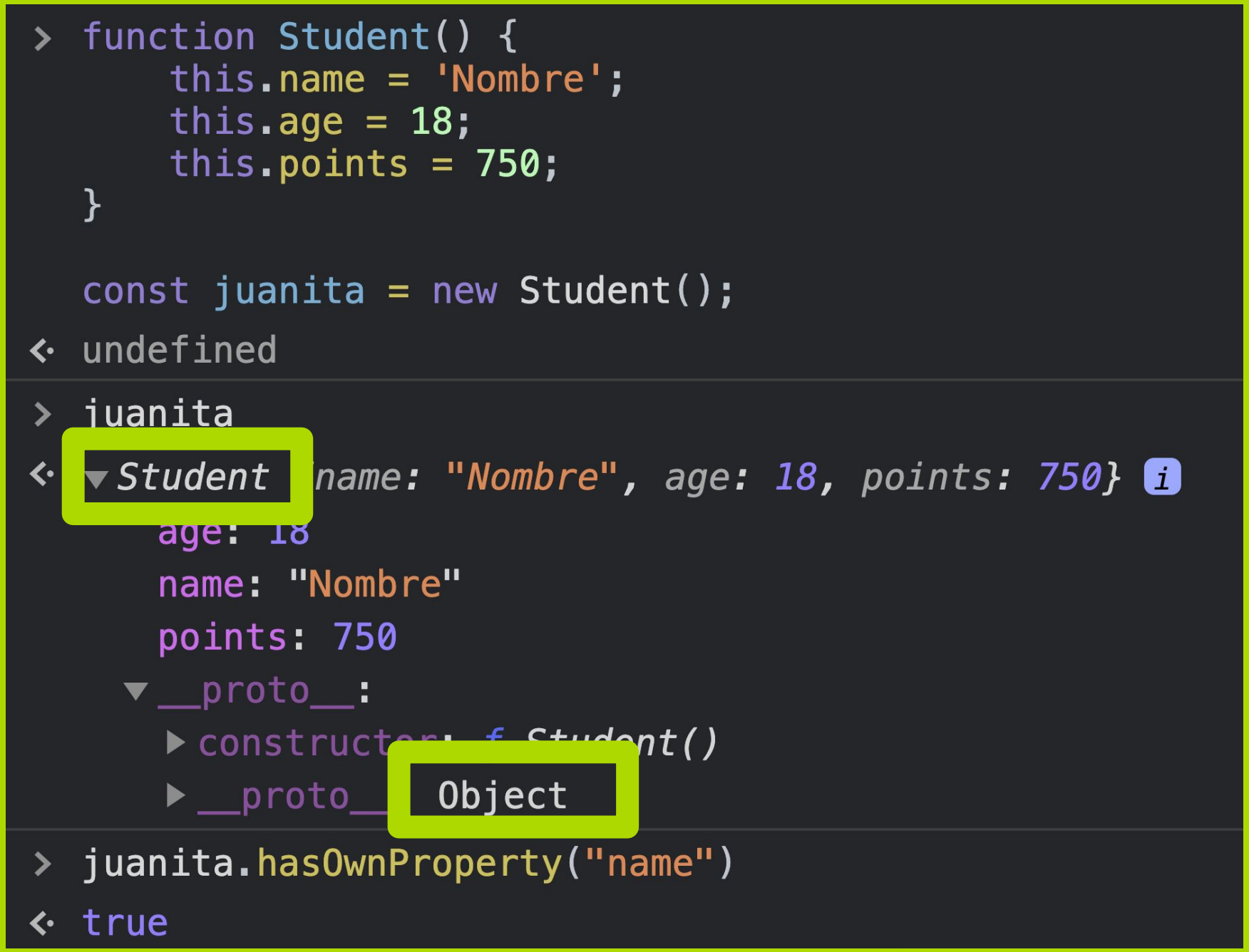
```
▼ __proto__: Array(0)
  ▶ concat: ƒ concat()
  ▶ constructor: ƒ Array()
  ▶ copyWithin: ƒ copyWithin()
  ▶ entries: ƒ entries()
  ▶ every: ƒ every()
  ▶ fill: ƒ fill()
  ▶ filter: ƒ filter()
  ▶ find: ƒ find()
  ▶ findIndex: ƒ findIndex()
  ▶ flat: ƒ flat()
  ▶ flatMap: ƒ flatMap()
  ▶ forEach: ƒ forEach()
  ▶ includes: ƒ includes()
  ▶ indexOf: ƒ indexOf()
  ▶ join: ƒ join()
  ▶ keys: ƒ keys()
  ▶ lastIndexOf: ƒ lastIndexOf()
    length: 0
  ▶ map: ƒ map()
  ▶ pop: ƒ pop()
  ▶ push: ƒ push()
  ▶ reduce: ƒ reduce()
  ▶ reduceRight: ƒ reduceRight()
  ▶ reverse: ƒ reverse()
  ▶ shift: ƒ shift()
  ▶ slice: ƒ slice()
  ▶ some: ƒ some()
  ▶ sort: ƒ sort()
  ▶ splice: ƒ splice()
  ▶ toLocaleString: ƒ toLocaleString()
  ▶ toString: ƒ toString()
  ▶ unshift: ƒ unshift()
  ▶ values: ƒ values()
  ▶ Symbol(Symbol.iterator): ƒ values()
  ▶ Symbol(Symbol.unscopables): {copyWithin: true, entries: true, fill: true, find: true, findIndex: true, …}
  ▶ __proto__: Object
```

