

[Área personal](#) / [Mis cursos](#) / [Progr2](#) / [Bienvenidos!](#) / [Examen Parcial_práctica](#)

Comenzado el	Friday, 14 de October de 2022, 14:19
Estado	Finalizado
Finalizado en	Friday, 14 de October de 2022, 15:32
Tiempo empleado	1 hora 12 minutos
Calificación	Sin calificar aún

Pregunta 1

Finalizado

Puntuá como 0,40

Dado el siguiente diagrama de clases:

Profesor
<<Atributos de instancia>> legajo: entero edad: entero dedicación: char <<Atributos de clase>> edadJubilatoria: entero
<<Constructor>> Profesor(l: entero, e: entero, ded: char) <<Comandos>> establecerLegajo(l: entero) establecerEdad(e: entero) establecerDedicacion(d: char) copy(p: Profesor) <<Consultas>> obtenerLegajo(): entero obtenerEdad(): entero obtenerDedicacion(): char obtenerEdadJubilatoria(): entero equals(p: Profesor): boolean clone(): Profesor

copy(p: Profesor)

Requiere **p** ligado. Modifica el estado interno del objeto que recibe el mensaje con los valores de los atributos de instancia del objeto ligado a **p**.

equals(p: Profesor): boolean

Requiere **p** ligado. Retorna verdadero si el estado interno es igual al estado interno del objeto ligado a **p**.

clone(): Profesor

Crea y retorna un nuevo objeto con el mismo estado interno que el objeto que recibe el mensaje

Implemente la clase Profesor con sus correspondientes atributos y servicios. Considere 65 la edad jubilatoria.

```
class Profesor():
```

```
    #Atributos de clase
```

```
    edadJubilatoria = 65
```

```
    #Atributos de instancia
```

```
    def __init__(self, leg, ed, ded):
```

```
        self.legajo = leg
```

```
        self.edad = ed
```

```
        self.dedicacion = ded
```

```
    #Comandos
```

```
    def establecerLegajo(self, leg):
```

```
        self.legajo = leg
```

```
    def establecerEdad(self, ed):
```

```
        self.edad = ed
```

```
    def establecerDedicacion(self, ded):
```

```
        self.dedicacion = ded
```

```
    def copy(self, prof):
```

```
        self.legajo = prof.obtenerLegajo()
```

```
        self.edad = prof.obtenerEdad()
```

```
        self.dedicacion = prof.obtenerDedicacion()
```

```
    #Consultas
```

```
    def obtenerLegajo(self):
```

```
        return self.legajo
```

```
    def obtenerEdad(self):
```

```
        return self.edad
```

```
    def obtenerDedicacion(self):
```

```
        return self.dedicacion
```

