

# Programación 2

**Tecnicatura en Desarrollo de Aplicaciones  
Informáticas**

# Parcialito 2

# ¿Qué imprime el siguiente código?

```
public class Persona {  
    private String nombre;  
    private String ciudad;  
  
    public Persona(String nombre, String ciudad) {  
        this.nombre = nombre;  
        this.ciudad = ciudad;  
    }  
  
    public String toString() {  
        return "Nombre: " + nombre + " Ciudad: " + ciudad;  
    }  
}
```

```
-----  
public class Cliente extends Persona {  
    private int nroCliente;  
  
    public Cliente(String nombre, String ciudad, int nroCliente) {  
        super(nombre, ciudad);  
        this.nroCliente = nroCliente;  
    }  
  
    public String toString(double deuda) {  
        return super.toString() + " Nro.: " + nroCliente + " Debe: "  
    }  
}
```

```
-----  
public class Main {  
    public static void main(String[] args) {  
        Cliente c1 = new Cliente("Juan", "Tandil", 123);  
        System.out.println(c1);  
    }  
}
```

☐ Nombre: Juan Ciudad: Tandil



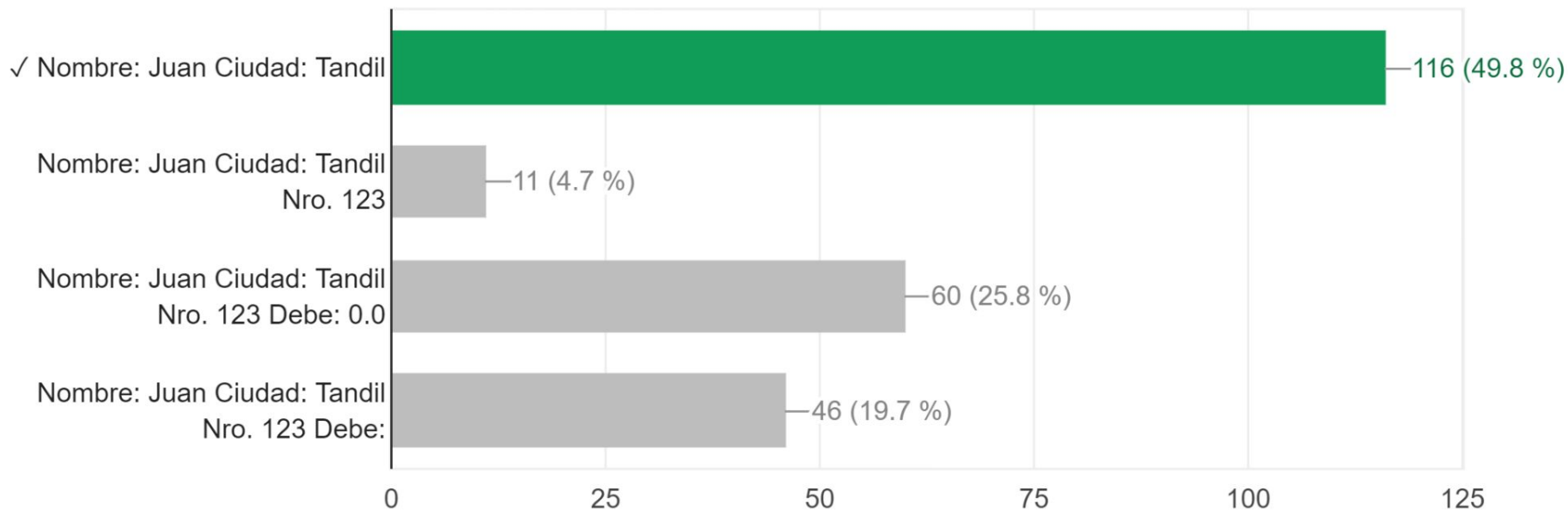
☐ Nombre: Juan Ciudad: Tandil Nro. 123

☐ Nombre: Juan Ciudad: Tandil Nro. 123 Debe: 0.0

☐ Nombre: Juan Ciudad: Tandil Nro. 123 Debe:

## ¿Qué imprime el siguiente código?


116/233 respuestas correctas



# Suponiendo que la clase Docente hereda de la clase Personal,

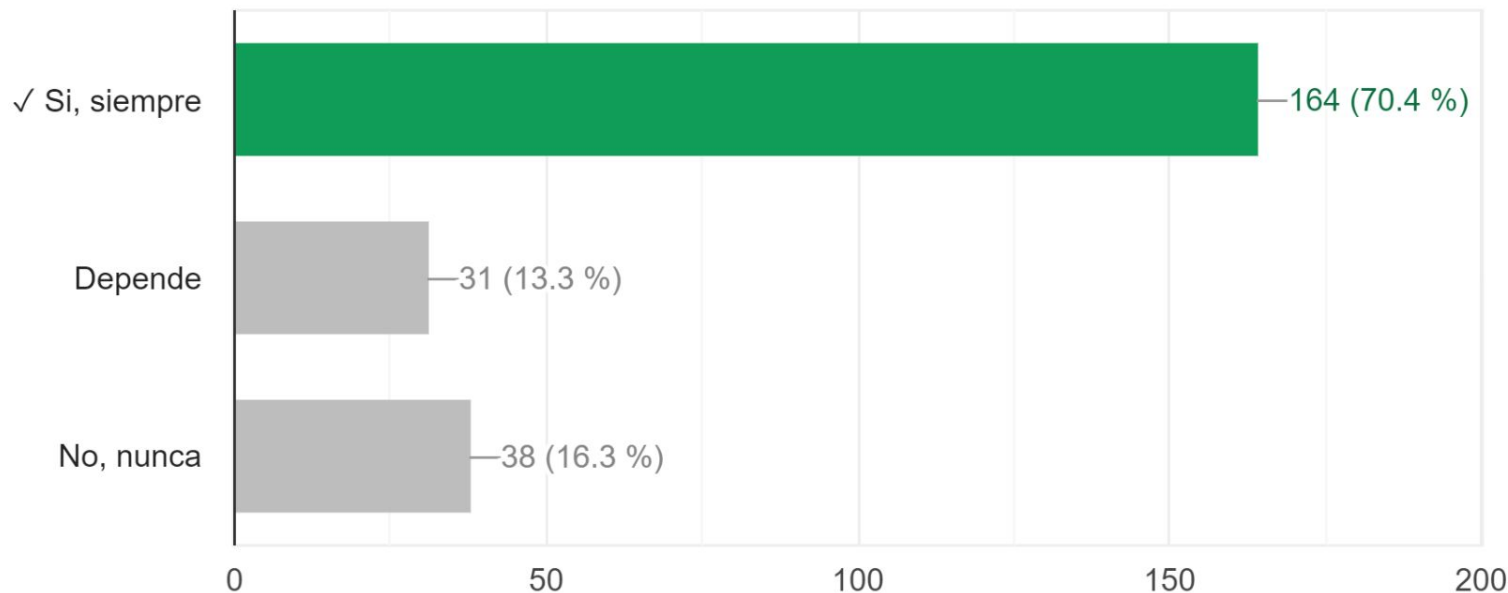
---

¿el compilador de Java permite que a una variable declarada del tipo Personal se le asigne una instancia de la clase Docente?

- Si, siempre 
- Depende
- No, nunca

Suponiendo que la clase Docente hereda de la clase Personal, ¿el compilador de Java permite que a una variable declarada del tipo Personal se le asigne una instancia de la clase Docente?

164/233 respuestas correctas



# Suponiendo que la clase Docente hereda de la clase Personal,

---

¿el compilador de Java permite que a una variable declarada de tipo Docente se le asigne una instancia de la clase Personal?

