JAVA

COLLECTIONS JAVA

ALBERTO ESCAMILLA JASSO

TIPES OF COLLECTIONS

ARRAYLIST

SET

MAPS

QUEUE

EXPLANATION

ARRAYLIST

It is an implementation of the List interface. It is a list that allows duplicate elements and maintains the insertion order. It is based on a dynamic array that can automatically resize itself when elements are added or removed.

SET

It is an interface that represents a collection that does not allow duplicate elements. It is ideal for situations where you want to store unique elements.

MAP

It is an interface that represents a collection of key-value pairs. Each key is unique and is associated with exactly one value. It is ideal for quickly retrieving values associated with specific keys.

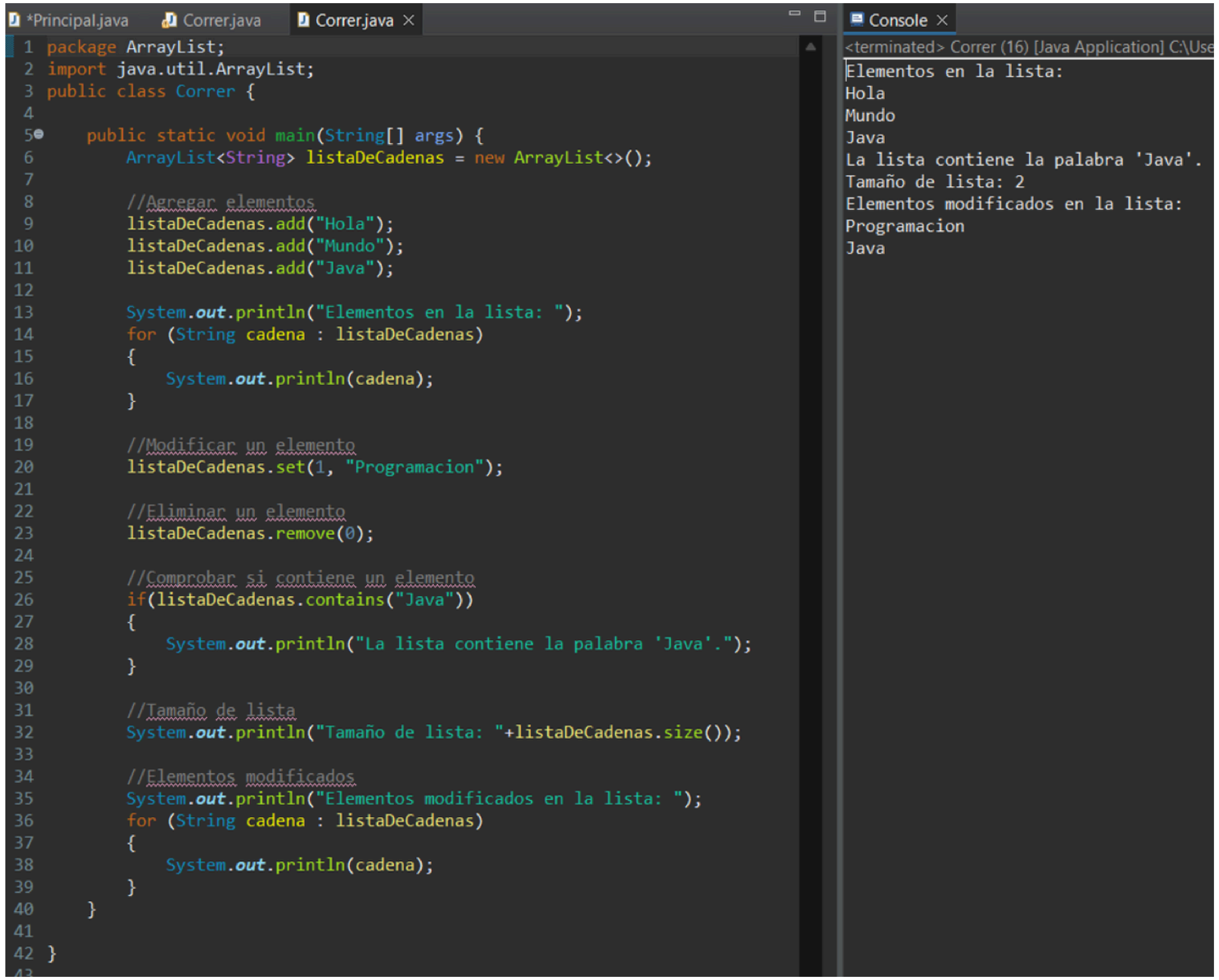
QUEUE

It is an interface that represents a collection following the queue data structure. A queue is a data structure where elements are inserted at one end and removed from the other end, following the FIFO (First In, First Out) principle

