Trabajo 1 de Estructura de Datos

Luis Felipe Romero Parra

Hennys Sebastian Tous De Avila

Universidad de Cartagena

**Texto, Carta

Descripción generada automáticamente**

Programa: Ingeniería de Software

Asignatura: Estructura de Datos

Tutor: Jhon Arrieta Arrieta

Cartagena de Indias – Julio 30 del año 2022

# Tabla de Contenido

1. Introducción.
2. Objetivo.
3. Justificación.
4. Mostrar un ejemplo de uso de arreglos en java así:
   1. Declarar un arreglo por cada uno de los tipos de datos primitivos (byte, Byte, char, Character, short, Short, int, Integer, long, Long, float, Float, double, Double, Object, String.
   2. Definir cada uno de los arreglos anteriores con tamaño de 5 elementos
   3. Asignar valores a cada arreglo usando la notación de posiciones o índices
   4. Mostrar los valor es cada Arreglo
   5. Declarar y definir otro arreglo de tipo String en una variable llamada dataStructs, e inicializarlo con los siguientes datos, usando la sintaxis de llaves.
      1. Listas
      2. Colas
      3. Pilas
      4. Mapas
      5. Conjuntos
   6. Declarar y definir otro arreglo de tipo String en una variable llamada características, con 5 elementos.
   7. Ingresar los siguientes datos por teclado:
      1. Simples, Circulares y Enlazadas
      2. Primero en entrar, Primero en Salir **COLAS**.
      3. Primero en entrar, Último en salir **PILAS.**
      4. Parejas de clave y valor
      5. Elementos no repetidos
   8. Mostrar los elementos del arreglo dataStructs con su equivalente elemento del arreglo características.
   9. Declarar un arreglo de enteros en una variable llamada ventas,
   10. Inicializar el arreglo ventas con una cantidad aleatoria de elementos.
   11. Llenar el arreglo ventas con valores aleatorios entre 1 y 1000
   12. Mostrar la cantidad de ventas
   13. Mostrar el valor de cada una de las ventas
   14. Mostrar el total de ventas
   15. Mostrar el promedio de las ventas.
   16. Diseñar una función que permita buscar la venta más alta e invocar la función para probarla.
   17. Diseñar una función que permita buscar la venta más baja e invocar la función para probarla.
   18. Diseñar una función que permita obtener el total de las ventas e invocar la función para probarla.
   19. Diseñar una función que permita obtener el promedio de las ventas e invocar la función para probarla.
   20. Diseñar una función que devuelva la venta cuyo valor se acerque más al valor que le sigue al promedio e invocar la función para probarla.
   21. Diseñar una función que permita ordenar las ventas de forma descendente e invocar la función para probarla.
   22. Diseñar una función que permita ordenamiento que permita ordenar las ventas de forma ascendente e invocar la función para probarla.
   23. Diseñar una función que permite desordenar las ventas e invocar la función para probarla.
   24. Diseñar una función que permita ordenar las ventas primero las partes y luego en impares e invocar la función para probarla.
   25. Declarar e inicializar una matriz de 5 x 8 de tipo Object en una variable llamada lenguajes
   26. A la primera línea de la matriz se de debe asignar los siguientes valores en ese orden:

[NOMBRE, AÑO, AUTOR, DETALLES, FRAMEWORKS]

* 1. Realizar una función llamada **infoLenguajes** que permita llenar la matriz con datos por teclado a partir de la segunda fila.
  2. Realizar una función que permita mostrar los datos de la matriz según la siguiente logia:
     1. La función de debe recibir como parámetro un arreglo con 5 elementos de tipo boolean
     2. Al invocar la función se debe mostrar solo la información de la matriz de acuerdo al valor (false o true) que tenga el arreglo que recibe como parámetro. Por ejemplo:

boolean **info**[] = **{**true**,** false**,** true**,** true**,** false**}**;

**infoLenguajes**(info)

Debe mostrar los nombres de los lenguajes, los autores, los frameworks.

