1. **Write a Java program that reads a string from the user and uses StringTokenizer to split the string into individual words. Print each word on a new line.**

**Program:**

**package** myPackage;

**import** java.util.Scanner;

**import** java.util.StringTokenizer;

**public** **class** String\_Tokenizer1 {

**public** **static** **void** main(String[] args) {

// Print a message to the user

System.***out***.println("Enter the String");

// Create a Scanner object to read input from the user

Scanner s = **new** Scanner(System.***in***);

// Read the entire line of input from the user

String sentence = s.nextLine();

// Create a StringTokenizer object to split the input string based on ", " delimiters

StringTokenizer tokenizer = **new** StringTokenizer(sentence, ", ");

// Iterate through all the tokens and print each token

**while** (tokenizer.hasMoreTokens()) {

System.***out***.println(tokenizer.nextToken());

}

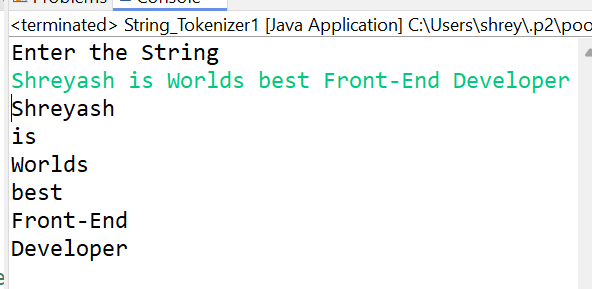
// Close the scanner object to avoid resource leak

s.close();

}

}

**Output:**



1. **Write a Java program that reads a string from the user and uses StringTokenizer to count the number of words in the string.**

**package** myPackage;

**import** java.util.Scanner;

**import** java.util.StringTokenizer;

**public** **class** String\_Tokennizer2 {

**public** **static** **void** main(String[] args) {

// Print a message to prompt user input

System.***out***.println("Enter a string:");

// Create a Scanner object to read input from the user

Scanner scanner = **new** Scanner(System.***in***);

// Read the entire line of input from the user

String inputString = scanner.nextLine();

// Create a StringTokenizer object to split the input string using default delimiters (whitespace)

StringTokenizer tokenizer = **new** StringTokenizer(inputString);

// Use the countTokens method to count the number of tokens (words) in the input string

**int** wordCount = tokenizer.countTokens();

// Print the number of words in the input string

System.***out***.println("The number of words in the string is: " + wordCount);

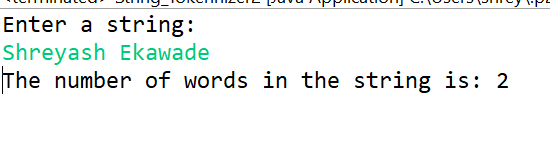
// Close the scanner object to avoid resource leak

scanner.close();

}

}

**Output:**



1. **Write a Java program to create a LinkedList of strings, add elements at specific positions (beginning, middle, end), and print the list.**

**Program:**

**package** myPackage;

**import** java.util.\*;

**public** **class** Linked\_list {

**public** **static** **void** main(String[] args) {

// Create a LinkedList of String type

LinkedList<String> linkedlist = **new** LinkedList<>();

// Add elements to the LinkedList

linkedlist.add("Beginning Name");

linkedlist.add("Middle Name");

linkedlist.add("End name");

// Print the LinkedList before updating

System.***out***.println("Before the Update List: " + linkedlist);

// Update elements in the LinkedList at specified positions

linkedlist.set(0, "Shreyash");

linkedlist.set(1, "Uday");

linkedlist.set(2, "Ekawade");

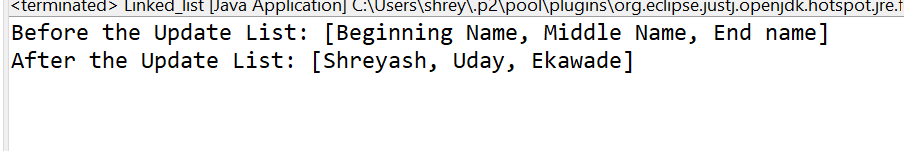
// Print the LinkedList after updating

System.***out***.println("After the Update List: " + linkedlist);

}

}

**Output:**



**4.     Write a Java program to sort a given array list.**

**Program:**

**package** myPackage;

**import** java.util.ArrayList;

**import** java.util.Collections;

**public** **class** Sort\_ArrayList {

**public** **static** **void** main(String[] args) {

// Create an ArrayList of Integer type

ArrayList<Integer> list1 = **new** ArrayList<>();

// Add elements to the ArrayList at specified positions

list1.add(0, 333); // Add the integer 333 at index 0

list1.add(1, 5); // Add the integer 5 at index 1

list1.add(2, 90); // Add the integer 90 at index 2

list1.add(3, 45); // Add the integer 45 at index 3

list1.add(4, 77); // Add the integer 77 at index 4

list1.add(5, 1); // Add the integer 1 at index 5

// Print the ArrayList before sorting

System.***out***.println("Before List: " + list1);

// Sort the ArrayList in ascending order using Collections.sort

Collections.*sort*(list1);

