



Navegación por el cuestionario



Finalizar revisión

Comenzado el	lunes, 4 de marzo de 2019, 19:50
Estado	Finalizado
Finalizado en	miércoles, 6 de marzo de 2019, 12:40
Tiempo empleado	1 día 16 horas
Calificación	9,47 de 10,00 (95%)

Pregunta 1
Correcta
Puntúa 1,00 sobre 1,00
🚩 Marcar pregunta

¿Qué palabra reservada hay que utilizar en Java para referirse a la superclase de la clase actual?

Seleccione una:

- ☐ a. `superClass` .
- ☐ b. `this` .
- ☒ c. `super` . ✓
- ☐ d. `that` .

La respuesta correcta es: `super` .

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 2
Correcta
Puntúa 1,00 sobre 1,00
🚩 Marcar pregunta

En Java no está permitida la herencia múltiple de clases. ¿Verdadero o falso?

Seleccione una:

- ☒ a. Verdadero ✓
- ☐ b. Falso

La respuesta correcta es 'Verdadero'

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 3
Correcta
Puntúa 1,00 sobre 1,00
🚩 Marcar pregunta

En Programación Orientada a Objetos, ¿con qué nombre es conocido el mecanismo que permite crear clases basadas en otras existentes?

Seleccione una:

- ☐ a. Encapsulación.
- ☒ b. Herencia. ✓
- ☐ c. Derivación.
- ☐ d. Polimorfismo.

La respuesta correcta es: Herencia.

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 4
Correcta
Puntúa 1,00 sobre 1,00
🚩 Marcar pregunta

La composición consiste en la inclusión de objetos como atributos de una clase. ¿Verdadero o falso?

Seleccione una:

- ☒ a. Verdadero ✓
- ☐ b. Falso

La respuesta correcta es 'Verdadero'

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 5
Correcta
Puntúa 1,00 sobre 1,00
🚩 Marcar pregunta

Tenemos la clase `Vehiculo` :

```
public class Vehiculo {
    private String matricula;

    public Vehiculo(String matricula) {
        this.matricula = matricula;
    }

    public void metodo() {
        System.out.println("Método");
    }

    public String toString() {
        return "toString";
    }
}
```

Escribe el constructor de la subclase `Coche` para que el siguiente código funcione:

```
Coche unCoche = new Coche("4567ABC");
unCoche.metodo();
System.out.println(unCoche);
```

Respuesta: (penalty regime: 10, 20, ... %)

Reiniciar respuesta

```
1 public class Coche extends Vehiculo {
2
3     public Coche(String matricula){
4         super(matricula);
5     }
6
7 }
```

Test	Expected	Got	
✓ Coche unCoche = new Coche("4567ABC"); unCoche.metodo(); System.out.println(unCoche);	Metodo toString	Metodo toString	✓

Todas las pruebas superadas. ✓

Question author's solution:

```
public class Coche extends Vehiculo {  
    public Coche(String matricula) {  
        super(matricula);  
    }  
}
```

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 6

Correcta

Puntúa 0,67 sobre 1,00

🚩 Marcar pregunta

Tenemos las siguientes clases:

```
public class Vehiculo {  
    private String matricula;  
  
    public void metodo1() {  
        System.out.println("Método 1");  
    }  
  
    public void metodo2() {  
        System.out.println("Método 2");  
    }  
  
    public String toString() {  
        return "toString";  
    }  
}  
  
public class Coche extends Vehiculo {  
  
    public void metodo2() {  
        System.out.println("Coche 2");  
    }  
}
```

Si declaramos los siguientes objetos:

```
Vehiculo unVehiculo = new Vehiculo();  
Coche unCoche = new Coche();
```

¿Cuál será el resultado de las siguientes sentencias?

Sentencia	Resultado
unVehiculo.metodo1();	Método 1 ✓
unVehiculo.metodo2();	Método 2 ✓
System.out.println(unVehiculo);	toString ✓
unCoche.metodo1();	Método 1 ✓
unCoche.metodo2();	Coche 2 ✓
System.out.println(unCoche);	toString ✓

Correcta

Puntos para este envío: 1,00/1,00. Contando con los intentos anteriores, daría 0,67/1,00.