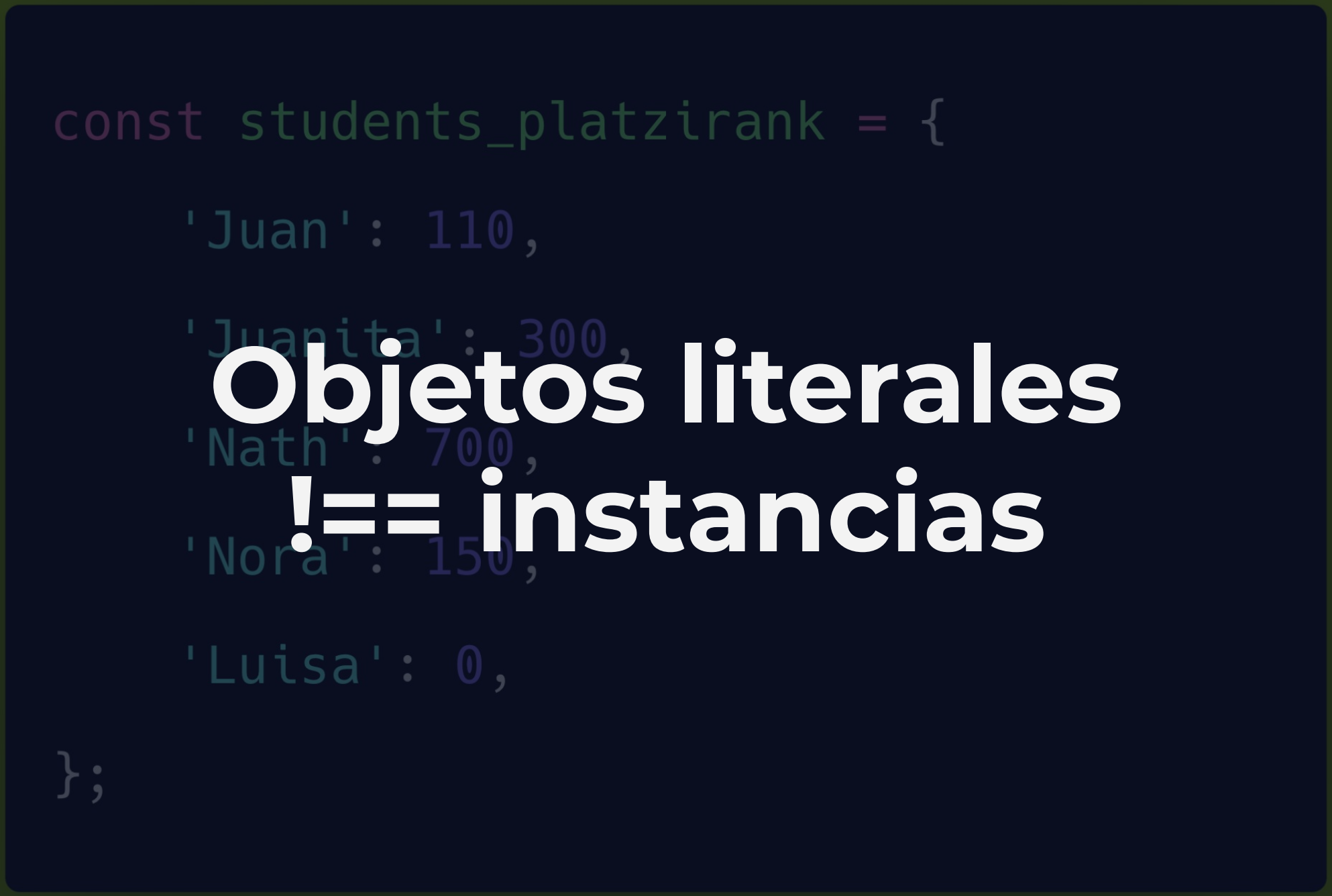
```
> estudiantes.push("JuanDC")
< 3
> estudiantes
< ▶ (3) ["Natalia", "Juanita", "JuanDC"]
> estudiantes.length
< 3
```


function Student() {
this.name = 'Nombre';
this.age = 18;
this.points = 750;
}
const juanita = new Student();
undefined
juanita
Student {name: "Nombre", age: 18, points: 750} i
age: 18
name: "Nombre"
points: 750
__proto__: Object