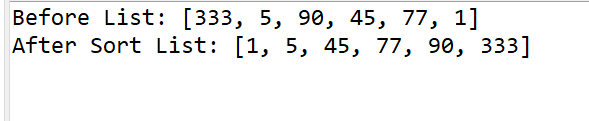
// Print the ArrayList after sorting

System.***out***.println("After Sort List: " + list1);

}

}

**Output:**



**5. Write a Java program to replace the second element of an ArrayList with the specified element.**

**Program:**

**package** myPackage;

**import** java.util.ArrayList;

**import** java.util.Collections;

**public** **class** Array\_List\_Replace\_Method {

**public** **static** **void** main(String[] args) {

// Create an ArrayList of String type

ArrayList<String> list2 = **new** ArrayList<>();

// Add elements to the ArrayList at specified positions

list2.add(0, "Swift"); // Add "Swift" at index 0

list2.add(1, "Bugati"); // Add "Bugati" at index 1

list2.add(2, "Range-Rover"); // Add "Range-Rover" at index 2

// Print the ArrayList before replacing an element

System.***out***.println("Before List: " + list2);

// Replace the element at index 0 with "Harrier"

list2.set(0, "Harrier");

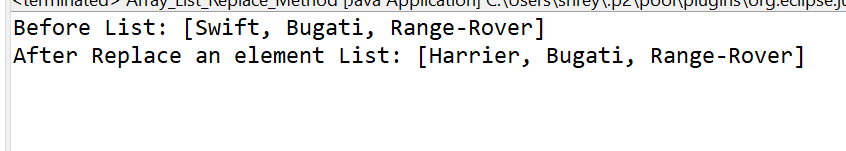
// Print the ArrayList after replacing an element

System.***out***.println("After Replace an element List: " + list2);

}

}

**Output:**



1. **Write a Java program to iterate a linked list in reverse order.**

**Program:**

**package** myPackage;

**import** java.util.Collections;

**import** java.util.LinkedList;

**public** **class** Linked\_List\_ReverseOrder {

**public** **static** **void** main(String[] args) {

// Create a LinkedList of Integer type

LinkedList<Integer> linkedlist3 = **new** LinkedList<>();

// Add elements to the LinkedList

linkedlist3.add(1); // Add the integer 1 to the list

linkedlist3.add(2); // Add the integer 2 to the list

linkedlist3.add(3); // Add the integer 3 to the list

linkedlist3.add(4); // Add the integer 4 to the list

linkedlist3.add(5); // Add the integer 5 to the list

linkedlist3.add(6); // Add the integer 6 to the list

// Print the LinkedList before reversing

System.***out***.println("Before the Update List: " + linkedlist3);

// Reverse the order of elements in the LinkedList

Collections.*reverse*(linkedlist3);

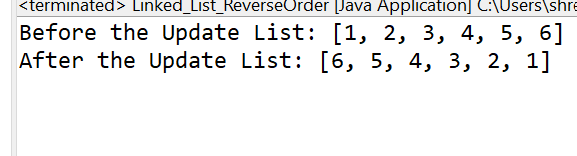
// Print the LinkedList after reversing

System.***out***.println("After the Update List: " + linkedlist3);

}

}

**Output:**



1. **Write a Java program to retrieve, but not remove, the last element of a linked list.**

**Program:**

**package** myPackage;

**import** java.util.\*;

**public** **class** Retrive\_LastElement\_LinkedList {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

// Create and populate a LinkedList

LinkedList<String> list = **new** LinkedList<>();

list.add("Apple");

list.add("Banana");

list.add("Cherry");

list.add("Mango");

list.add("Orange");

// Print the original LinkedList

System.***out***.println("Original LinkedList: " + list+"\n");

// Retrieve but not remove the last element

String lastElement = list.peekLast();

// Print the last element

System.***out***.println("Last element (retrieved but not removed): " + lastElement+"\n");

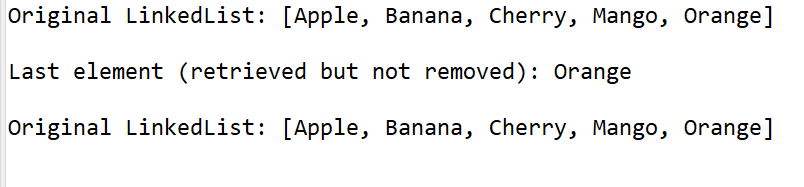
// Print the original LinkedList after retrieved of element

System.***out***.println("Original LinkedList: " + list+"\n");

}

}

**Output:**



**8.     Write a Java program to create a LinkedList of integers and print all the elements.**

**Program:**

**package** myPackage;

**import** java.util.LinkedList;

**public** **class** Linked\_List\_Print\_All\_Elem {

**public** **static** **void** main(String[] args) {

// Create a LinkedList of Integer type

LinkedList<Integer> linkedlist4 = **new** LinkedList<>();

// Add elements to the LinkedList

linkedlist4.add(1); // Add the integer 1 to the list

linkedlist4.add(2); // Add the integer 2 to the list

linkedlist4.add(3); // Add the integer 3 to the list

linkedlist4.add(4); // Add the integer 4 to the list

linkedlist4.add(5); // Add the integer 5 to the list

linkedlist4.add(6); // Add the integer 6 to the list

// Print all elements of the LinkedList

System.***out***.println(linkedlist4);

}

}

**Output:**

