



---

# UNIDAD 4. ARREGLOS

**MATERIA: Programación orientada a objetos I**

---



5 DE JUNIO DE 2020

Guadalupe García Vázquez  
MATRICULA: ES1921008556  
Desarrollo de Software

**Docente: Claudia Erika González Gómez**

## Actividad 2. Operaciones con arreglos

## Contenido

<b>Introducción</b> .....	3
Investigación.....	4
Desarrollo Caso de estudio .....	7
Ejecución del programa .....	11
<b>Conclusiones</b> .....	13
<b>Bibliografía</b> .....	14

## Introducción

En el siguiente trabajo se abordará de nuevo el tema de los arreglos en Java, pero ahora se hará la utilización de arreglos unidimensional junto con un multidimensional.

Recordemos que los arreglos en Java, también conocidos como arrays, agrupan valores del mismo tipo guardados en una misma variable y se puede acceder de manera independiente.

La principal práctica en este programa será el array multidimensional.

Un **array multidimensional** son más de un **array unidimensional** unidos, para que te hagas una idea, es como si fuera una tabla. Se crea igual que un **array unidimensional**, solo hay que añadir un corchete más con un tamaño en la definición del **array**. Veamos un ejemplo:

```
1      public class PruebaApp {  
2          public static void main(String[] args) {  
3              //Definimos un array de 3 filas x 5 columnas  
4              int array[][]=new int [3][5];  
5          }  
6      }
```

Como vemos, añadimos un corchete más después del nombre y del tamaño. El primer corchete son las filas y el segundo son las columnas.

## Investigación

1. Considerando la investigación anterior y la lectura que te proporciona el material de estudio para la realización de esta actividad.
2. Utilizarás un arreglo de una dimensión y otro arreglo multidimensional.
3. Implementarás las operaciones básicas de lectura, recorrido y asignación en ambos arreglos para el siguiente caso de estudio:

## Caso de Estudio

Bonne Ice es una empresa que se dedica a la venta de productos de helado, para ello ocupa las franquicias en diferentes puntos de la ciudad, la operación que realiza es la siguiente:



- Los vendedores del producto (sencillos, dobles, bonisote, megas), se surten en la franquicia que les corresponde según lleguen, su horario es variable,
- **se requiere se emplee una estructura del tipo arreglo unidimensional** para capturar los **nombres de los vendedores** y
- un **arreglo multidimensional para capturar las cantidades de producto que cargan que deben ser de los 4 productos** mencionados entre paréntesis. Ocupando esa misma estructura, obtener los totales de productos llevados por los vendedores por categoría.
- Permitir un menú como el siguiente para la actividad con las siguientes opciones para el usuario:

Bonne Ice Registro de Cargas Versión 1.0

- 1.- Capturar las cargas del vendedor.
- 2.- Mostrar las cargas de todos los vendedores.
- 3.- Obtener totales de carga por categoría de producto.
- 4.- Salir del programa.

Ejemplo de la actividad:

Arreglo 1		Arreglo 2			
Vendedor		Sencillo	Doble	BonIsote	Mega
Cesar		30	30	50	20
Alex		20	40	50	20
Total		50	70	100	40
Categoría					

Responde las siguientes preguntas:

¿Conoces los métodos de ordenación?

Los métodos de ordenación: es la operación de arreglar los elementos de un determinado vector en algún orden secuencial de acuerdo a un criterio de ordenamiento

Los principales métodos de ordenación son

1) Bubble Sort (Ordenamiento Burbuja):

Es el algoritmo de ordenamiento más sencillo de todos, conocido también como método del intercambio directo, el funcionamiento se basa en la revisión de cada elemento de la lista que va a ser ordenada con el elemento siguiente, intercambiando sus posiciones.

2) Quick Sort (Ordenamiento Rápido):

Es el algoritmo de ordenamiento más eficiente de todos, se basa en la técnica de "Divide y Vencerás", que permite en promedio, ordenar  $n$  elementos en un tiempo proporcional a  $n \cdot \log(n)$ .

3) Heap Sort (Ordenamiento por Montículos):

Es un algoritmo de ordenamiento no recursivo, no estable, consiste en almacenar todos los elementos del vector a ordenar en un montículo (heap), y luego extraer el nodo que queda como nodo raíz del montículo (cima) en sucesivas iteraciones obteniendo el conjunto ordenado.

