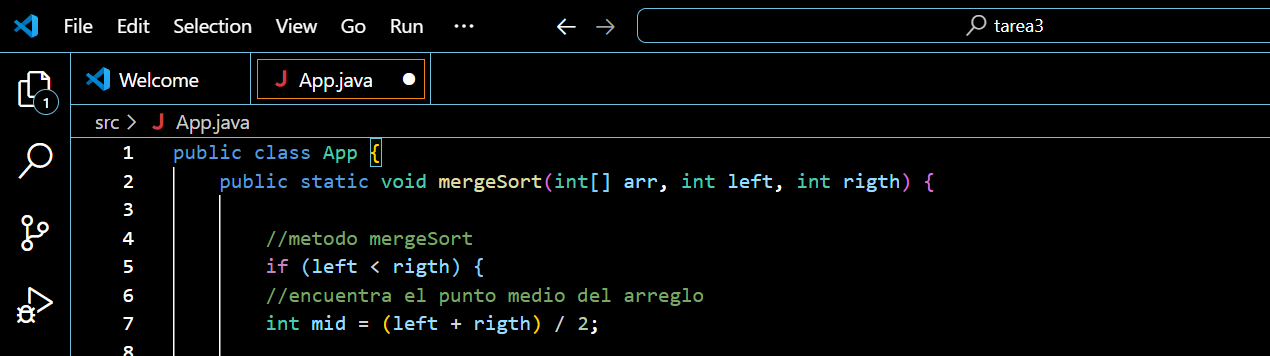
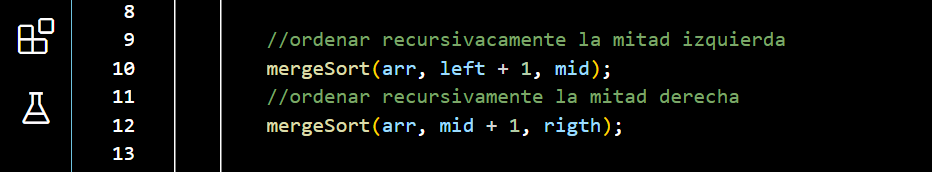
Método mergeSort

Lo que entiendo es que calcula el punto medio del arreglo que esta siendo dividido en dos partes en el proceso de dividir y de conquistar el algoritmo merge Sort. De este paso es que este punto es importante para dividir el arreglo en dos partes iguales o casi iguales, lo que facilita la recursión y la combinación de las partes ordenadas posteriormente.

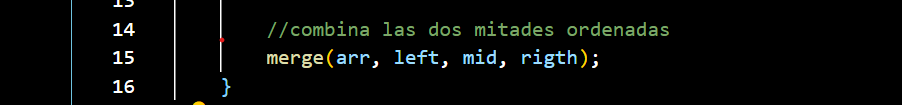


Divide el arreglo en dos mitades, desde el índice “left” hasta el índice “mid” y ordena recursivamente la primera mitad que es la parte izquierda. También divide el arreglo en dos mitades que va desde el índice “mid + 1” hasta el índice “right” y igualmente ordena recursivamente la segunda mitad que es la parte derecha.

Esto continúa dividiendo el arreglo en mitades mas pequeñas hasta que cada subarreglo tenga un solo elemento.



Se utiliza para funcionar dos subarreglos ordenados en un solo arreglo ordenado después de que las llamadas recursivas de mergeSort. Hallan ordenada las mitades de derecha a izquierda del arreglo ya que este paso es importante para combinar los subarreglos ordenados en uno solo ordenados correctamente.

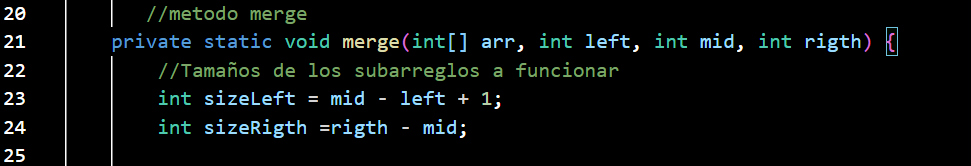


Método merge

Lo que entiendo de este método es que calcula el tamaño del subarreglo izquierdo sumando 1 al resultado de (mid – left) esto se hace para asegurarse de incluir el elemento en el índice “mid” en el subarreglo izquierdo.

Calcula el tamaño del subarreglo derecho restando “mid” del índice “right” esto para determinar cuántos elementos hay en el subarreglo derecho.