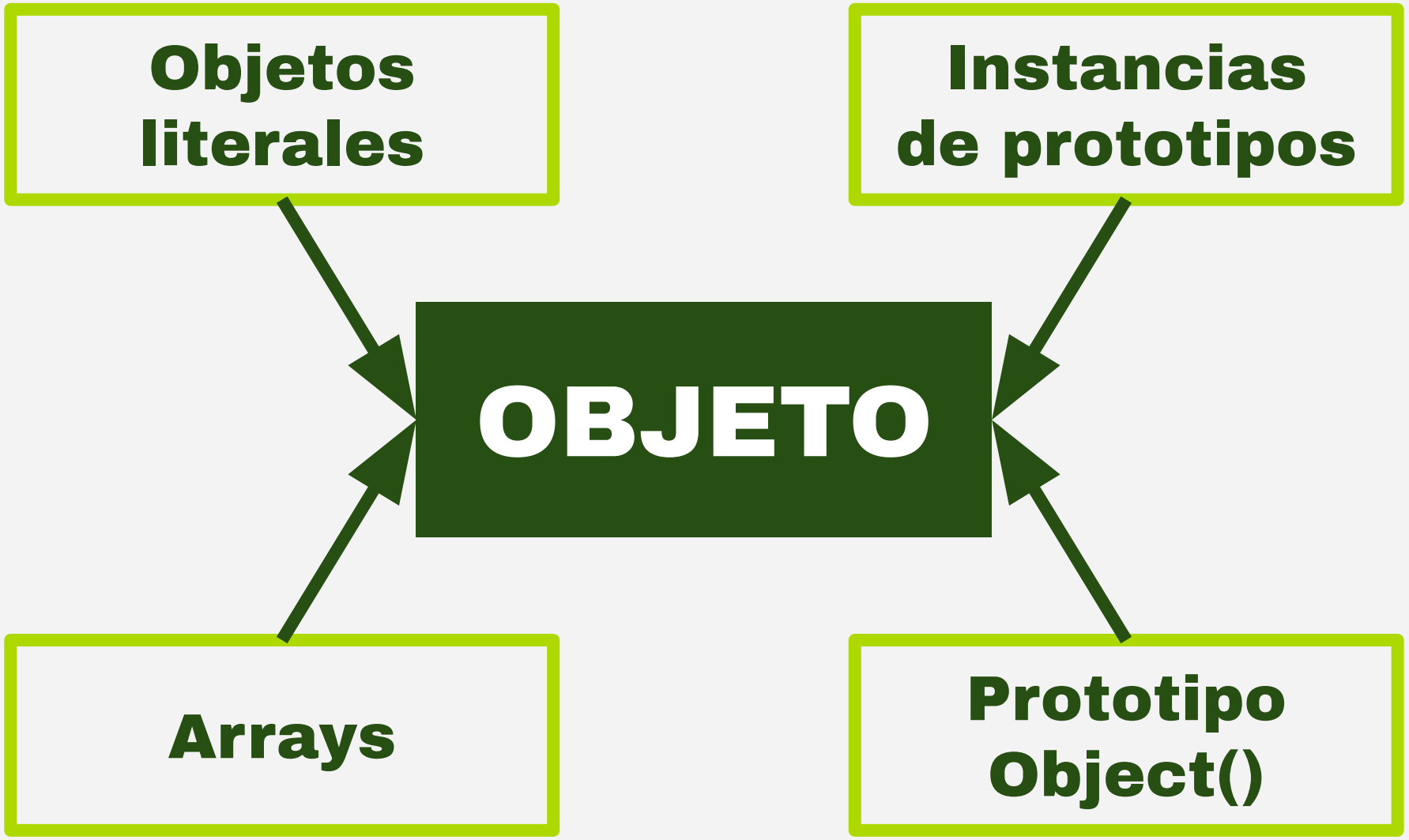

> const arraycito = [];
undefined
> const objetito = {};
undefined
> arraycito
[] i
length: 0
__proto__
Array(0)
> objetito
{} i
__proto__
: Object

