Evidencia 1 Estructura de Datos

Igor Sung Min Kim Juliao
AL02829189
Juan Pablo Hernández Parra
AL02887299
Mauricio Estrada de la Garza
AL02976904

Clases de Objetos:

La clase cliente tiene como atributos el ID de tipo int, y la membresía de tipo String, también incluye los getters y el constructor. La clase de product tiene 1 solo atributo llamado type de tipo String, es para almacenar el tipo de categoría del producto, incluye un constructor y el getter de la clase.

Main:

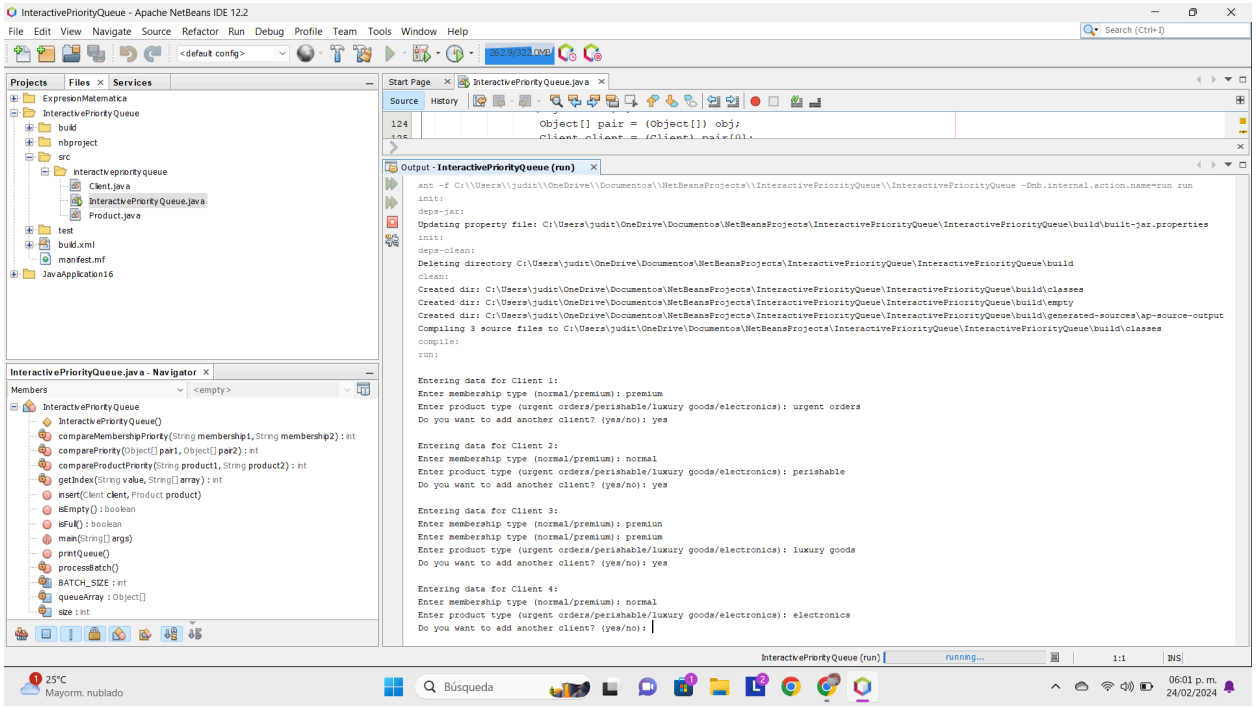
En el main se instancia un objeto de la clase principal, después se declara una variable de tipo int llamada clientCount que es igual a 0 por default, esta variable representa el número de cliente, dentro de un ciclo While se ejecutará el programa principal siempre y cuando el ciclo se declare como True. Se utiliza un ciclo do-while para que se ejecute el dato de entrada y luego se determine si es true o false, el primer dato que pide el programa es si el cliente será normal o premium, y si no se proporciona una de las 2 respuestas, entonces se declara como falso y se vuelve a repetir la pregunta. Lo mismo se aplicó para el tipo de productos, también para los 2 datos de entrada se hizo uso de operadores lógicos(AND y NOT). Una vez proporcionados los datos de entrada, se crean 2 objetos, uno de la clase cliente que se le agregan como argumentos las variables de clientCount y membership, y el otro es de la clase Producto que se le pasa como argumento la variable productType, finalmente se meten los 2 objetos en la cola de prioridad, después hay una condicional que siempre y cuando la prioridad tenga elementos, se imprima en consola, después al final se pregunta si se desea agregar otro cliente a través de Strings, si el usuario escribe "no" entonces ya se rompe el ciclo While principal.

