

## Tarea #8

Se llama a la función crearCuentaBancaria:

- Se invoca la función crearCuentaBancaria definida anteriormente.
- Se le pasa un argumento 1000, que representa el saldo inicial de la cuenta.

Asignación de la cuenta a la variable miCuenta:

- El resultado de la llamada a la función crearCuentaBancaria (que es un objeto que representa la cuenta bancaria) se asigna a la variable miCuenta.

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot shows the Python Tutor interface for JavaScript (ES6). The code being executed is as follows:

```
28     }  
29   }  
30 }  
31  
32 let miCuenta = crearCuentaBancaria(1000);  
33 console.log("Saldo inicial: " + miCuenta.consultarSaldo());  
34 miCuenta.realizarDeposito(500);  
35 console.log("Saldo después del depósito:" + miCuenta.consultarSaldo());  
36 miCuenta.realizarRetiro(200);  
37 console.log("Saldo después del retiro: " + miCuenta.consultarSaldo());  
38 //A continuación se presenta un ejemplo de como manejar excepciones  
39 try {  
40   miCuenta.depositar(100);  
41 } catch (e) {  
42   console.log(e.message);  
43 }  
44 try {  
45   miCuenta.retirar(100);  
46 } catch (e) {  
47   console.log(e.message);  
48 }
```

The interface includes a "Print output" area at the top right, a "Frames" panel on the left showing the "Global frame" and the "crearCuentaBancaria" function, and an "Objects" panel on the right showing the function's internal state and methods. The "Global frame" panel shows the "crearCuentaBancaria" function being called. The "Objects" panel shows the function's internal state, including the "saldo" variable and the "consultarSaldo" method.

Esta línea inicializa la variable saldo con el valor del saldo inicial de la cuenta.

1. let saldo = saldoInicial;

Esta línea declara una variable llamada saldo y le asigna el valor de la variable saldoInicial.

- let: Es una palabra clave de JavaScript que se utiliza para declarar variables con alcance de bloque. Esto significa que la variable saldo solo estará disponible dentro del bloque de código donde se declara.
- saldo: Es el nombre de la variable que se va a utilizar para almacenar el saldo de la cuenta bancaria.
- saldoInicial: Es una variable que se presume que contiene el valor del saldo inicial de la cuenta bancaria, proporcionado al crear la cuenta.

2. function depositar(cantidad) {

Esta línea define una función llamada depositar que recibe un parámetro llamado cantidad.

3. if (cantidad > 0) {

Esta línea inicia una instrucción condicional if que verifica si la cantidad a depositar (cantidad) es mayor que cero.

4. saldo += cantidad;

Esta línea aumenta el valor del saldo actual (saldo) en la cantidad a depositar (cantidad).

- saldo += cantidad;: Es un operador de asignación compuesto que suma el valor de cantidad al valor actual de saldo y almacena el resultado en saldo.

5. } else {

Esta línea inicia la rama else de la instrucción condicional if. Se ejecuta si la condición cantidad > 0 es false, es decir, si la cantidad a depositar no es mayor que cero.

6. console.log ("La cantidad depositada debe ser mayor a cero.");

Esta línea imprime un mensaje en la consola indicando que la cantidad depositada debe ser mayor a cero.

7. }

Esta línea cierra la instrucción condicional if.

8. function retirar (cantidad) {

Esta línea define una función llamada retirar que recibe un parámetro llamado cantidad. Similar a la función depositar, esta función simula la acción de realizar un retiro de la cuenta bancaria.

9. if (cantidad > 0 && cantidad <= saldo) {

Esta línea inicia una instrucción condicional if que verifica si la cantidad a retirar (cantidad) es mayor que cero y no excede el saldo disponible (saldo).

- cantidad > 0 && cantidad <= saldo: Es la condición que se evalúa. Si es true, significa que la cantidad a retirar es positiva y no supera el saldo actual.

10. saldo -= cantidad;

Esta línea disminuye el valor del saldo actual (saldo) en la cantidad a retirar (cantidad).

11. } else {

Esta línea inicia la rama else de la instrucción condicional if. Se ejecuta si la condición cantidad > 0 && cantidad <= saldo es false, es decir, si la cantidad a retirar no es válida.

12. console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")

Esta línea imprime un mensaje en la consola indicando que la cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot displays the Python Tutor interface for JavaScript (ES6). On the left, the code editor shows a function `crearCuentaBancaria` that initializes a balance and defines `depositar` and `retirar` methods. The execution is paused at line 2. Below the code editor, navigation controls and a progress indicator (Step 2 of 34) are visible. On the right, the 'Frames' pane shows the global frame with `crearCuentaBancaria` and `miCuenta` (undefined). The 'Objects' pane shows the object created by `crearCuentaBancaria`, which has properties `saldoInicial` (1000), `depositar`, and `retirar`. Arrows indicate the relationship between the function call in the code and the objects created in memory.

```
1 function crearCuentaBancaria (saldoInicial) {
2   let saldo = saldoInicial;
3   function depositar(cantidad) {
4     if (cantidad > 0) {
5       saldo += cantidad;
6     } else {
7       console.log ("La cantidad depositada debe ser mayor a cero.");
8     }
9   }
10  //Método privado para retirar dinero
11  function retirar (cantidad) {
12    if (cantidad > 0 && cantidad <= saldo) {
13      saldo -= cantidad;
14    } else {
15      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
16    }
17  }
18  //Retornamos un objeto con métodos públicos
19  return {
20    consultarSaldo: function () {
21      return saldo;
22    },
23    realizarDeposito: function (cantidad) {
24      depositar (cantidad);
25    },
26    realizarRetiro: function (cantidad) {
27      retirar (cantidad);
28    }
29  }
30 }
```