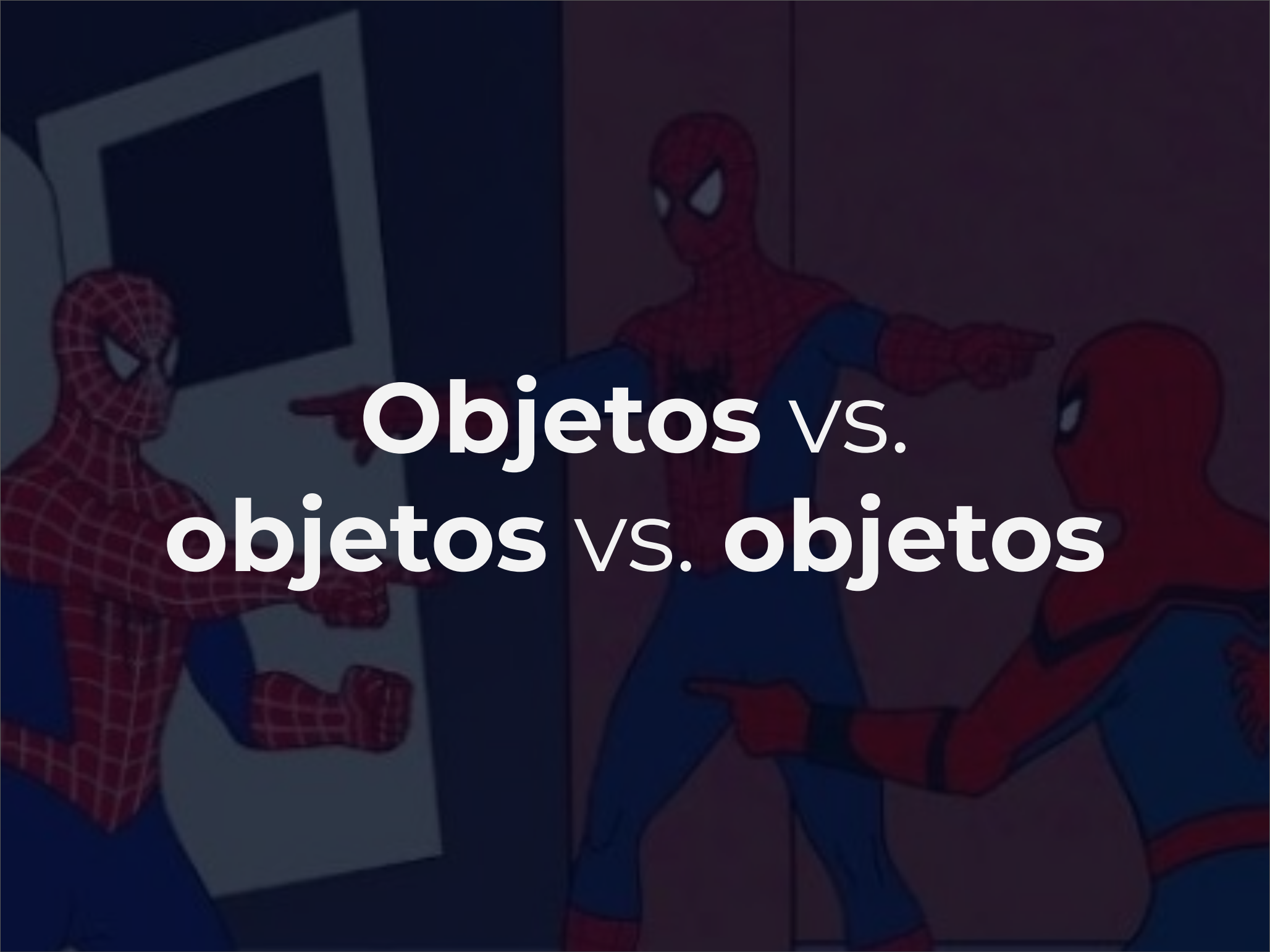

> const arraycito = new Array();
undefined
> const objetito = new Object();
undefined
> arraycito
[] i
length: 0
__proto__: Array(0)
> objetito
{} i
__proto__: Object


function Student() {
this.name = 'Nombre';
this.age = 18;
this.points = 750;
}
const juanita = new Student();
undefined
juanita
Student {name: "Nombre", age: 18, points: 750}
age: 18
name: "Nombre"
points: 750
__proto__:
constructor: f Student()
__proto__ Object
juanita.hasOwnProperty("name")
true


const students_platzirank = {
'Juan': 110,
'Juanita': 300,
'Nath': 700,
'Nora': 150,
'Luisa': 0,
};
Objetos literales
!== instancias


const students_platzirank = {
'Juan':
};
FAKE NEWS


Objetos literales
Instancias de prototipos
OBJETO
Arrays
Prototipo Object()