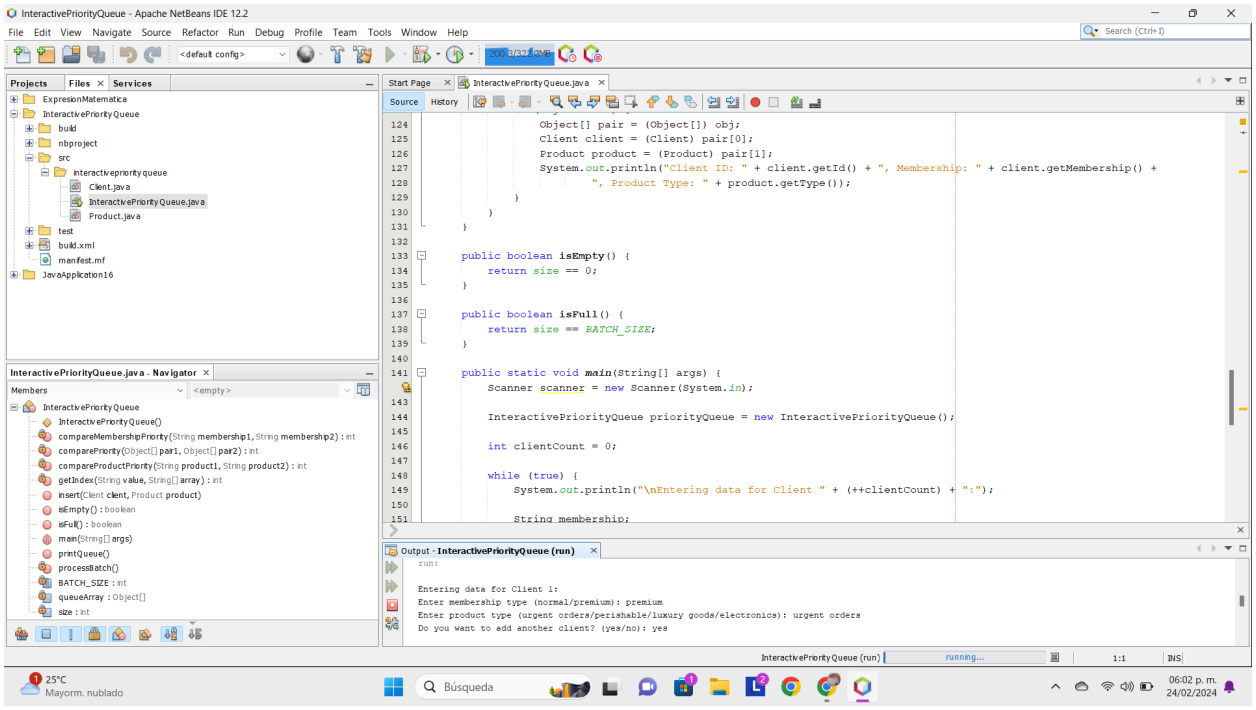
Clase principal:

