En primer lugar, en el paquete de administración encontramos la clase Administrador. Esta clase tiene métodos muy útiles para gestionar el personal del parque. Por ejemplo, el método agregarEmpleado se encarga de añadir un empleado a la lista de empleados que se administran,; esta función recibe un objeto empleado y lo inserta. Además, el método consultarEmpleado recorre la lista y muestra información básica del empleado que se encuentra con el identificador indicado. Todo esto nos permite que la clase Administrador coordine de manera simplificada la incorporación y revisión del personal.

En el paquete de atracciones, la clase Atraccion es la base para representar cualquier atracción del parque. Los métodos de esta clase, principalmente getters y setters, permiten acceder y modificar información como el cupo, el número mínimo de empleados necesarios, el nivel, tipo y nombre de la atracción. A partir de esta clase derivamos dos especializaciones: AtraccionCultural y AtraccionMecanica. En la clase AtraccionCultural se encuentran métodos que permiten asignar y consultar la edad mínima permitida y detalles relacionados con la temporada , mientras que la clase AtraccionMecanica dispone de métodos que permiten establecer el peso y la altura mínimos, además de verificar si la atracción es adecuada para personas con discapacidad o si puede provocar vértigo.

En el paquetcione de clientes, la clase Cliente ofrece varias funciones importantes. Entre ellas, tenemos el método carnetSalud, el cual crea un registro en forma de mapa con información sobre la salud del cliente. Igualmente, el método tiqueteComprado se utiliza para registrar en su historial la compra de una entrada o tiquete, asociando al usuario los detalles de la transacción. Otra función esencial es cargarUsuariosDesdeArchivo, que se encarga de leer los datos de un archivo de texto, separar la información y crear instancias de Cliente que se guardan en un mapa, garantizando así la persistencia de la información.Otra funcion clave es mostrarTiposDeTiquetesComprados la cual se encarga de mostrar por cliente los clientes que se han comprado desde la interfaz dada.Esta información se encuentra almacenada en un mapa de mapas que almacena los tiquetes.

Para los empleados usamos el paquete de empleado, donde la clase empleado es la base para gestionar el personal y contiene métodos para agregar empleados a un mapa global, buscar empleados por nombre, y modificar datos como el turno, el salario o el lugar asignado. Por

ejemplo, el método cambiarLugarAsignado nos permite actualizar el área de trabajo de un empleado. Además, se incluye el método mostrarTodosEmpleados, que recorre el mapa y presenta los datos básicos de cada empleado en consola. Tambien se tiene la funcion getInfoPor Empleado la cual almacena la información dada con base en cada empleado por su nombre,es decir el, id, salario, rango etc. se guardan en un mapa donde la llave es el nombre del empleado Las clases derivadas, como cajero, cocinero y operadorMecanica, extienden esta funcionalidad añadiendo métodos propios. En cajero, se incorpora un método para obtener el punto asignado, en cocinero se incluye el método para obtener el certificado de alimentos, y en operadorMecanica se cuenta con un método que permite certificar nuevas atracciones en las que puede operar.

En el paquete encargado del parque de diversiones, la clase ParqueDeDiversiones maneja la carga y gestión de las atracciones. Su método cargarAtracciones lee línea por línea un archivo de texto, y en función del tipo de atracción, crea instancias específicas (ya sea de atracciones culturales o mecánicas). Este método no solo se encarga de crear las instancias correctas, sino que también gestiona posibles errores, como datos incompletos, y asigna valores por defecto cuando es necesario. Además, se encuentra el método mostrarAtracciones, que permite visualizar el catálogo de atracciones filtrado por tipo , y el método getNivelExclusividad, que consulta el nivel asignado a cada atracción.

El paquete de tiquetes define cómo se manejan las entradas que los clientes compran para disfrutar del parque. La clase abstracta Tiquete agrupa ciertos métodos, como el de generar un identificador único, obtener el tipo, la fecha de compra o comprobar el uso del tiquete mediante validarUso. Ya luego, tenemos especializaciones como TiqueteConRango y TiqueteTemporada, que además de heredar estos métodos, incluyen funciones específicas. Por ejemplo, en TiqueteConRango, existe el método esValidoEnFecha, que convierte la fecha de uso a un formato que permite comparar si el tiquete es válido para una fecha especificada.