Pregunta 7

Correcta

Puntúa 1,00 sobre 1,00

🚩 Marcar pregunta

Indica las afirmaciones que describen la herencia en Java:

Seleccione una o más de una:

- ☐ a. Podemos aplicar la herencia múltiple
- ☒ b. Facilita la reutilización de código. ✓
- ☐ c. Solo permite añadir nuevo código, nunca modificar el código heredado
- ☒ d. Para crearla usamos la palabra reservada extends. ✓

Respuesta correcta

La respuesta correcta es: Facilita la reutilización de código., Para crearla usamos la palabra reservada extends.

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 8

Correcta

Puntúa 1,00 sobre 1,00

🚩 Marcar pregunta

Tenemos las siguientes clases:

```
public class Vehiculo {  
    private String matricula;  
  
    public void metodo1() {  
        System.out.print("Método 1");  
    }  
  
    public void metodo2() {  
        System.out.print("Método 2");  
    }  
  
    public String toString() {  
        return "toString";  
    }  
}  
  
public class Coche extends Vehiculo {  
  
    public void metodo1() {  
        super.metodo2();  
    }  
  
    public void metodo2() {  
        super.metodo1();  
        System.out.print("Coche 2");  
    }  
  
    public String toString() {  
        return super.toString() + super.toString();  
    }  
}
```

}

Si declaramos los siguientes objetos:

Vehículo unVehículo = new Vehículo();

Coche unCoche = new Coche();

¿Cuál será el resultado de las siguientes sentencias?

Sentencia	Resultado
unVehiculo.metodo1();	Método 1 ✓
unVehiculo.metodo2();	Método 2 ✓
System.out.println(unVehiculo);	toString ✓
unCoche.metodo1();	Método 2 ✓
unCoche.metodo2();	Método 1Coche 2 ✓
System.out.println(unCoche);	toStringtoString ✓

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 9

Correcta

Puntuía 0.90 sobre 1,00

🚩 Marcar pregunta

Partiendo de la última versión de la clase [Empleado](#) de nuestros vídeos, crea la subclase Chofer con las siguientes características:

- Cobra la mitad del salario más 50€ por cada viaje que hace. Usará un atributo viajes para llevar la cuenta.
- Al constructor además de la antigüedad se le pasará el número de viajes que ha realizado.
- Tiene el metodo adicional viajar. Recibirá el número de viajes que ha hecho, actualizará el atributo viajes y mostrará por consola el siguiente mensaje: "Mas viajes al bolsillo"

Recuerda interactuar correctamente con la superclase usando super.

Comprobaremos la subclase con el siguiente código:

System.out.println("Chofer: ");
Chofer raul = new Chofer(10, 50);
raul.viajar(100);
System.out.println(raul.getHoras() + " horas a la semana, ");
System.out.printf("%.2f euros, \n", raul.getSalario());
System.out.println(raul.getdiasVacaciones() + " dias de vacaciones, usando el formulario ");
System.out.println(raul.getFormularioVacaciones());

Respuesta: (penalty regime: 10, 20, ... %)

1-
2-
3-
4-
5-
6-
7-
8-
9-
10-
11-
12-
13-
14-
15-
16-
17-
18-
19-
20-
}

```
public class Chofer extends Empleado {  
    private int viajes;  
  
    public Chofer(int antigüedad, int viajes) {  
        super(antigüedad);  
        this.viajes = viajes;  
    }  
  
    @Override  
    public double getSalario() {  
        double salarioBase = super.getSalario();  
        salarioBase = salarioBase / 2 + (50 + viajes);  
        return salarioBase;  
    }  
  
    public void viajar(int viajes) {  
        this.viajes = viajes;  
        System.out.println( "Mas viajes al bolsillo");  
    }  
}
```

	Test	Expected	Got
✓	System.out.println("Chofer: "); Chofer raul = new Chofer(0, 0); raul.viajar(0); System.out.println(raul.getHoras() + " horas a la semana, "); System.out.printf("%.2f euros, \n", raul.getSalario()); System.out.println(raul.getdiasVacaciones() + " dias de vacaciones, usando el formulario "); System.out.println(raul.getFormularioVacaciones());	Chofer: Mas viajes al bolsillo 40 horas a la semana, 11500.00 euros, 20 dias de vacaciones, usando el formulario amarillo	Chofer: Mas v Mas v 40 hc 11500 20 d amari
✓	System.out.println("Chofer: "); Chofer raul = new Chofer(10, 50); raul.viajar(100); System.out.println(raul.getHoras() + " horas a la semana, "); System.out.printf("%.2f euros, \n", raul.getSalario()); System.out.println(raul.getdiasVacaciones() + " dias de vacaciones, usando el formulario "); System.out.println(raul.getFormularioVacaciones());	Chofer: Mas viajes al bolsillo 40 horas a la semana, 16500.00 euros, 30 dias de vacaciones, usando el formulario amarillo	Chofer: Mas v Mas v 40 hc 16500 30 d amari