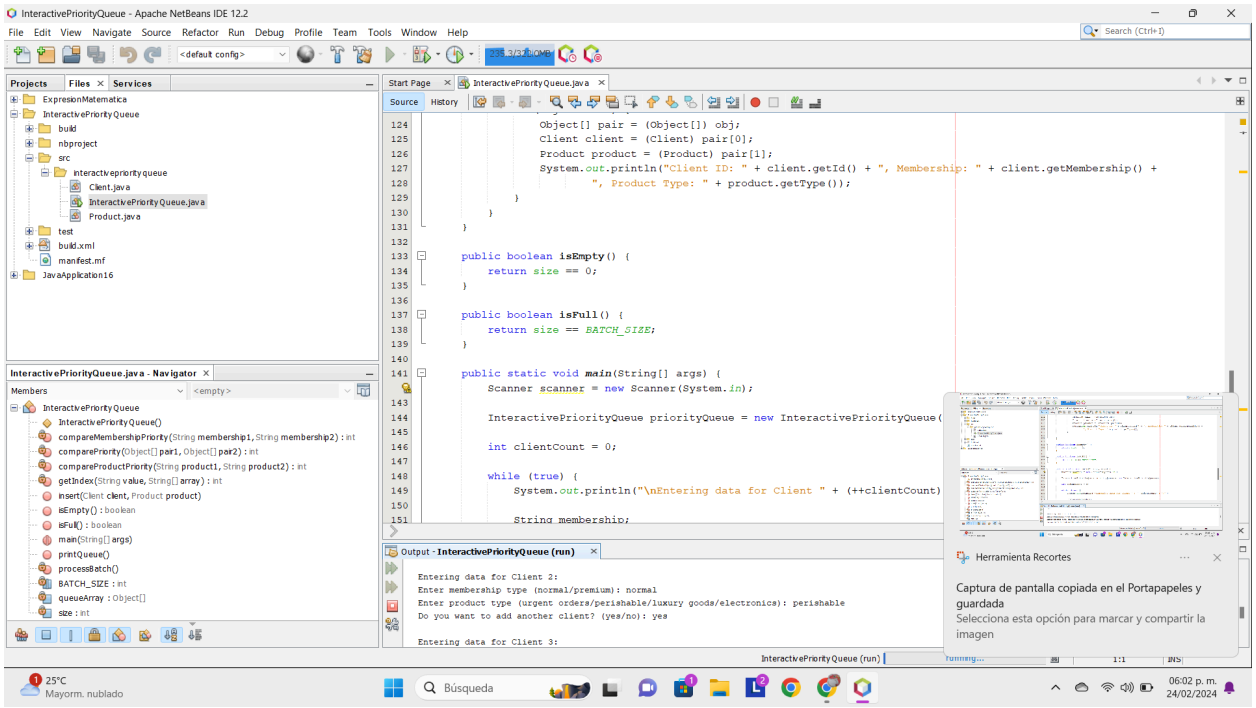
La clase se llama InteractivePriorityQueue, tiene 3 atributos, uno de tipo int llamado BatchSize que es para definir la cantidad de elementos que se guardaran en la cola, en este caso será 6. el segundo atributo es un arreglo de objetos donde se guardaran las instancias y el tercer atributo es del tamaño actual de la cola de prioridad. En el constructor, el arreglo es igual a un objeto con un tamaño de 6 elementos (batch size) y el size por default será igual a 0. El primer método se llama "isEmpty()" y regresa el tamaño de la cola igual a 0, el segundo método se llama "isFull()" y regresa el tamaño igual al Batch Size, o sea 6. El tercer método es para imprimir la cola de prioridad, se iteran sobre los objetos guardados en el arreglo, y mientras la cola no sea nulo, se imprima la información del pedido (Cliente y producto que pidió, etc) el siguiente método es para el dato de salida final en el main, sale un texto llamado "processing batch" y además se manda a llamar el método anterior. Luego se realizó un método de tipo int para encontrar los índices en los arreglos, pasamos una variable y un arreglo de tipo String como parámetros, y dentro del ciclo For, mientras el índice sea menor al tamaño del arreglo, se iterara el ciclo y si el parámetro value es igual al índice del arreglo pasado como parámetro, se regresa el índice, de lo contrario el -1 representa que no se encontró el valor en el arreglo. Luego se hizo un metodo para comparar la prioridad de los productos, se pasan como parámetros de entrada unas variables de tipo String llamado producto 1 y producto 2, dentro del método se declara un arreglo de tipo String donde almacena todos los productos, luego se definen 2 variables de tipo int para los índices de los productos y equivalen al método de getIndex y se pasan como argumentos los parámetros del método actual y el arreglo de PriorityOrder, y al final regresa una función llamada Integer compare y se pasan como argumentos las variables de tipo int declaradas, y el índice que sea mayor en valor, tendrá más prioridad en la cola, lo mismo se realizó para el método se comparar membresias. Ya después se implementó un método de tipo int para comprar las prioridades, se pasan como parámetros 2 arreglos de objetos, después se definen 2 objetos de la clase Cliente y producto, las instancias de la clase cliente se guardan en la posición 0 de los arreglos y las instancias de la clase producto se guardan en la posición 1 de los arreglos, luego se define una condicional donde si el tipo de producto es igual a "urgent orders", se regresa 1 y eso indica que tendrá mayor prioridad en la cola, y si regresa -1, aunque el segundo objeto agregado sea también "urgent orders" tendrá menos prioridad que el primer objeto agregado, más abajo se declara una variable de tipo int llamada productPriority y equivale al método de comparar prioridad de productos, se pasan como argumentos los getters de los productos y luego en la condicional si la prioridad del producto No equivale a 0, entonces regresa la prioridad del producto que sería 1 o -1, lo mismo sucede con la membresía de prioridad donde también está declarado el método en una variable de tipo int con la diferencia que no se implementó una condicional para esa parte del algoritmo. El último método