#### 4) Ordenación por Selección

El proceso de la selección es el siguiente:

- Captura el Número más pequeño (si se desea ordenar los elementos de menor a mayor) o el número más grande (si se desea ordenar de mayor a menor).
- El número capturado es colocado en la primera posición, teniendo en cuenta que un Array empieza desde la posición Cero.
- El Proceso se repite para todos los datos sobrantes hasta llegar al último de ellos.
- Finalmente, los datos quedan ordenados ya sea en forma ascendente o descendente.

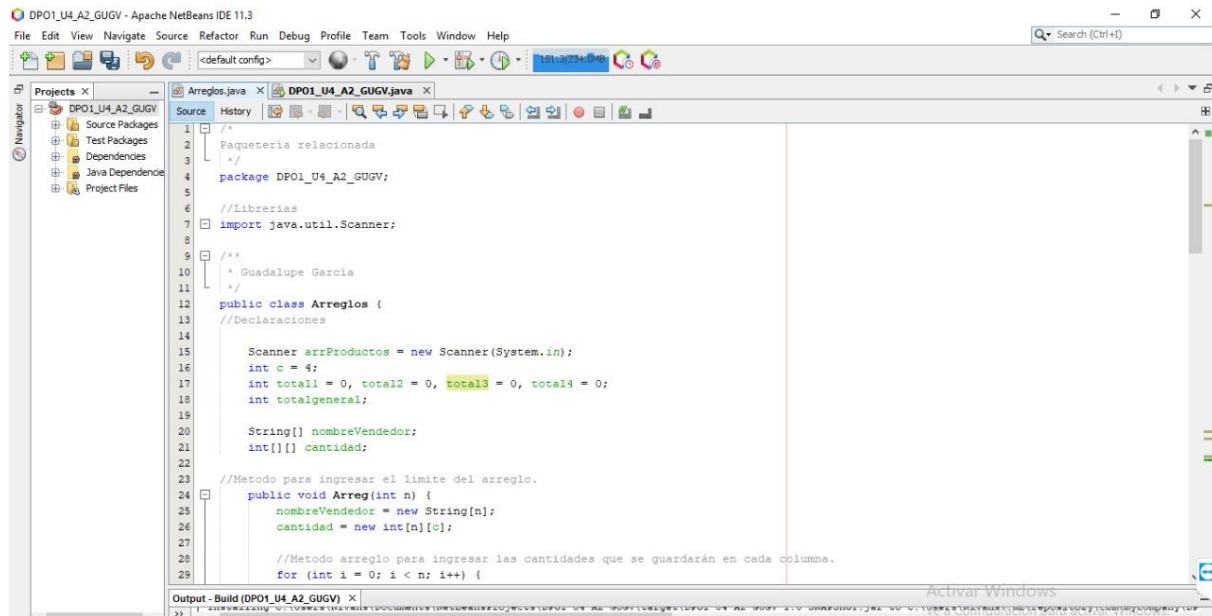
¿Cuál emplearías para esta actividad y por qué?

La ordenación por selección, ayudaría en las cargas generales, para saber cuál es el producto que más de carga y el que menos.