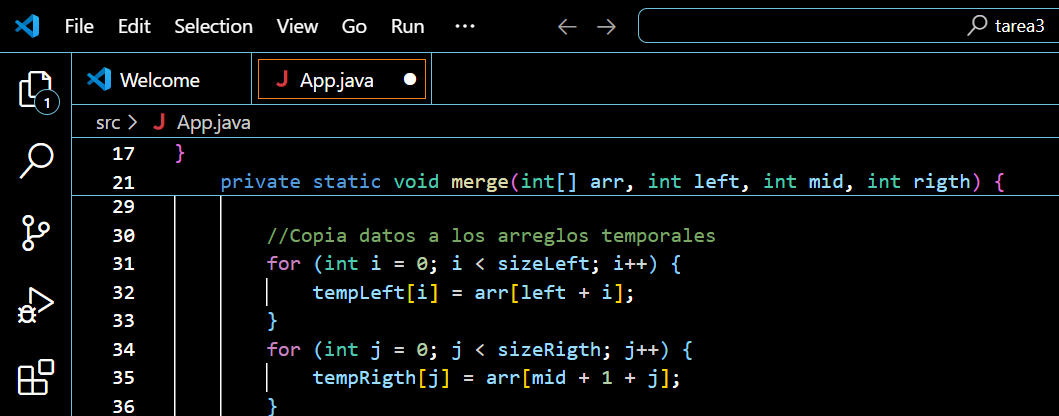
Se utiliza más adelante para el proceso de la función para iterar sobre los subarreglos izquierdo y derechos que están funcionando mientra se combina en un solo arreglo ordenado.



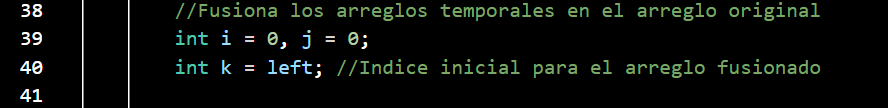
Se crea un nuevo arreglo temporal llamado “templeft” con tamaño iguala “sizeLeft”, que es el tamaño del subarreglo izquierdo. Este arreglo se utiliza para almacenar temporalmente los elementos del subarreglo izquierdo antes de funcionarlos.

De manera similar se crea un nuevo arreglo temporal llamado “tempRight” con tamaño del subarreglo derecho. Este arreglo se utiliza para almacenar temporalmente los elementos del subarreglo derecho antes de funcionarlos.

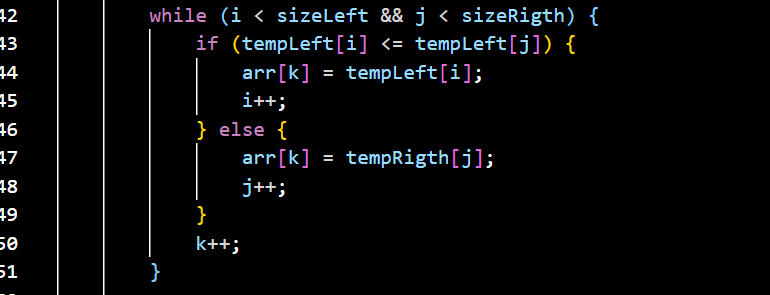
Estos arreglos temporales son necesarios para el proceso de función, ya que permite realizar la función de manera eficiente sin modificar el arreglo original.



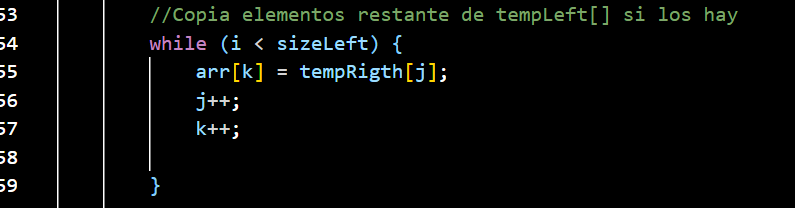
Estas variables se utilizan en el proceso de fusión para recorrer los arreglos temporales y colocar los elementos fusionados en el arreglo original en la posición adecuada. El bucle de fusión utiliza estas variables para comparar los elementos en los arreglos temporales y colocarlos en orden en el arreglo original.



garantiza que los elementos de los subarreglos temporales se fusionen en el arreglo original en orden ascendente.

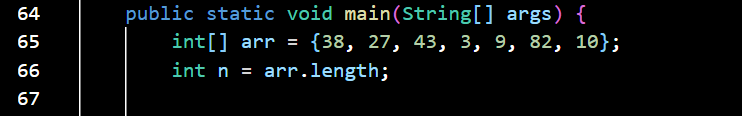


asegura que, si uno de los subarreglos temporales tiene elementos sobrantes después de que se haya agotado el otro subarreglo temporal, esos elementos se agreguen correctamente al final del arreglo original “arr” en el orden correcto.



Método main

Esto calcula la longitud del arreglo “arr” y la asigna a la variable n. La longitud del arreglo es el número de elementos que contiene. En este caso, la variable n tendría el valor 7, ya que hay 7 elementos en el arreglo “arr”.



sirve para ordenar el arreglo “arr” completo utilizando el algoritmo “merge sort”, comenzando desde el índice “0” y terminando en el índice “n – 1”.



este código forma parte de la salida del programa que muestra el arreglo originalmente desordenado seguido por el arreglo ya ordenado.