COLLECTIONS

EXAMPLE

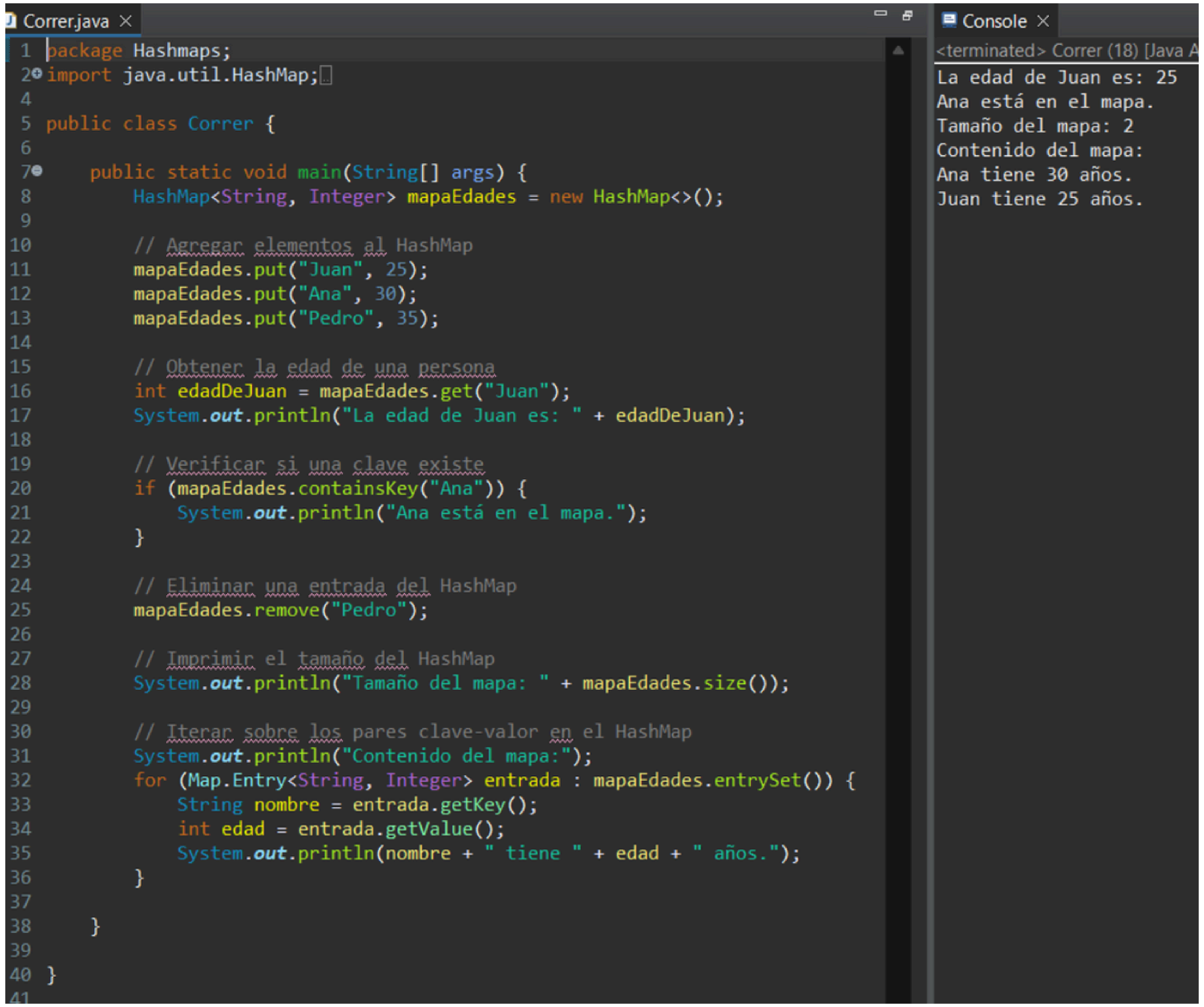
ARRAYLIST



```java
package ArrayList;
import java.util.ArrayList;
public class Correr {

    public static void main(String[] args) {
        ArrayList<String> listaDeCadenas = new ArrayList<>();

        //Agregar elementos
        listaDeCadenas.add("Hola");
        listaDeCadenas.add("Mundo");
        listaDeCadenas.add("Java");

        System.out.println("Elementos en la lista: ");
        for (String cadena : listaDeCadenas)
        {
            System.out.println(cadena);
        }

        //Modificar un elemento
        listaDeCadenas.set(1, "Programacion");

        //Eliminar un elemento
        listaDeCadenas.remove(0);

        //Comprobar si contiene un elemento
        if(listaDeCadenas.contains("Java"))
        {
            System.out.println("La lista contiene la palabra 'Java'.");
        }

        //Tamaño de lista
        System.out.println("Tamaño de lista: "+listaDeCadenas.size());

        //Elementos modificados
        System.out.println("Elementos modificados en la lista: ");
        for (String cadena : listaDeCadenas)
        {
            System.out.println(cadena);
        }
    }

}
```

```
<terminated> Correr (16) [Java Application] C:\Use
Elementos en la lista:
Hola
Mundo
Java
La lista contiene la palabra 'Java'.
Tamaño de lista: 2
Elementos modificados en la lista:
Programacion
Java
```

