PARCIAL OBJETOS – CURSO K2054 NOMBRE:

La Gerencia de Recursos Humanos de "Rabufetti Construcciones" quiere informatizar las búsquedas de personal para cubrir puestos de trabajo. El informe del relevamiento es el siguiente:

Los postulantes son las personas que acceden al proceso de selección de un cargo, en principio pueden ser:

- **Personal de planta:** son las personas que están en relación de dependencia con la empresa, cobran mensualmente un sueldo fijo determinado por el cargo que ocupan (jefe de planta, supervisor, operario junior, operario semi-senior, gerente línea media, gerente ejecutivo, etc.) y trabajan en un sector (Administración, Sistemas, Recursos Humanos, etc.)

- **Personal contratado que trabaja para la compañía:** son empleados que están en relación de dependencia con otra consultora. Rabufetti paga los servicios de estas personas por hora trabajada pero cada empleado cobra un sueldo propio (que no depende del escalafón de cargos de Rabufetti). Las personas contratadas trabajan en un sector y dependen de un jefe que forma parte del personal de planta de la empresa.

  Se debe registrar para el personal de planta y para los contratados cuántos empleados a cargo tienen (no hace falta saber quiénes, sino su cantidad).

- **Externos:** son personas desempleadas o bien que están trabajando para otra compañía, que dejaron sus datos en una web: nombre y apellido, teléfono, e-mail, su currículum (conformado por las empresas en las que trabajaron + el cargo que ocuparon), fecha de nacimiento, etc.

La empresa realiza distintos tipos de búsqueda:

- **Búsquedas internas:** participan solamente
  - personal de planta
  - personal contratado si la búsqueda es para el mismo sector en el que trabajan.

- **Búsquedas externas:** solamente pueden participar
  - los externos
  - el personal contratado que tenga menos de un año de trabajo en la empresa.

- **Búsquedas especiales:** se utiliza para puestos de alta jerarquía. En ese caso pueden participar:
  - Personal de planta con sueldo actual menor al que se ofrece (según el cargo a elegir) y que tenga al menos 10 personas a cargo
  - Personal contratado que tenga más de 20 personas a cargo
  - Externos que hayan trabajado anteriormente en el mismo puesto

1) Realice un script en un Workspace que permita crear:

   - Al empleado Erico Lavallén, empleado de planta, con sueldo $ 2.500, que es lo que gana un *Project Leader*, tiene 12 personas a cargo y trabaja en el sector Desarrollo de Sistemas.
   - Al empleado Patricio García, personal contratado por la consultora "Little People". Su jefe es Erico Lavallén y no tiene personal a cargo. Trabaja en el sector Desarrollo de Sistemas.
   - Al externo Rodrigo Girlando, rgirlandus@gmail.com que trabajó 3 años como Analista Junior en MedCenter y 2 años como Analista Junior en Telecom.
   - Una búsqueda externa para cubrir un puesto de Programador Semi-Senior, a la cual se postularon Patricio García y Rodrigo Girlando.

   No hace falta que codifique los métodos en los objetos receptores, sólo indique cómo armaría el script.

2) Realice el diagrama de objetos según el punto 1). Recuerde que no hace falta comunicar todos los objetos en el diagrama, sólo los que ud. considere importantes.

3) Realice un diagrama de clases, mostrando las relaciones de generalización y asociación. Explique qué significa cada una de las relaciones.

4) Codifique los siguientes métodos, indicando en qué clase se ubican:

   a) **empleadosQueTrabajanEn:** unSector
   Devuelve los postulantes de una búsqueda que trabajan en un sector particular (recordar que puede haber externos entre los postulantes).

   b) **busquedasCopadas**
   Devuelve las búsquedas copadas en las que se anotó un postulante. Las búsquedas copadas son:
   - *Para empleados de planta:* aquellas en donde ofrecen un sueldo dos o más veces superior al sueldo del empleado. Recordamos que el sueldo ofrecido por la compañía viene dado por el cargo que hay que cubrir.
   - *Para empleados contratados:* aquellas en donde ofrecen un sueldo superior al sueldo del empleado.
   - *Para externos:* aquellas búsquedas cuyo sueldo es superior a $ 1.000

   c) **puedePostularseABusquedaInterna:** unaBusqueda
   Indica si un postulante puede inscribirse en una búsqueda interna, según las reglas de negocio definidas en la página anterior).