- Si, siempre
- Depende
- No, nunca

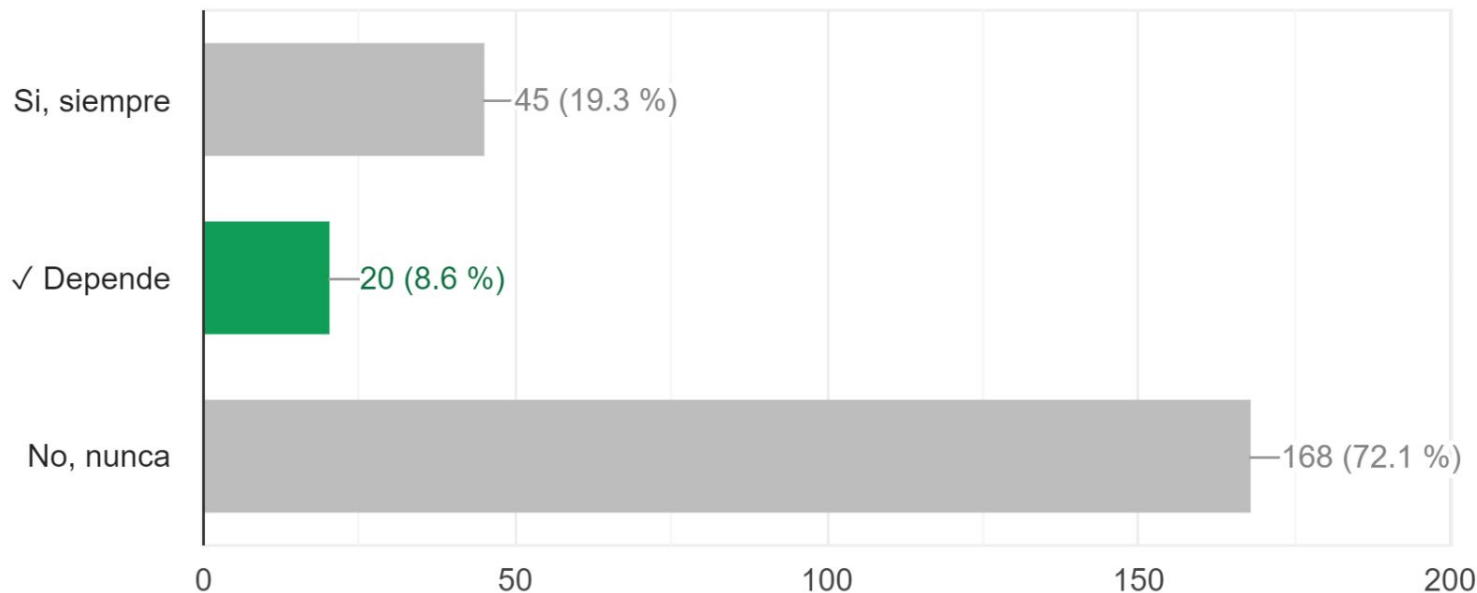


```
Docente d;  
Personal a;
```

```
d = a; //ERROR  
d = (Docente) a; //Permitido por  
el compilador
```

Suponiendo que la clase Docente hereda de la clase Personal, ¿el compilador de Java permite que a una variable declarada de tipo Docente se le asigne una instancia de la clase Personal?

20/233 respuestas correctas






# Considere las siguientes dos clases, A y B. ¿Cuál de las siguientes afirmaciones es correcta?

— — —

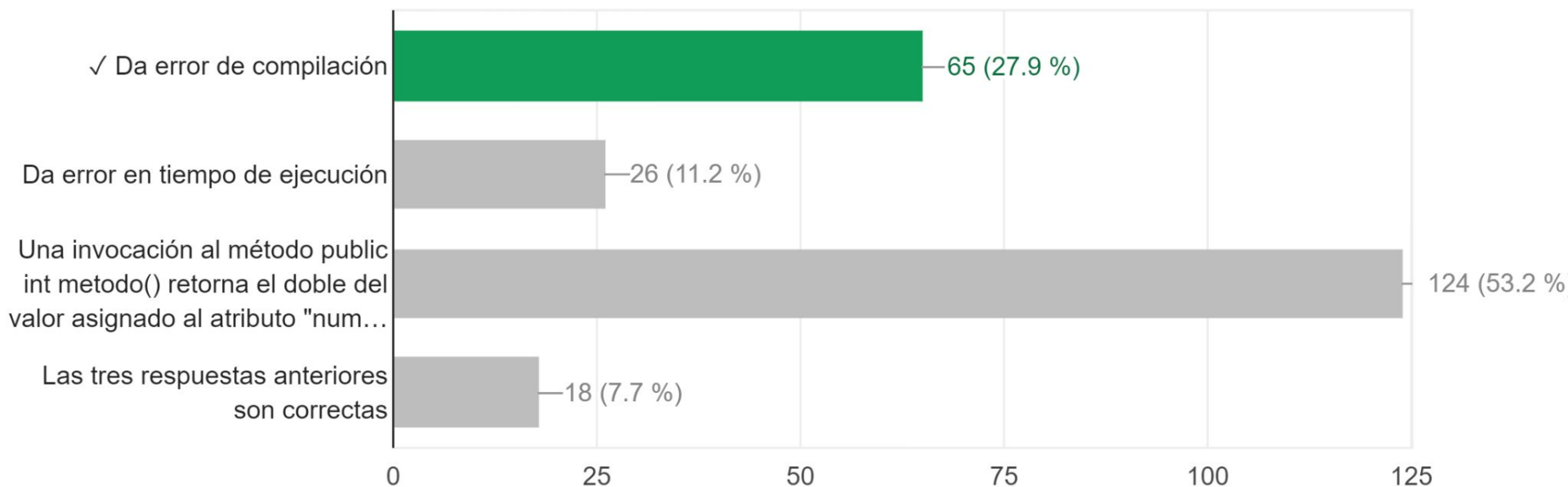
```
1 package test1;
2
3 public class A {
4     int numerito;
5
6     public int getNumerito() {
7         return numerito;
8     }
9
10    public void setNumerito(int numerito) {
11        this.numerito = numerito;
12    }
13
14    public A(int numerito) {
15        this.numerito = numerito;
16    }
17 }
18
```

```
1 package test2;
2
3 import test1.A;
4
5 public class B extends A {
6
7     public B(int numerito) {
8         super(numerito);
9     }
10
11    public int metodo(){
12        return this.numerito*2;
13    }
14 }
15
16
17
18
19
```

- Da error de compilación 
- Da error en tiempo de ejecución
- Una invocación al método public int metodo() retorna el doble del valor asignado al atributo "numerito"
- Las tres respuestas anteriores son correctas

Considere las siguientes dos clases, A y B. ¿Cuál de las siguientes afirmaciones es correcta?

65/233 respuestas correctas




```

public class MagicClass {
    private static int magicNumber = 23;
    ...
}

public static void x() {
    //Opción 1
    MagicClass temp = new MagicClass();
    temp.magicNumber = 66;

    //Opción 2
    MagicClass.magicNumber = 23;

    //Opción 3
    debería existir un método en la clase MagicClass
    public void setMagicNumber(int value){
        this.magicNumber = value;
    }
    y llamarlo con MagicClass.setMagicNumber(66);

    //Opción 4 
    debería existir un método en la clase MagicClass
    public static void setMagicNumber(int value){
        MagicClass.magicNumber = value;
    }
    y llamarlo con MagicClass.setMagicNumber(66);
}

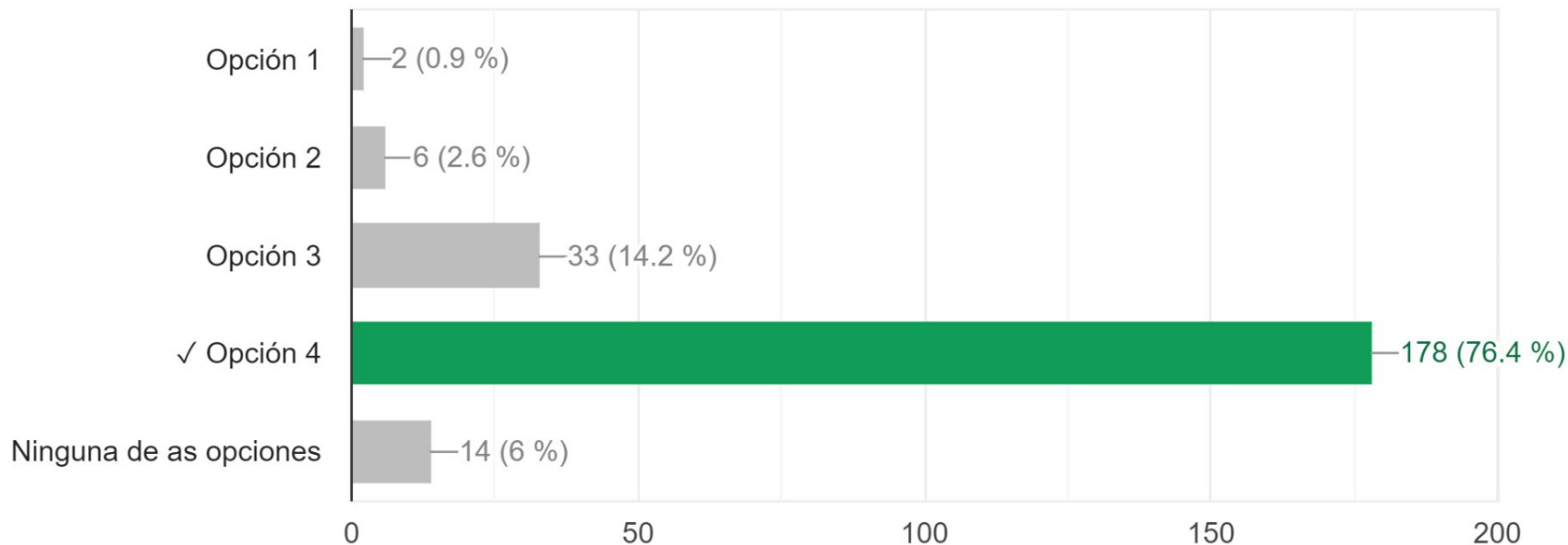
```