MAPS



Correr.java

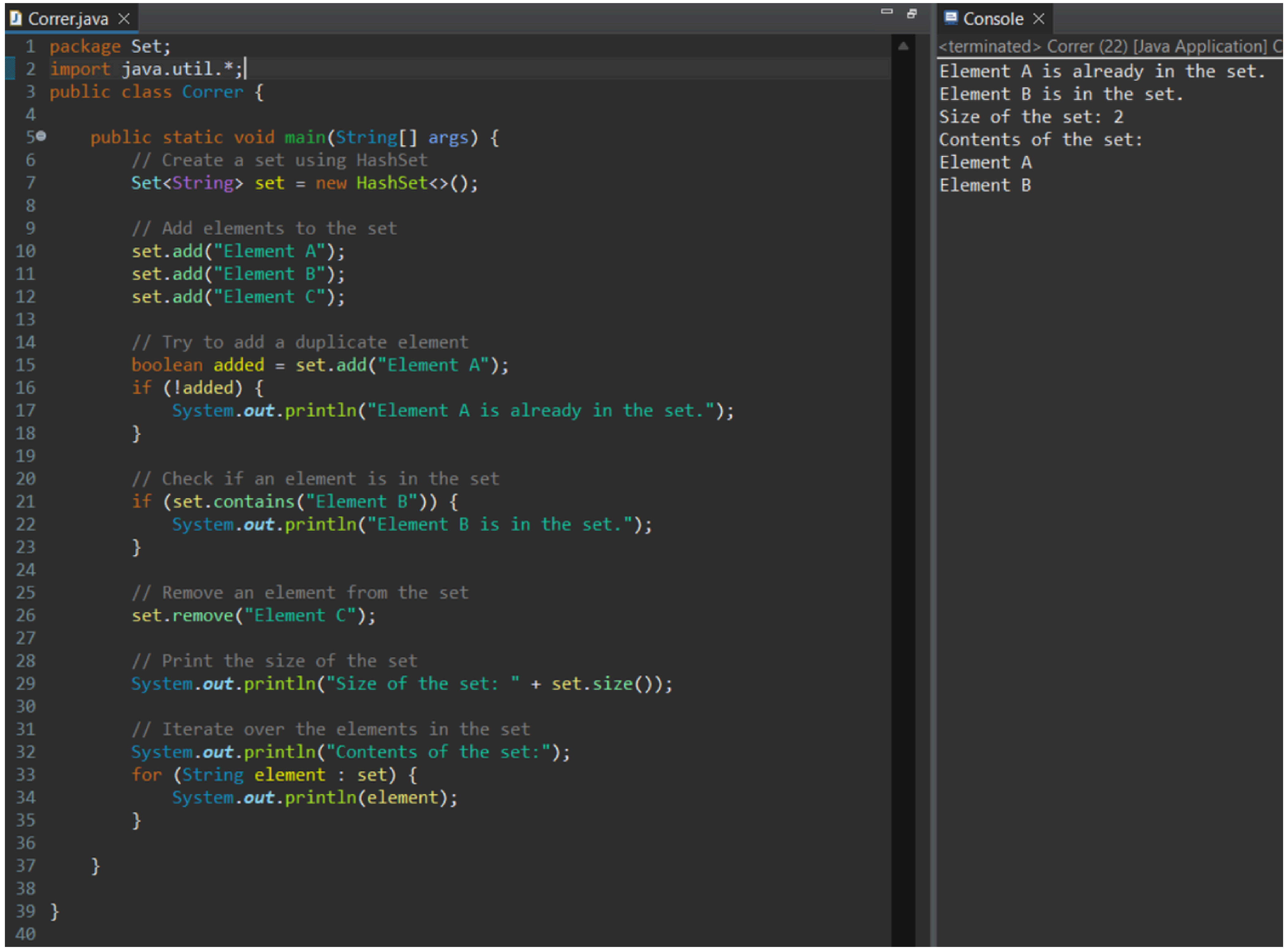
```java
package Hashmaps;
import java.util.HashMap;

public class Correr {

    public static void main(String[] args) {
        HashMap<String, Integer> mapaEdades = new HashMap<>();

        // Agregar elementos al HashMap
        mapaEdades.put("Juan", 25);
        mapaEdades.put("Ana", 30);
        mapaEdades.put("Pedro", 35);

        // Obtener la edad de una persona
        int edadDeJuan = mapaEdades.get("Juan");
        System.out.println("La edad de Juan es: " + edadDeJuan);

        // Verificar si una clave existe
        if (mapaEdades.containsKey("Ana")) {
            System.out.println("Ana está en el mapa.");
        }

        // Eliminar una entrada del HashMap
        mapaEdades.remove("Pedro");

        // Imprimir el tamaño del HashMap
        System.out.println("Tamaño del mapa: " + mapaEdades.size());

        // Iterar sobre los pares clave-valor en el HashMap
        System.out.println("Contenido del mapa:");
        for (Map.Entry<String, Integer> entrada : mapaEdades.entrySet()) {
            String nombre = entrada.getKey();
            int edad = entrada.getValue();
            System.out.println(nombre + " tiene " + edad + " años.");
        }

    }

}
```