1. Síntesis y argumentos individuales.
2. Bibliografía

# Introducción

En el siguiente trabajo vamos a mostrar de manera detallada y justificada la manera básica todo sobre estructura de datos desde cómo hacer un arreglo, como llenar el arreglo hasta como hacer arreglos bidimensionales.

También encontraras como como se maneja los arreglos con cada tipo de datos primitivos y diferentes sistemas principales que trabajaras en esta asignatura.

# Objetivos

- Aprender a como crea e inicializar los arreglos con diferentes tipos de datos.

- Identificar los tipos de arreglos y las ocasiones en donde su uso es especifico

- Comprender la estructuración de datos por medio de Arrays en Java

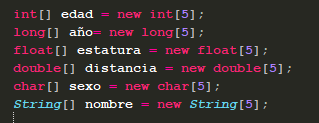
- Saber cómo operar con los datos almacenados en uno o más Arrays para obtener distintos outputs.

# Justificación

Nuestras investigaciones nos demuestran que para lograr un aprendizaje más amplio es compartiendo lo aprendido y de esta manera tener más conocimiento. De esta manera hemos trabajo en este trabajo de investigación y elaboración para que usuario que lo obtenga lo entienda de manera directa.

# Resultados

* 1. Declarar un arreglo por cada uno de los tipos de datos primitivos (byte, Byte, char, Character, short, Short, int, Integer, long, Long, float, Float, double, Double, Object, String.
  2. Definir cada uno de los arreglos anteriores con tamaño de 5 elementos.



De esta forma declaramos cada arreglo con el tipo de datos primitivos y entre los últimos corchetes definimos el tamaño de arreglo. **Puntos (a & b)**

* 1. Asignar valores a cada arreglo usando la notación de posiciones o índices

Texto

Descripción generada automáticamente

Asignación de valores de tipo entero corto.

Texto

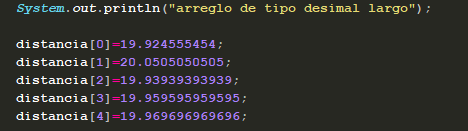
Descripción generada automáticamente

Entero largo.

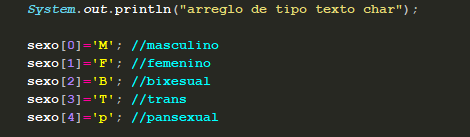
Imagen que contiene Texto

Descripción generada automáticamente

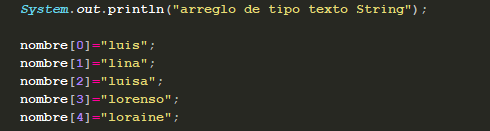
Datos de tipo float (flotante o decimal corto)



Ingresamos datos de tipo dounle (decimal largo) es el más utilizado para trabajar con decimales



Los datos de tipo char son utilizados para tipos de datos texto con una sola letra además para poder declararlas necesitas colocarlas entre comillas simples (‘ ’) ; en la imagen puedo mostrar un ejemplo.



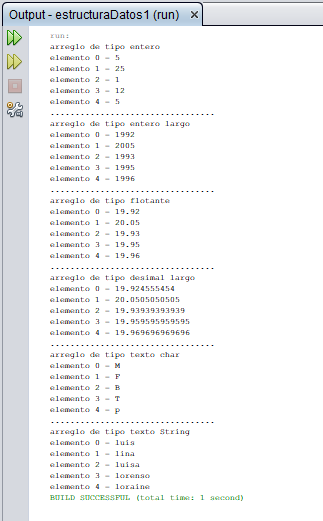
Para asignar los valores de tipo texto o string a diferencia de los tipo char aquí tienes que declarar los datos entre comillas dobles.

* 1. Mostrar los valor es cada Arreglo

Para poder mostrar los valores dela arreglo debemos primero declara un siclo repetitivo en este caso ciclo for para que nos corra todos los datos de nuestro arreglo para imprimirlos en pantalla.