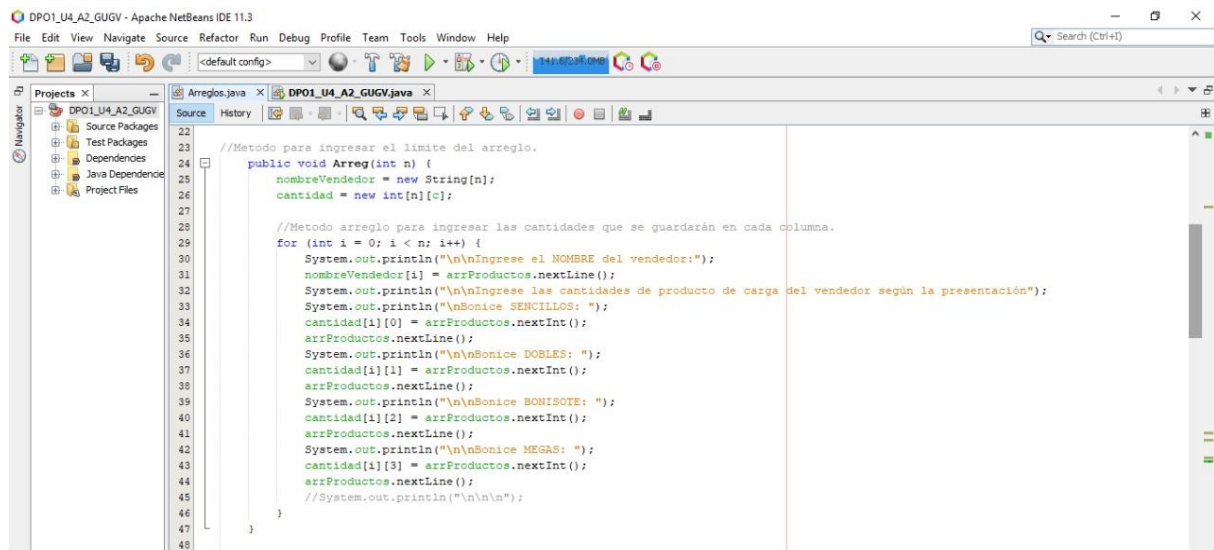
Así cómo los proveedores distinguir el comportamiento de sus cargas.

## Desarrollo Caso de estudio

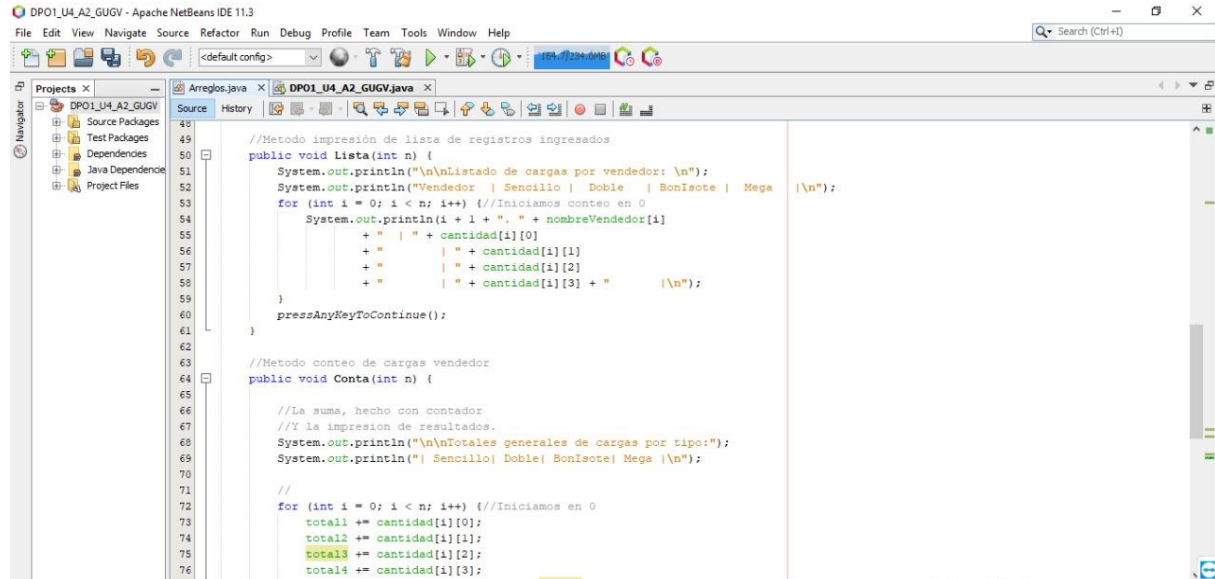
Clase arreglos, en ella se desarrolla el código de los elementos para el registro de la cantidad de producto que cargan los vendedores



Ingreso de código para pedirle al usuario el ingreso de cantidades, de acuerdo a la carga del tipo de Productos