   puedePostularseABusquedaExterna: unaBusqueda
   Indica si un postulante puede inscribirse a una búsqueda externa, según las reglas de negocio definidas en la página anterior).

   puedePostularseABusquedaEspecial: unaBusqueda
   Indica si un postulante puede inscribirse a una búsqueda especial, según las reglas de negocio definidas en la página anterior).

   d) **postularA:** unPostulante
   Permite que un postulante se anote en una búsqueda. Si el empleado no cumple los requisitos, el sistema debe devolver un mensaje de error indicando “No se puede postular a la búsqueda”.

Nota: es <u>importante</u> que la solución de todos los métodos muestre las herramientas del POO aprendidas durante la cursada (no vale con que “ande”).

5) Señale dónde aparecieron en el punto anterior los siguientes conceptos:
   - Polimorfismo
   - Delegación

   ¿Quién aprovechó el polimorfismo?

Recomendaciones
No usen una clase Empresa o Sistema que conozca a todo el mundo, que cada objeto conozca a los que tiene que conocer y que cada objeto haga <u>sólo</u> lo que tiene que hacer.

P.ej. si cada postulante tiene que conocer a las búsquedas en las que participó, pues que las conozca. No preocuparse por conocimientos bidireccionales. Obviamente, tampoco hacer que se conozcan más objetos de lo necesario.

Puntos:

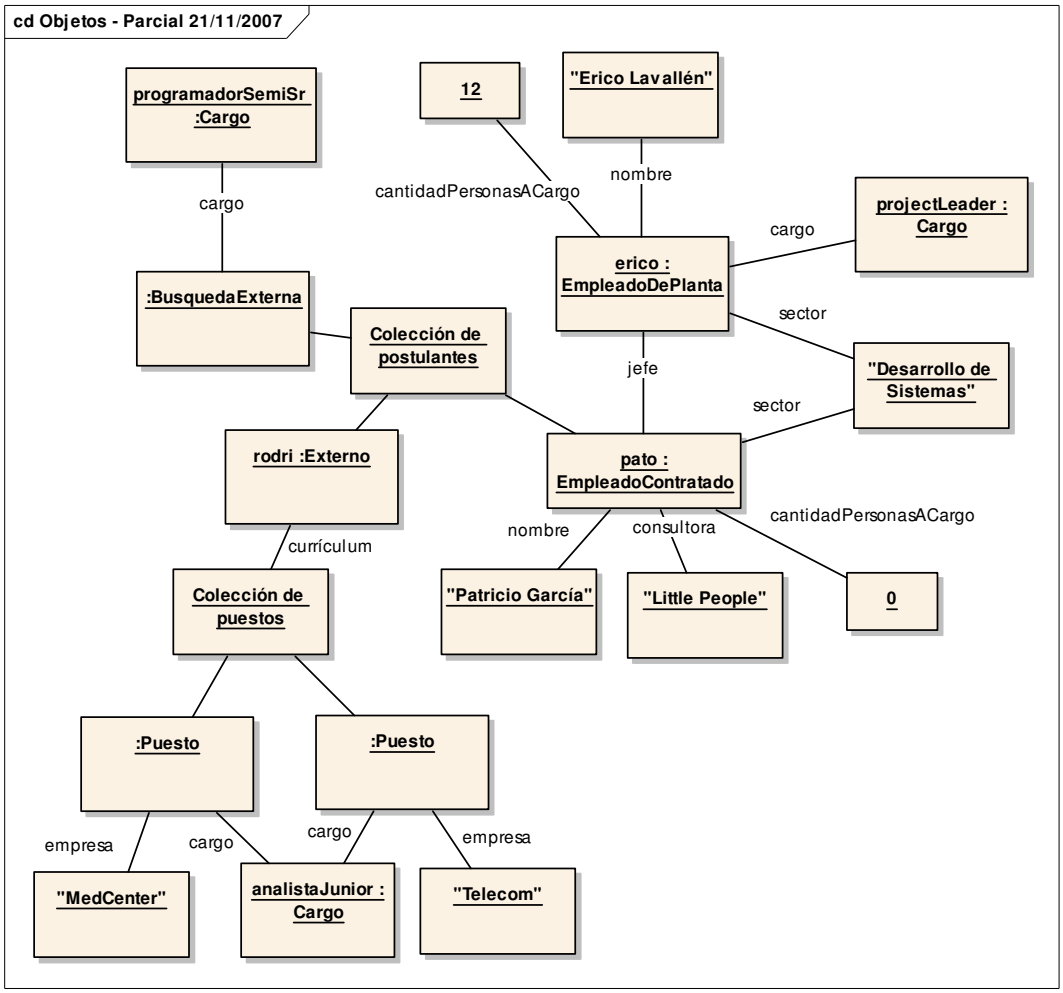
	Qué se evalúa
1	Script en un Workspace
2	Diagrama de objetos. Cómo se relacionan en el ambiente.
3	Diagrama de clases. Tiene vs. "Es un". Relaciones estáticas.
4	Implementación siguiendo conceptos del paradigma.
5	Conceptos teóricos fundamentales para las materias que vienen: polimorfismo y delegación.

