Excursión a Isla Mágica

Examen - 3er Trimestre

1º DAW - Programación

Mayo 27, 2025

Descripción del ejercicio

Los profesores Gabriel e Israel, con motivo de fin de curso, han decidido llevar a los alumnos de 1º de Desarrollo de Aplicaciones Web a una gran aventura, una excursión al parque temático Isla Mágica, ¡Una excursión muy merecida por el gran curso que han hecho!

Durante la jornada en el parque, los alumnos deberán resolver distintos retos de programación: desde gestionar entradas, controlar accesos a atracciones, realizar pedidos en restaurantes, participar en juegos, hasta registrar todo lo vivido.

1: Gestión de Entradas (2 puntos)

Cada visitante necesita una entrada válida para acceder al parque. El precio y tipo de entrada varían según su edad, altura, si tiene acceso VIP y la provincia de origen.

1. Crea una clase `Visitante` con los atributos: `nombre` (String), `edad` (int), `altura` (double), `esVip` (boolean), `provinciaOrigen` (String).
2. Crea un enumerado `TipoEntrada` con los valores: `INFANTIL`, `GENERAL`, `VIP`.
3. Crea una clase `Entrada` con los atributos: `visitante` (objeto de clase Visitante), `tipoEntrada` (TipoEntrada), `precio` (double).
4. Implementa en `Entrada` los métodos:
   - `calcularPrecio()`: Este método debe usar los atributos del visitante (`edad`, `altura`, `esVip`, `provinciaOrigen`) para devolver el precio de la entrada. Sigue estas reglas:
     - Si el visitante es VIP → precio = 0 €.
     - Si edad menor de 12:
       - altura menor de 1.20 m → 10 €
       - altura mayor igual de 1.20 m → 15 €
     - Edad 12–17:
       - General: 20 €
     - Edad mayor o igual 18:
       - General: 25 €
       - Si mide más de 1.85 m y es VIP: 20 €
     - Nota: Si el visitante es de Sevilla, tendrá un 15% de descuento adicional.
   - `esTipoValido()`: Este método debe validar si el tipo de entrada (`tipoEntrada`) es coherente con las características del visitante:
     - "Infantil": solo si edad menor de 12 años y altura es menor de 1.40 m
     - "General": si edad mayor o igual a 12 años o altura es mayor de 1.40 m, y no es VIP
     - "VIP": solo si `esVip == true`
   - `resumenEntrada()`: Devuelve una cadena de texto que resuma la información del visitante y la entrada, por ejemplo: "Nombre: Ana | Edad: 10 años| Tipo: Infantil | Provincia: Cádiz | Precio: 10,0 €"

2: Atracciones del Parque (1.5 puntos)

Las atracciones del parque tienen requisitos de acceso que dependen de la edad, altura, tipo de visitante, capacidad actual y si están en funcionamiento.

1. Crea una clase `Atraccion` con los atributos: `nombre`, `zona`, `alturaMinima`, `edadMinima`, `enFuncionamiento`, `requiereVip`, `capacidadMaxima`, `aforoActual`.
2. Implementa los métodos:
   - `puedeSubirse(Visitante v)`: Devuelve true si:
     - La atracción está funcionando.
     - La altura y edad del visitante son mayor o igual a los mínimos.
     - Si requiere VIP, el visitante debe serlo.
     - Si la zona es "Zona de Aventuras" y el visitante mide más de 2.00 m, devuelve false.
     - Si `aforoActual mayor o igual capacidadMaxima`, devuelve false.
   - `abrir()` / `cerrar()`: Cambian el valor de `enFuncionamiento`.
   - `mostrarEstado()`: Devuelve una cadena que incluya nombre, zona, estado, requisitos de edad y altura, y aforo actual.
   - `registrarSubida()`: Aumenta `aforoActual` si aún hay espacio.
   - `reiniciarAforo()`: Pone `aforoActual = 0`.

3: Pedidos en Restaurantes (1.5 puntos)

Cada visitante puede realizar un pedido en un restaurante. Se aplican menus, descuentos acumulativos y condiciones específicas por tipo y provincia.