Global frame

- crearCuentaBancaria
- miCuenta undefined

Objects

- function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar (cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar (cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Este objeto literal define tres métodos públicos que permiten a los usuarios:

- Consultar el saldo actual de la cuenta.
- Realizar depósitos en la cuenta.
- Realizar retiros de la cuenta.

Al retornar este objeto desde la función `crearCuentaBancaria`, se proporciona una interfaz para interactuar con la cuenta bancaria simulada.

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

```

15 console.log ("La cantidad a retirar debe
16 }
17 }
18 //Retornamos un objeto con métodos públicos
19 return {
20   consultarSaldo: function () {
21     return saldo;
22   },
23   realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25   },
26   realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28   }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34

```

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria	undefined
miCuenta	undefined
crearCuentaBancaria	Object
saldoInicial	1000
depositar	function
retirar	function
saldo	1000

Objects

```

function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log ("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Metodo privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}

function depositar(cantidad) {
  if (cantidad > 0) {
    saldo += cantidad;
  } else {
    console.log ("La cantidad depositada debe ser mayor a cero.");
  }
}

function retirar(cantidad) {
  if (cantidad > 0 && cantidad <= saldo) {
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
  }
}

```

Step 3 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

```

15 console.log ("La cantidad a retirar debe
16 }
17 }
18 //Retornamos un objeto con métodos públicos
19 return {
20   consultarSaldo: function () {
21     return saldo;
22   },
23   realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25   },
26   realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28   }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34

```

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria	undefined
miCuenta	Object
saldoInicial	1000
depositar	function
retirar	function
saldo	1000

Objects

```

function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log ("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Metodo privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}

function depositar(cantidad) {
  if (cantidad > 0) {
    saldo += cantidad;
  } else {
    console.log ("La cantidad depositada debe ser mayor a cero.");
  }
}

function retirar(cantidad) {
  if (cantidad > 0 && cantidad <= saldo) {
    saldo -= cantidad;
  } else {
    console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
  }
}

```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 4 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

var miCuenta = crearCuentaBancaria(1000);

- Esta línea crea una cuenta bancaria utilizando la función crearCuentaBancaria y almacena la referencia a la cuenta en la variable miCuenta.

- crearCuentaBancaria(1000): Se invoca la función crearCuentaBancaria pasándole como argumento el valor inicial del saldo (1000).
- var miCuenta: Se declara una variable llamada miCuenta utilizando la palabra clave var.
- =: Se asigna el valor retornado por la función crearCuentaBancaria (que es la referencia a la cuenta creada) a la variable miCuenta.

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot shows the Python Tutor interface with JavaScript code. The code defines functions for depositing and withdrawing money, and creates a bank account object. The execution is at line 33, where the initial balance is logged.

```

22     },
23     realizarDeposito: function (cantidad) {
24         depositar (cantidad);
25     },
26     realizarRetiro: function (cantidad) {
27         retirar (cantidad);
28     }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSaldo());
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como manejar
40 try {
41     miCuenta.depositar(100);

```

The right panel shows the 'Frames' and 'Objects' sections. The 'Global frame' contains the functions and the 'miCuenta' object. The 'Objects' section shows the 'miCuenta' object with its methods: 'consultarSaldo', 'realizarDeposito', and 'realizarRetiro'.

Object	Method	Code
miCuenta	consultarSaldo	function () { return saldo; }
	realizarDeposito	function (cantidad) { depositar (cantidad); }
	realizarRetiro	function (cantidad) { retirar (cantidad); }

console.log("Saldo inicial: " + miCuenta.consultarSaldo());

- Esta línea imprime el saldo inicial de la cuenta en la consola del navegador.

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane
40 try {
41   miCuenta.depositar(100);
42 } catch (e) {
43   console.log(e.message);
44 }
45 try {
46   miCuenta.retirar(100);
47 } catch (e) {
48   console.log(e.message)

```

Print output (drag lower right corner to resize)

Frames

Objects

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Metodo privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos publicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar(cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar(cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 6 of 34

Sponsor: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Se retorna el saldo

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)

```
11 function retirar (cantidad) {
12   if (cantidad > 0 && cantidad <= saldo) {
13     saldo -= cantidad;
14   } else {
15     console.log ("La cantidad a retirar debe
16   }
17 }
18 //Retornamos un objeto con métodos públicos
19 return {
20   consultarSaldo: function () {
21     return saldo;
22   },
23   realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25   },
26   realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28   }
29 }
```

Print output (drag lower right corner to resize)

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo

parent:depositar

parent:retirar

Return value

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar (cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 7 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se imprime saldo inicial, saldo después del depósito y saldo después del retiro

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
23     realizarDeposito: function (cantidad) {
24         depositar (cantidad);
25     },
26     realizarRetiro: function (cantidad) {
27         retirar (cantidad);
28     }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane}
40 try {
41     miCuenta.depositar(100);
```

Print output (drag lower right corner to resize)

Frames

- Global frame
- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log ("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar (cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 8 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se realiza un depósito de 500 a la cuenta.



## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)  
[known limitations](#)

```
18 //Retornamos un objeto con métodos públicos
19 return {
20     consultarSaldo: function () {
21         return saldo;
22     },
23     realizarDeposito: function (cantidad) {
24         depositar (cantidad);
25     },
26     realizarRetiro: function (cantidad) {
27         retirar (cantidad);
28     }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this	
parent:saldo	1000
parent:depositar	
parent:retirar	
cantidad	500

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 10 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Si la cantidad es mayor a 0 suma se actualiza a suma mas cantidad y si no se imprime "La cantidad depositada debe ser mayor a cero."

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)  
[known limitations](#)

```
1 function crearCuentaBancaria(saldoInicial) {
2   let saldo = saldoInicial;
3   function depositar(cantidad) {
4     if (cantidad > 0) {
5       saldo += cantidad;
6     } else {
7       console.log("La cantidad depositada debe ser mayor a cero.");
8     }
9   }
10  //Método privado para retirar dinero
11  function retirar(cantidad) {
12    if (cantidad > 0 && cantidad <= saldo) {
13      saldo -= cantidad;
14    } else {
15      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
16    }
17  }
18  //Retornamos un objeto con métodos públicos
19  return {
20    consultarSaldo: function() { return saldo; },
21    realizarDeposito: function(cantidad) { depositar(cantidad); },
22    realizarRetiro: function(cantidad) { retirar(cantidad); }
23  };
24 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

cantidad 500

depositar

parent:saldo 1000

parent:depositar

parent:retirar

cantidad 500

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function() {  
 return saldo;  
 },  
 realizarDeposito: function(cantidad) {  
 depositar(cantidad);  
 },  
 realizarRetiro: function(cantidad) {  
 retirar(cantidad);  
 }  
 };  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 11 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)  
[known limitations](#)

```
1 function crearCuentaBancaria(saldoInicial) {
2   let saldo = saldoInicial;
3   function depositar(cantidad) {
4     if (cantidad > 0) {
5       saldo += cantidad;
6     } else {
7       console.log("La cantidad depositada debe ser mayor a cero.");
8     }
9   }
10  //Método privado para retirar dinero
11  function retirar(cantidad) {
12    if (cantidad > 0 && cantidad <= saldo) {
13      saldo -= cantidad;
14    } else {
15      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
16    }
17  }
18  //Retornamos un objeto con métodos públicos
19  return {
20    consultarSaldo: function() { return saldo; },
21    realizarDeposito: function(cantidad) { depositar(cantidad); },
22    realizarRetiro: function(cantidad) { retirar(cantidad); }
23  };
24 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1000

parent:depositar

parent:retirar

cantidad 500

depositar

parent:saldo 1000

parent:depositar

parent:retirar

cantidad 500

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function() {  
 return saldo;  
 },  
 realizarDeposito: function(cantidad) {  
 depositar(cantidad);  
 },  
 realizarRetiro: function(cantidad) {  
 retirar(cantidad);  
 }  
 };  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 12 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)  
[known limitations](#)

```
1 function crearCuentaBancaria (saldoInicial) {
2   let saldo = saldoInicial;
3   function depositar(cantidad) {
4     if (cantidad > 0) {
5       saldo += cantidad;
6     } else {
7       console.log ("La cantidad depositada debe ser mayor a cero.");
8     }
9   }
10  //Método privado para retirar dinero
11  function retirar (cantidad) {
12    if (cantidad > 0 && cantidad <= saldo) {
13      saldo -= cantidad;
14    } else {
15      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
16    }
17  }
18  //Retornamos un objeto con métodos públicos
19  return {
20    consultarSaldo: function () {
21      return saldo;
22    },
23    realizarDeposito: function (cantidad) {
24      depositar (cantidad);
25    },
26    realizarRetiro: function (cantidad) {
27      retirar (cantidad);
28    }
29  };
30 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 500

depositar

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 500

Return value undefined

Objects

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 13 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se realiza el deposito exitosamente:

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

known limitations

```

15 console.log("La cantidad a retirar debe
16 }
17 }
18 //Retornamos un objeto con métodos públicos
19 return {
20   consultarSaldo: function () {
21     return saldo;
22   },
23   realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25   },
26   realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28   }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa

```

line that just executed

next line to execute

Step 14 of 34

<< First

< Prev

Next >

Last >>

Sponsor: interested in a [free Python tip every week?](#)