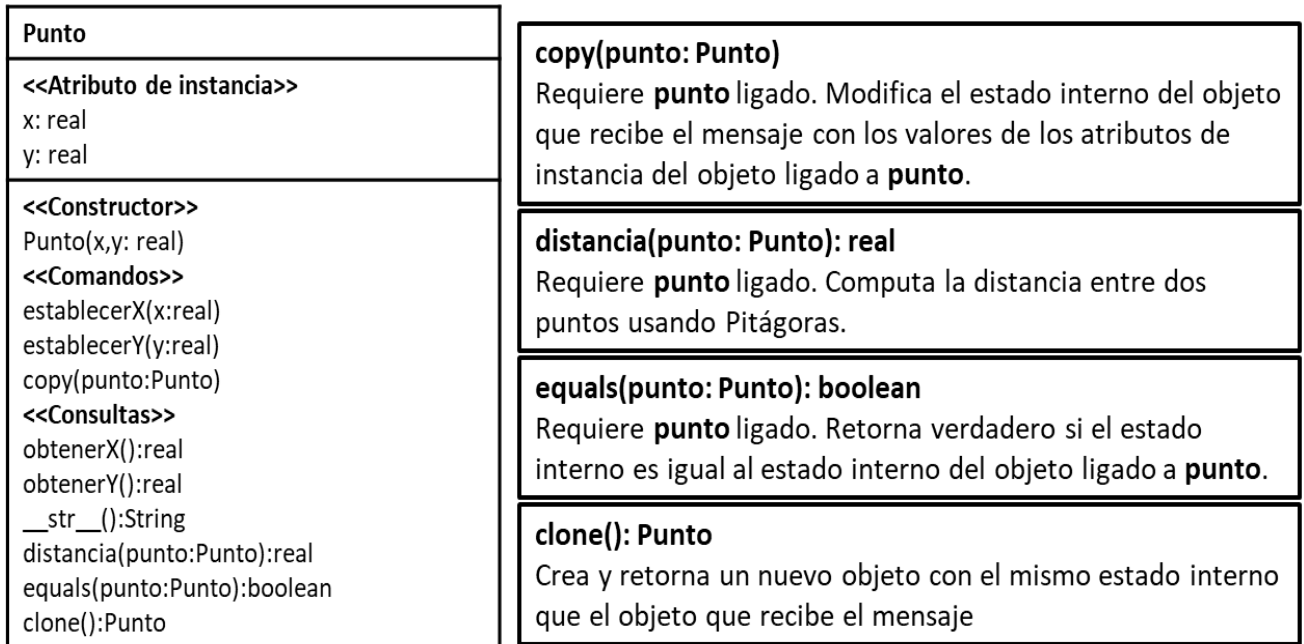
```
def obtenerEdadJubilatoria(self):  
    return self.edadJubilatoria  
  
def equals(self, prof):  
    return self.legajo == prof.obtenerLegajo() and self.edad == prof.obtenerEdad() and self.dedicacion ==  
prof.obtenerDedicacion()  
  
def clone(self):  
    profe = Profesor(self.legajo, self.edad, self.dedicacion)  
    return profe
```

Pregunta 2

Finalizado

Puntúa como 0,40

Dado el siguiente diagrama de clases:



Considere el siguiente programa:

```
p1 = Punto(2.0, 5.0)
p2 = Punto(3.0, 5.0)
p3 = Punto(2.0, 5.0)
p4 = p2.clone()
p5 = p4
```

Considerando que en la clase Punto el método **clone()** se encuentra implementado e **equals()** implementado en profundidad, muestre los valores que computan las siguientes expresiones

1. p1==p2
2. p1.equals(p2)
3. p2==p3
4. p3.equals(p1)
5. p2.equals(p1)
6. p1.equals(p4)
7. p4==p2
8. p5==p4
9. p3.equals(p5)
10. p4.equals(p3)

1. False
2. False
3. False
4. True
5. False
6. False
7. False
8. True
9. False
10. False

Pregunta **3**

Finalizado

Puntúa como 0,40

Dada la clase Robot:

```
class Robot():
    #Atributos de clase
    energiaMaxima=5000
    energiaMinima=100
    def __init__(self,nro,caja):
        # requiere caja ligada
        self.nroSerie = nro
        self.energia = self.energiaMaxima
        self.ruedas = caja.obtenerRuedas()
        self.opticas = caja.obtenerOpticas()
        self.chasis = caja.obtenerChasis()
        caja.vaciar()
    #Comandos
    def recargar(self):
        self.energia = self.energiaMaxima
    def armarAuto(self):
        """Requiere que se haya controlado si hay
        piezas disponibles"""
        self.ruedas -= 4
        self.opticas -= 6
        self.energia -= 70
        self.chasis -= 1
        #Controla si es necesario recargar energía
        if self.energia < self.energiaMinima:
            self.recargar()
    def abrirCaja(self,caja):
        """Aumenta sus cantidades según las de la caja
        y la vacía.
        Requiere caja ligada"""
        self.ruedas += caja.obtenerRuedas()
        self.opticas += caja.obtenerOpticas()
        self.chasis += caja.obtenerChasis()
        self.energia += 50
        caja.vaciar()
        #Controla si es necesario recargar energía
        if self.energia < self.energiaMinima:
            self.recargar()
    #Consultas
    def obtenerRuedas(self):
        return self.ruedas
    def obtenerOpticas(self):
        return self.opticas
    def obtenerChasis(self):
        return self.chasis
    def obtenerNroSerie(self):
        return self.nroSerie
    def obtenerEnergia(self):
        return self.energia
```