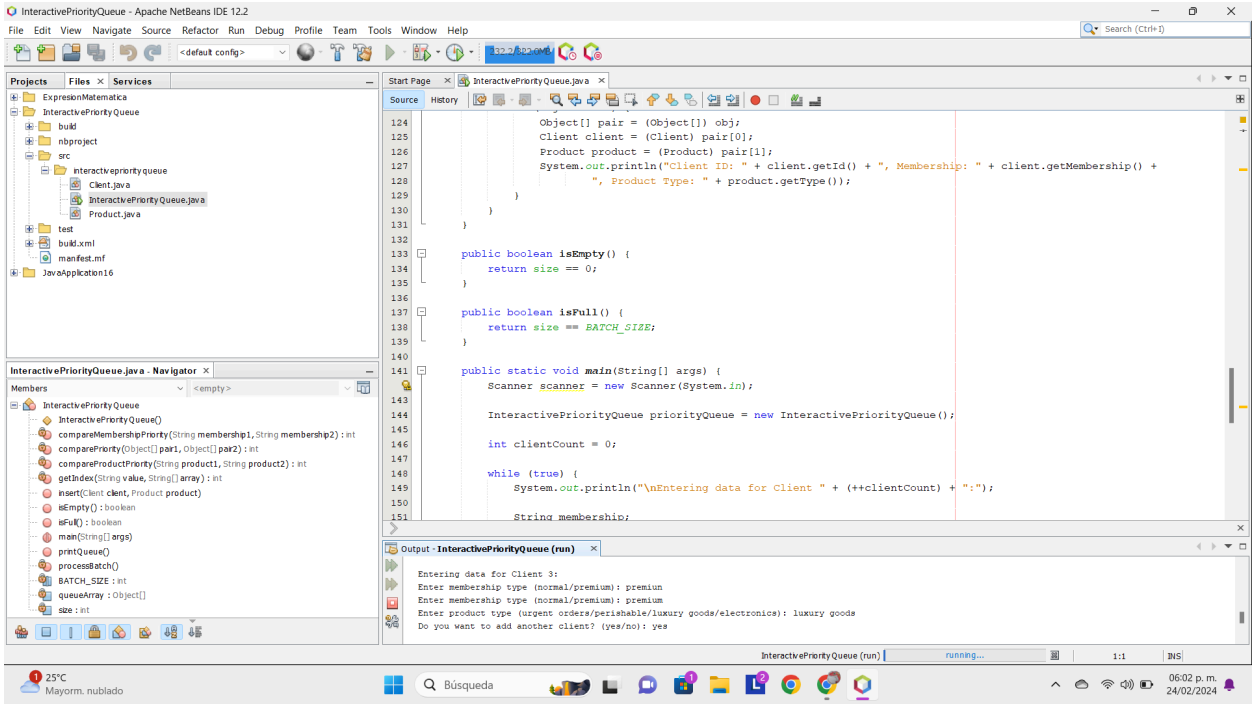
es para insertar elementos en la cola de prioridad, se pasan como parámetros objetos de las otras 2 clases creadas, en el primer IF, si la cola de prioridad contiene elementos, entonces se manda a llamar el método para el dato de salida, luego en la segunda condicional si la cola está vacía O el residuo del tamaño y el BatchSize es igual a 0, entonces el arreglo es igual a un Objeto de arreglo que tiene como argumentos los parámetros proporcionados en el método, y el tamaño se va iterando por 1. En el Else se crea un arreglo de tipo Objeto que tiene como argumentos los parámetros proporcionados y luego en el ciclo For mientras el índice sea igual al tamaño de la cola - 1 y mientras sea mayor o igual a cero, y el índice decrementa, entonces se itera con los elementos en el arreglo del Objeto. Luego hay una condicional donde si el método de comparar prioridad que tiene como argumentos los 2 últimos objetos declarados es mayor a 0, entonces se imprime las iteraciones de los elementos, en el else solamente se termina el algoritmo, luego se declara que el primer objeto creado de este último elemento es igual al arreglo y el tamaño incrementa por 1, al final del método si el tamaño y el batch size son iguales, se imprime efectivamente el dato de salida.