¿Cómo cambiaría el valor de un atributo llamado "magicNumber" declarado como "static" en una clase MagicClass como se muestra a continuación?

Elegir cuál de las opciones que se enuncian es la correcta

¿Cómo cambiaría el valor de un atributo llamado "magicNumber" declarado como "static" en una clase MagicClass como se muestra a continuación? ...l de las opciones que se enuncian es la correcta

178/233 respuestas correctas



¿Gracias a qué concepto visto (además de herencia) este código funciona para la Clase X y cualquier clase que herede de X?


— — —

```
public void imprimirNombre(X nn) {  
    System.out.println(nn.getNombre());  
}
```

Polimorfismo y  
binding dinámico!

# Si la clase "Legajo" NO implementa el método "equals", el siguiente código...

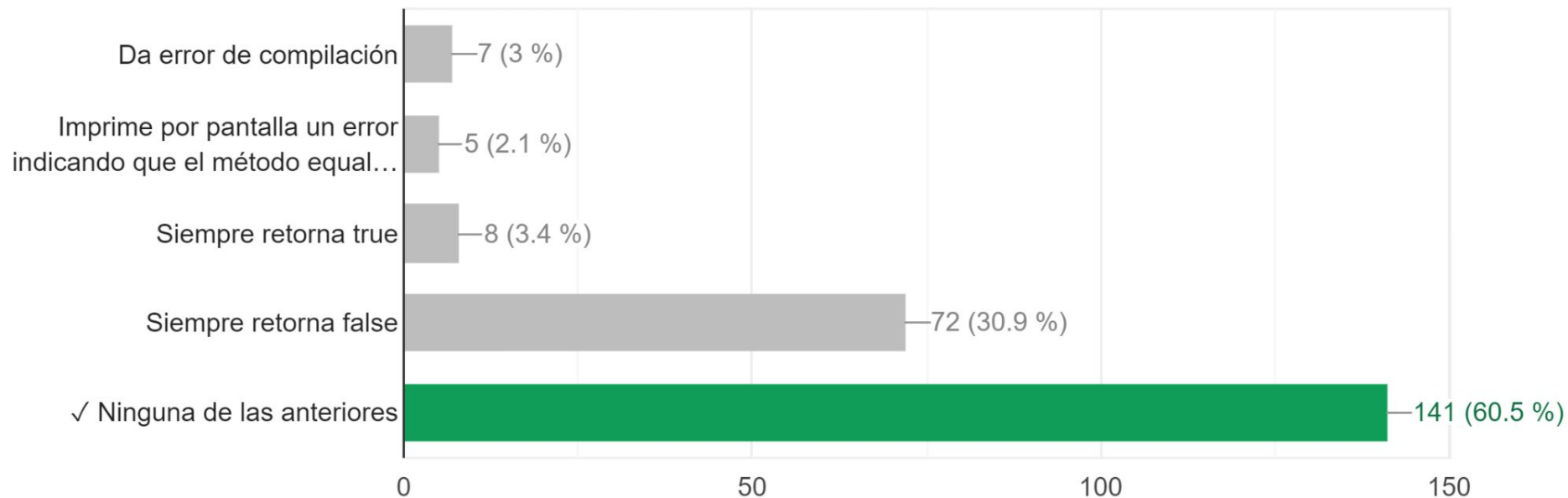
```
public boolean compararIgualdad(Legajo l1, Legajo l2){  
    return l1.equals(l2);  
}
```

- Da error de compilación.
- Imprime por pantalla un error indicando que el método equals no existe en la clase Legajo
- Siempre retorna true
- Siempre retorna false
- Ninguna de las anteriores 

```
Legajo pepe = new Legajo(...);  
compararIgualdad(pepe, pepe);  
// retorna true  
Legajo cacho = new Legajo(...);  
compararIgualdad(pepe, cacho);  
//retorna false;
```

Si la clase "Legajo" NO implementa el método "equals", el siguiente código...

141/233 respuestas correctas



En una Clase X, ¿cómo debo declarar un atributo si quiero que el mismo sea accesible solo por X y por cualquier clase que herede de X?

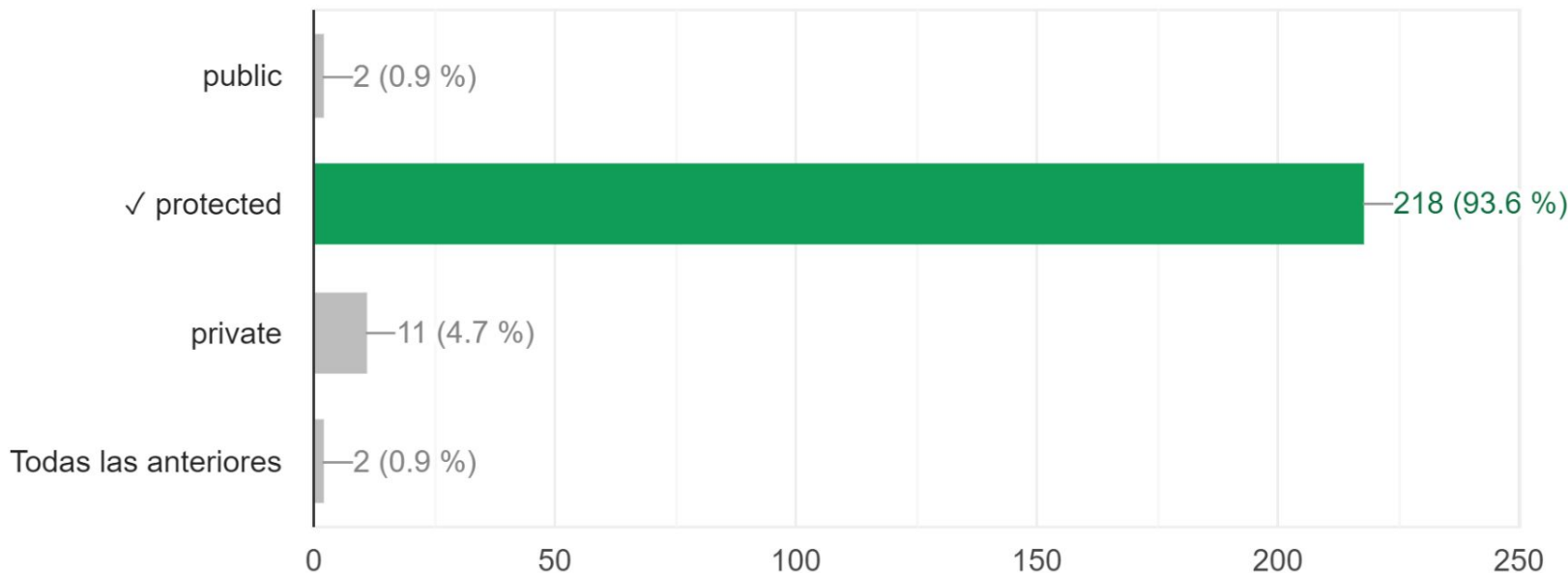
— — —

- public
- protected 
- private
- Todas las anteriores



En una Clase X, ¿cómo debo declarar un atributo si quiero que el mismo sea accesible solo por X y por cualquier clase que herede de X?