Se requiere que se implemente el método **cantAutos()** que retornará la cantidad de autos que puede armar el robot con las piezas que tiene disponibles. Armar un auto consume 70 unidades de energía, 4 ruedas, 6 ópticas y 1 chasis.

```
def cantAutos(self):
    ruedas = self.ruedas // 4
    opticas = self.opticas // 6
    chasis = self.chasis
    energia = self.energia // 70
```



```
autos = min(ruedas, opticas, chasis, energia)  
return autos
```

Pregunta 4

Finalizado

Puntuá como 0,40

Dada la clase Robot:

```
class Robot():
    #Atributos de clase
    energiaMaxima=5000
    energiaMinima=100
    def __init__(self,nro,caja):
        # requiere caja ligada
        self.nroSerie = nro
        self.energia = self.energiaMaxima
        self.ruedas = caja.obtenerRuedas()
        self.opticas = caja.obtenerOpticas()
        self.chasis = caja.obtenerChasis()
        caja.vaciar()
    #Comandos
    def recargar(self):
        self.energia = self.energiaMaxima
    def armarAuto(self):
        """Requiere que se haya controlado si hay
        piezas disponibles"""
        self.ruedas -= 4
        self.opticas -= 6
        self.energia -= 70
        self.chasis -= 1
        #Controla si es necesario recargar energía
        if self.energia < self.energiaMinima:
            self.recargar()
    def abrirCaja(self,caja):
        """Aumenta sus cantidades según las de la caja
        y la vacía.
        Requiere caja ligada"""
        self.ruedas += caja.obtenerRuedas()
        self.opticas += caja.obtenerOpticas()
        self.chasis += caja.obtenerChasis()
        self.energia -= 50
        caja.vaciar()
        #Controla si es necesario recargar energía
        if self.energia < self.energiaMinima:
            self.recargar()
    #Consultas
    def obtenerRuedas(self):
        return self.ruedas
    def obtenerOpticas(self):
        return self.opticas
    def obtenerChasis(self):
        return self.chasis
    def obtenerNroSerie(self):
        return self.nroSerie
    def obtenerEnergia(self):
        return self.energia
```

Se requiere que se implemente el método **clone()** que crea y retorna un robot con los mismos valores en cada uno de los atributos que el robot que recibió el mensaje.

```
def clone(self):
    robot = Robot(self.nroSerie, Caja(self.ruedas, self.opticas, self.chasis))
    return robot
```

Pregunta **5**

Correcta

Se puntúa 0,40 sobre 0,40

Elegir la opción que corresponda.

Composición modular

: favorece la integración de softwares para crear nuevos sistemas

Respuesta correcta

La respuesta correcta es:

Elegir la opción que corresponda.

[Composición modular]: favorece la integración de softwares para crear nuevos sistemas

Pregunta 6

Correcta

Se puntúa 0,40 sobre 0,40

Enlace los enunciados con los Factores que determinan la Calidad de Software.

Esfuerzo requerido para que partes de una aplicación sean utilizadas en otras aplicaciones.

Reusabilidad



Grado en que una aplicación o sistema puede ser transferido a otro hardware o sistema operativo.

Portabilidad



Cantidad de recursos que necesita una aplicación o sistema para realizar las operaciones con tiempos de respuesta óptimos.

Eficiencia



Grado con que puede controlarse el acceso al software y/o a los datos de un sistema a personal no autorizado.