Get AI Help

Move and hide objects

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Objects

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo

parent:depositar

parent:retirar

cantidad

Return value

function crearCuentaBancaria(saldoInicial) {

let saldo = saldoInicial;

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

}

//Método privado para retirar dinero

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

}

//Retornamos un objeto con métodos públicos

return {

consultarSaldo: function () {

return saldo;

},

realizarDeposito: function (cantidad) {

depositar (cantidad);

},

realizarRetiro: function (cantidad) {

retirar (cantidad);

}

}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

}

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

}

Deposito de 500

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

```
25     },
26     realizarRetiro: function (cantidad) {
27         retirar (cantidad);
28     }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como mane
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log ("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar (cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 15 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se retorna el saldo

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)

```
11 function retirar (cantidad) {
12   if (cantidad > 0 && cantidad <= saldo) {
13     saldo -= cantidad;
14   } else {
15     console.log ("La cantidad a retirar debe
16   }
17 }
18 //Retornamos un objeto con métodos públicos
19 return {
20   consultarSaldo: function () {
21     return saldo;
22   },
23   realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25   },
26   realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28   }
29 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

crearCuentaBancaria

miCuenta

parent:saldo 1500

parent:depositar

parent:retirar

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar (cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log ("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 16 of 34

Sponsor: Interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se imprime saldo despues de deposito

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

```
25 },
26 realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28 }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como maneja
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000

Frames

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar(cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}
```

object

property	value
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 18 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Retiro de 200

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)

```
25 },
26 realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28 }
29 }
30 }
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como maneja
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar(cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}
```

object

property	value
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 19 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Funcion de retiro

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

```

25      },
26      realizarRetiro: function (cantidad) {
27          retirar (cantidad);
28      }
29  }
30  }
31
32  var miCuenta = crearCuentaBancaria (1000);
33  console.log("Saldo inicial: " + miCuenta.consultarSa
34  miCuenta.realizarDeposito(500);
35  console.log("Saldo después del depósito:" + miCuenta
36  miCuenta.realizarRetiro(200);
37  console.log("Saldo después del retiro: " + miCuenta.
38
39  //A continuación se presenta un ejemplo de como maneja
40  try {
41      miCuenta.depositar(100);
42  } catch (e) {
43      console.log(e.message);

```

[Edit this code](#)

Step 20 of 34

[Get AI Help](#)

[Move and hide objects](#)

Print output (drag lower right corner to resize)
 

Saldo inicial: 1000  
 Saldo después del depósito:1500

Frames

Global frame	
crearCuentaBancaria	
miCuenta	
this	
parent:saldo	1500
parent:depositar	
parent:retirar	
cantidad	200

Objects

```

function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Método privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}

```

object	
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

```

function depositar(cantidad) {
  if (cantidad > 0) {
    saldo += cantidad;
  } else {
    console.log("La cantidad depositada debe ser mayor a cero.");
  }
}

function retirar(cantidad) {
  if (cantidad > 0 && cantidad <= saldo) {
    saldo -= cantidad;
  } else {
    console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
  }
}

```

Si la cantidad es mayor que cero y menor o igual que saldo entonces se actualiza el saldo a saldo menos cantidad de otra manera se imprime: "La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible."



## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)

```
1 function crearCuentaBancaria(saldoInicial) {
2   let saldo = saldoInicial;
3   function depositar(cantidad) {
4     if (cantidad > 0) {
5       saldo += cantidad;
6     } else {
7       console.log("La cantidad depositada debe ser mayor a cero.");
8     }
9   }
10  //Método privado para retirar dinero
11  function retirar(cantidad) {
12    if (cantidad > 0 && cantidad <= saldo) {
13      saldo -= cantidad;
14    } else {
15      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
16    }
17  }
18  //Retornamos un objeto con métodos públicos
19  return {
20    consultarSaldo: function () {
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 200

retirar

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 200

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Método privado para retirar dinero
  function retirar(cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar(cantidad);
    }
  }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar(cantidad); }
realizarRetiro	function (cantidad) { retirar(cantidad); }

```
function depositar(cantidad) {
  if (cantidad > 0) {
    saldo += cantidad;
  } else {
    console.log("La cantidad depositada debe ser mayor a cero.");
  }
}

function retirar(cantidad) {
  if (cantidad > 0 && cantidad <= saldo) {
    saldo -= cantidad;
  } else {
    console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
  }
}
```

→ line that just executed  
→ next line to execute

Step 21 of 34

Sponsor: Interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)

```

1 let saldo = saldoInicial;
2 function depositar(cantidad) {
3   if (cantidad > 0) {
4     saldo += cantidad;
5   } else {
6     console.log("La cantidad depositada debe ser mayor a cero.");
7   }
8 }
9 //Método privado para retirar dinero
10 function retirar(cantidad) {
11   if (cantidad > 0 && cantidad <= saldo) {
12     saldo -= cantidad;
13   } else {
14     console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
15   }
16 }
17 //Retornamos un objeto con métodos públicos
18 return {
19   consultarSaldo: function () {
20     return saldo;
21   },
22   realizarDeposito: function (cantidad) {
23     depositar(cantidad);
24   },
25   realizarRetiro: function (cantidad) {
26     retirar(cantidad);
27   }
28 };

```

Step 22 of 34

Print output (drag lower right corner to resize)