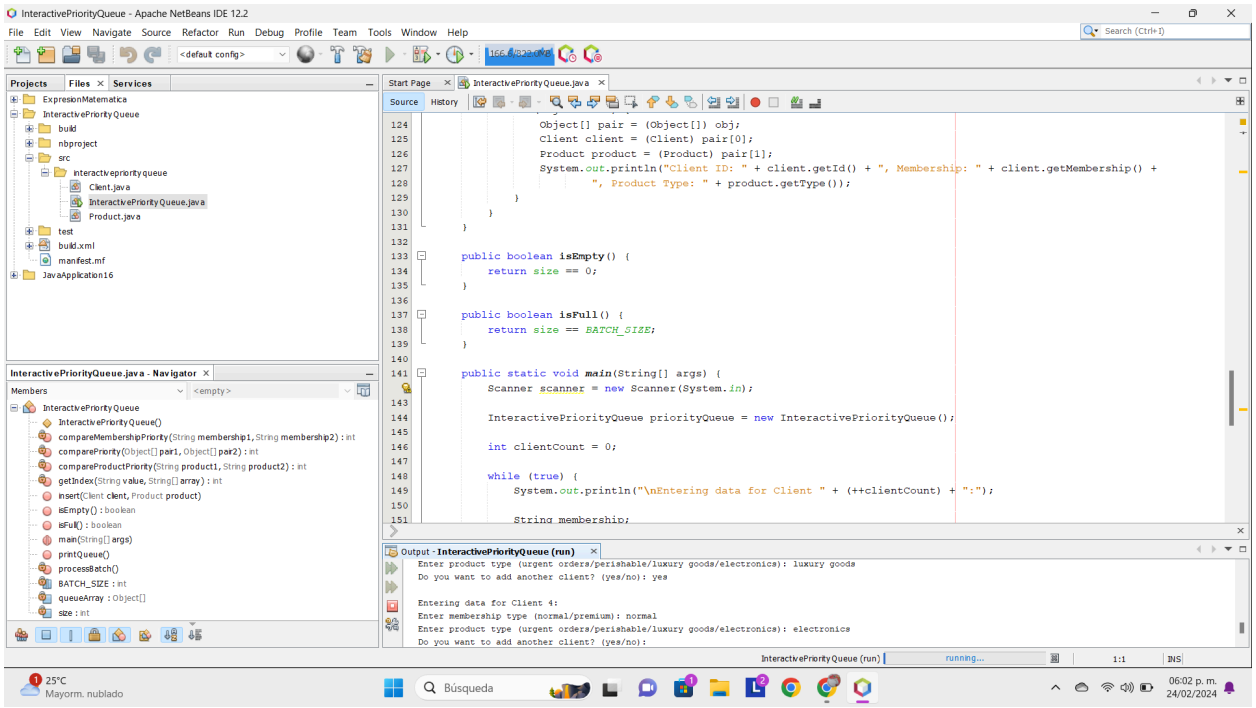
Pruebas:




InteractivePriorityQueue - Apache NetBeans IDE 12.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Projects Files Services
ExpresionMatematica
InteractivePriorityQueue
build
nbproject
src
interactivepriorityqueue
Client.java
InteractivePriorityQueue.java
Product.java
test
build.xml
manifest.mf
JavaApplication16
InteractivePriorityQueue.java - Navigator
Members
InteractivePriorityQueue
InteractivePriorityQueue()
compareMembershipPriority(String membership1, String membership2) : int
comparePriority(Object[] pair1, Object[] pair2) : int
compareProductPriority(String product1, String product2) : int
getIndex(String value, String[] array) : int
insert(Client client, Product product)
isEmpty() : boolean
isFull() : boolean
main(String[] args)
printQueue()
processBatch()
BATCH_SIZE : int
queueArray : Object[]
size : int
Start Page
InteractivePriorityQueue.java
Source History
Output - InteractivePriorityQueue (run)
run:
Entering data for Client 1:
Enter membership type (normal/premium): premium
Enter product type (urgent orders/perishable/luxury goods/electronics): urgent orders
Do you want to add another client? (yes/no): yes
InteractivePriorityQueue (run) running...
1:1 INS
25°C Mayorm. nublado
Búsqueda
06:02 p. m. 24/02/2024


InteractivePriorityQueue - Apache NetBeans IDE 12.2
Herramienta Recortes
Captura de pantalla copiada en el Portapapeles y guardada
Selecciona esta opción para marcar y compartir la imagen
Output - InteractivePriorityQueue (run)
Entering data for Client 2:
Enter membership type (normal/premium): normal
Enter product type (urgent orders/perishable/luxury goods/electronics): perishable
Do you want to add another client? (yes/no): yes
Entering data for Client 3:
25°C Mayorm. nublado
Búsqueda
06:02 p. m. 24/02/2024





Reflexión Individual - Mau: Mientras estuve creando mi propia version del codigo, la verdad si me estuve batallando para poder implementar ArrayLists y Colas de Prioridad sin necesidad de las librerías, yo pienso que no viene tanto al caso aprender implementar colecciones desde 0 si contamos con librerías y nos ayudan a reducir

código, que a final de cuentas en un trabajo como desarrollador de software nos pediran hacer codigo mas limpio para reducir el uso de la memoria en los servidores.

-Igor: Yo creo que con lo que más batalle, fue que como no tenía la librería de Arrays, tuve que hacer todo a través de índices, y el batch size ya siendo fijo fue un poco más difícil de utilizar, justamente porque, no podía crecer la cola, entonces decidí resolverlo reiniciando la cola después de mostrar el print de la prioridad de las colas. En sí al hacer este ejercicio me di cuenta que hay varias cosas que aún no entiendo bien de las colas y las colas priorizadas, como el cómo es que funcionan específicamente, en cuanto a las adiciones de nuevos elementos, porque es que pueden añadirse infinitamente, como delimitar o restringir el rango o número de elementos a reorganizar, etc, y este tipo de ejercicios donde intento hacer algo nuevo como esto y en el que nos hacen pensar en ideas de cómo hacer el proyecto, y dejarnos jugar con el código como este tipo de retos me hace darme cuenta que a pesar de que conozco los conceptos básicos, hay muchas más cosas que no se hacer, o que nunca siquiera se me ocurrieron que se podrían implementar a partir de estos mismo conceptos básicos, y se me hizo algo muy innovador y divertido.

Juan Pablo:lo más difícil de realizar el código y que sin duda fue un verdadero desafío fue la ausencia de uso de librerías tales como Array, aparte que hubo un problema con el batch size ya que llegada a los 6 casos sucedía un error, pero mis compañeros lograron resolver el problemas, fue muy divertido el probar la interacción del código, el poner diferentes casos entre los tipos de membresías, productos y el seguir con otro cliente.