Código para imprimir la pantalla, de la lista de la carga por vendedor

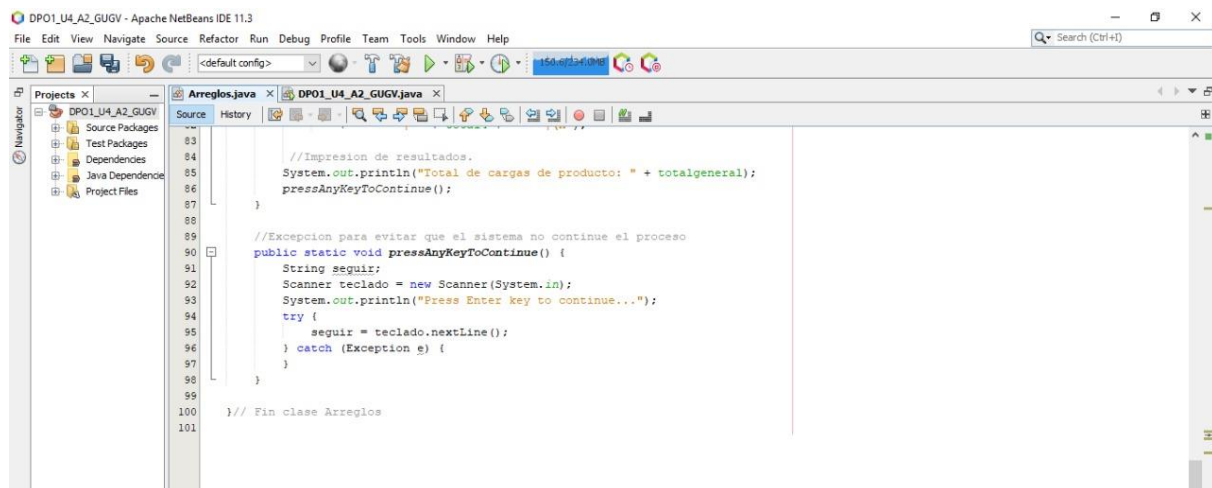


```

48 //Metodo impresion de lista de registros ingresados
49
50 public void Lista(int n) {
51     System.out.println("\n\nListado de cargas por vendedor: \n");
52     System.out.println("Vendedor | Sencillo | Doble | Bonisote | Mega | \n");
53     for (int i = 0; i < n; i++) { //Iniciamos conteo en 0
54         System.out.println(i + 1 + " | " + nombreVendedor[i]
55             + " | " + cantidad[i][0]
56             + " | " + cantidad[i][1]
57             + " | " + cantidad[i][2]
58             + " | " + cantidad[i][3] + " | \n");
59     }
60     pressAnyKeyToContinue();
61 }
62
63 //Metodo conteo de cargas vendedor
64 public void Conta(int n) {
65
66     //La suma, hecho con contador
67     //Y la impresion de resultados.
68     System.out.println("\n\nTotales generales de cargas por tipo:");
69     System.out.println(" | Sencillo | Doble | Bonisote | Mega | \n");
70
71     //
72     for (int i = 0; i < n; i++) { //Iniciamos en 0
73         total1 += cantidad[i][0];
74         total2 += cantidad[i][1];
75         total3 += cantidad[i][2];
76         total4 += cantidad[i][3];
77     }
78 }

```

Manejo de excepción para evitar que el sistema concluya, sin que el usuario lo requiera