Solución

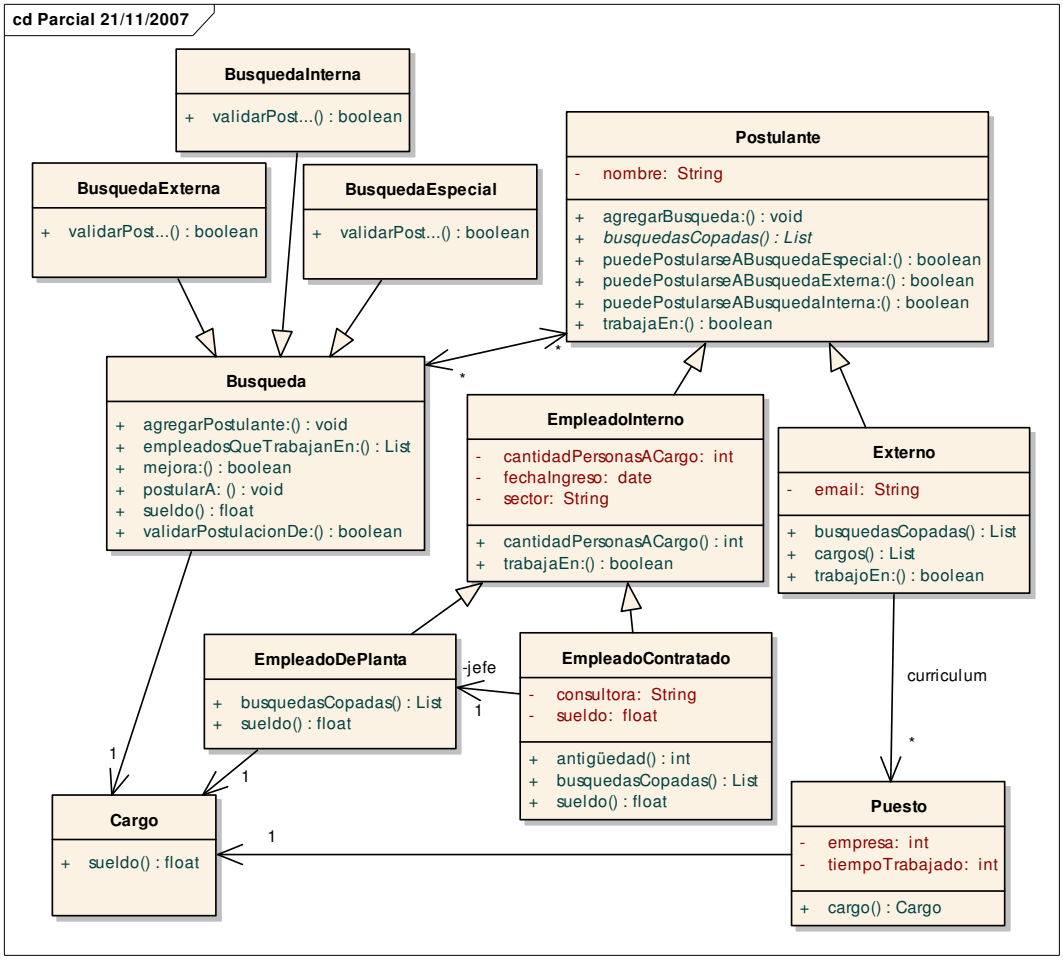
1)

```
analistaJunior := Cargo new nombre: 'Analista Junior'; sueldo: 1750.
projectLeader := Cargo new nombre: 'Project Leader'; sueldo: 2500.

erico := EmpleadoDePlanta new
        nombre: 'Erico Lavallén';
        cargo: projectLeader;
        cantidadPersonasACargo: 12;
        sector: 'Desarrollo de Sistemas'.
pato := EmpleadoContratado new
        nombre: 'Patricio García';
        consultora: 'Little People';
        jefe: erico;
        cantidadPersonasACargo: 0;
        sector: 'Desarrollo de Sistemas'.
rodri := Externo new
        nombre: 'Rodrigo Girlando';
        email: 'rgirlandus@gmail.com'.

rodri agregarTrabajoRealizadoDe: analistaJunior en: 'MedCenter' durante: 3.
rodri agregarTrabajoRealizadoDe: analistaJunior en: 'Telecom' durante: 2.

programadorSemiSr := Cargo new nombre: 'Programador Semi Senior'; sueldo: 1600.

busquedaProgramadorSemiSr := BusquedaExterna new
        cargo: programadorSemiSr;
        postularA: pato;
        postularA: rodri.
```

