“ Caso de los Números adyacentes”

Integrantes:

Antonia Paredes (21194240121)

Leandro Matamoros (20978573021)

Docente: Samuel Sepulveda

Asignatura: Programación orientada a objetos

Fecha: 22 de septiembre de 2023

Índice

Introducción

El siguiente informe aborda un programa Java denominado "Producto de Números Adyacentes", diseñado para encontrar el producto máximo entre números adyacentes en un arreglo de números enteros. Este programa permite al usuario especificar el tamaño del arreglo, generando números aleatorios para su posterior análisis. Durante el desarrollo de este proyecto, se ha prestado especial atención a la implementación de pruebas unitarias utilizando JUnit y la gestión de errores mediante el uso de excepciones. A lo largo de este informe, se detalla el tiempo real consumido en cada etapa, los pasos seguidos, un auto-análisis y comentarios de la experiencia.

Descripción de caso

El escenario planteado requiere la creación de un método denominado productoAdyacentes, el cual será utilizado para encontrar el mayor producto entre números adyacentes en un arreglo de enteros proporcionado como entrada.

Para ilustrar este desafío, se presenta el siguiente ejemplo:

Dado el arreglo de enteros: {1, -4, 2, 2, 5, -1}, se busca determinar cuál es el mayor producto de números adyacentes presente en el conjunto. En este caso específico, el mayor producto se obtiene al multiplicar los números 2 y 5, resultando en un valor de 10.

Es esencial tener en cuenta que este método debe ser capaz de operar de manera eficiente en arreglos de diversas longitudes y con valores que oscilan entre -1000 y 1000, abarcando un amplio espectro de escenarios posibles.

El éxito de esta implementación radica en la capacidad de desarrollar una solución robusta y eficaz que garantice la correcta identificación y cálculo del producto máximo de números adyacentes, cumpliendo con las buenas prácticas de programación y contemplando la gestión de posibles errores durante su ejecución.

Análisis del caso

Se comenzó leyendo lo solicitado en el caso, luego de un tiempo de análisis se determinó lo necesario para para poder cumplir con lo solicitado, se determinó los parámetros de entrada, los valores de retorno y los principales métodos a utlizar.

Los parámetros de entrada son el largo del array (int) y los valores que llenan este array (int); array que es la entrada del método multiplicador (productoAdyacentes).

El valor de retorno es el mayor de los multiplicados (valor de tipo int).

El programa comienza solicitando al usuario el tamaño del arreglo, asegurándose de que esté dentro de los límites establecidos. Luego, genera y llena el arreglo con números aleatorios, imprime su contenido y calcula el producto máximo entre números adyacentes en el arreglo.

Inicialmente se consideraron los siguientes Pasos/Métodos principales a seguir:

- Método validarEntrada()
- Método largoArray()
- Método llenarArray()
- Método generarRandom()
- Método productoAdyacentes()
- Método identificarMayor()

Estos se desarrollaron siguiendo un orden secuencial y evolucionaron durante la implementación.

Tiempo estimado: 10 minutos.

Tiempo real utilizado: 9 minutos aproximadamente.

Implementación de la solución

El código se implementó utilizando el lenguaje de programación Java. Cada función se diseñó para realizar una tarea específica, lo que facilita el mantenimiento y la comprensión del programa. Se ha prestado especial atención a la implementación de pruebas unitarias utilizando el marco de pruebas JUnit para garantizar la calidad del código y su comportamiento correcto.

Los métodos utilizados son los siguientes:

- iniciar()
- entradaUsuario()
- validarLargoArreglo()
- generarArregloVacio()
- generarRandom()
- llenarArreglo()
- imprimirArreglo()
- productoAdyacentes()

Inicialmente se consideraron menos métodos y se agregaron los necesarios durante el desarrollo del código para mantener la unicidad de estos y las buenas prácticas.

Tiempo estimado: 1 hora y 30 minutos

Tiempo real utilizado: Aproximadamente 2 horas y 20 minutos.