218/233 respuestas correctas



# Indicar qué imprime el siguiente código

```
public class Empleado {
    private int cantidadEmpleados;
    private String nombre;
    private int legajo;
    public Empleado(String nombre){
        this.nombre = nombre;
        legajo = cantidadEmpleados;
        cantidadEmpleados++;
    }
    public int getLegajo(){
        return legajo;
    }
    public int getNombre(){
        return legajo;
    }
    ....
    ....
    public static void main(String[] args){
        Empleado e1 = new Empleado("Marcelo");
        Empleado e2 = new Empleado("Luis");
        Empleado e3 = new Empleado("Ariel");
        System.out.println(e3.getLegajo());
    }
}
```

## - Imprime 0

El valor por defecto de la variable contadorEmpleados que es un int es 0. Como es un atributo de instancia siempre arranca en 0 en cada objeto.

# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }  
}
```

```
    public String x2(){  
        return "tenis";  
    }  
}
```

```
    public String toString() {  
        return this.x1()+" - "+  
               this.x2();  
    }  
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }  
}
```

```
    public String y1(){  
        return "voley";  
    }  
}
```

```
}
```

```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }  
}
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
               this.y1();  
    }  
}
```

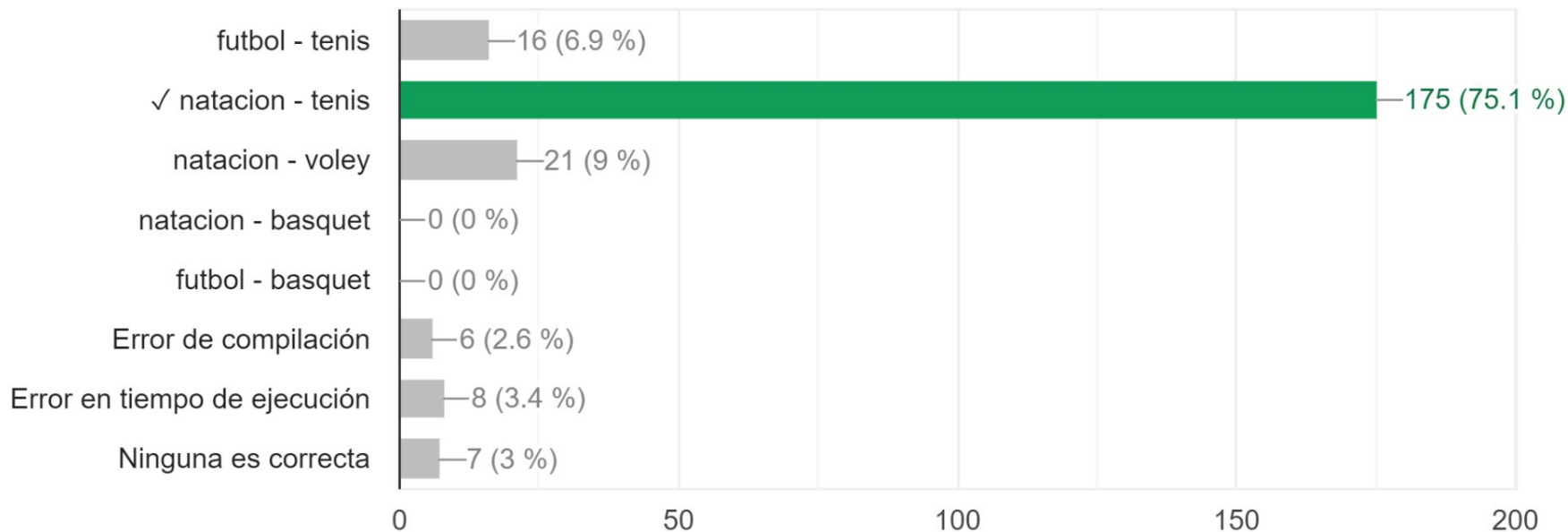
```
}
```

```
YY aux1 = new YY();  
System.out.println(aux1);
```

- futbol - tenis
- natacion - tenis ✓
- natacion - voley
- natacion - basquet
- Error de compilación
- Error en tiempo de ejecución
- Ninguna es correcta

## ¿Qué imprime el siguiente código?

175/233 respuestas correctas



# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }  
}
```

```
    public String x2(){  
        return "tenis";  
    }  
}
```

```
    public String toString() {  
        return this.x1()+" - "+  
               this.x2();  
    }  
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }  
}
```

```
    public String y1(){  
        return "voley";  
    }  
}
```

```
}
```

```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }  
}
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
               this.y1();  
    }  
}
```

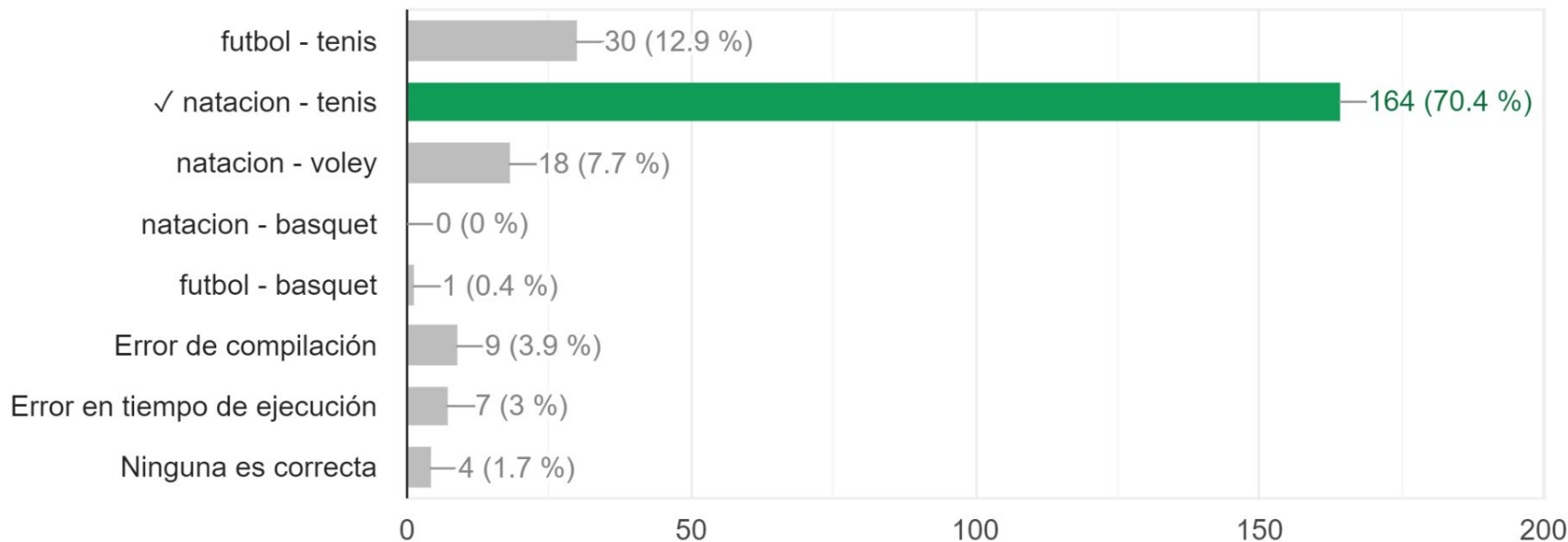
```
}
```

```
XX aux2 = new YY();  
System.out.println(aux2);
```

- futbol - tenis
- natacion - tenis ✓
- natacion - voley
- natacion - basquet
- Error de compilación
- Error en tiempo de ejecución
- Ninguna es correcta