2) *Diagrama de objetos (para verlo mejor, agrandar con el Zoom el documento)*



3) *Diagrama de clases (para verlo mejor, agrandar con el Zoom el documento)*



El Sector puede o no ser una clase, no importa.

Hay relaciones de **generalización** (“es un”) entre:

- la superclase Busqueda y las subclases BusquedaInterna, BusquedaExterna y BusquedaEspecial
- la superclase Postulante y las subclases EmpleadoInterno y Externo
- la superclase EmpleadoInterno y las subclases EmpleadoDePlanta y Contratado.

No es obligatoria la existencia de EmpleadoInterno, aunque hay muchas características y comportamiento común entre el empleado de planta y el contratado que lo justifica.

La relación de generalización permite que las subclases hereden la definición de los atributos y el comportamiento (el código) de la superclase.

Hay relaciones de **asociación** (“tiene”) entre las clases:

- Búsqueda y Postulante, y es bidireccional
- Búsqueda y Cargo
- Puesto y Cargo
- EmpleadoDePlanta y Cargo

En la relación bidireccional Búsqueda y Postulante, la búsqueda conoce a sus postulantes y cada postulante conoce en qué búsquedas se anotó. En el caso de Búsqueda y Cargo, cada objeto Búsqueda tiene una variable de instancia cargo que referencia a un objeto Cargo. Lo mismo ocurre con los otros dos casos (la relación es de la primera entidad hacia la segunda).

4)

a)

```
empleadosQueTrabajanEn: unSector (#Busqueda)
   ^postulantes select: [ :postulante | postulante trabajaEn: unSector ]

trabajaEn: unSector (#Externo o #Postulante)
   ^false

trabajaEn: unSector (#Interno)
   ^sector = unSector
```

b)

```
busquedasCopadas (#EmpleadoDePlanta)
   ^busquedas select: [ :busqueda | busqueda mejora: self sueldo * 2[1] ]

busquedasCopadas (#EmpleadoContratado)
   ^busquedas select: [ :busqueda | busqueda mejora: self sueldo[2] ]

busquedasCopadas (#EmpleadoContratado)
   ^busquedas select: [ :busqueda | busqueda sueldo > 1000[3] ]

sueldo (#Busqueda y también #EmpleadoDePlanta)
   ^cargo sueldo

sueldo (#EmpleadoContratado)
    ^sueldo

mejora: unMonto (#Busqueda)
   ^cargo sueldo > unMonto[4]
```

c)

```
puedePostularseABusquedaInterna: unaBusquedaInterna (#EmpleadoDePlanta)
    ^true

puedePostularseABusquedaInterna: unaBusquedaInterna (#Postulante)
    ^false

puedePostularseABusquedaInterna: unaBusquedaInterna (#EmpleadoContratado)
    ^unaBusqueda sector = sector

puedePostularseABusquedaExterna (#Externo)
    ^true

puedePostularseABusquedaExterna (#Postulante)
    ^false

puedePostularseABusquedaExterna (#EmpleadoContratado)
```

[1] también pueden hacer busqueda sueldo > self sueldo * 2

[2] también pueden hacer busqueda sueldo > self sueldo
Menos copado: busqueda cargo sueldo > self sueldo, pero vale.