```

Saldo inicial: 1000
Saldo después del depósito:1500

```

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 200

retirar

parent:saldo 1500

parent:depositar

parent:retirar

cantidad 200

Objects

function crearCuentaBancaria(saldoInicial) {

let saldo = saldoInicial;

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

//Método privado para retirar dinero

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

//Retornamos un objeto con métodos públicos

return {

consultarSaldo: function () {

return saldo;

},

realizarDeposito: function (cantidad) {

depositar(cantidad);

},

realizarRetiro: function (cantidad) {

retirar(cantidad);

}

}

}

object

consultarSaldo	function () {	return saldo;	}
realizarDeposito	function (cantidad) {	depositar(cantidad);	}
realizarRetiro	function (cantidad) {	retirar(cantidad);	}

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

}

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

}

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

JavaScript (ES6)

```

1 let saldo = saldoInicial;
2 function depositar(cantidad) {
3   if (cantidad > 0) {
4     saldo += cantidad;
5   } else {
6     console.log("La cantidad depositada debe ser mayor a cero.");
7   }
8 }
9 //Método privado para retirar dinero
10 function retirar(cantidad) {
11   if (cantidad > 0 && cantidad <= saldo) {
12     saldo -= cantidad;
13   } else {
14     console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
15   }
16 }
17 //Retornamos un objeto con métodos públicos
18 return {
19   consultarSaldo: function () {
20     return saldo;
21   },
22   realizarDeposito: function (cantidad) {
23     depositar(cantidad);
24   },
25   realizarRetiro: function (cantidad) {
26     retirar(cantidad);
27   }
28 };

```

Step 23 of 34

Print output (drag lower right corner to resize)

```

Saldo inicial: 1000
Saldo después del depósito:1500

```

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1300

parent:depositar

parent:retirar

cantidad 200

retirar

parent:saldo 1300

parent:depositar

parent:retirar

cantidad 200

Return value

undefined

Objects

function crearCuentaBancaria(saldoInicial) {

let saldo = saldoInicial;

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

//Método privado para retirar dinero

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

//Retornamos un objeto con métodos públicos

return {

consultarSaldo: function () {

return saldo;

},

realizarDeposito: function (cantidad) {

depositar(cantidad);

},

realizarRetiro: function (cantidad) {

retirar(cantidad);

}

}

}

object

consultarSaldo	function () {	return saldo;	}
realizarDeposito	function (cantidad) {	depositar(cantidad);	}
realizarRetiro	function (cantidad) {	retirar(cantidad);	}

function depositar(cantidad) {

if (cantidad > 0) {

saldo += cantidad;

} else {

console.log("La cantidad depositada debe ser mayor a cero.");

}

}

function retirar(cantidad) {

if (cantidad > 0 && cantidad <= saldo) {

saldo -= cantidad;

} else {

console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");

}

}

Se retira la cantidad y se imprime saldo despues de retiro

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6) [known limitations](#)

```

18 //Retornamos un objeto con métodos públicos
19 return {
20     consultarSaldo: function () {
21         return saldo;
22     },
23     realizarDeposito: function (cantidad) {
24         depositar (cantidad);
25     },
26     realizarRetiro: function (cantidad) {
27         retirar (cantidad);
28     }
29 }
30
31 var miCuenta = crearCuentaBancaria(1000);
32 console.log("Saldo inicial: " + miCuenta.consultarSa
33 miCuenta.realizarDeposito(500);
34 console.log("Saldo después del depósito:" + miCuenta
35 miCuenta.realizarRetiro(200);

```

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo

parent:depositar

parent:retirar

cantidad

Return value

undefined

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
}

function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
}

Step 24 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6) [known limitations](#)