Integridad



Esfuerzo requerido para aprender el manejo de una aplicación, trabajar con ella, introducir datos y conseguir resultados.

Facilidad de uso



Esfuerzo necesario para realizar modificaciones sobre una aplicación o sistema que ya se encuentra en funcionamiento.

Flexibilidad



Grado en que una aplicación o sistema cumple con la definición y lo encomendado por el cliente.

Corrección



Respuesta correcta

La respuesta correcta es:

Esfuerzo requerido para que partes de una aplicación sean utilizadas en otras aplicaciones. → Reusabilidad,

Grado en que una aplicación o sistema puede ser transferido a otro hardware o sistema operativo. → Portabilidad,

Cantidad de recursos que necesita una aplicación o sistema para realizar las operaciones con tiempos de respuesta óptimos. → Eficiencia,

Grado con que puede controlarse el acceso al software y/o a los datos de un sistema a personal no autorizado. → Integridad,

Esfuerzo requerido para aprender el manejo de una aplicación, trabajar con ella, introducir datos y conseguir resultados. → Facilidad de uso,

Esfuerzo necesario para realizar modificaciones sobre una aplicación o sistema que ya se encuentra en funcionamiento. → Flexibilidad,

Grado en que una aplicación o sistema cumple con la definición y lo encomendado por el cliente. → Corrección

Pregunta 7

Correcta

Se puntúa 0,40 sobre 0,40

Una los Criterios con los Factores de Calidad que estos favorecen

Protección Modular	Robustez	✓
Entendimiento Modular	Mantenimiento	✓
Composición Modular	Reusabilidad	✓
Continuidad Modular	Extensibilidad	✓

Respuesta correcta

La respuesta correcta es:

Protección Modular → Robustez,

Entendimiento Modular → Mantenimiento,

Composición Modular → Reusabilidad,

Continuidad Modular → Extensibilidad

Pregunta 8

Correcta

Se puntúa 0,40 sobre 0,40

Una los términos con sus respectivas definiciones.

Formada por la signatura de los servicios públicos.	Interfaz	✓
Servicio que modifica el valor de al menos un atributo.	Comando	✓
Mantienen valores compartidos por todos los objetos de una clase.	Atributos de clase	✓
Propiedad o cualidad relevante que caracteriza a todos los objetos de una clase.	Atributo	✓
Representa la abstracción del conjunto de objetos o instancias.	Nombre	✓
Dependiendo del lenguaje, se usa para crear un objeto o inicializar la instancia de una clase.	Constructor	✓

Respuesta correcta

La respuesta correcta es:

Formada por la signatura de los servicios públicos. → Interfaz,

Servicio que modifica el valor de al menos un atributo. → Comando,

Mantienen valores compartidos por todos los objetos de una clase. → Atributos de clase,

Propiedad o cualidad relevante que caracteriza a todos los objetos de una clase. → Atributo,

Representa la abstracción del conjunto de objetos o instancias. → Nombre, Dependiendo del lenguaje, se usa para crear un objeto o inicializar la instancia de una clase. → Constructor

Pregunta 9

Correcta

Se puntúa 0,40 sobre 0,40

Enlace los 5 principios de construcción de software con sus respectivas definiciones

No debe revelarse la forma en los servicios ofrecidos por un módulo son implementados.

Acceso Uniforme



Propiedades que describen a un módulo capaz de ser extendido por el usuario o utilizado por otro módulo.

Abierto-Cerrado



Un único módulo debe conocer una lista exhaustiva de alternativas a utilizarse por el programa.

Cambio Simple



La documentación sobre un módulo debe estar contenida en su propio código.

Auto-documentación



Los módulos deben corresponderse con las unidades sintácticas que ofrece el lenguaje utilizado.