Console

<terminated> Correr (18) [Java A

```
La edad de Juan es: 25
Ana está en el mapa.
Tamaño del mapa: 2
Contenido del mapa:
Ana tiene 30 años.
Juan tiene 25 años.
```

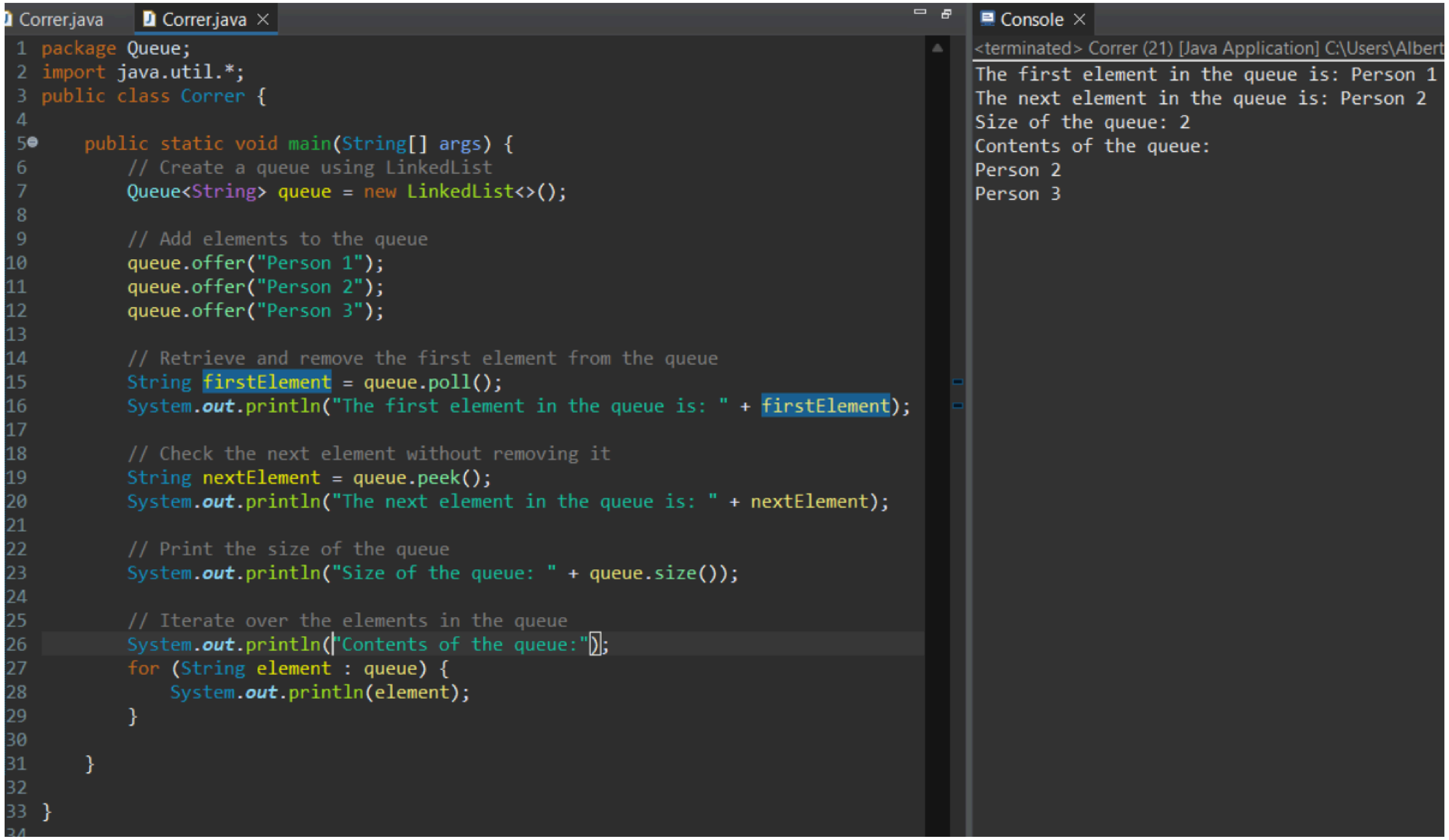
SET



```java
package Set;
import java.util.*;
public class Correr {

    public static void main(String[] args) {
        // Create a set using HashSet
        Set<String> set = new HashSet<>();

        // Add elements to the set
        set.add("Element A");
        set.add("Element B");
        set.add("Element C");

        // Try to add a duplicate element
        boolean added = set.add("Element A");
        if (!added) {
            System.out.println("Element A is already in the set.");
        }

        // Check if an element is in the set
        if (set.contains("Element B")) {
            System.out.println("Element B is in the set.");
        }

        // Remove an element from the set
        set.remove("Element C");

        // Print the size of the set
        System.out.println("Size of the set: " + set.size());

        // Iterate over the elements in the set
        System.out.println("Contents of the set:");
        for (String element : set) {
            System.out.println(element);
        }

    }

}
```

```
<terminated> Correr (22) [Java Application] C
Element A is already in the set.
Element B is in the set.
Size of the set: 2
Contents of the set:
Element A
Element B
```

QUEUE



Correr.java Correr.java

```java
package Queue;
import java.util.*;
public class Correr {

    public static void main(String[] args) {
        // Create a queue using LinkedList
        Queue<String> queue = new LinkedList<>();