```

20 consultarSaldo: function () {
21     return saldo;
22 },
23 realizarDeposito: function (cantidad) {
24     depositar (cantidad);
25 },
26 realizarRetiro: function (cantidad) {
27     retirar (cantidad);
28 }
29 }
30
31 var miCuenta = crearCuentaBancaria(1000);
32 console.log("Saldo inicial: " + miCuenta.consultarSa
33 miCuenta.realizarDeposito(500);
34 console.log("Saldo después del depósito:" + miCuenta
35 miCuenta.realizarRetiro(200);
36 console.log("Saldo después del retiro: " + miCuenta.
37 //A continuación se presenta un ejemplo de como manejar

```

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame

crearCuentaBancaria

miCuenta

Objects

function crearCuentaBancaria(saldoInicial) {  
 let saldo = saldoInicial;  
 function depositar(cantidad) {  
 if (cantidad > 0) {  
 saldo += cantidad;  
 } else {  
 console.log("La cantidad depositada debe ser mayor a cero.");  
 }  
 }  
 //Método privado para retirar dinero  
 function retirar(cantidad) {  
 if (cantidad > 0 && cantidad <= saldo) {  
 saldo -= cantidad;  
 } else {  
 console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
 }  
 }  
 //Retornamos un objeto con métodos públicos  
 return {  
 consultarSaldo: function () {  
 return saldo;  
 },  
 realizarDeposito: function (cantidad) {  
 depositar (cantidad);  
 },  
 realizarRetiro: function (cantidad) {  
 retirar (cantidad);  
 }  
 }  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 25 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

20

consultarSaldo: function () {

21

return saldo;

22

},

23

realizarDeposito: function (cantidad) {

24

depositar (cantidad);

25

},

26

realizarRetiro: function (cantidad) {

27

retirar (cantidad);

28

}

29

}

30

}

31

var miCuenta = crearCuentaBancaria (1000);

32

console.log("Saldo inicial: " + miCuenta.consultarSa

33

miCuenta.realizarDeposito(500);

34

console.log("Saldo después del depósito:" + miCuenta

35

miCuenta.realizarRetiro(200);

36

console.log("Saldo después del retiro: " + miCuenta.

37

38

39

// la continuación se representa un elemento de como mane

40

[Edit this code](#)

==> line that just executed

==> next line to execute

<< First

< Prev

Next >

Last >>

Step 26 of 34

Sponsors: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Objects

Global frame

crearCuentaBancaria

miCuenta

this

parent:saldo 1300

parent:depositar

parent:retirar

function crearCuentaBancaria(saldoInicial) {  
let saldo = saldoInicial;  
function depositar(cantidad) {  
if (cantidad > 0) {  
saldo += cantidad;  
} else {  
console.log ("La cantidad depositada debe ser mayor a cero.");  
}  
}  
//Metodo privado para retirar dinero  
function retirar (cantidad) {  
if (cantidad > 0 && cantidad <= saldo) {  
saldo -= cantidad;  
} else {  
console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")  
}  
}  
//Retornamos un objeto con métodos publicos  
return {  
consultarSaldo: function () {  
return saldo;  
},  
realizarDeposito: function (cantidad) {  
depositar (cantidad);  
},  
realizarRetiro: function (cantidad) {  
retirar (cantidad);  
}  
}  
}

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

function depositar(cantidad) {  
if (cantidad > 0) {  
saldo += cantidad;  
} else {  
console.log ("La cantidad depositada debe ser mayor a cero.");  
}  
}

function retirar(cantidad) {  
if (cantidad > 0 && cantidad <= saldo) {  
saldo -= cantidad;  
} else {  
console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")  
}  
}

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

20     consultarSaldo: function () {  
21         return saldo;  
22     },  
23     realizarDeposito: function (cantidad) {  
24         depositar (cantidad);  
25     },  
26     realizarRetiro: function (cantidad) {  
27         retirar (cantidad);  
28     }  
29   }  
30 }  
31  
32 var miCuenta = crearCuentaBancaria (1000);  
33 console.log("Saldo inicial: " + miCuenta.consultarSa  
34 miCuenta.realizarDeposito(500);  
35 console.log("Saldo después del depósito:" + miCuenta  
36 miCuenta.realizarRetiro(200);  
37 console.log("Saldo después del retiro: " + miCuenta.  
38  
39 // continuación se presenta un ejemplo de como mane  
40

⇒ line that just executed  
→ next line to execute

<< First   < Prev   Next >   Last >>

Step 27 of 34

Sponsors: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame  
crearCuentaBancaria  
miCuenta  
this  
parent:saldo 1300  
parent:depositar  
parent:retirar  
Return value 1300

Objects

function crearCuentaBancaria(saldoInicial) {  
  let saldo = saldoInicial;  
  function depositar(cantidad) {  
    if (cantidad > 0) {  
      saldo += cantidad;  
    } else {  
      console.log ("La cantidad depositada debe ser mayor a cero.");  
    }  
  }  
  //Método privado para retirar dinero  
  function retirar (cantidad) {  
    if (cantidad > 0 && cantidad <= saldo) {  
      saldo -= cantidad;  
    } else {  
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
    }  
  }  
  //Retornamos un objeto con métodos públicos  
  return {  
    consultarSaldo: function () {  
      return saldo;  
    },  
    realizarDeposito: function (cantidad) {  
      depositar (cantidad);  
    },  
    realizarRetiro: function (cantidad) {  
      retirar (cantidad);  
    }  
  }  
}  
  
object  
consultarSaldo   function () {  
  return saldo;  
}  
realizarDeposito   function (cantidad) {  
  depositar (cantidad);  
}  
realizarRetiro   function (cantidad) {  
  retirar (cantidad);  
}  
  
function depositar(cantidad) {  
  if (cantidad > 0) {  
    saldo += cantidad;  
  } else {  
    console.log ("La cantidad depositada debe ser mayor a cero.");  
  }  
}  
  
function retirar(cantidad) {  
  if (cantidad > 0 && cantidad <= saldo) {  
    saldo -= cantidad;  
  } else {  
    console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
  }  
}

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

20     consultarSaldo: function () {  
21         return saldo;  
22     },  
23     realizarDeposito: function (cantidad) {  
24         depositar (cantidad);  
25     },  
26     realizarRetiro: function (cantidad) {  
27         retirar (cantidad);  
28     }  
29   }  
30 }  
31  
32 var miCuenta = crearCuentaBancaria (1000);  
33 console.log("Saldo inicial: " + miCuenta.consultarSa  
34 miCuenta.realizarDeposito(500);  
35 console.log("Saldo después del depósito:" + miCuenta  
36 miCuenta.realizarRetiro(200);  
37 console.log("Saldo después del retiro: " + miCuenta.  
38  
39 // continuación se presenta un ejemplo de como mane  
40

⇒ line that just executed  
→ next line to execute

<< First   < Prev   Next >   Last >>

Step 28 of 34

Sponsors: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500

Frames

Global frame  
crearCuentaBancaria  
miCuenta

Objects

function crearCuentaBancaria(saldoInicial) {  
  let saldo = saldoInicial;  
  function depositar(cantidad) {  
    if (cantidad > 0) {  
      saldo += cantidad;  
    } else {  
      console.log ("La cantidad depositada debe ser mayor a cero.");  
    }  
  }  
  //Método privado para retirar dinero  
  function retirar (cantidad) {  
    if (cantidad > 0 && cantidad <= saldo) {  
      saldo -= cantidad;  
    } else {  
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");  
    }  
  }  
  //Retornamos un objeto con métodos públicos  
  return {  
    consultarSaldo: function () {  
      return saldo;  
    },  
    realizarDeposito: function (cantidad) {  
      depositar (cantidad);  
    },  
    realizarRetiro: function (cantidad) {  
      retirar (cantidad);  
    }  
  }  
}  
  
object  
consultarSaldo   function () {  
  return saldo;  
}  
realizarDeposito   function (cantidad) {  
  depositar (cantidad);  
}  
realizarRetiro   function (cantidad) {  
  retirar (cantidad);  
}

Se intenta realizar un depósito de 100 a la cuenta utilizando el método `miCuenta.depositar(100)`.

- Si la operación se realiza correctamente, el código continúa su ejecución sin entrar al bloque `catch`.
- Si la operación falla por una excepción (por ejemplo, la cantidad a depositar no es válida), se genera una excepción y el flujo del código salta al bloque `catch`.

Dentro del bloque `catch`, se captura la excepción (`e`) y se imprime su mensaje (`e.message`) en la consola.

- El mensaje de error proporciona información sobre la causa de la excepción, lo que puede ser útil para depurar el código y detectar posibles errores en la lógica de las operaciones.

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot displays the Python Tutor interface for JavaScript (ES6). The code editor on the left shows a script for a bank account. The execution flow is indicated by a green arrow pointing to line 37 and a red arrow pointing to line 41. The console output on the right shows the initial balance, the result of a deposit, and the result of a withdrawal. The 'Frames' pane shows the global frame with variables `crearCuentaBancaria` and `miCuenta`. The 'Objects' pane shows the function objects for `consultarSaldo`, `realizarDeposito`, and `realizarRetiro`.