## ¿Qué imprime el siguiente código?

164/233 respuestas correctas



# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }  
}
```

```
    public String x2(){  
        return "tenis";  
    }  
}
```

```
    public String toString() {  
        return this.x1()+" - "+  
               this.x2();  
    }  
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }  
}
```

```
    public String y1(){  
        return "voley";  
    }  
}
```

```
}
```

```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }  
}
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
               this.y1();  
    }  
}
```

```
}
```

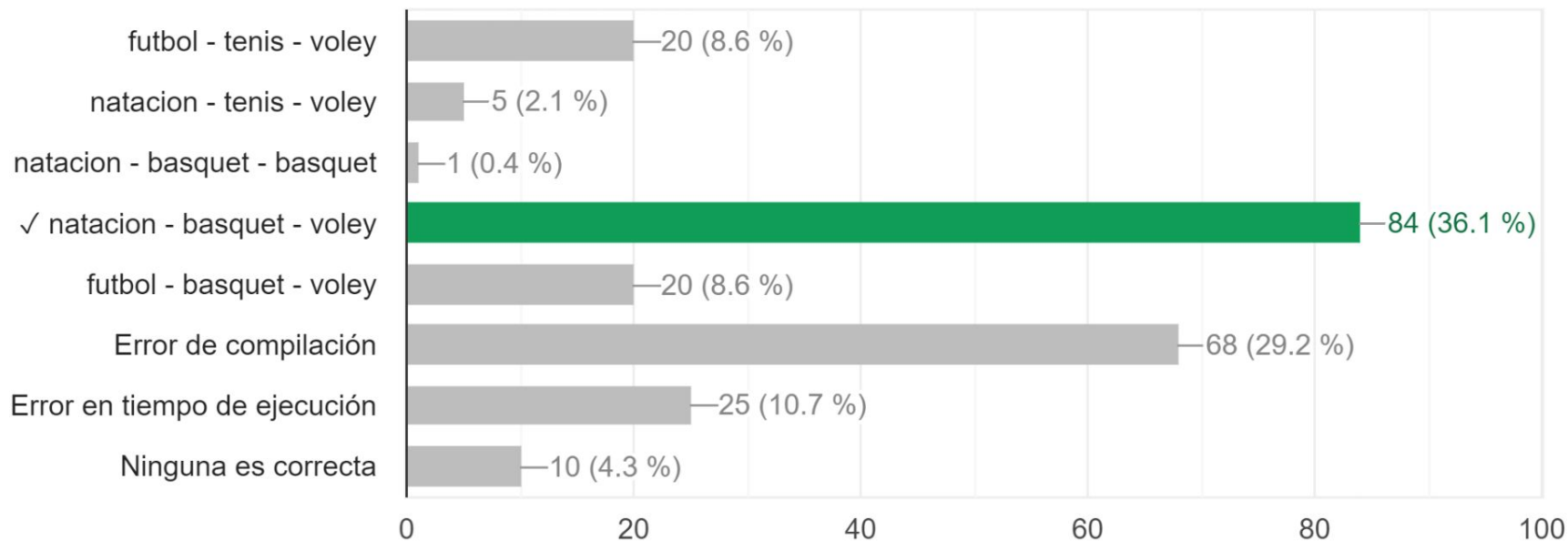
```
XX aux3 = new ZZ();  
System.out.println(aux3);
```

- futbol - tenis - voley
- natacion - tenis -voley
- natacion - basquet - basquet
- natacion - basquet - voley
- Error de compilación
- Error en tiempo de ejecución
- Ninguna es correcta



## ¿Qué imprime el siguiente código?

84/233 respuestas correctas





# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }  
}
```

```
    public String x2(){  
        return "tenis";  
    }  
}
```

```
    public String toString() {  
        return this.x1()+" - "+  
            this.x2();  
    }  
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }  
}
```

```
    public String y1(){  
        return "voley";  
    }  
}
```

```
}
```


```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }  
}
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
            this.y1();  
    }  
}
```

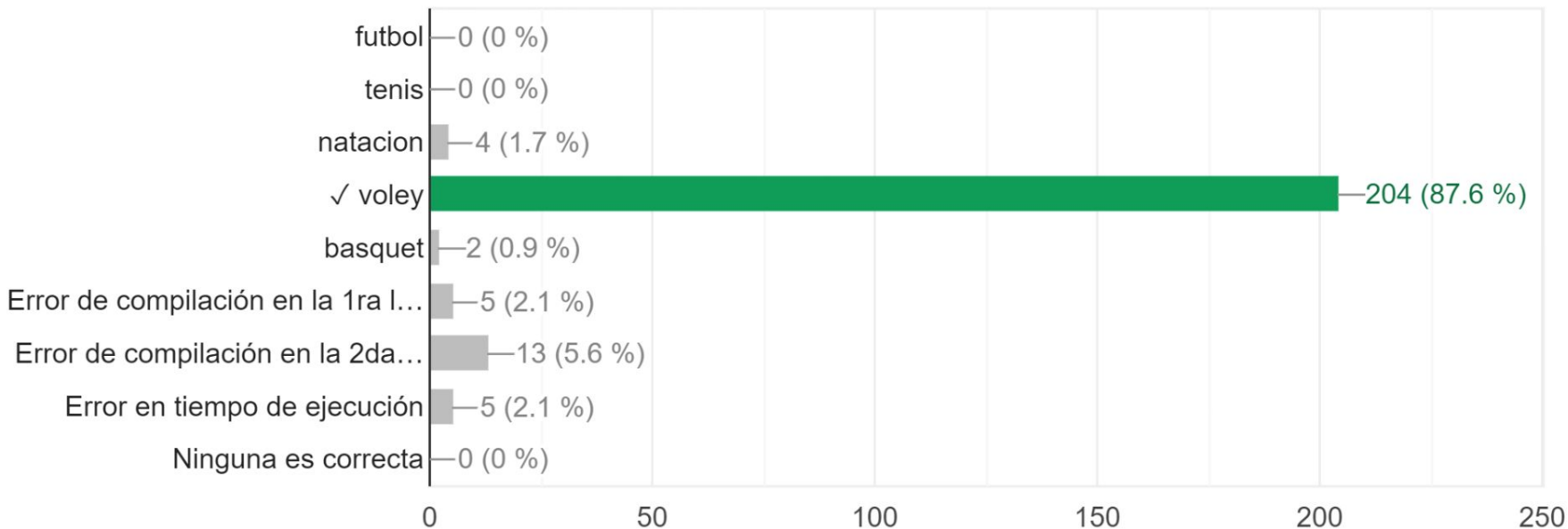
```
}
```

```
YY aux4 = new YY();  
System.out.println(aux4.y1());
```

- futbol
- tenis
- natacion
- voley 
- basquet
- Error compil. 1ra linea
- Error compil. 2da linea
- Error en tiempo de ejecución
- Ninguna es correcta

## ¿Qué imprime el siguiente código?

204/233 respuestas correctas



# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }
```

```
    public String x2(){  
        return "tenis";  
    }
```

```
    public String toString() {  
        return this.x1()+" - "+  
            this.x2();  
    }
```

```
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }
```

```
    public String y1(){  
        return "voley";  
    }
```

```
}
```

```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
            this.y1();  
    }
```