Aquí mostramos la de tipo de datos String pero es la misma para cada tipo de datos.



* 1. Declarar y definir otro arreglo de tipo String en una variable llamada dataStructs, e inicializarlo con los siguientes datos, usando la sintaxis de llaves.

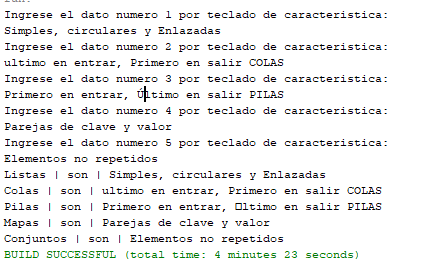


* 1. Ingresar los siguientes datos por teclado:
     1. Simples, Circulares y Enlazadas
     2. Último en entrar, Primero en salir **COLAS**
     3. Primero en entrar, Último en salir **PILAS**.
     4. Parejas de clave y valor.
     5. Elementos no repetidos

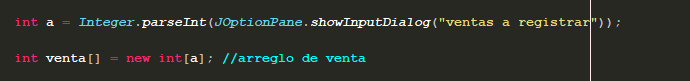
Texto

Descripción generada automáticamente

* 1. Mostrar los elementos del arreglo dataStructs con su equivalente elemento del arreglo características.

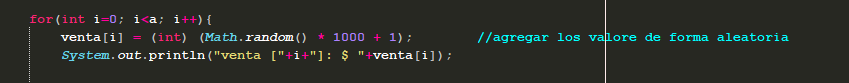


* 1. Declarar un arreglo de enteros en una variable llamada ventas.
  2. Inicializar el arreglo ventas con una cantidad aleatoria de elementos.



Aquí esta declara e inicializado el arreglo con una variable a al inicio con la cual podremos ingresarle la cantidad de datos que va a tener mi arreglo. **(punto J & K).**

* 1. Llenar el arreglo ventas con valores aleatorios entre 1 y 1000.



Con esta función random llenamos de manera automática nuestro arreglo con valores determinados entre 1 y el 1000.

En este ejemplo mostraremos 5 datos para nuestro arreglo:

* Definimos la cantidad de datos q tendrá el arreglo.

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

* Los datos llenados automáticamente para el ejemplo de este arreglo.

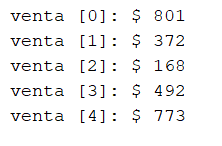
Texto

Descripción generada automáticamente

* 1. Mostrar la cantidad de ventas



* 1. Mostrar el valor de cada una de las ventas



* 1. Mostrar el total de ventas



* 1. Mostrar el promedio de las ventas.



* 1. Diseñar una función que permita buscar la venta más alta e invocar la función para probarla.
* Iniciamos una variable llamada ventaMayor y le damos un valor inicial en cero.



* Agredamos la función que nos permitirá hallar el valor de la venta mayor.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

* Este es el valor que nos muestre en pantalla.



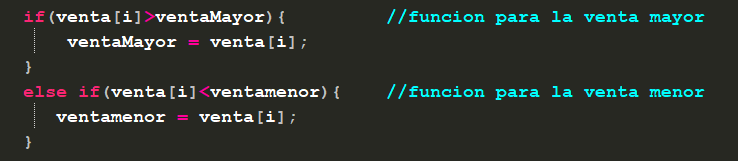


Que al ver los datos anteriores muestra que es el correcto.

* 1. Diseñar una función que permita buscar la venta más baja e invocar la función para probarla.
* Iniciamos una variable llamada ventaMenor y le damos como valor inicial un número mayor a 1000.



* Agregamos la función que nos permitirá hallar el valor de la venta más baja.



Seguimos con la función para hallar la venta menor donde denotamos que si no se cumple la función principal se cumpla la segundaria donde decimos que si la si el valor de la venta es menor a ventamenor entonces venta menor va hacer igual al valor de la venta.

* Valor mostrado en pantalla.





* 1. Diseñar una función que permita obtener el total de las ventas e invocar la función para probarla.