Pruebas unitarias

Se realizaron 4 casos de pruebas para el método productoAdyacente, considerando que los valores ingresados al arreglo vienen dados por el método generarRandom los casos fueron:

1. testProductoAdyacenteConCeros(): En este test se ingresó un arreglo donde todos los valores ingresados son 0, el valor esperado de retorno es 0 el cual se retorna como se esperaba.

2. testProductoAdyacenteMultiplesPares(): El arreglo ingresado son solamente 2 números repetidos en varias ocasiones, donde todas las multiplicaciones tienen el mismo valor.

3. testProductoAdyacenteRepetidos(): Se tiene un arreglo de un solo número repetido muchas ocasiones. Como ejemplo se utilizó el número 9 un total de 12 veces, se espera como resultado 81, que es el valor que finalmente retorna.

4. testProductoAdyacenteValoresPositivosNegativos(): El arreglo solamente tiene una mezcla de números negativos y positivos, donde todas las multiplicaciones son negativas, el método es capaz de elegir correctamente el número mayor entre los negativos.

Tiempo estimado: 50 minutos.

Tiempo real utilizado: 45 minutos.

Excepciones

1. InputMismatchException: Considerando que el usuario ingresara un valor que no fuera un entero para establecer el largo deseado del arreglo, se implementó un try-catch solicitando un valor entero, para evitar que el código presentase errores.

2. Valor fuera de rango: Para evitar que el arreglo fuera menor a 2 o mayor a 20, según la condición “2 ≤ arreglo.length ≤ 20” solicitada, se implementó un método que verifica que el valor ingresado por el usuario se encuentra dentro de este rango. (Illegalargumentexception).

3. ArrayIndexOutOfBoundsException: Posible excepción en caso de que el método Multiplicación de adyacentes intentase multiplicar un arreglo de un largo menor a 2, este problema fue manejado gracias al método mencionado en el punto anterior y la condición de que los arreglos deben ser de un largo igual o mayor a 2; por lo que no se incluyó un try-catch para esta excepción.

4. Valores del arreglo: Para confirmar que los valores del arreglo creados de forma aleatoria cumpliesen con la condición “-1000 ≤ arreglo[i] ≤ 1000” solicitada, se implementó un método para generar números aleatorios sólo dentro de este rango, con los que más tarde se llena la matriz. (Illegalargumentexception).

Experiencia

En retrospectiva, al analizar el tiempo dedicado a las diversas etapas de este proyecto, hemos observado que el análisis del problema inicial se completó de manera más rápida de lo que inicialmente habíamos anticipado. Sin embargo, reconocemos que subestimamos la importancia de considerar exhaustivamente ciertos métodos y enfoques en esta fase. Esta falta de consideración inicial se tradujo en un aumento en el tiempo requerido para la implementación del código, ya que tuvimos que abordar aspectos que no se habían previsto de manera completa en la etapa de análisis. Este hallazgo subraya la importancia de realizar un análisis completo y detenido en las etapas iniciales del desarrollo de proyectos, ya que puede ahorrar tiempo y esfuerzo en fases posteriores.

Conclusión

En resumen, el proyecto "Producto de Números Adyacentes" ha sido una valiosa experiencia que nos ha permitido aprender importantes lecciones en cuanto al desarrollo de software. Hemos identificado la relevancia crítica de un análisis exhaustivo y detallado en las etapas iniciales del proyecto, ya que esto puede reducir considerablemente los desafíos y el tiempo necesario en las fases posteriores. La implementación de pruebas unitarias utilizando JUnit ha demostrado ser una práctica efectiva para garantizar la funcionalidad del código y su confiabilidad. Asimismo, la gestión de excepciones ha mejorado la experiencia del usuario al hacer que el programa sea más resistente a errores. En conclusión, este proyecto ha enriquecido nuestras habilidades de desarrollo y nos ha preparado para futuros desafíos en la creación de software de calidad.