```
JavaScript (ES6)
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane
40 try {
41   miCuenta.depositar(100);
42 } catch (e) {
43   console.log(e.message);
44 }
45 try {
46   miCuenta.retirar(100);
47 } catch (e) {
48   console.log(e.message)
49 }
```

Print output (drag lower right corner to resize)

```
Saldo inicial: 1000
Saldo después del depósito:1500
Saldo después del retiro: 1300
```

Frames

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Método privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
    }
  }
  //Retornamos un objeto con métodos publicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}
```

object

property	value
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Step 29 of 34

Sponsor: Interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como mane
40 try {
41   miCuenta.depositar(100);
42 } catch (e) {
43   console.log(e.message);
44 }
45 try {
46   miCuenta.retirar(100);
47 } catch (e) {
48   console.log(e.message)
49 }
```

Edit this code

==> line that just executed

→ next line to execute

<< First

< Prev

Next >

Last >>

Step 30 of 34

TypeError: miCuenta.depositar is not a function

see [unsupported features](#) or click "Get AI Help" to debug

Sponsor: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500  
Saldo después del retiro: 1300

Frames

Global frame  
crearCuentaBancaria  
miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log ("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Metodo privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como mane
40 try {
41   miCuenta.depositar(100);
42 } catch (e) {
43   console.log(e.message);
44 }
45 try {
46   miCuenta.retirar(100);
47 } catch (e) {
48   console.log(e.message)
49 }
```

Edit this code

==> line that just executed

→ next line to execute

<< First

< Prev

Next >

Last >>

Step 31 of 34

Sponsor: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500  
Saldo después del retiro: 1300

Frames

Global frame  
crearCuentaBancaria  
miCuenta  
e

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log ("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Metodo privado para retirar dinero
  function retirar (cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar (cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar (cantidad);
    }
  }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[Known limitations](#)

```

31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se presenta un ejemplo de como mane
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
45 try {
46     miCuenta.retirar(100);
47 } catch (e) {
48     console.log(e.message)
49 }

```

Print output (drag lower right corner to resize)

```

Saldo inicial: 1000
Saldo después del depósito:1500
Saldo después del retiro: 1300
miCuenta.depositar is not a function

```

Frames

Global frame  
crearCuentaBancaria  
miCuenta

Objects

```

function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar(cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}

```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Time that just executed  
next line to execute

Step 32 of 34

Sponsors: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Se intenta realizar un retiro de 100 de la cuenta utilizando el método `miCuenta.retirar(100)`.