Iniciamos una variable llamada sumaventa la cual iniciamos con igual a cero ya que va hacer nuestro acumulador.



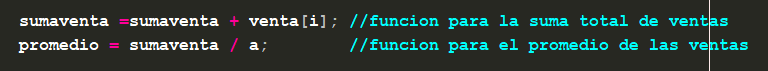
Y con esta función nuestro acumulador sumaventa va sumando cada valor que va ingresando al arreglo.



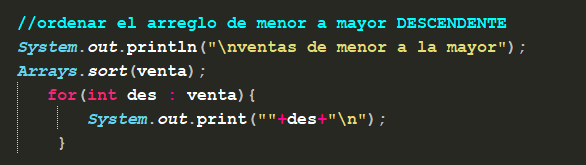


Y aquí mostramos por pantalla le valor total de nuestro acumulador

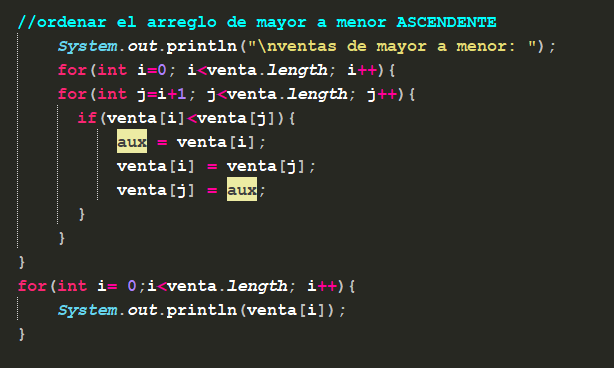
* 1. Diseñar una función que permita obtener el promedio de las ventas e invocar la función para probarla.



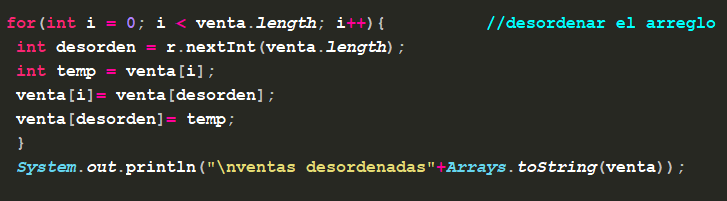
* 1. Diseñar una función que devuelva la venta cuyo valor se acerque más al valor que le sigue al promedio e invocar la función para probarla.
  2. Diseñar una función que permita ordenar las ventas de forma descendente e invocar la función para probarla.



* 1. Diseñar una función que permita ordenamiento que permita ordenar las ventas de forma ascendente e invocar la función para probarla.

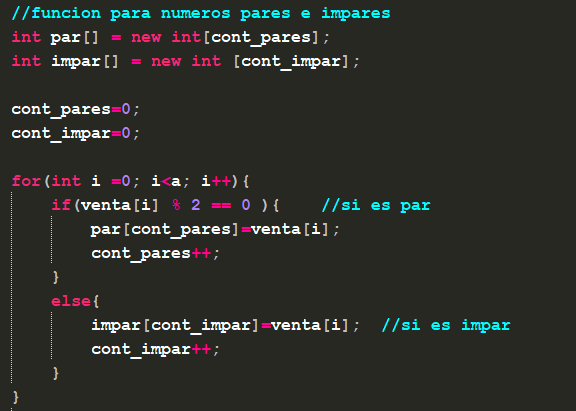


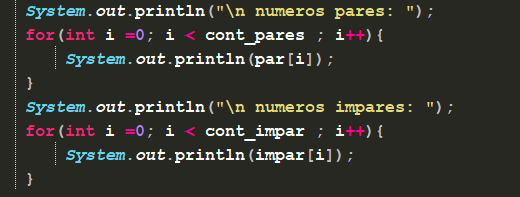
* 1. Diseñar una función que permite desordenar las ventas e invocar la función para probarla.



* 1. Diseñar una función que permita ordenar las ventas primero las partes y luego en impares e invocar la función para probarla.





