Examen Final

Debemos hacer un sistema cuyo objetivo consiste en que un empresario pueda liquidar el sueldo del mes de sus obreros y poder conocer el total a pagar para todos los empleados.
Hay diferentes tipos de obreros:

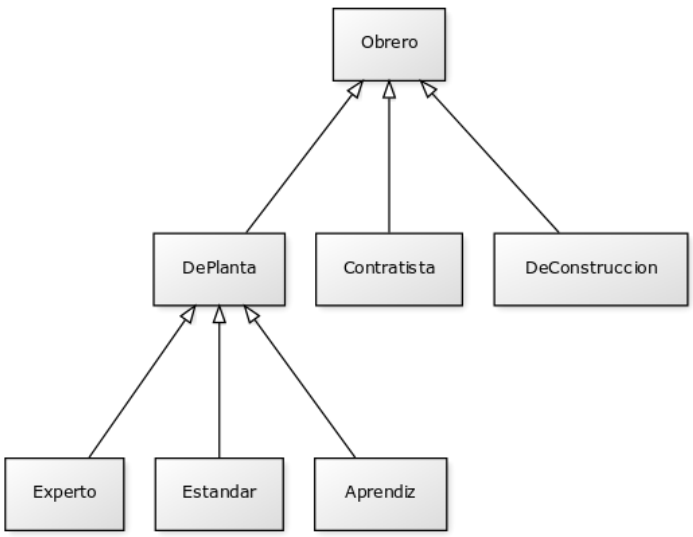
- de la **construcción**: el monto se establece por quincena y puede ser diferente para cada obrero; pero todos cobran al mes el equivalente a dos quincenas.
- de **planta**: trabajan 40 horas semanales y cobran un sueldo mensual que depende del básico de la categoría que tenga, más lo que le corresponda por las horas extras que haya hecho. Hay tres categorías: aprendiz, estándar y experto, y a cada una de ellas le corresponde un único valor. Además para el cálculo de horas extra, salvo para los aprendices que lamentablemente no les pagan las horas extra, cada hora se paga al 150% del valor normal (que es el básico / 40). Los obreros de planta, con el tiempo, pueden cambiar de categoría.
- **contratistas**: cobran la sumatoria de los importes de los trabajos que hicieron en el mes.

De cada empleado se conoce: su nombre, qué tipo de obrero es y las particularidades necesarias para la liquidación de su sueldo.

Parte A

Para una solución en objetos, en la cual tenemos una clase **Empresario** con un atributo **obreros** que es una colección con objetos que representan a todos los empleados que trabajan para él, y al que se espera poder mandarle el mensaje totalAPagarPorMes:

1. ¿Cuáles serían las ventajas de que los obreros sean polimórficos para el empresario?
2. ¿Qué opinás de utilizar herencia para modelar los distintos tipos de obreros y los distintos tipos de obreros de planta como se muestra en el siguiente diagrama? **Justificar**, indicando ventajas y/o desventajas, de usar herencia **en cada caso**.



3. Implementar el cálculo de sueldo para un obrero de planta y para un obrero contratista. Si se introducen cambios al modelo, justificarlos.
4. ¿Qué conceptos del paradigma se destacan en la solución del punto anterior? Indicar dónde se ponen en evidencia.

Parte B

Para una solución en lógico, querríamos armar una base de conocimientos en donde se registre la información de cada empleado mediante un predicado llamado **obrero**.

1. ¿Cómo podrían modelarse los empleados para una solución en Prolog? Escribir tres cláusulas de ejemplo para el predicado obrero, una por cada tipo de obrero.
2. Implementar el cálculo de sueldo para un obrero de construcción.
3. ¿El predicado definido en el punto anterior es inversible? Justificar por qué sí o por qué no y cómo impactaría esto a su uso en un predicado totalAPagarPorMes/1 que obtenga la sumatoria de lo que debe pagarse a todos los obreros registrados en la base de conocimientos.

Parte C

Tenemos la siguiente función:

```
f x _ [] = x
f x y (z:zs)
    | y z > 0 = z + f x y zs
    | otherwise = f x y zs
```

1. Declarar el tipo de la función f.
2. Definir nuevamente la función f de modo que sea más declarativa.
3. Indicar qué conceptos se aplicaron en la nueva solución mostrando dónde aparecen. ¿Qué es lo que hace que la misma sea más declarativa?
4. ¿Qué pasaría si a f se le aplica como último parámetro una lista infinita?