[3] de esta manera, el que define si una búsqueda es copada es el Externo, pero el que sabe el sueldo de una búsqueda es la Búsqueda. Con esto nos referimos a que cada objeto haga sólo lo que tiene que hacer.

[4] También puedo delegar si mejora: unMonto al objeto Cargo

```
    ^self antigüedad > 1

antigüedad (#EmpleadoContratado)
    ^Date today yearsSince: fechaIngreso (si no que usen el – y a la lona)

puedePostularseABusquedaEspecial: unaBusquedaEspecial (#Postulante)
    ^false

puedePostularseABusquedaEspecial: unaBusquedaEspecial (#Externo)
    ^self trabajoEn: unaBusquedaEspecial cargo

trabajoEn: unCargo (#Externo)
    ^self cargos includes: unCargo

cargos (#Externo)
    ^currículum collect: [ :puesto | puesto cargo ]

puedePostularseABusquedaEspecial: unaBusquedaEspecial (#EmpleadoContratado)
    ^self cantidadPersonasACargo > 20

puedePostularseABusquedaEspecial: unaBusquedaEspecial (#EmpleadoDePlanta)
    ^(self cantidadPersonasACargo > 10) && (unaBusquedaEspecial mejora: self sueldo)
```

d)

```
postularA: unPostulante (#Busqueda)
    (self validarPostulacionDe: unPostulante)
        ifFalse: [ self error: 'El postulante no cumple los requisitos' ].
    unPostulante agregarBusqueda: self.
    self agregarPostulante: unPostulante.

validarPostulacionDe: unPostulante (#BusquedaInterna)
    ^unPostulante puedePostularseABusquedaInterna: self

validarPostulacionDe: unPostulante (#BusquedaExterna)
    ^unPostulante puedePostularseABusquedaExterna

validarPostulacionDe: unPostulante (#BusquedaEspecial)
    ^unPostulante puedePostularseABusquedaEspecial: self

agregar: unPostulante (#Busqueda)
    postulantes add: unPostulante

agregar: unaBusqueda (#Postulante)
    busquedas add: unaBusqueda
```

Menos exquisito (pero igualmente válido) es definir tres métodos postular que validen en forma particular y agreguen las relaciones entre postulante y búsqueda

6)

Polimorfismo: yo puedo intercambiar objetos postulantes para validar si se pueden anotar en una búsqueda o para saber si trabajan en un sector, pero también puedo intercambiar las búsquedas internas, externas o especiales para saber si un postulante puede anotarse en dicha búsqueda. También puedo intercambiar empleados de planta y contratados para saber qué sueldo tienen.

El beneficio es siempre <u>para el que usa</u> a los objetos polimórficos. Yo no se si un empleado es contratado o es de planta, yo le pido que me de su sueldo. En el caso de los empleados de

planta, el sueldo depende del cargo, en el caso de los contratados, lo conoce el contratado mismo, pero al que le pide el sueldo *no le importa (y está bien que sea así)*.

Delegación: en muchos casos se da self delegation (todos los métodos que validan la postulación de un postulante se envían a sí mismo y delegan en el postulante esa validación), en otros se "parte" un método largo en varios métodos más cortos dentro de la misma clase (ej: trabajoEn: de #Externo), y en casi los demás casos cuando dos objetos colaboran, el que termina resolviendo cada cosa es el que es responsable de hacerlo:

```
empleadosQueTrabajanEn: unSector (#Busqueda)
    ^postulantes select: [ :postulante | postulante trabajaEn: unSector ]
```

Acá el postulante y la búsqueda colaboran para que la búsqueda devuelva todos los empleados que trabajan en un sector, pero el Postulante es el que determina si trabaja en un sector.
¿Quién conoce a los postulantes? La búsqueda.
¿Quién sabe si trabaja en un sector? El postulante.
Con la delegación me aseguro que sólo tengo que cumplir que el postulante entienda el mensaje trabajaEn: