Ejercicio 1

El dueño de un hotel te pide a desarrollar un programa para consultar sobre las habitaciones disponibles y reservar habitaciones de su hotel

El hotel posee tres tipos de habitaciones: simple, doble y matrimonial, y dos tipos de clientes: habituales y esporádicos. Una reserva almacena datos del cliente, de la habitación reservada, la fecha de comienzo y el número de días que será ocupada la habitación

De cada habitación almacenamos un código correspondiente piso+letra (2ª, segundo piso letra A) y su precio por dia.

Para los clientes almacenamos la información ,. Dni, nombre y apellidos . Los cliente habituales tienen un descuento para todos iguales almacenado como cte.

El recepcionista del hotel debe poder hacer las siguientes operaciones:

- Obtener un listado de las habitaciones disponible de acuerdo a su tipo
- Preguntar por el precio de una habitación de acuerdo a su codigo
- Preguntar por el descuento ofrecido a los clientes habituales
- Preguntar por el precio total para un cliente dado, especificando su numero de DNI, tipo de habitación y número de noches.
- Reservar una habitación especificando el número de la habitación, DNI y nombre del cliente.
- Eliminar una reserva especificando el número de la habitación

El administrador puede usar el programa para:

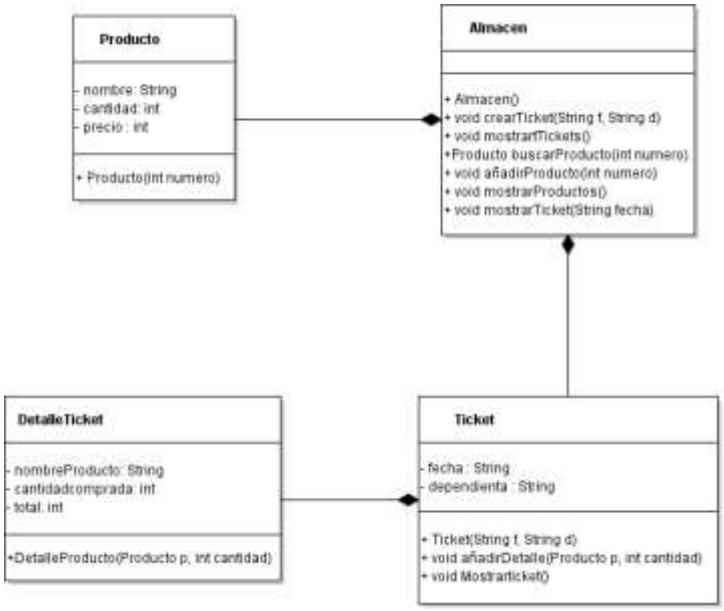
- Cambiar el precio de una habitación de acuerdo a su tipo
- Cambiar el valor del descuento ofrecido a los clientes habituales

Las habitaciones se almacenan com un conjunto implemetnado con la colección TreeSet mientras que los clientes los almacenamos en un conjunto implementado con la colección HashSet con independencia del tipo de cliente que sea (habitual o esporadico)

Ejercicio 2

Se pretende crear una aplicación para la gestión de las cajas de un supermercado.

Diagrama de clases



Clase Producto

- Atributos:
  - Nombre: String de la siguiente forma “productoX” con x un numero entero
  - Cantidad : valor aleatorio entre [1-50]
  - Precio : int valor aleatorio entre [1-100]
- Metodos
  - Producto(int numero): recibe como parámetro un numero y consigue de forma automática el nombre. Los valores para el precio y cantidad se generan de forma aleatoria

Clase DetalleTicket

- Atributos:
  - NombreProducto: string que almacena el nombre del producto que compramos
  - Cantidad: numero entero
  - Total : calcula a partir del precio * cantidad el importe total del producto comprado
- Métodos
  - DetalleTiket : constructor que recibe como parámetro un objeto producto. A partir de el sacamos el nombre y el precio para calcualr el total. Recibe como parámetro la cantidad comprada.

    Deberemos tener en cuenta los siguiente
    - El objeto producto no sea nulo
    - La cantidad comprado no debe superar a la cantidad de producto almacenada

Clase Ticket

Almacena una lista con los detalles de los productos comprados

- Atributos
  - String fecha
  - String nombreDependienta
- Métodos
  - Constructor: recibe como parámetro la fecha y el nombre de la dependienta
  - void añadirDetallePedido: recibe como parámetro un producto y la cantidad.
  - Void mostrarTicket : mestra lso datos del ticket y la lista de detalles. Finalmente calcula el importefinal del ticket sumando el importe de cada detalle

Clase Almacen:

Almacena dos conjuntos: productos y tickets

- Productos : utilizará el tipo de colección TreeSet
  Recordar que si almacenamos TreeSet<X> X tiene qu ser un tipo de clase que implemente el interfaz comparable
- Tickets: utilizara el tipo de colección HashSet
  Recordar que si almacenamos un HashSet<X> , X tiene que implmeentar el metodo hashCode y equals para comparar objetos correctamente

Metodos:

- Crearticket :recibmos los datos de la fecha y la dependienta y vamos añadiendo diferentes detalles de productos
- mostrarTickets: mostramos el conjunto de tickets
- buscarProducto(int numero) recibe como paramtro un numro , crea el nombre completo asociado a un producto y lo busca. Si nol o encuentra retorna null. Si loencuentra retorna el producto
- mostrarTicket(String fecha) recibe como parámetro una fecha y muestra el ticket con esa fecha

Ejercicio 3

El objetivo es desarrollar una aplicación para el control de las llamadas realizadas en una centralita

De las llamadas queremos almacenar:

- Id : valor único generado por el sistema de forma automática e incremental
- numeroOrigen : String de 9 caracteres
- numeroDestino : String de 9 caracteres
- duración en segundos

Existen dos tipos de llamadas : locales y provinciales

- Las llamadas locales tienen una tarifa única de 15 céntimos el segundo

- Las llamadas provinciales tienen una tarifa por segundo que depende de la duración:
    - Menos de 1 minutos: 30 céntimos/seg
    - Entre 1 y 5 minutos Franja 2: 25 céntimos/seg
    - Más de 5 minutos: 20 céntimos/seg

Todas las llamadas tendrán un método getImporte que retorna el valor del importe final de la llamada según su tipo y tarifa si aplica, multiplicado por su duración

Se pide:

- Crear una clase centralita que almacene 10 llamadas (5 locales y 5 provinciales) en un conjunto
- Mostrar las llamadas almacenadas en le conjunto
- Método registroDeLlamadas que recibe como parámetro un teléfono y busca las llamadas realizadas por ese numero(numeroOrigen) así como el importe total de las mismas.

Ejercicio 4

Gestión de los empleados de una empresa.
Representar mediante un diagrama de clases la siguiente especificación:

- La aplicación de gestión en concreto necesita almacenar información sobre empresas, sus empleados y sus clientes, en ambos casos debemos almacenar el nombre y la edad.
- Los empleados tendrán a mayores un sueldo bruto. Los empleados que tienen el cargo de directivo tienen una categoría, asi como un conjunto de empleados a su cargo.
- De los clientes además de la información mencionada anteriormente se debe almacenar el teléfono de contacto.
- La aplicación necesita mostrar los datos de empleados y clientes.