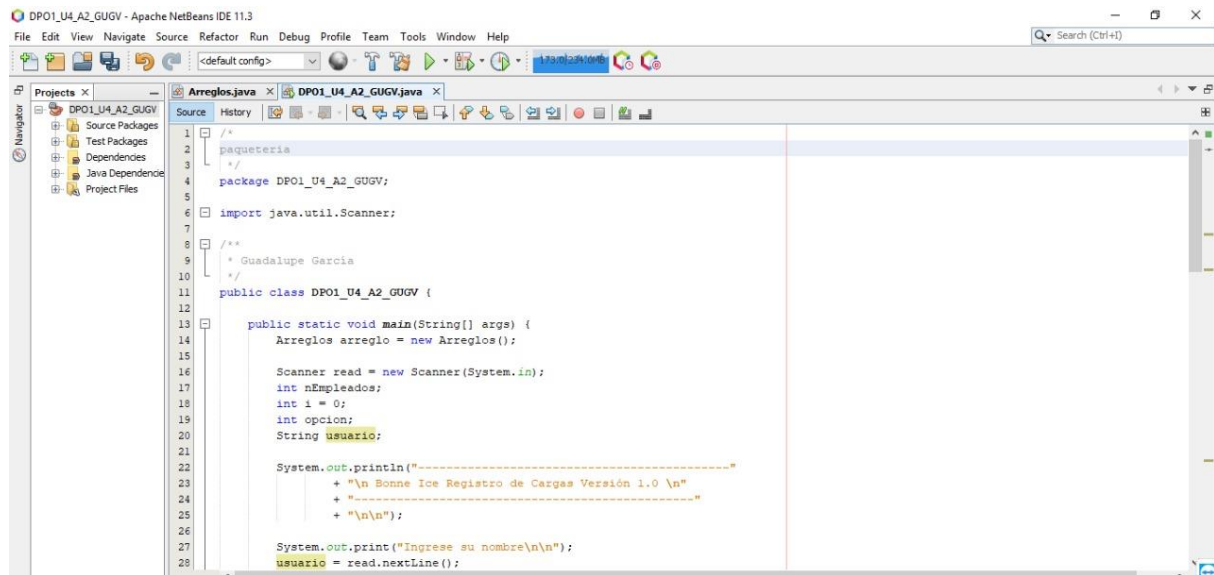
```

83
84 //Impresion de resultados.
85 System.out.println("Total de cargas de producto: " + totalgeneral);
86 pressAnyKeyToContinue();
87
88
89
90 //Excepcion para evitar que el sistema no continúe el proceso
91 public static void pressAnyKeyToContinue() {
92     String seguir;
93     Scanner teclado = new Scanner(System.in);
94     System.out.println("Press Enter key to continue...");
95     try {
96         seguir = teclado.nextLine();
97     } catch (Exception e) {
98     }
99 }
100
101 // Fin clase Arreglos

```



Código de la clase principal, dónde se hará llamar los arreglos.

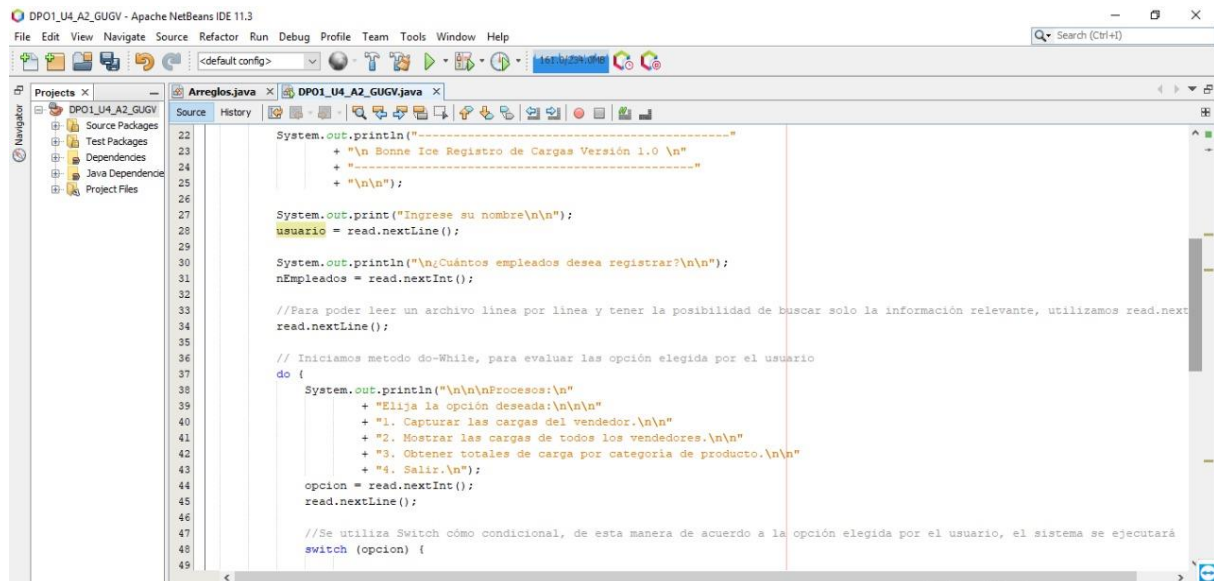


```

1  /**
2   * paquete
3   */
4   package DP01_U4_A2_GUGV;
5
6   import java.util.Scanner;
7
8   /**
9    * Guadalupe Garcia
10   */
11  public class DP01_U4_A2_GUGV {
12
13      public static void main(String[] args) {
14          Arreglos arreglo = new Arreglos();
15
16          Scanner read = new Scanner(System.in);
17          int nEmpleados;
18          int i = 0;
19          int opcion;
20          String usuario;
21
22          System.out.println("-----")
23              + "\n Bonne Ice Registro de Cargas Versión 1.0 \n"
24              + "-----"
25              + "\n\n";
26
27          System.out.print("Ingrese su nombre\n\n");
28          usuario = read.nextLine();

```

Código opciones a elegir



```

22  System.out.println("-----")
23      + "\n Bonne Ice Registro de Cargas Versión 1.0 \n"
24      + "-----"
25      + "\n\n";
26
27  System.out.print("Ingrese su nombre\n\n");
28  usuario = read.nextLine();
29
30  System.out.println("\n¿Cuántos empleados desea registrar?\n\n");
31  nEmpleados = read.nextInt();
32
33  //Para poder leer un archivo línea por línea y tener la posibilidad de buscar solo la información relevante, utilizamos read.next
34  read.nextLine();
35
36  // Iniciamos metodo do-While, para evaluar las opción elegida por el usuario
37  do {
38      System.out.println("\n\nProcesos:\n"
39          + "Elija la opción deseada:\n\n"
40          + "1. Capturar las cargas del vendedor.\n\n"
41          + "2. Mostrar las cargas de todos los vendedores.\n\n"
42          + "3. Obtener totales de carga por categoría de producto.\n\n"
43          + "4. Salir.\n\n");
44      opcion = read.nextInt();
45      read.nextLine();
46
47      //Se utiliza Switch como condicional, de esta manera de acuerdo a la opción elegida por el usuario, el sistema se ejecutará
48      switch (opcion) {
49

```

## Método Switch

```
//Instancia metodo Arreglo, delimitando al parámetro de tal, mediante nEmpleados
case 1:
    if (i < nEmpleados) {
        arreglo.Arreg(nEmpleados);
        i++;
    } else {
        System.out.println("Datos completos \n");
    }
    break;
//Instancia metodo para generar lista
case 2:
    arreglo.Lista(nEmpleados);
    break;
//Instancia contar totales de carga por categoria de producto
case 3:
    arreglo.Conta(nEmpleados);
    break;
//Finalizar el programa
case 4:
    System.out.println("\n\nUsuario: "+usuario+
        "\n\nGracias por utilizar el programa\n");
    break;
}
} while (opcion != 4);
}
} //Fin clase principal
```

## Ejecución del programa

DP01\_U4\_A2\_GUGV - Apache NetBeans IDE 11.3

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+F)

Projects: DP01\_U4\_A2\_GUGV, Source Packages, Test Packages, Dependencies, Java Dependencies, Project Files

Output - Run (DP01\_U4\_A2\_GUGV)

```
cd C:\Users\Nivans\Documents\NetBeansProjects\DP01_U4_A2_GUGV; "JAVA_HOME=C:\\Program Files\\Java\\jdk-11.0.7" cmd /c "%C:\\Program Files\\NetBeans-11.3\\netbe
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be us
Scanning for projects...

Building DP01_U4_A2_GUGV 1.0-SNAPSHOT

--- exec-maven-plugin:1.5.0:exec (default-cli) @ DP01_U4_A2_GUGV ---
Bonne Ice Registro de Cargas Versión 1.0

Ingrese su nombre
Guadalupe
¿Cuántos empleados desea registrar?
1

Procesos:
Elija la opción deseada:

1. Capturar las cargas del vendedor.
```

DP01\_U4\_A2\_GUGV - Apache NetBeans IDE 11.3

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+F)

Projects: DP01\_U4\_A2\_GUGV, Source Packages, Test Packages, Dependencies, Java Dependencies, Project Files

Output - Run (DP01\_U4\_A2\_GUGV)

```
Procesos:
Elija la opción deseada:

1. Capturar las cargas del vendedor.
2. Mostrar las cargas de todos los vendedores.
3. Obtener totales de carga por categoría de producto.
4. Salir.
1

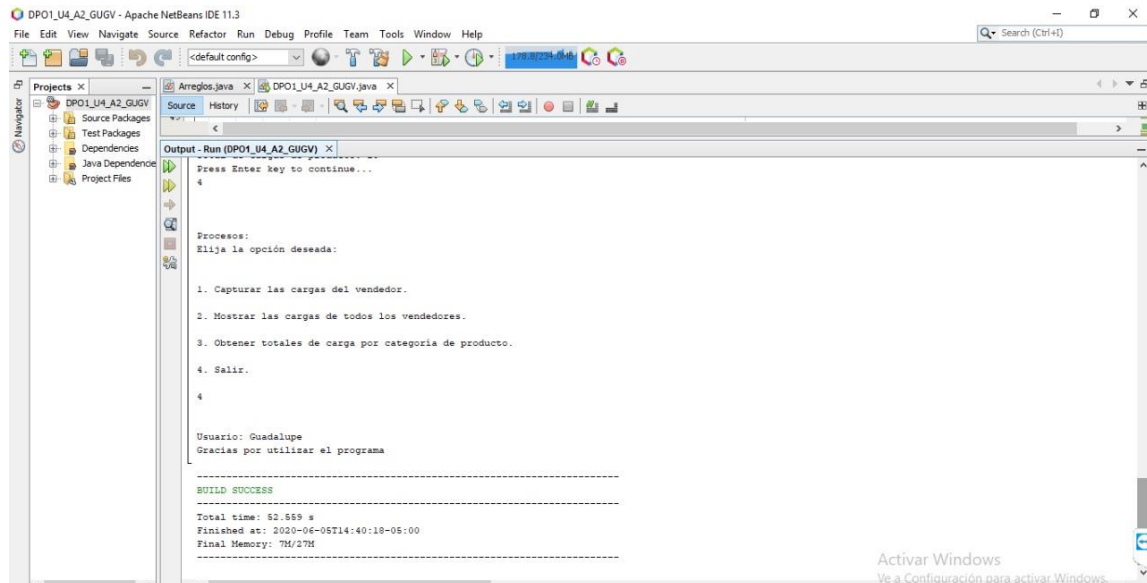
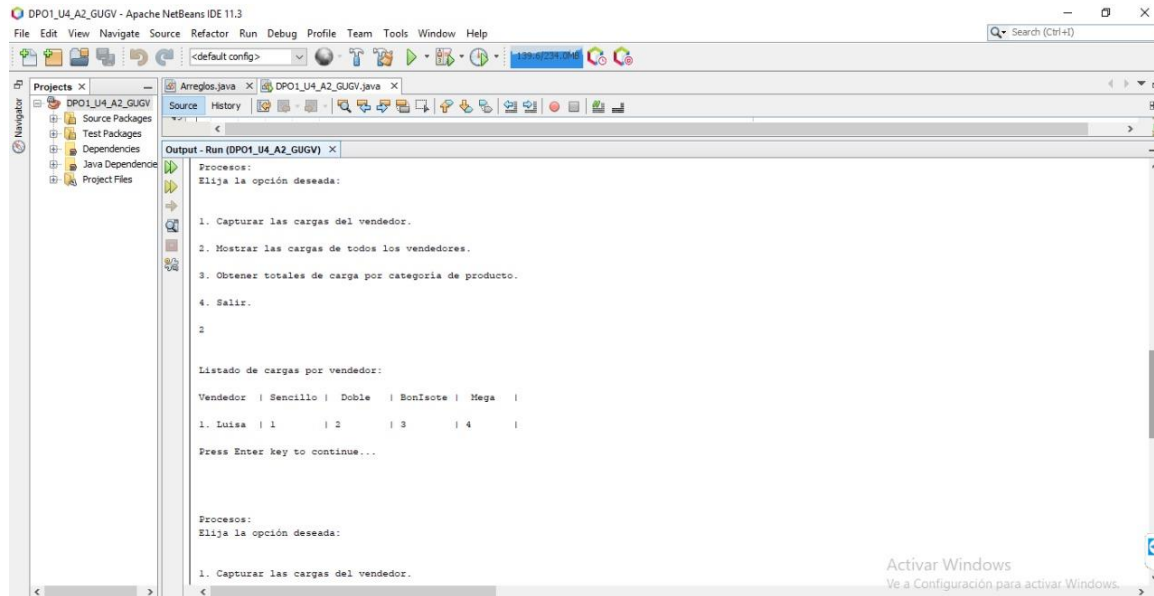
Ingrese el NOMBRE del vendedor:
Luisa

Ingrese las cantidades de producto de carga del vendedor según la presentación

Bonice SENCILLOS:
1

Bonice DOBLES:
2

Bonice BONISOTIE:
```



## Retos:

Los retos que se presentaron al desarrollar el sistema, fue en primero lugar la iniciación de los arreglos, ya que el conteo debía comenzar de 0 por lo que, al momento de iniciarlo en 0, el programa se rompía y saltaba al siguiente punto.