```
}
```

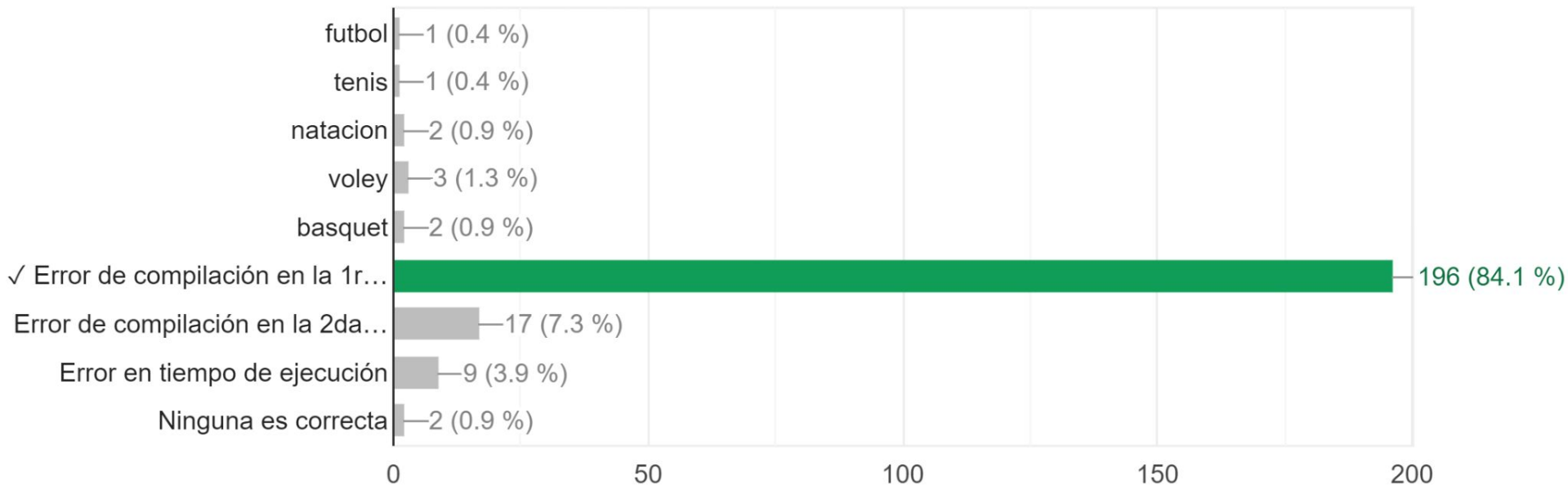
```
YY aux5 = new XX();  
System.out.println(aux5.y1());
```

- futbol
- tenis
- natacion
- voley
- basquet

- Error compil. 1ra linea ✓
- Error compil. 2da linea
- Error en tiempo de ejecución
- Ninguna es correcta

## ¿Qué imprime el siguiente código?

196/233 respuestas correctas



# ¿Qué imprime el siguiente código?

```
public class XX {
```

```
    public String x1(){  
        return "futbol";  
    }  
}
```

```
    public String x2(){  
        return "tenis";  
    }  
}
```

```
    public String toString() {  
        return this.x1()+" - "+  
               this.x2();  
    }  
}
```

```
public class YY extends XX{
```

```
    public String x1(){  
        return "natacion";  
    }  
}
```

```
    public String y1(){  
        return "voley";  
    }  
}
```

```
}
```

```
public class ZZ extends YY{
```

```
    public String x2(){  
        return "basquet";  
    }  
}
```

```
    @Override  
    public String toString() {  
        return super.toString()+" - "+  
               this.y1();  
    }  
}
```

```
}
```

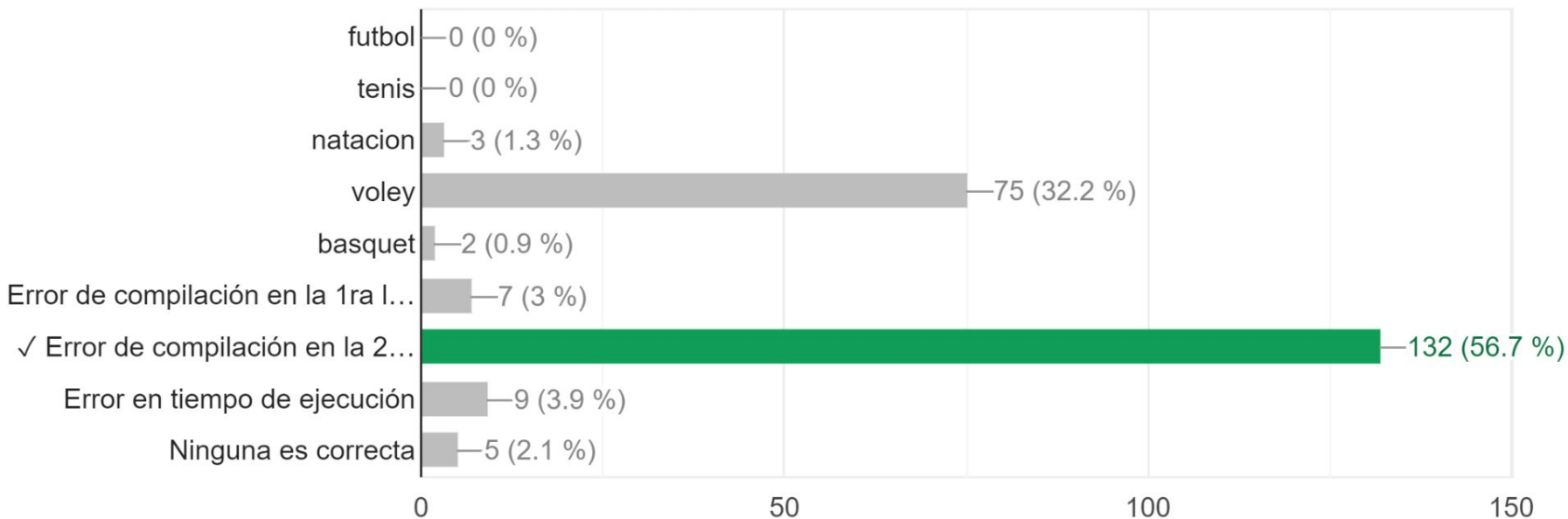
```
XX aux6 = new YY();  
System.out.println(aux6.y1());
```

- futbol
- tenis
- natacion
- voley
- basquet

- Error compil. 1ra linea
- Error compil. 2da linea ✓
- Error en tiempo de ejecución
- Ninguna es correcta

## ¿Qué imprime el siguiente código?

132/233 respuestas correctas



# Si invoco `super(...)` en el constructor de una clase,

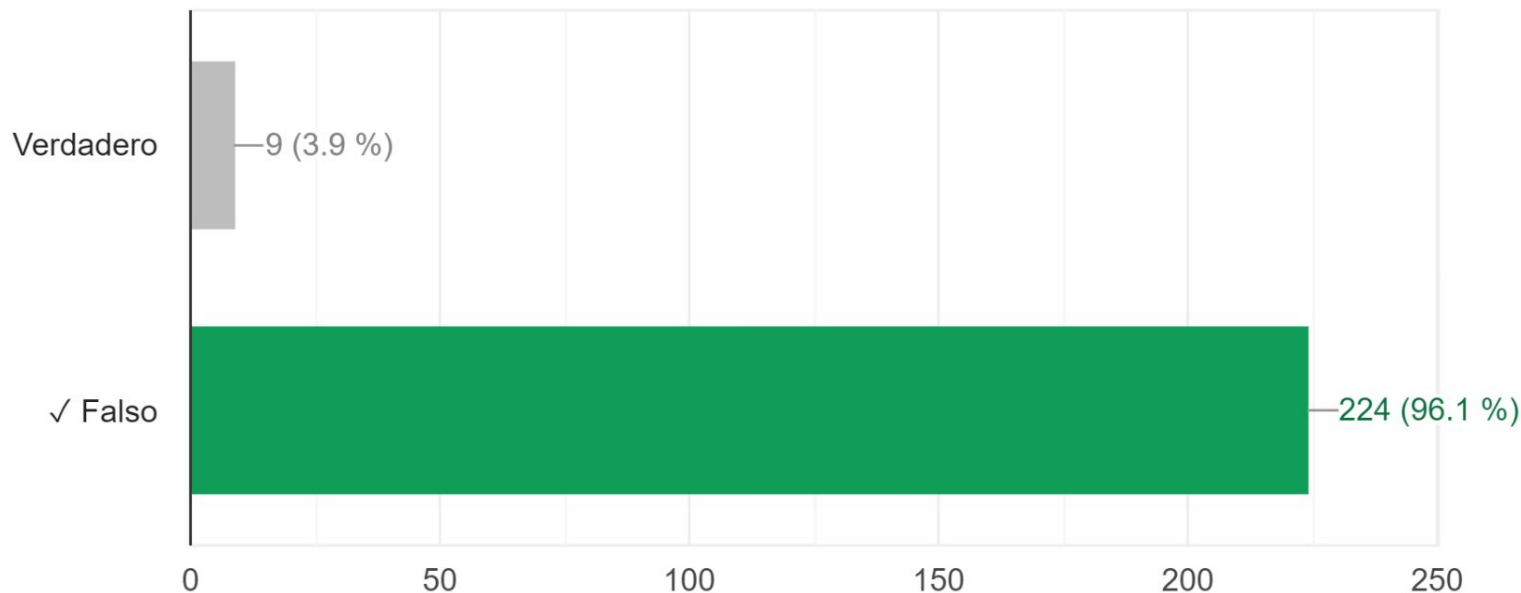
---

debo hacerlo indefectiblemente en la última línea del código del mismo

- Verdadero
- Falso 

Si invoco super(...) en el constructor de una clase, debo hacerlo indefectiblemente en la última línea del código del mismo

224/233 respuestas correctas





## Dado el siguiente código