Unidad Modular Lingüística



Respuesta correcta

La respuesta correcta es:

No debe revelarse la forma en los servicios ofrecidos por un módulo son implementados. → Acceso Uniforme,

Propiedades que describen a un módulo capaz de ser extendido por el usuario o utilizado por otro módulo. → Abierto-Cerrado,

Un único módulo debe conocer una lista exhaustiva de alternativas a utilizarse por el programa. → Cambio Simple,

La documentación sobre un módulo debe estar contenida en su propio código. → Auto-documentación,

Los módulos deben corresponderse con las unidades sintácticas que ofrece el lenguaje utilizado. → Unidad Modular Lingüística

Pregunta 10

Correcta

Se puntúa 0,40 sobre 0,40

Hoy te regalan un centavo. Mañana, recibirás el doble (2 centavos). Al próximo día, volverás a recibir el doble de ello (4 centavos). Finalmente, una vez que hayas recibido un millón o más, no recibirás más regalos. El siguiente código calcula el día y monto que se recibe al romper la barrera del millón.

```
dinero, regalo, dia = 0, 0.01, 1
while True:
    dinero += regalo
    if dinero >= 1000000:
        break
    regalo *= 2
    dia += 1
```

¿Cuál es la salida esperada?

En el día

27

✓ , tu regalo de \$671,088.

64

✓ te dejará un total de \$1,

342

✓ ,177.27.

Pregunta 11

Correcta

Se puntúa 0,40 sobre 0,40

Dadas las clases PresionArterial y SignosVitales de la semana 9, evalúe la ejecución del siguiente programa

```
presion1 = PresionArterial(90,185)
presion2 = presion1
sv1 = SignosVitales(36.5,presion1)
sv2 = SignosVitales(36.5,presion2)
```

Los objetos referenciados por sv1 y sv2 están asociados con:

- ☒ a. El mismo objeto de tipo PresionArterial. ✓
- ☐ b. Objetos equivalentes de tipo PresionArterial.
- ☐ c. Objetos distintos de tipo PresionArterial.

Respuesta correcta

La respuesta correcta es:

El mismo objeto de tipo PresionArterial.

Pregunta 12

Correcta

Se puntúa 0,40 sobre 0,40

En un esquema de clases proveedoras y clases clientes, cada clase:

- ☐ a. No necesita conocer los servicios que brindan sus clases proveedoras, pero si conocer quienes son sus clientes.
- ☒ b. Debe conocer los servicios que brindan sus clases proveedoras, pero no necesita conocer quienes son sus clientes. ✓
- ☒ c. Debe conocer los servicios que brindan sus clases proveedoras pero no cómo estos están implementados. ✓
- ☐ d. No necesita conocer los servicios que brindan sus clases proveedoras ni quienes son sus clientes.

Respuesta correcta

Las respuestas correctas son:

Debe conocer los servicios que brindan sus clases proveedoras, pero no necesita conocer quienes son sus clientes.,

Debe conocer los servicios que brindan sus clases proveedoras pero no cómo estos están implementados.

Pregunta 13

Correcta

Se puntúa 0,40 sobre 0,40

Elija, entre las siguientes combinaciones, la opción que crea correcta.

- ☐ a.
- Correctitud & Extensibilidad - Modularidad
 - Robustez & Reusabilidad - Confiabilidad
- ☐ b.
- Correctitud & Robustez - Modularidad
 - Extensibilidad & Reusabilidad - Confiabilidad
- ☒ c. 
- Correctitud & Robustez - Confiabilidad
 - Extensibilidad & Reusabilidad - Modularidad
- ☐ d.
- Reusabilidad & Robustez - Confiabilidad
 - Extensibilidad & Correctitud - Modularidad

Respuesta correcta

La Correctitud y la Robustez aportan a la Confiabilidad de una aplicación o sistema. Es decir, que mientras ellos se cumplan, este hará lo que se requiere de forma fiable. Por otro lado, la extensibilidad y reusabilidad ayudan a modularizar el software logrando un código organizado y limpio.