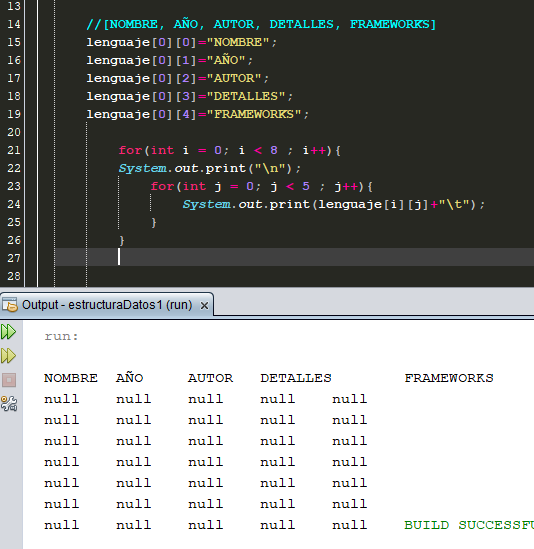


* 1. Declarar e inicializar una matriz de 5 x 8 de tipo Object en una variable llamada lenguajes



* 1. A la primera línea de la matriz se de debe asignar los siguientes valores en ese orden:

[NOMBRE, AÑO, AUTOR, DETALLES, FRAMEWORKS]



* 1. Realizar una función llamada **infoLenguajes** que permita llenar la matriz con datos por teclado a partir de la segunda fila.
  2. Realizar una función que permita mostrar los datos de la matriz según la siguiente logia:
     1. La función de debe recibir como parámetro un arreglo con 5 elementos de tipo boolean
     2. Al invocar la función se debe mostrar solo la información de la matriz de acuerdo al valor (false o true) que tenga el arreglo que recibe como parámetro. Por ejemplo:

boolean **info**[] = **{**true**,** false**,** true**,** true**,** false**}**;

**infoLenguajes**(info)

Debe mostrar los nombres de los lenguajes, los autores, los frameworks.

# Síntesis

Una array o arreglo es una colección de variables del mismo tipo, a la que se hace referencia por un nombre común. En Java, los arrays pueden tener una o más dimensiones, aunque el array unidimensional es el más común.

Por ejemplo, puede usar una matriz para mantener un registro de la temperatura alta diaria durante un mes, una lista de promedios de precios de acciones o una lista de tu colección de libros de programación.

La ventaja principal de un array es que organiza los datos de tal manera que puede ser manipulado fácilmente. Además, los arrays organizan los datos de tal manera que se pueden ordenar fácilmente.

Aunque los arrays en Java se pueden usar como matrices en otros lenguajes de programación, tienen un atributo especial: se implementan como objetos. Este hecho es una de las razones por las que la discusión de los arrays se pospuso hasta que se introdujeron los objetos.

# Conclusión

Podemos decir que se a partir de la realización de este trabajo pudimos comprender a profundidad como podemos estructurar datos mediante el uso de Arrays unidimensionales y Bidimensionales.

# Referencias

Abrirllave.com. (s.f.). *Abrirllave.com*. Obtenido de https://www.abrirllave.com/java/leer-los-valores-de-un-array-desde-el-teclado.php

ATS, P. (28 de Mayo de 2017). *YouTube*. Obtenido de https://www.youtube.com/watch?v=gAKX3UiWYaY&list=WL&index=6

ATS, P. (4 de Abril de 2017). *YouTube*. Obtenido de https://www.youtube.com/watch?v=79LOsYYZm\_8&t=177s

DelftStack. (2019). *DelftStack*. Obtenido de https://www.delftstack.com/es/

UskroKruM2010. (19 de Marzo de 2015). *YouTube*. Obtenido de https://www.youtube.com/watch?v=5k6xDsSyVA8&t=221s

Walton, A. (10 de Marzo de 2018). *javadesdecero*. Obtenido de https://javadesdecero.es/arrays/unidimensionales-multidimensionales/