```
public class Dado {  
    protected int maxCaras;  
    ...  
    public int tirar() {  
        return (int)(1 + Math.random() * this.maxCaras);  
    }  
    ...  
}
```

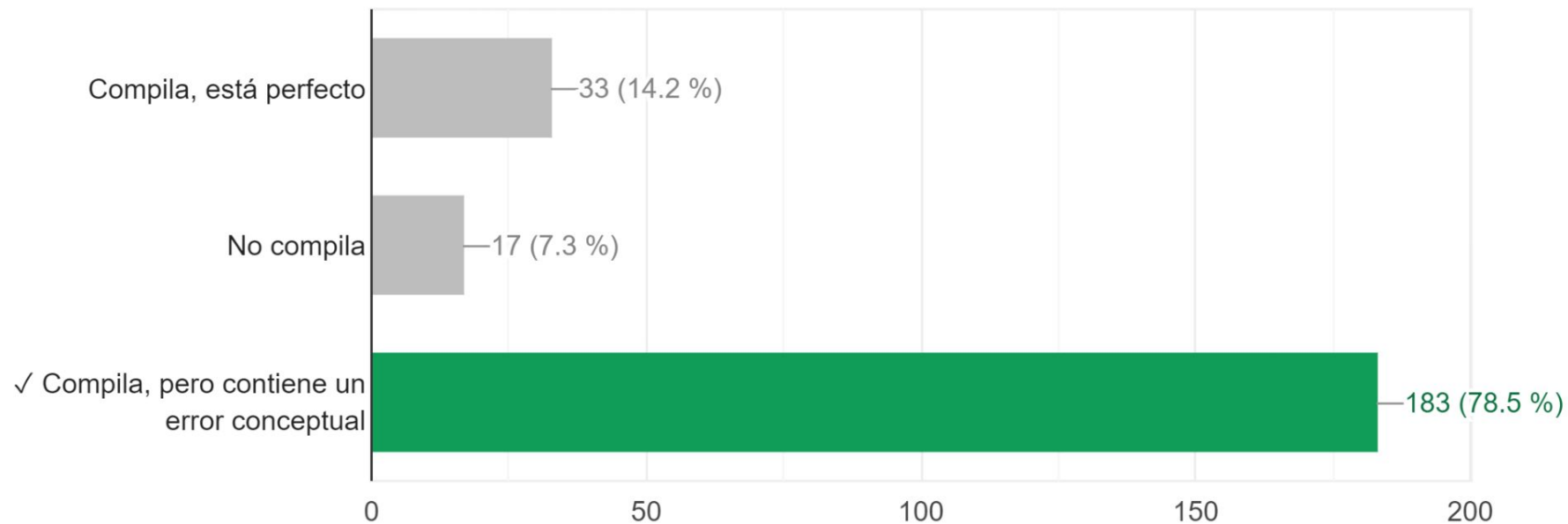
```
public class DadoCargado extends Dado {  
    protected double probCarga;  
    ...  
    public int calcularValor() {  
        if (Math.random() > probCarga)  
            return super.tirar();  
        else  
            return this.maxCaras;  
    }  
    ...  
}
```

- Compila, está perfecto
- No compila
- Compila, pero contiene un error conceptual



## Dado el siguiente código


183/233 respuestas correctas



# Suponiendo que la clase DadoCargado extiende la clase Dado,

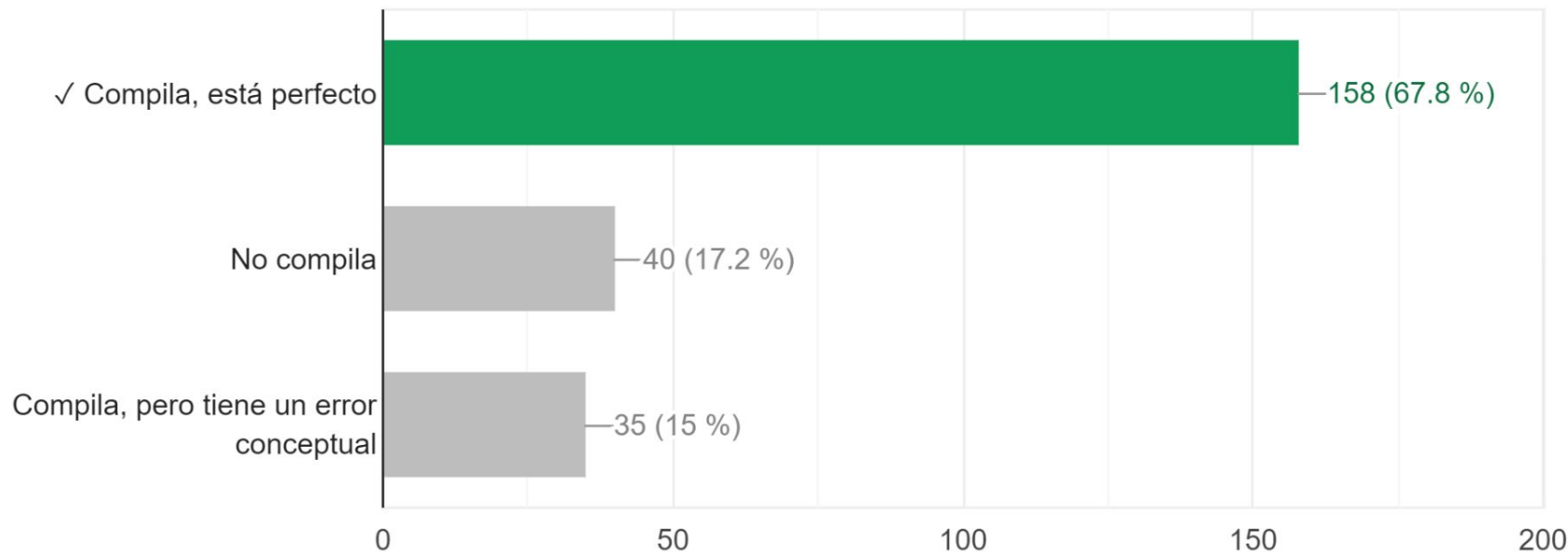
---

la siguiente asignación: `Dado cargado = new DadoCargado();`

- Compila, está perfecto 
- No compila
- Compila, pero tiene un error conceptual

Suponiendo que la clase DadoCargado extiende la clase Dado, la siguiente asignación: Dado cargado = new DadoCargado();


158/233 respuestas correctas



# Suponiendo que la clase DadoCargado extiende la clase Dado,

---

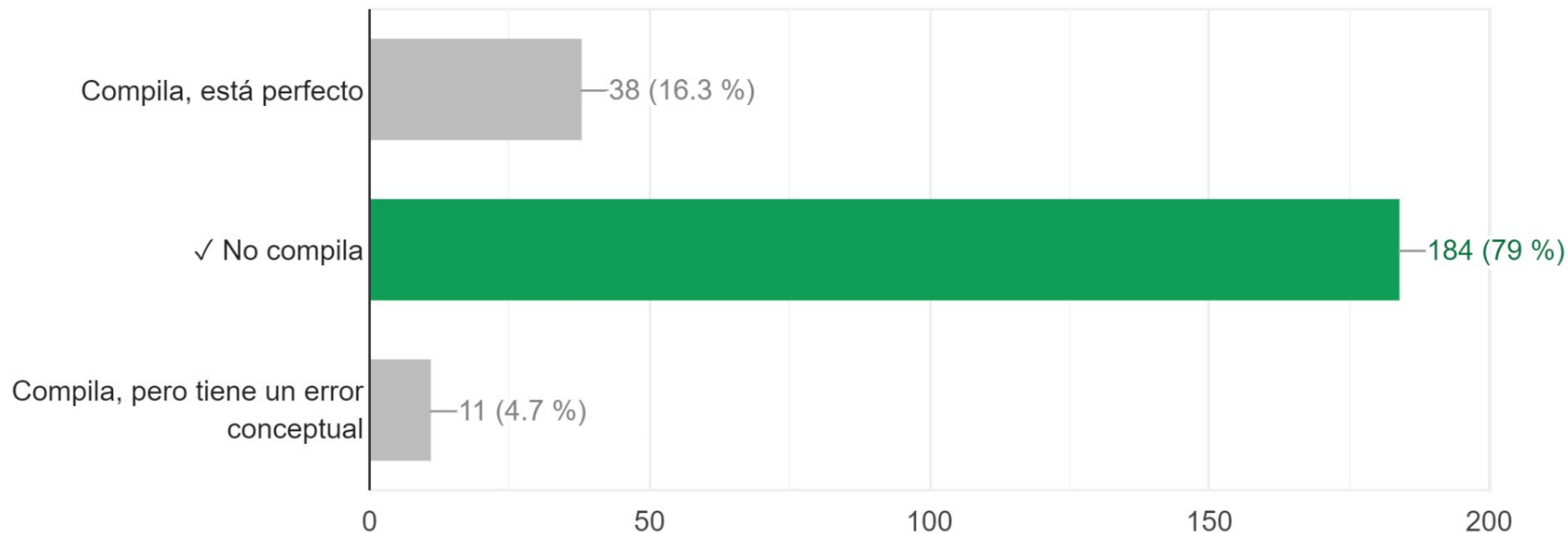
la siguiente asignación: `DadoCargado cargado = new Dado()`

- Compila, está perfecto
- No compila 
- Compila, pero tiene un error conceptual

Suponiendo que la clase DadoCargado extiende la clase Dado, la siguiente asignación:

`DadoCargado cargado = new Dado()`

184/233 respuestas correctas



## Qué imprime el siguiente código

- 6
- 12
- 24
- 36
- 48

No imprime nada

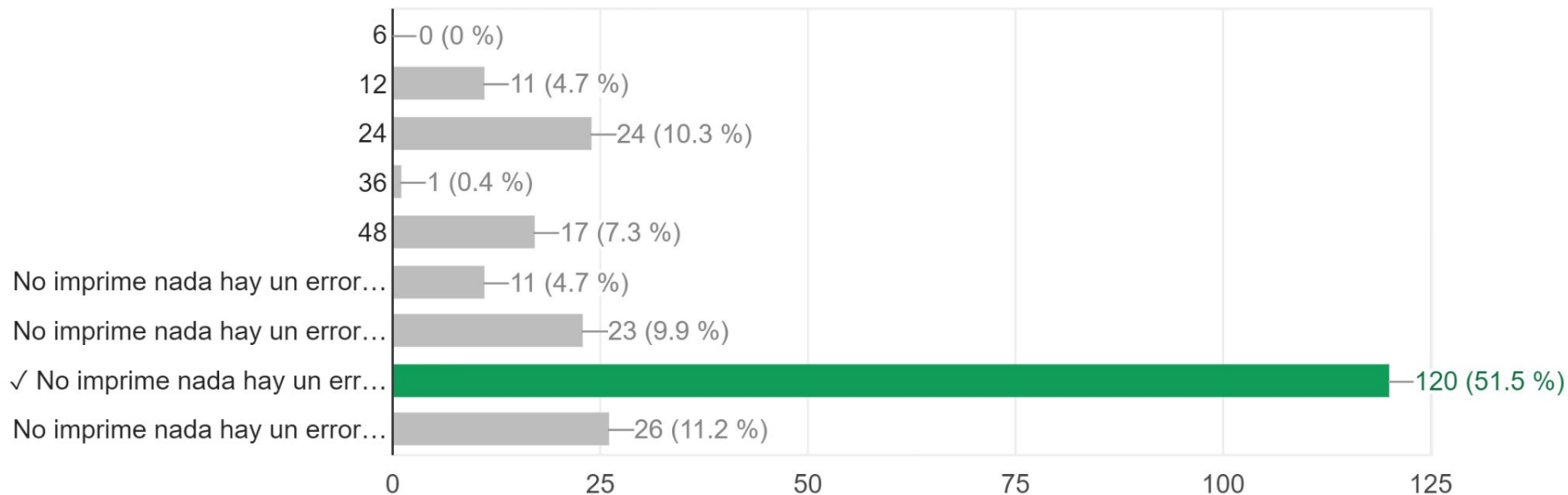
- hay un error en tiempo de ejecución.
- hay un error en tiempo de compilación en línea 2
- hay un error en tiempo de compilación en línea 11
- hay un error en tiempo de compilación en línea 21