La respuesta correcta es:




- Correctitud & Robustez - Confiabilidad
- Extensibilidad & Reusabilidad - Modularidad

Pregunta 14

Correcta

Se puntúa 0,40 sobre 0,40

Seleccione las sentencias que cree son correctas

- ☒ a. El Entendimiento Modular se relaciona a la facilidad para comprender el comportamiento de un módulo con solo leer su código.  Un método favorece el Entendimiento Modular si facilita que quien lea un módulo pueda comprenderlo sin necesidad de acudir a otros módulos.
- ☒ b. En un programa donde se satisface el criterio de Protección Modular, los cambios que impactan un módulo NO se propagan al resto.  La Protección Modular especifica que las excepciones en tiempo de ejecución detectadas en un módulo, es decir, mientras el programa está corriendo, son contenidas y NO se propagan al resto de los módulos.
- ☐ c. La Descomposición Modular sigue la idea de poder descomponer un programa en módulos menos complejos, rigurosamente dependientes unos de otros.
- ☐ d. El criterio de Continuidad Modular especifica que los cambios que impactan un módulo se propaguen al resto.
- ☒ e. La Composición Modular trata de la creación de nuevas unidades de software creadas a partir de la combinación de otras ya existentes.  Se satisface el criterio de Composición Modular si se favorece la producción de elementos de software que pueden ser combinados para crear nuevos sistemas, posiblemente en un entorno diferente a aquel en el que se idearon.

Respuesta correcta

Las respuestas correctas son:

El Entendimiento Modular se relaciona a la facilidad para comprender el comportamiento de un módulo con solo leer su código.,

La Composición Modular trata de la creación de nuevas unidades de software creadas a partir de la combinación de otras ya existentes.

Pregunta 15

Correcta

Se puntúa 0,40 sobre 0,40

Dada la clase:

```
84 class NaveEspacial:
85
86     # Atributos de clase
87     max_deposito = 1000
88     parsec = 100
89
90     # Método de inicialización
91     def __init__(self, co, comb):
92         self.estado_alertas = False
93
94         # Atributos de instancia
95         self.color=co
96         if (comb > self.max_deposito):
97             self.combustible = self.max_deposito
98         else:
99             self.combustible = comb
100
101     def establecerEstadoAlertas(self, habilitar):
102         self.estado_alertas = habilitar
103
104     def obtenerCombustible(self):
105         return self.combustible
106
107     def agregarCombustible(self, comb):
108         if self.combustible + comb > self.max_deposito:
109             if self.estado_alertas:
110                 print('¡De los ' + str(comb) + ' litros, solo se pudieron cargar '
111                       + str(self.max_deposito - self.combustible) + ' litros!')
112             self.combustible = self.max_deposito
113         else:
114             if self.estado_alertas:
115                 print('¡Se cargaron ' + str(comb) + ' litros!')
116             self.combustible += comb
```

¿Qué sucede si se ejecuta el siguiente programa?

```
nave_espacial1 = NaveEspacial('R',100)
print('Combustible de Nave 1: ' + nave_espacial1.obtenerCombustible())
nave_espacial1.agregarCombustible(700)
print('Combustible de Nave 1: ' + str(nave_espacial1.obtenerCombustible()))
```

- ☐ a. Funciona normalmente
- ☒ b. La primer instrucción print falla porque se intenta concatenar una cadena con un valor entero. ✓
- ☐ c. Arroja una error, ya que el método __init__ espera que se proporcionen los parámetros co y comb.

Respuesta correcta

La respuesta correcta es:

La primer instrucción print falla porque se intenta concatenar una cadena con un valor entero.

