Orientación a Objetos I

Agenda de la
semana del 16 de Septiembre

Actividades de la semana anterior

- Ejercicio 3 (Presupuestos) y haberlo testeado con los tests provistos
- Ejercicio 4 (Balanza Mejorada)
- Ejercicio 5 (Figuras y cuerpos) y haberlo testeado con los tests provistos

Cuadernillo Semestral de Actividades

Actualizado: 8 de septiembre de 2024

El presente cuadernillo posee un compilado con todos los ejercicios que se usarán durante el semestre en la asignatura. Los ejercicios están organizados en forma secuencial, siguiendo los contenidos que se van viendo en la materia.

Cada semana les indicaremos cuáles son los ejercicios en los que deberían enfocarse para estar al día y algunos de ellos serán discutidos en la explicación de práctica.

Recomendación importante:
Los contenidos de la materia se incorporan y fijan mejor cuando uno intenta aplicarlos **- no alcanza con ver un ejercicio resuelto por alguien más**. Para sacar el máximo provecho de los ejercicios, es importante que asistan a las consultas de práctica habiendo intentado resolverlos (tanto como les sea posible). De esa manera podrán hacer consultas más enfocadas y el docente podrá darles mejor feedback.

FACULTAD DE INFORMATICA | UNIVERSIDAD NACIONAL DE LA PLATA

Actividades esperadas para esta semana

- Ejercicio 6: Genealogía Salvaje
- Ejercicio 7: Red de Alumbrado
- Ejercicio 8: Method lookup con Empleados

Ejercicio 6: Genealogía Salvaje

- **Ejercicio 6: Genealogía Salvaje**
- Ejercicio 7: Red de Alumbrado
- Ejercicio 8: Method lookup con Empleados

Ejercicio 6: Genealogía salvaje

En una reserva de vida salvaje (como la estación de cría ECAS, en el camino Centenario), los cuidadores quieren llevar registro detallado de los animales que cuidan y sus familias. Para ello nos han pedido ayuda. Debemos:

Tareas:

a) **Complete el diseño e implemente**

Modelar una solución en objetos e implementar la clase Mamífero (como subclase de Object). El siguiente diagrama de clases (incompleto) nos da una idea de los mensajes que un mamífero entiende.

Proponga una solución para el método *tieneComoAncestroA(...) y deje la implementación para el final y discuta su solución con el ayudante.*

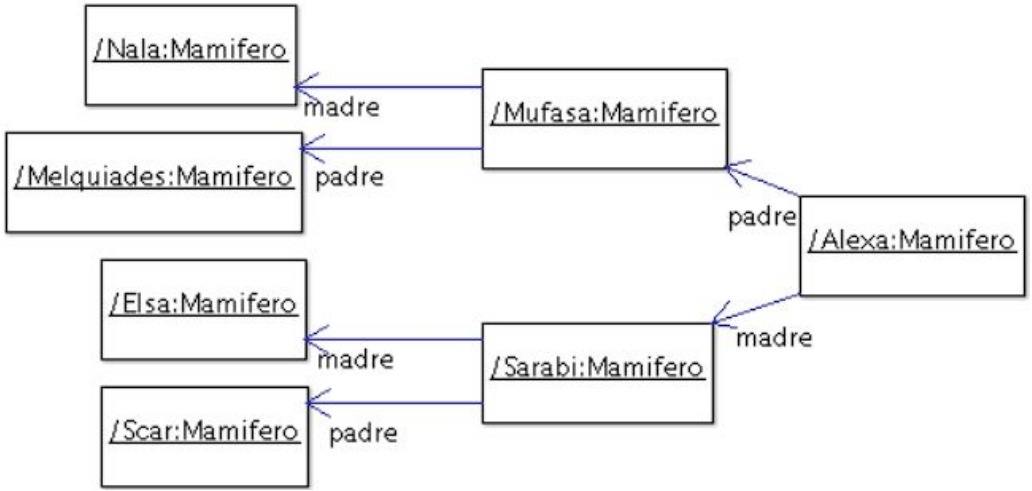
FACULTAD DE INFORMATICA | UNIVERSIDAD NACIONAL DE LA PLATA

Ejercicio 6: Genealogía Salvaje

Siguiendo los ejemplos de ejercicios anteriores, ejecute las pruebas automatizadas provistas.

En este caso, se trata de una clase, **MamiferoTest**, que debe agregar dentro del paquete tests.

En esta clase se trabaja con la familia mostrada en la siguiente figura.



Ejercicio 6 - Genealogía Salvaje

Clase Mamifero

¿Atributos?
¿Relaciones?

¿Variables de instancia?