De igual modo, TiqueteTemporada tiene métodos para comprobar si la fecha en que se intenta usar la entrada se encuentra dentro de un rango definido, haciendo posible validar la vigencia de la entrada a lo largo del tiempo. Por otro lado, la clase EntradaIndividual representa un tiquete

para una atracción específica, y la clase VentaOnline gestiona la venta de tiquetes tanto de manera virtual como presencial.

Finalmente, toda la integración del sistema se encuentra en la clase Main, ubicada en el paquete consola. Esta es la parte que interactúa directamente con el usuario a través de un menú. En el menú de Main se reúnen todos los métodos explicados anteriormente. Otras opciones en el menú permiten cargar datos desde archivos, vender tiquetes que en base a una información dada por el usuario asigna un tiquete al usuario que se almacena en un mapa . Al finalizar la ejecución, Main se encarga de llamar a métodos para guardar toda la información en archivos .txt, asegurando que la próxima vez que se inicie el programa la información se recupere correctamente.

DECISIONES A LA HORA DE REALIZAR E IMPLEMENTAR EL CÓDIGO:

Inicialmente, se decide iniciar a desarrollar el código implementando las clases dadas por el uml, se decide posterior a esto crear una clase main que se encargue de dar una interfaz por la cual se pueda usar el código y ejecutar los requerimientos funcionales,los cuales son explicitamente los siguientes:

1).un programa para la compra de tiquetes que simula el proceso de compra. Este programa crea un cliente al cual le solicita la selección del tipo de pase (ya sea un tiquete con rango, de temporada o una entrada individual) y procede a validar la información.Además, se incluye la simulación del proceso de pago, donde se calcula el costo total.

2). Otro programa se enfoca en la gestión de atracciones. En esta prueba se simula la carga de datos desde un archivo o mediante la creación directa de objetos que representen atracciones mecánicas o culturales. El sistema muestra un catálogo completo de atracciones, mostrando al cliente las diferencias entre esas atracciones con más restricciones que otras. (Cabe mencionar que en este caso se decidió dar la carga mediante un archivo).

3). De igual manera, tenemos un programa para la gestión de empleados y turnos, cuyo objetivo es demostrar la correcta asignación y funcionamiento del personal. En esta prueba se crean empleados de distintos roles y se asignan a turnos específicos. Se valida que los empleados estén en sus respectivas áreas de servicio de acuerdo con las restricciones del sistema, por ejemplo, un cajero debe estar asignado a una taquilla.

4). Otro aspecto crucial que vimos es la validación y uso de tiquetes, por lo que se diseñó un programa específico para este fin. En este caso, se simula el proceso en el que un visitante

intenta acceder a una atracción presentando su tiquete. El sistema, por ejemplo, verifica si el tiquete se encuentra dentro del rango de fechas con validez.
5)Finalmente, disponemos de un programa que prueba la asignación de empleados a los lugares de servicio, como cafeterías, taquillas o tiendas. Esta simulación muestra cómo el sistema asigna de manera automática y segura a los empleados de acuerdo con las funciones requeridas en cada establecimiento. Por ejemplo, se verifica que cada taquilla cuente con al menos un cajero y que las cafeterías dispongan de un cocinero certificado.(ESTE REQUERIMIENTO ESTÁ IMPLÍCITO EN LA FUNCIÓN DE REGISTRO DE EMPLEADOS Y GESTIÓN DE EMPLEADOS).

MANEJO DEL PROGRAMA:

Para el manejo del programa es necesario tener en cuenta dos aspectos fundamentales,primero la carga de datos la cual al ingresar la dirección del archivo no debe ir en comillas,seguido a esto es clave tener en cuenta que al guardar el archivo quedan almacenados en la carpeta proyecto1,también de igual manera para ejecutar los requerimientos funcionales es necesario hacerlo desde el package consola y la clase main donde se encuentran la interfaz para operar la aplicación.

RECURSOS USADOS:

Los recursos usados para poder implementar el código giraron en torno a consultas que permitieron implementar y usar métodos que ayuden a que el código sea más óptimo y mucho más fácil de implementar,se usó principalmente la ayuda de stackoverflow y videos en Youtube,también para los datos que se usaron de prueba para cargar la aplicació, se usó IA ya que crearlos a mano era una labor muy tediosa al ser datos aleatorios.Tambien usamos otras herramientas para corregir errores que no entendíamos cómo solucionarlos dentro del código.