- Similar al caso anterior, si la operación se realiza correctamente, el código continúa sin entrar al bloque `catch`.
- Si la operación falla por una excepción (por ejemplo, saldo insuficiente), se genera una excepción y el flujo del código salta al bloque `catch`.

Dentro del bloque `catch`, se captura la excepción (`e`) y se imprime su mensaje (`e.message`) en la consola.

- Al igual que en el caso del depósito, el mensaje de error proporciona información sobre la causa de la excepción, permitiendo identificar el problema.



## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
45 try {
46     miCuenta.retirar(100);
47 } catch (e) {
48     console.log(e.message)
49 }
```

[Edit this code](#)

Step 33 of 34

**TypeError: miCuenta.retirar is not a function**  
see [unsupported features](#) or click "Get AI Help" to debug

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Print output (drag lower right corner to resize)

```
Saldo inicial: 1000
Saldo después del depósito:1500
Saldo después del retiro: 1300
miCuenta.depositar is not a function
```

Frames

Global frame

- crearCuentaBancaria
- miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Método privado para retirar dinero
  function retirar(cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar(cantidad);
    }
  }
}
```

object	
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar(cantidad); }
realizarRetiro	function (cantidad) { retirar(cantidad); }

## Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
31
32 var miCuenta = crearCuentaBancaria(1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
45 try {
46     miCuenta.retirar(100);
47 } catch (e) {
48     console.log(e.message)
49 }
```

[Edit this code](#)

Step 34 of 34

Sponsor: interested in a [free Python tip every week?](#)

[Get AI Help](#)

[Move and hide objects](#)

Print output (drag lower right corner to resize)

```
Saldo inicial: 1000
Saldo después del depósito:1500
Saldo después del retiro: 1300
miCuenta.depositar is not a function
```

Frames

Global frame

- crearCuentaBancaria
- miCuenta
- e

Objects

```
function crearCuentaBancaria(saldoInicial) {
  let saldo = saldoInicial;
  function depositar(cantidad) {
    if (cantidad > 0) {
      saldo += cantidad;
    } else {
      console.log("La cantidad depositada debe ser mayor a cero.");
    }
  }
  //Método privado para retirar dinero
  function retirar(cantidad) {
    if (cantidad > 0 && cantidad <= saldo) {
      saldo -= cantidad;
    } else {
      console.log("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.");
    }
  }
  //Retornamos un objeto con métodos públicos
  return {
    consultarSaldo: function () {
      return saldo;
    },
    realizarDeposito: function (cantidad) {
      depositar(cantidad);
    },
    realizarRetiro: function (cantidad) {
      retirar(cantidad);
    }
  }
}
```

object	
consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar(cantidad); }
realizarRetiro	function (cantidad) { retirar(cantidad); }

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

JavaScript (ES6)  
[known limitations](#)

```
31
32 var miCuenta = crearCuentaBancaria (1000);
33 console.log("Saldo inicial: " + miCuenta.consultarSa
34 miCuenta.realizarDeposito(500);
35 console.log("Saldo después del depósito:" + miCuenta
36 miCuenta.realizarRetiro(200);
37 console.log("Saldo después del retiro: " + miCuenta.
38
39 //A continuación se pesenta un ejemplo de como mane
40 try {
41     miCuenta.depositar(100);
42 } catch (e) {
43     console.log(e.message);
44 }
45 try {
46     miCuenta.retirar(100);
47 } catch (e) {
48     console.log(e.message)
49 }
```

Print output (drag lower right corner to resize)

Saldo inicial: 1000  
Saldo después del depósito:1500  
Saldo después del retiro: 1300  
miCuenta.depositar is not a function  
miCuenta.retirar is not a function

Frames

Global frame  
crearCuentaBancaria  
miCuenta

Objects

```
function crearCuentaBancaria(saldoInicial) {
    let saldo = saldoInicial;
    function depositar(cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
        } else {
            console.log ("La cantidad depositada debe ser mayor a cero.");
        }
    }
    //Método privado para retirar dinero
    function retirar (cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
        } else {
            console.log ("La cantidad a retirar debe ser mayor a cero y no exceder el saldo disponible.")
        }
    }
    //Retornamos un objeto con métodos públicos
    return {
        consultarSaldo: function () {
            return saldo;
        },
        realizarDeposito: function (cantidad) {
            depositar (cantidad);
        },
        realizarRetiro: function (cantidad) {
            retirar (cantidad);
        }
    }
}
```

object

consultarSaldo	function () { return saldo; }
realizarDeposito	function (cantidad) { depositar (cantidad); }
realizarRetiro	function (cantidad) { retirar (cantidad); }

Done running (34 steps)

Sponsors: interested in a [free Python tip every week?](#)

Get AI Help

[Move and hide objects](#)