2°Parcial/2022 - TEMA 2

Apellido y Nombres: .....

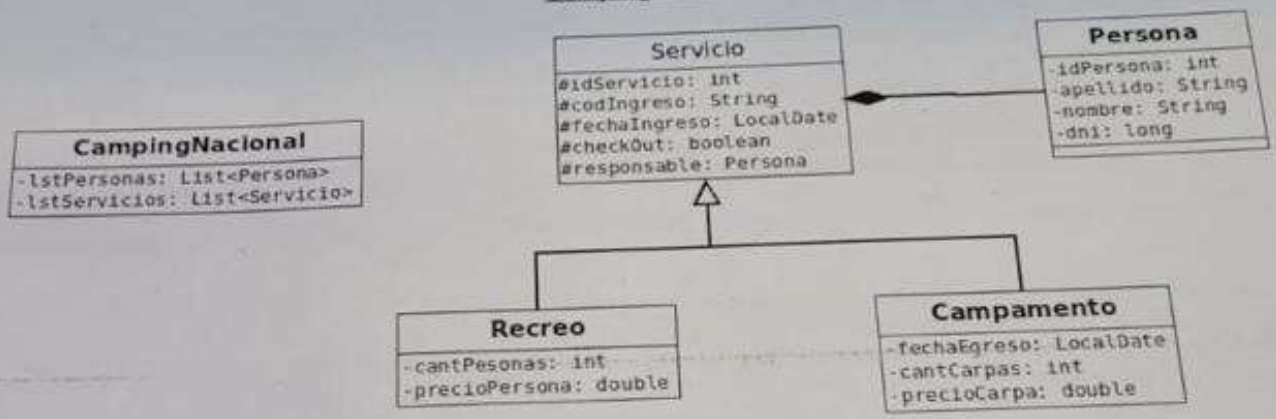
DNI: .....

Firma: .....

UNIVERSIDAD NACIONAL DE LANUS
UNLa.
Licenciatura en Sistemas
Orientación a Objetos 1

IDE: Eclipse **Proyecto:** Crear una carpeta /tuApellidoNombre/tuNroDni (el nombre del proyecto es tu DNI)
Entrega: Aula Virtual subir la carpeta comprimida con tuApellidoNombre
Adjunto: La carpeta a enviar comprimida es tuApellidoNombre

Camping Nacional



Nota: El examen acredita puntos por la resolución del modelo y test de cada CU.

Clase Persona:
- int idPersona;
- String apellido;
- String nombre;
- long dni;

Clase Servicio (Abstracta):
int idServicio;
String codIngreso;
LocalDate fechaIngreso;
boolean checkOut;
Persona responsable;

Clase Recreo:
- int cantPersonas;
- double precioPersona;

Clase Campamento:
- LocalDate fechaEgreso;
- int cantCarpas;
- double precioCarpa;

Clase CampingNacional:
- List<Persona> lstPersonas;
- List<Servicio> lstServicios;

Casos de uso:

1) + traerPersona(long dni) : Persona

2) + agregarPersona(String apellido, String nombre, long dni) : boolean
Lanzar excepción si ya existe una Persona con el mismo dni.
El idPersona se calcula de forma autoincremental, teniendo en cuenta que la lista puede tener altas y bajas de

3) + esValidoDigitoControl(String codIngreso) : boolean
Para calcular el dígito de verificador se suman los 6 primeros dígitos, resultando el dígito verificador congruenc
(resto de dividir la suma en 10).
Ejemplo: si el codIngreso="7432163"
a) Se suman los dígitos seis primeros dígitos: 7 + 4 + 3 + 2 + 1 + 6 = 23
b) dVerificador=suma % 10 entonces dVerificador=3
c) Como el dígito verificador calculado coincide con el último dígito el método retorna true

4) + agregarRecreo(String codIngreso, LocalDate fechaIngreso, boolean checkOut, Persona responsable, int cantPersonas, double precioPersona) : boolean
Lanzar excepción si el codIngreso es inválido.
El idServicio se calcula de forma autoincremental, teniendo en cuenta que la lista puede tener altas y bajas de objetos.