```
1 public final class A{
2     int valor;
3     public A(int v){
4         valor = v;
5     }
6     public int getValor(){
7         return valor;
8     }
9 }
10
11 public class B extends A{
12     public B(int valor){
13         super(valor*2);
14     }
15     public void getValor(){
16         return super.getValor()*2;
17     }
18 }
19
20 public static void main(String[] args){
21     A b1 = (A) new B(12);
22     System.out.println(b1.getValor());
23 }
24
```

## Que imprime el siguiente código

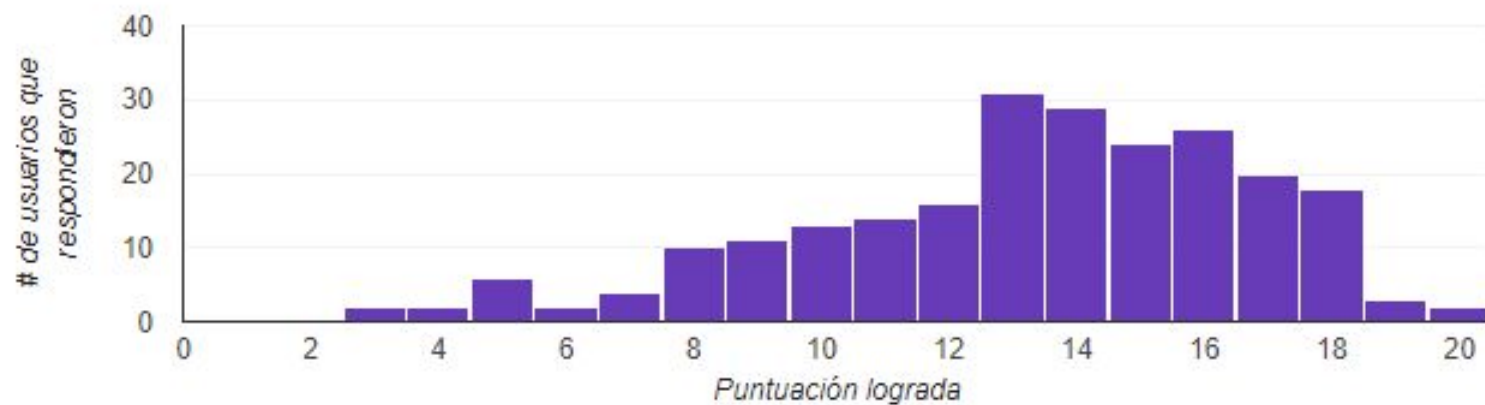
121/233 respuestas correctas





# Números Finales

Distribución de puntos totales



# Preguntas con respuestas incorrectas más frecuentes

## Preguntas con respuestas incorrectas más frecuentes

Pregunta	Respuestas correctas
¿Qué imprime el siguiente código? <code>#1 toString()</code>	116/233
Suponiendo que la clase Docente hereda de la clase Personal, ¿el compilador de Java permite que a una variable declarada de tipo Docente se le asigne una instancia de la clase Personal?	20/233
Considere las siguientes dos clases, A y B. ¿Cuál de las siguientes afirmaciones es correcta?	65/233
¿Qué imprime el siguiente código? <code>natacion - basquet - voley</code>	84/233