Objetos vs.
objetos vs. objetos


Objetos

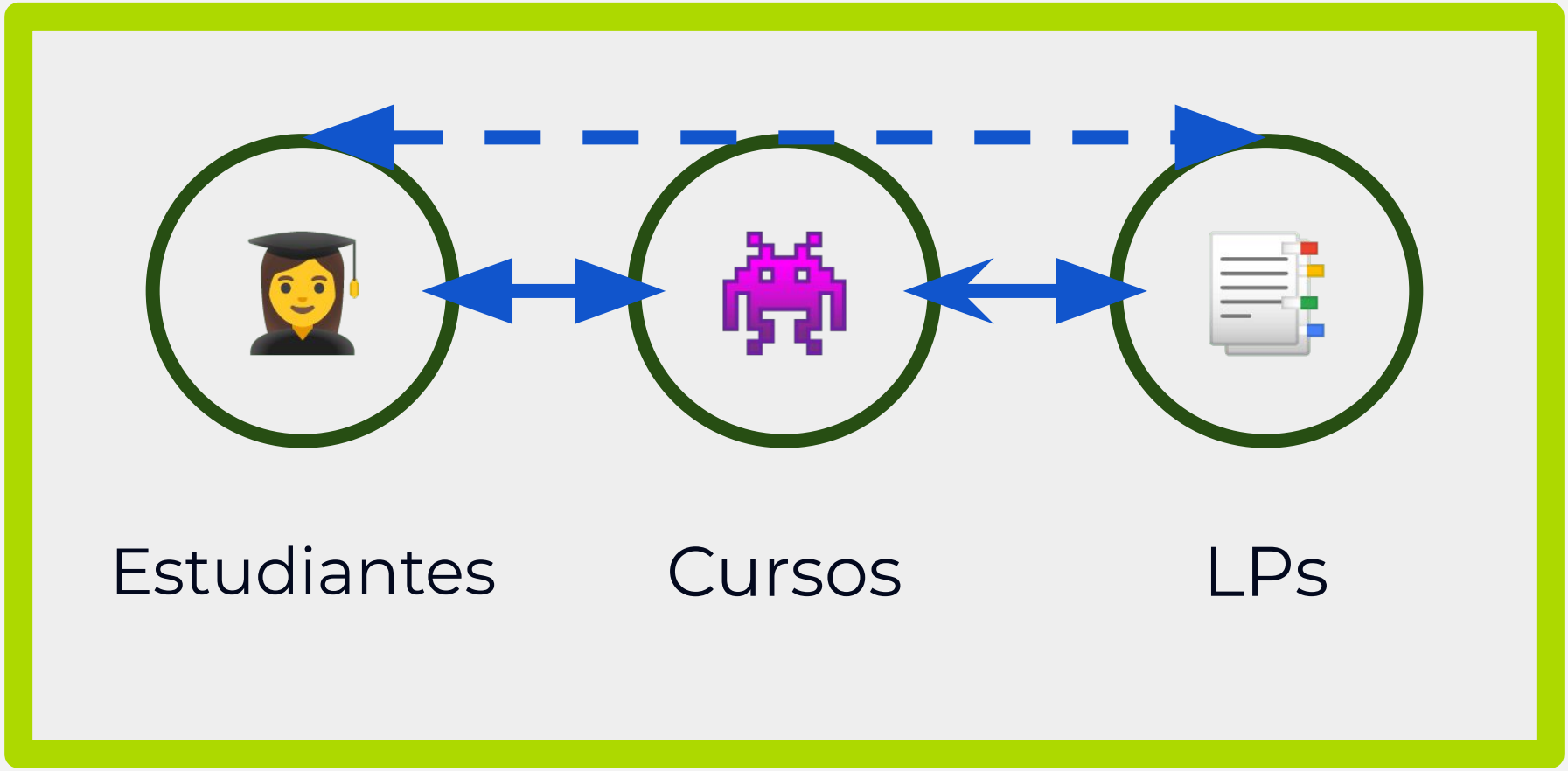
Cómo crear **objetos** y **prototipos** en **JavaScript**

Cómo crear **clases** en **JavaScript**

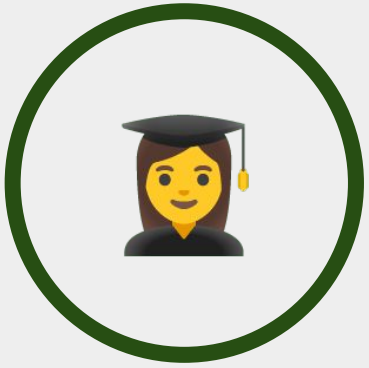
Ventajas de la **POO** en **JS**

Qué es abstracción

Platzi

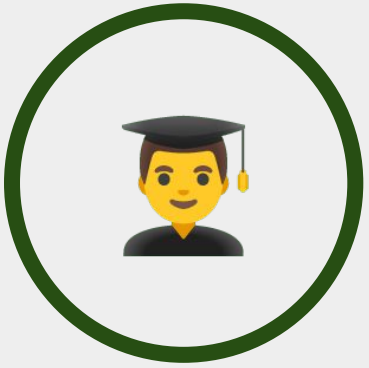


Estudiantes



Natalia

Miguel



JuanDC

Estudiante

Natalia

Nombre

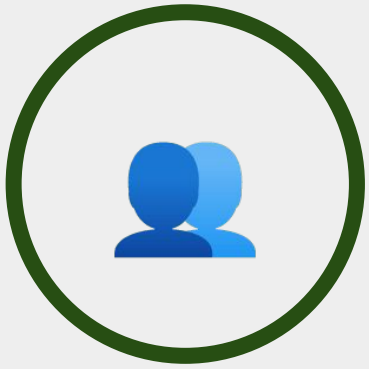
Edad

PlatziRank

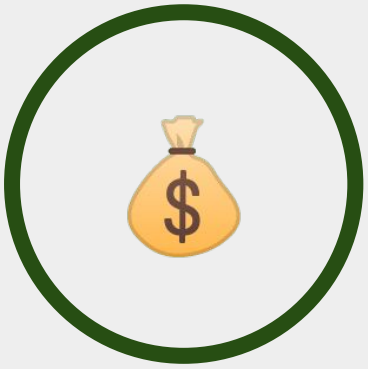
Abstracción en JavaScript

Qué es encapsulamiento

Guardianes

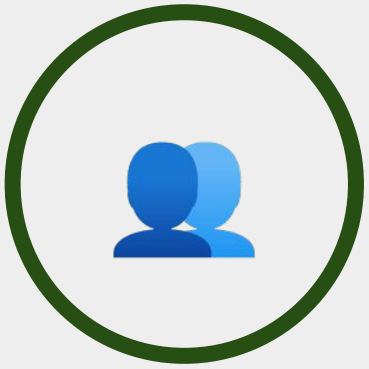


Personas

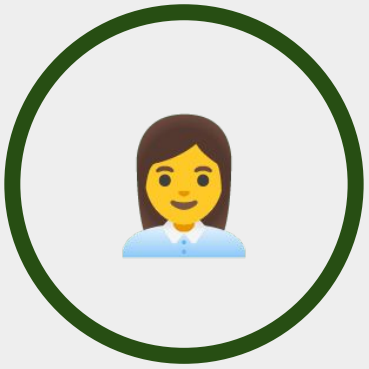


Banco

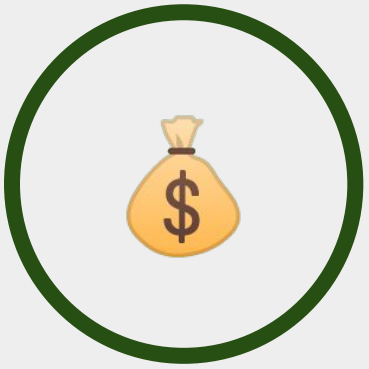
Guardianes



Personas



Banquera



Banco

Encapsular

- Esconder métodos y atributos 🙈
- No permitir la alteración de métodos y atributos 🤷

Encapsular... en JS

- Esconder métodos y atributos 🙈
- No permitir la alteración de métodos y atributos

Encapsular... en JS

- ✅ Getters y setters
- 🔜 Namespaces
- 🔜 Object.defineProperties
- ✅ Módulos de ES6