5) + agregarCampamento(String codIngreso, LocalDate fechaIngreso, boolean checkOut, Persona responsable, LocalDate fechaEgreso, int cantCarpas, double precioCarpa) : boolean
Lanzar excepción si el codIngreso es inválido.
El idServicio se calcula de forma autoincremental, teniendo en cuenta que la lista puede tener altas y bajas de objetos.

6) + cantidadDias() : int
Devuelve la cantidad de días entre la fecha de ingreso y la fecha de egreso del **Campamento**. Si no se realizó el checkOut, calcular respecto a la fecha de hoy "LocalDate.of(2022.10,18)"

HELP
Se puede utilizar la clase **Period** que es estática y está dentro de la librería java.time.Period
Ejemplo: **Period**.between(LocalDate startDateInclusive, LocalDate endDateExclusive).**getDays()**
Retorna el número de días que hay entre dos fechas.

7) ~ calcularPrecioFinal() : double // Implementar como método abstracto.
Si la clase es Recreo, se devuelve precioPersona*cantPersonas.
Si la clase es Campamento, se devuelve cantCarpas*precioCarpa*cantidadDias

8) + traerServiciosPorPrecioFinal(double mayorIgualA) : List<Servicio>
Devuelve todos los servicios que hayan realizado el check-out y cuyo precio final sea mayor al valor pasado como parámetro

9) + traerCampamentosPorDias(int mayorIgualA) : List<Campamento>
Devuelve todos los **Campamentos** cuya cantidad de días en el predio sea mayor o igual al valor pasado como parámetro.

Nota: Al comenzar cada test indicar el número a resolver ej: System.out.println("1"); y luego la implementación del mismo.

Test.java
1) Agregar e imprimir siguientes personas
Persona [apellido=Sergio, nombre=Lopez, dni=11111111],
Persona [apellido=Juan, nombre=Rodriguez, dni=22222222],
Persona [apellido=Maria, nombre=Fernandez, dni=33333333],
Persona [apellido=Juan, nombre=Vazquez, dni=44444444],
Persona [apellido=Ana, nombre=Martinez, dni=55555555]

2) Agregar e imprimir los siguientes servicios
Recreo [codIngreso=7358902, fechaIngreso=2022-09-02, checkOut=false, responsable=
Persona [apellido=Sergio, nombre=Lopez, dni=11111111], cantPersonas=5, precioPersona=300.0],

Campamento [codIngreso=6573540, fechaIngreso=2022-09-03, checkOut=false, responsable=
Persona [apellido=Juan, nombre=Rodriguez, dni=22222222], fechaEgreso=null, cantCarpas=1, precioCarpa=1500.0],

Recreo [codIngreso=3571398, fechaIngreso=2022-09-03, checkOut=false, responsable=
Persona [apellido=Maria, nombre=Fernandez, dni=33333333], cantPersonas=10, precioPersona=300.0],

Campamento [codIngreso=2583941, fechaIngreso=2022-09-03, checkOut=true, responsable=
Persona [apellido=Juan, nombre=Vazquez, dni=44444444], fechaEgreso=2022-09-08, cantCarpas=2, precioCarpa=1500.0],

Recreo [codIngreso=5243925, fechaIngreso=2022-09-03, checkOut=true, responsable=
Persona [apellido=Ana, nombre=Martinez, dni=55555555], cantPersonas=8, precioPersona=300.0]

3) Traer e imprimir los Servicios con precio final mayor a 1000 y que hayan finalizado su estadía

4) Traer e imprimir los Campamentos con más de 5 días en el predio

5) Intentar agregar a la siguiente Persona que ya existe en la lista:
Persona [apellido=María, nombre=Fernandez, dni=33333333]

6) Intentar agregar el siguiente servicio con codIngreso inválido:
Recreo [codIngreso=5732927, fechaIngreso=2022-10-02, checkOut=false, responsable=
Persona [apellido=Sergio, nombre=Lopez, dni=11111111], cantPersonas=5, precioPersona=300.0]