1. Clase `Producto`: `nombre`, `precio`, `tipo` (TipoProducto).
2. Crea un enumerado `TipoProducto` con los valores: `BEBIDA`, `COMIDA`, `POSTRE`.
3. Clase `PedidoVisitante`, quien recibe en su constructor a un `Visitante` y tiene una lista de posibles `productos` (Lista de Producto).
4. Métodos:
   - `agregarProducto(Producto p)`: Añade el producto a la lista.
   - `calcularTotal()`: Calcula el precio total del pedido. Aplica:
     - Si hay al menos 1 bebida y 1 comida → -1.50 €
     - Si hay más de 4 productos → 5% de descuento al subtotal
     - Si el visitante es VIP y de Sevilla → 20% adicional
     - Añadir 10% de IVA al final del total
   - `resumenPedido()`: Devuelve una cadena con los productos, subtotal, descuentos, IVA y total final.

4: Juegos y Puntajes (1.5 puntos)

El visitante puede participar en juegos con distintos niveles de dificultad y restricciones VIP.

1. Clase `Juego`: `nombre`, `puntuacionMaxima`, `dificultad` (TipoDificultad), `soloVip`.
2. Crea un enumerado `TipoDificultad` con los valores: `FACIL`,`MEDIA`, `DIFICIL`)
3. Clase `Participacion`: `jugador` (Visitante), `juego`, `puntuacion`.
4. Métodos:
   - `esGanador()`: Devuelve true si:
     - Dificultad
       - Fácil → mayor o igual a 70% de puntuación máxima
       - Media → mayor o igual a 80%
       - Difícil → mayor o igual a 90%, o mayor o igual a 80% si el visitante es VIP
       - Si el juego es solo para VIP y el visitante no lo es → false
   - `mostrarResultado()`: Devuelve una cadena con nombre del jugador, juego, puntuación, y si ha ganado.

5: Registro de Visitas (1.5 puntos)

Se necesita guardar un registro de todos los eventos en un fichero CSV, cuyo nombre del fichero debe ser: islamagica-aaaa-mm-dd.csv. (`aaaa` es el año, `mm` es el mes y `dd` es el día actual.

1. Clase `RegistroVisitas`:
   - Cada línea del CSV debe tener: `fecha`, `hora`, `tipoEvento`, `descripcion`
2. Métodos estáticos:
   - `guardarEvento(String tipoEvento, String descripcion)`: Añade al archivo visitas.csv una línea con la hora actual.
   - `mostrarHistorial()`: Lee el archivo y muestra su contenido.
   - `filtrarPorFecha(LocalDate fecha)`: Muestra solo los eventos ocurridos en esa fecha.
   - `filtrarPorTipo(String tipoEvento)`: Muestra solo los eventos de ese tipo.

6: Clase Parque (2 puntos)

Se necesita una clase principal que simule todo lo que ocurre en Isla Mágica a lo largo de un día: generación de visitantes, emisión de entradas, acceso a atracciones, realización de pedidos, participación en juegos y registro de eventos.

1. Crea una clase `Parque` con atributos:
   - `List<Visitante> visitantes`
   - `List<Atraccion> atracciones`
   - `List<Pedido> pedidos`
   - `List<Juego> juegos`
   - `List<Participacion> participaciones`
2. Implementa en `Parque` el método:
   - `void simularDia()`: este método debe:
     - Crear varios visitantes de forma simulada.
     - Generar sus entradas y calcular el precio.
     - Registrar acceso a una o varias atracciones.
     - Generar un pedido por visitante y calcular totales.
     - Registrar participaciones en juegos.
     - Guardar todos los eventos relevantes con `RegistroVisitas`.
3. Puedes añadir más métodos como:
   - `generarVisitantesAleatorios(int cantidad)`
   - `mostrarResumenFinal()`: muestra cuántos visitantes hubo, cuántos subieron a atracciones, cuántos juegos se jugaron, etc.

7: Manejo de Excepciones (1 punto extra)

1. Crea las clases `ExcepcionProductoInvalido` y `ExcepcionPuntuacionExcedida`.
2. Úsalas en `Pedido` y `Participacion` para lanzar errores si:
   - El producto es nulo o su precio negativo.
   - La puntuación supera la puntuación máxima del juego.