Pregunta 16

Correcta

Se puntúa 0,40 sobre 0,40

Dada la clase `robot.py` adjunta en el material de la Semana 5, ¿Cuál es el valor del atributo `energia` de cada objeto luego de ejecutar el siguiente programa?

```
r1 = Robot('Fabian')
r2 = Robot('Rosalia')
r1.energia=50
r2.energia=100
r1.recargar()
print(r1.obtenerEnergia())
print(r2.obtenerEnergia())
```

- ☐ a. 50
50
- ☐ b. 100
50
- ☒ c. 100 ✓
100
- ☐ d. El programa no funciona, arroja un error
- ☐ e. 50
100

Respuesta correcta

La respuesta correcta es:

100

100

Pregunta 17

Correcta

Se puntúa 0,40 sobre 0,40

Dada la clase `punto.py` adjunta en el material de la Semana 6, evalúe la ejecución del siguiente programa:

```
punto12 = Punto(1, 2)
punto13 = punto12
punto13.establecerY(3)
punto12bis = punto13.clone()
punto12bis.copy(punto12)
print "[" + str(id(punto12)) + "]" + str(punto12) + " | " + "[" + str(id(punto13)) + "]" + str(punto13) + " | " + "[" + str(id(punto12bis)) + "]" + str(punto12bis)
print(punto12.equals(punto12bis))
```

La última instrucción imprime **True** porque:

- ☐ a. `punto12` y `punto12bis` hacen referencia al mismo objeto.
- ☐ b. `punto12` y `punto12bis` hacen referencia a objetos distintos, pero el valor del atributo `__x` de ambos coincide.
- ☒ c. `punto12` y `punto12bis` hacen referencia a objetos distintos y sus estados internos son equivalentes. ✓

Respuesta correcta

La respuesta correcta es:

`punto12` y `punto12bis` hacen referencia a objetos distintos y sus estados internos son equivalentes.

Pregunta 18

Correcta

Se puntúa 0,40 sobre 0,40

El siguiente programa se supone que debe verificar si un archivo existe y, si su tamaño es mayor a 0, imprimir la leyenda "El archivo prueba.txt existe y no está vacío". De lo contrario, debería imprimir "El archivo prueba.txt no existe". Sin embargo, el programa no funciona cuando el archivo no existe y lanza un error. ¿Por qué?

```
import os.path
from os import path
if path.exists('prueba.txt') or os.stat("prueba.txt").st_size > 0:
    print('El archivo prueba.txt existe y no está vacío')
else:
    print('El archivo prueba.txt no existe')
```

- ☐ a. El archivo debe existir si o si
- ☒ b. La sentencia **or** debería ser cambiada por una sentencia **and** ✓
- ☐ c. Las sentencias contenidas en bloques **if / else** deben precederse y estar seguidas de llaves ({})

Respuesta correcta

La respuesta correcta es:

La sentencia **or** debería ser cambiada por una sentencia **and**

Pregunta 19

Correcta

Se puntúa 0,40 sobre 0,40

Dado el diagrama de clase:

Profesor
<<Atributos de instancia>> legajo: entero edad: entero dedicación: char <<Atributos de clase>> edadJubilatoria: entero
<<Constructor>> Profesor(l: entero, e: entero, ded: char) <<Comandos>> establecerLegajo(l: entero) establecerEdad(e: entero) establecerDedicacion(d: char) copy(p: Profesor) <<Consultas>> obtenerLegajo(): entero obtenerEdad(): entero obtenerDedicacion(): char obtenerEdadJubilatoria(): entero equals(p: Profesor): boolean clone(): Profesor

copy(p: Profesor)

Requiere **p** ligado. Modifica el estado interno del objeto que recibe el mensaje con los valores de los atributos de instancia del objeto ligado a **p**.

equals(p: Profesor): boolean

Requiere **p** ligado. Retorna verdadero si el estado interno es igual al estado interno del objeto ligado a **p**.

clone(): Profesor

Crea y retorna un nuevo objeto con el mismo estado interno que el objeto que recibe el mensaje

¿Cuál es la salida del siguiente programa?

```

p1 = Profesor(1253, 30, 'P')
p2 = p1.clone()
p3 = Profesor(1254, 35, 'E')
p3.copy(p2)
p1.establecerLegajo(1255)
p3 = p2
print(p3.obtenerLegajo())

