**Java Program to find longest substring without repeating characters in a string.**

**package** friday;

**import** java.util.LinkedHashMap;

**public** **class** MainClass

{

**static** **void** longestSubstring(String inputString)

{

//Convert inputString to charArray

**char**[] charArray = inputString.toCharArray();

//Initialization

String longestSubstring = **null**;

**int** longestSubstringLength = 0;

//Creating LinkedHashMap with characters as keys and their position as values.

LinkedHashMap<Character, Integer> charPosMap = **new** LinkedHashMap<Character, Integer>();

//Iterating through charArray

**for** (**int** i = 0; i < charArray.length; i++)

{

**char** ch = charArray[i];

//If ch is not present in charPosMap, adding ch into charPosMap along with its position

**if**(!charPosMap.containsKey(ch))

{

charPosMap.put(ch, i);

}

//If ch is already present in charPosMap, reposioning the cursor i to the position of ch and clearing the charPosMap

**else**

{

i = charPosMap.get(ch);

charPosMap.clear();

}

//Updating longestSubstring and longestSubstringLength

**if**(charPosMap.size() > longestSubstringLength)

{

longestSubstringLength = charPosMap.size();

longestSubstring = charPosMap.keySet().toString();

}

}

System.***out***.println("Input String : "+inputString);

System.***out***.println("The longest substring : "+longestSubstring);

System.***out***.println("The longest Substring Length : "+longestSubstringLength);

}

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

{

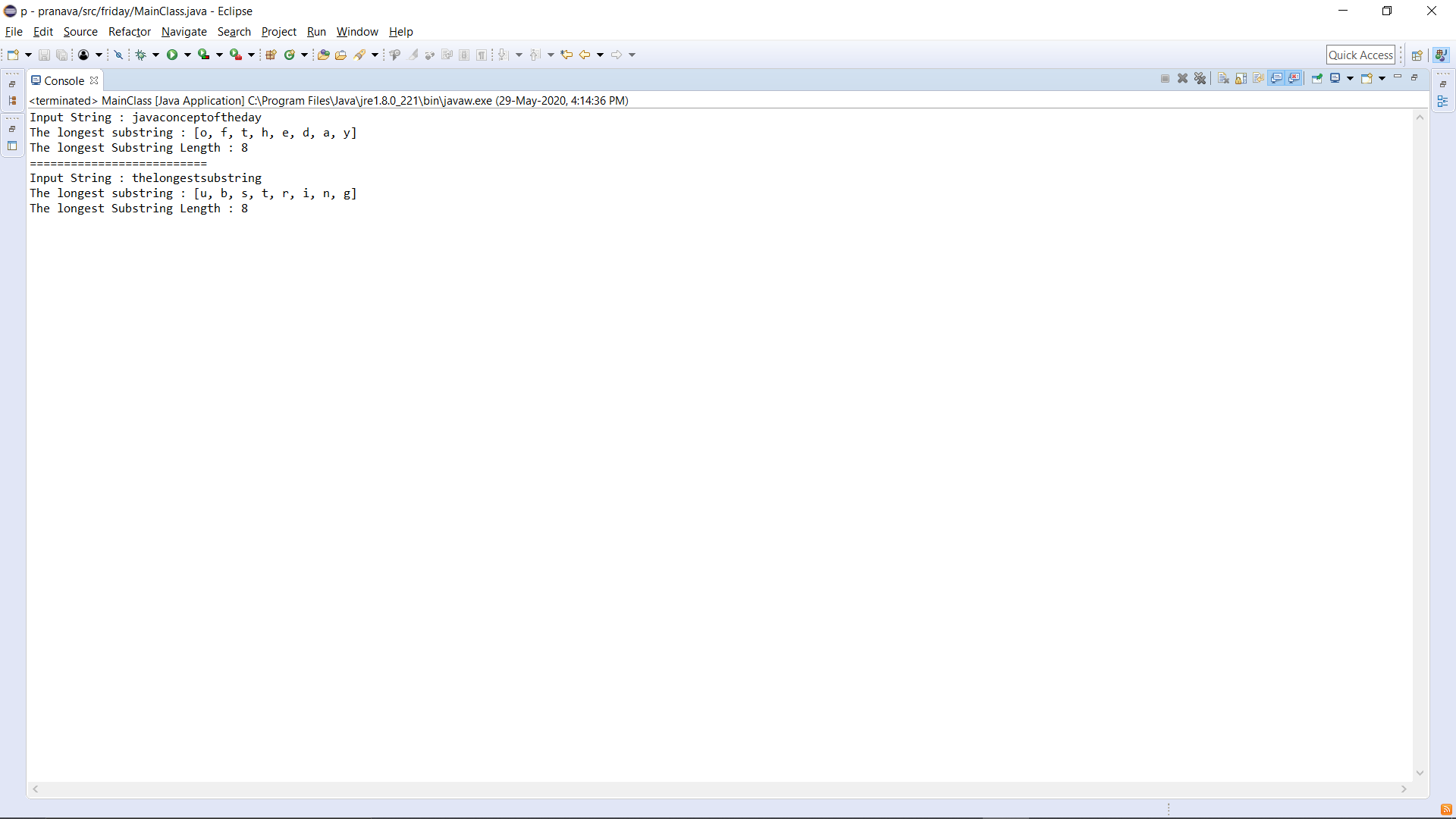
*longestSubstring*("javaconceptoftheday");

System.***out***.println("==========================");

*longestSubstring*("thelongestsubstring");

}

}

**Output:**