        // Add elements to the queue
        queue.offer("Person 1");
        queue.offer("Person 2");
        queue.offer("Person 3");

        // Retrieve and remove the first element from the queue
        String firstElement = queue.poll();
        System.out.println("The first element in the queue is: " + firstElement);

        // Check the next element without removing it
        String nextElement = queue.peek();
        System.out.println("The next element in the queue is: " + nextElement);

        // Print the size of the queue
        System.out.println("Size of the queue: " + queue.size());

        // Iterate over the elements in the queue
        System.out.println("Contents of the queue:");
        for (String element : queue) {
            System.out.println(element);
        }

    }

}
```

Console

<terminated> Correr (21) [Java Application] C:\Users\Albert

```
The first element in the queue is: Person 1
The next element in the queue is: Person 2
Size of the queue: 2
Contents of the queue:
Person 2
Person 3
```

SUBCOLLECTIONS

HASHSET

A set implementation backed by a hash table. It does not maintain any order of elements.

TREESET

A set implementation that uses a red-black tree to store elements in sorted order.

HASMAP

A map implementation that uses a hash table. It does not maintain any order of keys or values.

LINKEDHASHMAP

A map implementation that maintains insertion order using a linked list.

LINKEDLIST

A queue implementation that also implements the Deque interface, providing double-ended queue operations.

TREEMAP

A map implementation that uses a red-black tree to store keys in sorted order.