◀

Todas las pruebas superadas. ✓

▶

Question author's solution:

```
// Subclase Chofer  
public class Chofer extends Empleado {  
    int viajes;  
    // Constructor: super debe ser la primera línea de código  
    public Chofer(int antigüedad, int viajes) {  
        super(antigüedad);  
        this.viajes = viajes;  
    }  
  
    @Override  
    public double getSalario() {  
        return super.getSalario() / 2 + 50 * viajes;  
    }  
  
    // Simula la tarea: viajar.  
    public void viajar(int viajes) {  
        this.viajes = viajes;  
        System.out.println("Mas viajes al bolsillo");  
    }  
}
```

Correcta

Puntos para este envío: 1,00/1,00. Contando con los intentos anteriores, daría 0,90/1,00.

Pregunta 10

Correcta

Puntuía 0.90 sobre 1,00

🚩 Marcar pregunta

Partiendo de la última versión de la clase [Empleado](#) de nuestros vídeos, crea la subclase Comercial con las siguientes características:

- Gana 10000€ más que el salario básico de los empleados
- Tiene el metodo adicional publicitar que mostrará por consola el siguiente mensaje: "Hay que venderse más!"
- Resto no cambia

Recuerda interactuar correctamente con la superclase usando super.

Comprobaremos la subclase con el siguiente código:

System.out.println("Comercial: ");

```

Comercial lidia = new Comercial(5);
System.out.println(lidia.getHoras() + " horas a la semana, ");
System.out.printf("%.2f euros, \n", lidia.getSalario());
System.out.print(lidia.getdiasVacaciones() + " dias de vacaciones, usando el formulario ");
System.out.println(lidia.getFormularioVacaciones());
lidia.publicitar();

```

Respuesta: (penalty regime: 10, 20, ... %)

```

1 public class Comercial extends Empleado {
2
3     public Comercial(int antiguedad){
4         super(antiguedad);
5     }
6
7     @Override
8     public double getSalario() {
9         double salarioBase = super.getSalario();
10        return salarioBase + 10000;
11    }
12
13    public void publicitar() {
14        System.out.println("Hay que venderse mas!");
15    }
16 }

```

Test	Expected
✓ System.out.println("Comercial: "); Comercial lidia = new Comercial(5); System.out.println(lidia.getHoras() + " horas a la semana, "); System.out.printf("%.2f euros, \n", lidia.getSalario()); System.out.print(lidia.getdiasVacaciones() + " dias de vacaciones, usando el formulario "); System.out.println(lidia.getFormularioVacaciones()); lidia.publicitar();	Comercial: 40 horas a la semana, 33000.00 euros, 25 dias de vacaciones, usando el formulario amarillo Hay que venderse mas!
✓ System.out.println("Comercial: "); Comercial lidia = new Comercial(0); System.out.println(lidia.getHoras() + " horas a la semana, "); System.out.printf("%.2f euros, \n", lidia.getSalario()); System.out.print(lidia.getdiasVacaciones() + " dias de vacaciones, usando el formulario "); System.out.println(lidia.getFormularioVacaciones()); lidia.publicitar();	Comercial: 40 horas a la semana, 33000.00 euros, 20 dias de vacaciones, usando el formulario amarillo Hay que venderse mas!

Todas las pruebas superadas. ✓

Question author's solution:

```

// Subclase Comercial
public class Comercial extends Empleado {
    // Constructor: super debe ser la primera linea de codigo
    public Comercial(int antiguedad) {
        super(antiguedad);
    }
    // Devuelve el salario de un Comercial
    @Override
    public double getSalario() {
        return super.getSalario() + 10000.0;
    }

    // Simula la tarea de los comerciales: publicitar.
    public void publicitar() {
        System.out.println("Hay que venderse mas!");
    }
}

```

Correcta

Puntos para este envío: 1,00/1,00. Contando con los intentos anteriores, daría 0,90/1,00.

Finalizar revisión