Getters y setters en JavaScript

Módulos de **ECMAScript 6**

Qué es herencia

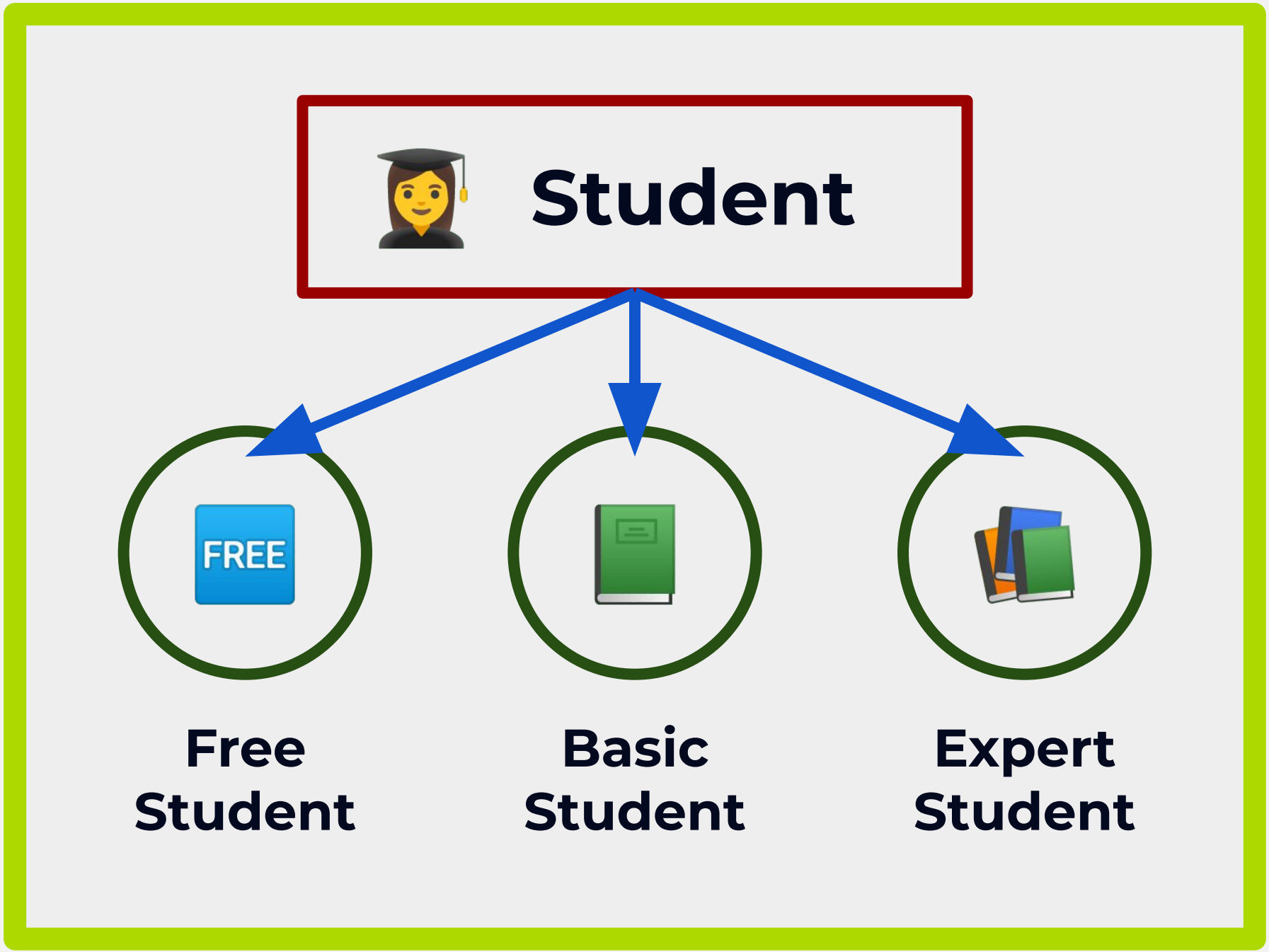

Estudiantes

Estudiantes

Suscripción Free

Suscripción Basic

Suscripción Expert

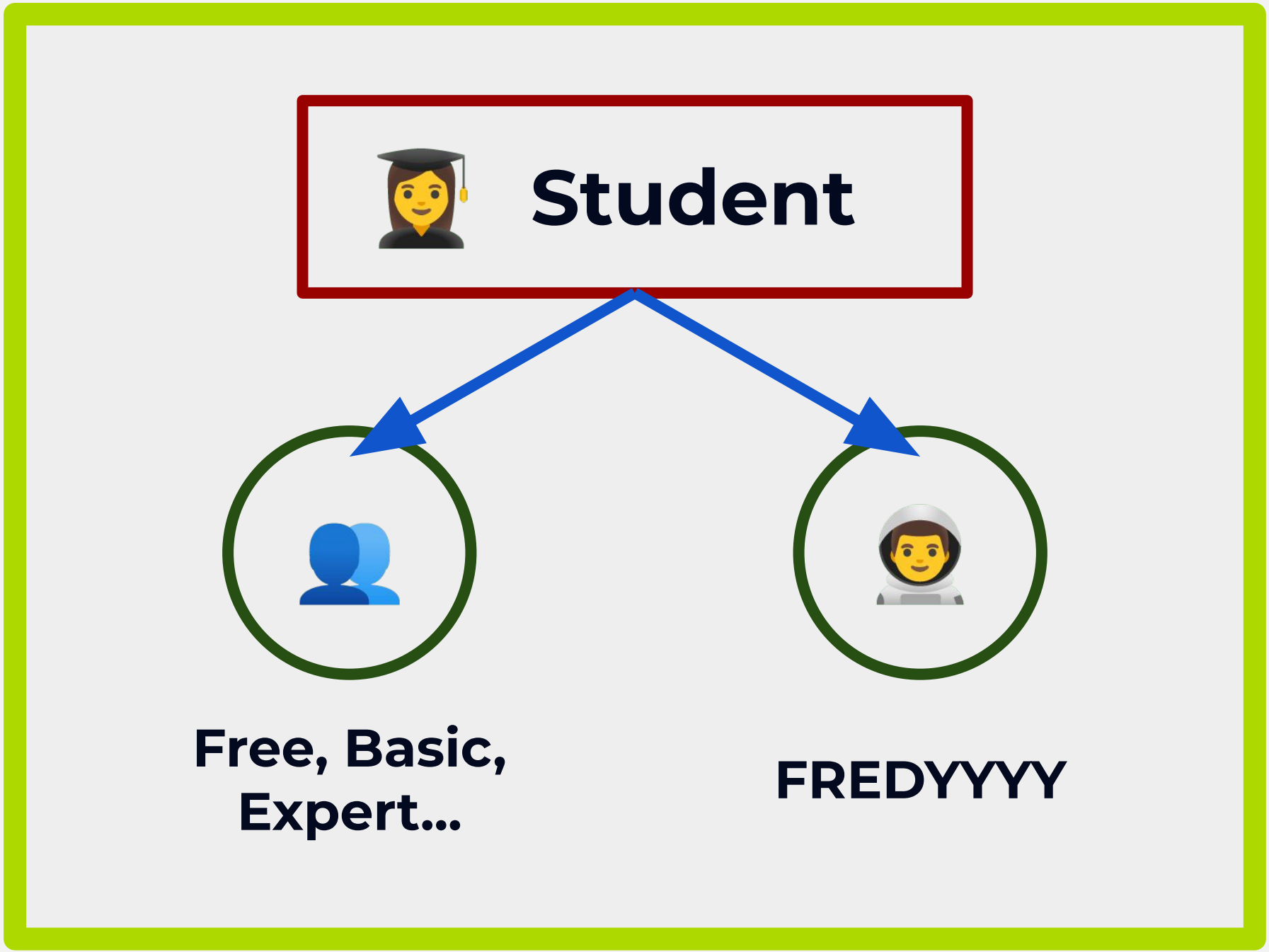

Student
FREE
Free Student
Basic Student
Expert Student

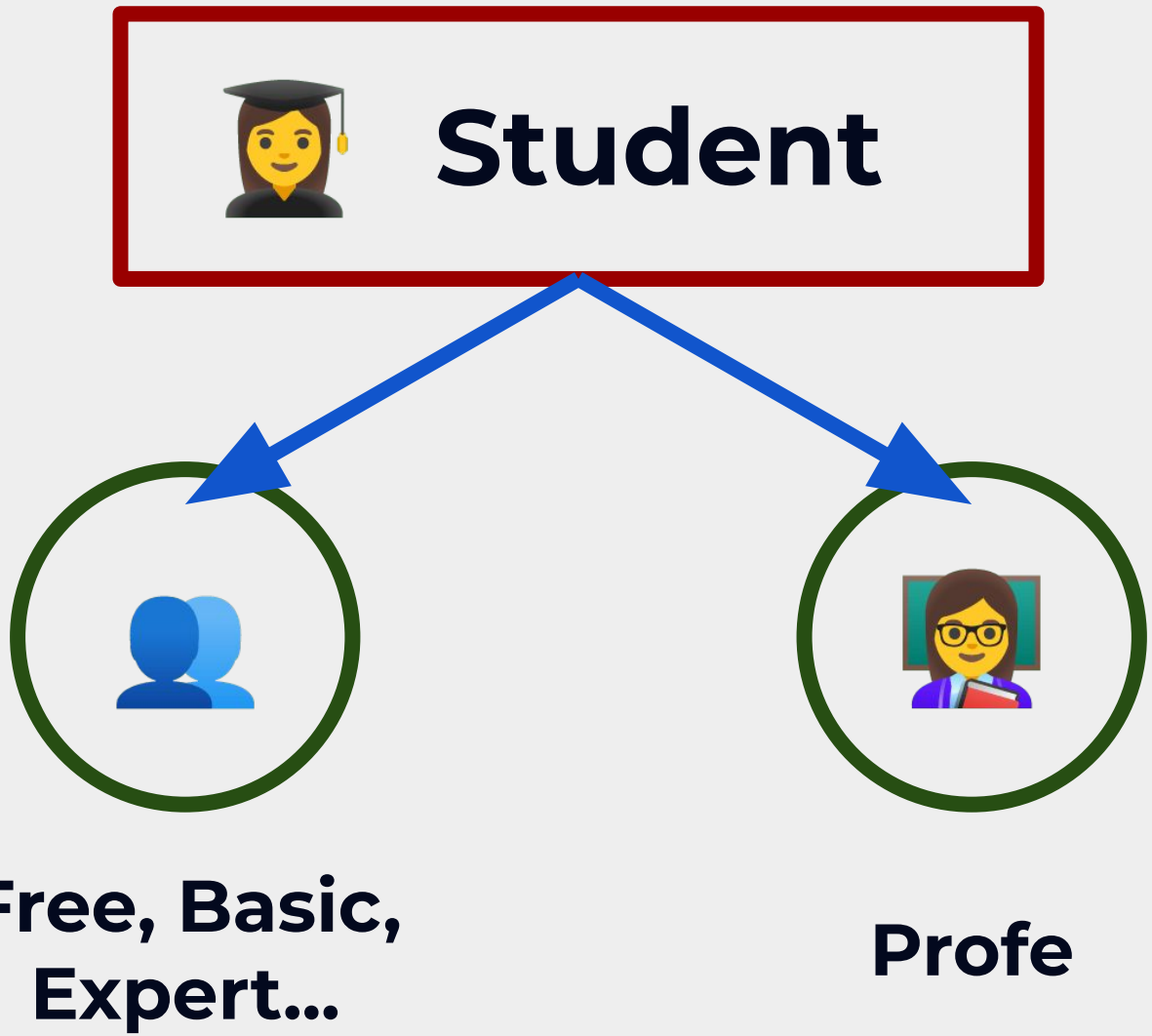
```
class Student {
  constructor(name) {
    this.name = name;
  }
}

class FreeStudent extends Student {}
```

Herencia en JavaScript

Qué es

polimorfismo


Student
Free, Basic, Expert...
FREDYYYY


Student
Free, Basic, Expert...
Profe

Tipos de polimorfismo

- Sobrecarga
- Paramétrico
- Inclusión

Polimorfismo en JavaScript

¿Quieres más
Cursos de POO
en **JavaScript**?