```

Respuesta: 1253



La respuesta correcta es: 1253

Pregunta **20**

Correcta

Se puntúa 0,40 sobre 0,40

¿Cuál es la salida del siguiente programa?

```
def factorial(n):  
    if n==0 or n==1:  
        resultado=1  
    elif n>1:  
        resultado=n*factorial(n-1)  
    return resultado  
  
fact=factorial(5)  
print(fact)
```

Respuesta: 120



La respuesta correcta es: 120

Pregunta **21**

Correcta

Se puntúa 0,40 sobre 0,40

Cuando una clase esta asociada a otra la implementación de la igualdad se puede hacer únicamente en forma superficial.

Seleccione una:

☐ Verdadero

☒ Falso ✓

Como se vio en clase, se puede hacer también en profundidad.

La respuesta correcta es 'Falso'

Pregunta 22

Correcta

Se puntúa 0,40 sobre 0,40

Dada la clase:

```
39 class PelotaConNombre:
40
41     def __init__(self, nombre):
42         self.nombre = nombre
43         self._establecerEstadoInicial()
44
45     def _establecerEstadoInicial(self):
46         self._establecerEstado('FRENADA')
47
48     def _establecerEstado(self, estado):
49         self.estado = estado
50
51     def establecerNombre(self, nombre):
52         self.nombre = nombre
53
54     def obtenerEstado(self):
55         return self.estado
56
57     def obtenerNombre(self):
58         return self.nombre
59
60     def rodar(self):
61         print('Rodando...')
62         self._establecerEstado('RODANDO')
63
64     def frenar(self):
65         print('Frenando...')
66         self._establecerEstado('FRENADA')
67
68     def imprimirEstado(self):
69         print('Estado de ' + self.nombre + ': ' + self.estado)
70
```

El programa:

```
pelota1 = PelotaConNombre('Pelota 1')
pelota2 = PelotaConNombre('Pelota 2')
pelota1.establecerNombre('Pelota 2')
pelota2.establecerNombre('Pelota 2')
print(pelota1.obtenerNombre())
print(pelota2.obtenerNombre())
```

Imprime:

```
Pelota 1
Pelota 2
```

Seleccione una:

☐ Verdadero☒ Falso ✓

La salida de este programa es:

```
Pelota 2
Pelota 2
```

La respuesta correcta es 'Falso'

Pregunta **23**

Correcta

Se puntúa 0,40 sobre 0,40

¿Las funciones son bloques de código que pueden ser llamadas cuantas veces sean necesarias?

Seleccione una:

- ☒ Verdadero ✓
- ☐ Falso

La respuesta correcta es 'Verdadero'

Pregunta **24**

Correcta

Se puntúa 0,40 sobre 0,40

Los comentarios en Python deben estar en un programa para que este pueda ejecutarse

Seleccione una:

- ☐ Verdadero
- ☒ Falso ✓

Los comentarios son una mera forma de documentación, que sirve como guía a los programadores para comprender el propósito y comportamiento de los módulos y modelos que componen un programa.

La respuesta correcta es 'Falso'

Pregunta **25**

Correcta

Se puntúa 0,40 sobre 0,40

Un tipo abstracto de datos es un tipo de datos que:

- Consta de Datos y Operaciones que pueden realizarse sobre estos datos
- Es definido por el programador
- Permite hasta cierto punto modelar el comportamiento de una entidad real

Seleccione una:

- ☒ Verdadero ✓
- ☐ Falso

Un Tipo Abstracto de Datos o TDA está constituido por una estructura de datos y operaciones que se pueden realizar sobre esos datos, y está definido por el programador.

- El lenguaje elegido para implementar un TDA debe permitir asociar la representación de la estructura de datos con las operaciones que la manipulan.
- La representación del TDA está oculta en las unidades de programa que lo utilizan.

La respuesta correcta es 'Verdadero'

◀ Programa de la Asignatura

Ir a...