Ejercicio 7: Red de Alumbrado

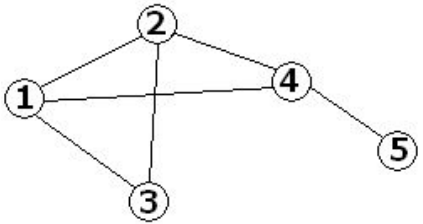
- Ejercicio 6: Genealogía Salvaje
- **Ejercicio 7: Red de Alumbrado**
- Ejercicio 8: Method lookup con Empleados

Ejercicio 7: Red de Alumbrado

Imagine una red de alumbrado donde cada farola está conectada a una o varias vecinas formando un grafo conexo[2]. Cada una de las farolas tiene un interruptor. Es suficiente con encender o apagar una farola cualquiera para que se enciendan o apaguen todas las demás. Sin embargo, si se intenta apagar una farola apagada (o si se intenta encender una farola encendida) no habrá ningún efecto, ya que no se propagará esta acción hacia las vecinas.

La funcionalidad a proveer permite:

1. crear farolas (inicialmente están apagadas)
2. conectar farolas a tantas vecinas como uno quiera (las conexiones son bi-direccionales)
3. encender una farola (y obtener el efecto antes descrito)
4. apagar una farola (y obtener el efecto antes descrito)



FACULTAD DE INFORMATICA | UNIVERSIDAD NACIONAL DE LA PLATA

Ejercicio 7: Red de Alumbrado

```
public class Farola {

// Variables de instancia??

/*

Crea la relación de vecinos entre las farolas. La relación de vecinos entre las farolas es recíproca, es
decir el receptor del mensaje será vecino de otraFarola, al igual que otraFarola también se convertirá en
vecina del receptor del mensaje

*/
public void pairWithNeighbor(Farola otraFarola){
    ...
}


/*
* Si la farola no está encendida, la enciende y propaga la acción.
*/
public void turnOn(){
   ...
}
```

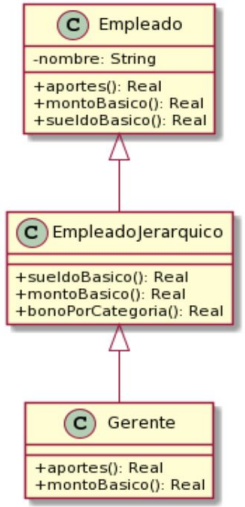
FACULTAD DE INFORMATICA

UNIVERSIDAD NACIONAL DE LA PLATA

Ejercicio 8: Method lookup con Empleados

- Ejercicio 6: Genealogía Salvaje
- Ejercicio 7: Red de Alumbrado
- **Ejercicio 8: Method lookup con Empleados**

Sea la jerarquía de `Empleado` como muestra la figura de la izquierda, cuya implementación de referencia se incluye en la tabla de la derecha.



Empleado	EmpleadoJerarquico	Gerente
`public double montoBasico() { return 35000; }`	`public double sueldoBasico() { return super.sueldoBasico()+ this.bonoPorCategoria(); }`	`public double aportes() { return this.montoBasico() * 0.05d; }`
`public double aportes(){ return 13500; }`	`public double montoBasico() { return 45000; }`	`public double montoBasico() { return 57000; }`
`public double sueldoBasico() { return this.montoBasico() + this.aportes();}`	`public double bonoPorCategoria() { return 8000; }`	

FACULTAD DE INFORMATICA | UNIVERSIDAD NACIONAL DE LA PLATA

Ejercicio 8: Method lookup con Empleados

Analice cada uno de los siguientes fragmentos de código y resuelva las tareas indicadas abajo:

```
Gerente alan = new Gerente("Alan Turing");
double aportesDeAlan = alan.aportes();
```

```
Gerente alan = new Gerente("Alan Turing");
double sueldoBasicoDeAlan = alan.sueldoBasico();
```

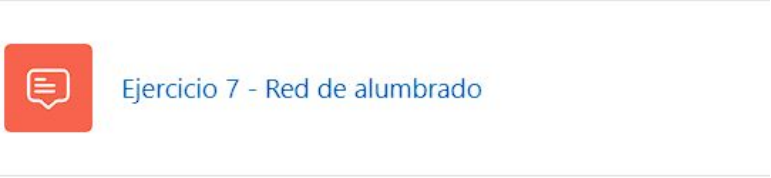
Tareas:

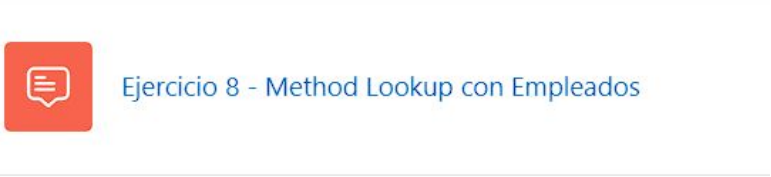
1. Liste todos los métodos, indicando nombre y clase, que son ejecutados como resultado del envío del último mensaje de cada fragmento de código (por ejemplo, (1) método +aportes de la clase Empleado, (2) ...)
2. ¿Qué valores tendrán las variables aportesDeAlan y sueldoBasicoDeAlan luego de ejecutar cada fragmento de código?

Foros de consulta

Cómo preguntar en el foro

Breve guía para poder sacar el mejor provecho al foro y a la convivencia a través de las preguntas y respuestas.





FACULTAD DE INFORMATICA

UNIVERSIDAD NACIONAL DE LA PLATA