En el caso del arreglo multidimensional, el saber cómo hacer la instancia en la clase principal, ya que debía de hacerlo también con el unidimensional, esto lo resolví utilizándolo dentro de la estructura del multidimensional.

El método que más de me complico fue el de contar, ya que debía tener cuidado con las variables y cuidar que el conteo fuera desde 0.

## Conclusiones

En este siguiente tema hemos abordamos el tema de los arreglos, pero ahora uniendo los dos tipos de arreglos. Para realizar la unión de ambos, es necesario establecer de forma óptima el objeto del arreglo, en este caso son las cargas de vendedores, junto con las cantidades por tipo de producto. Por lo que, al momento de hacer el llamado del arreglo, no necesitaremos hacer la instancia por separado, si no que el arreglo nos permitirá generalizarlo. En esto radica la importancia de los métodos, ya que estos representan los comportamientos de los objetos pertenecientes a una clase y constituyen el medio para el envío de mensajes entre ellos.

En este ejercicio pude notar la diferencia entre ambos arreglos, destacando que el arreglo multidimensional, se contempla como una tabla, en dónde para acceder a un dato, hay que establecer las “coordenadas”, para encontrarlo. Este tipo de arreglos tiene mucha utilidad en el manejo de datos extensos, con diferentes campos, pero con misma relación.

## Bibliografía

- Berzal, F. (Consultado 03/06/2020). *Vectores y matrices*. Obtenido de Vectores y matrices - Java: Obtenido de: <https://elvex.ugr.es/decsai/java/pdf/6A-Arrays.pdf>
- Canal: . (2017). *Curso Java - 5: Arreglos y Matrices (Arrays)*. Obtenido de Consultado el 04/06/2020 <https://www.youtube.com/watch?v=KZwMRL2q6O8>
- Canal: latincoder. (2014). *Arreglos multidimensionales/Matrices - Tutorial Java*. Obtenido de Consultado el 04/06/2020 [w.youtube.com/watch?v=Y3pJTyaMzac](https://www.youtube.com/watch?v=Y3pJTyaMzac)
- Creatividad Codificada. (Consultado el: 03/06/2020). *Arrays o arreglos en Java*. Obtenido de Obtenido de\_ <https://creatividadcodificada.com/java/arrays-o-arreglos-en-java/>
- David Hackro. (2014). *Registro de alumnos(arreglo de objetos) Resuelto con JAVA*. Obtenido de Consultado 04/06/2020 Obtenido de: <https://www.youtube.com/watch?v=Z4k582CoCvs>
- Grupo Codesi. (Consultado 03/06/2020). *Arreglos en Java*. Obtenido de Ventajas de utilizar Arreglos en Java: Obtenido de: <http://www.buscaminegocio.com/cursos-de-java/arreglos-java.html>
- Universidad Abierta y a Distancia de México. (Consultado el 03/06/2020). *Unidad 4. Arreglos*. Obtenido de Programación orientada a objetos I: Obtenido de: [https://ceit.unadmexico.mx/contenidos/DCEIT/BLOQUE2/DS/02/DPO1/U4/descargables/DPO1\\_U4\\_Contenido.pdf](https://ceit.unadmexico.mx/contenidos/DCEIT/BLOQUE2/DS/02/DPO1/U4/descargables/DPO1_U4_Contenido.pdf)
- Usuario:alexis\_0792@hotmail.com. (Consultado 04/05/2020). *Indice de Métodos*. Obtenido de Obtenido de: <https://programandoconjava.es.tl/Metodos-De-Busqueda-y-Orenacion.htm>
- V, J. (s.f.). *Arreglos en java*. Obtenido de Consultado 03/06/2020: Obtenido de: <http://codigoprogramacion.com/cursos/java/96-arreglos-en-java.html>