 *DEPARTMENT OF INFORMATION TECHNOLOGY*

Experiment No.5

|  |  |  |
| --- | --- | --- |
| **Experiment Number** | **5** | |
| **Experiment Title** | **Functions of String Class and Palindrome** | |
| **Resources / Apparatus Required** | Java Developer Kit , Command Prompt | Standard PC with Windows 7,8 or 10 |
| **Objectives**  **(Skill Set / Knowledge Tested / Imparted)** | To Learn the use of String class in java programs. | |
| **Theory** | A **Java String** contains an immutable sequence of Unicode characters. Unlike C/C++, where string is simply an array of char, A Java String is an object of the class java.lang.  Java String is, however, special. Unlike an ordinary class:   * String is associated with string literal in the form of double-quoted texts such as "Hello, world!” You can assign a string literal directly into a String variable, instead of calling the constructor to create a String instance. * The '+' operator is overloaded to concatenate two String operands. '+' does not work on any other objects such as Point and Circle. * **String is *immutable***. That is, its content cannot be modified once it is created. For example, the method toUpperCase() constructs and returns a new String instead of modifying the its existing content.   // Length  int length() // returns the length of the String  boolean isEmpty() // same as *thisString*.length == 0    // Comparison  boolean equals(String another) // CANNOT use '==' or '!=' to compare two Strings in Java  boolean equalsIgnoreCase(String another)  int compareTo(String another) // return 0 if this string is the same as another;  // <0 if lexicographically less than another; or >0  int compareToIgnoreCase(String another)  boolean startsWith(String another)  boolean startsWith(String another, int fromIndex) // search begins at fromIndex  boolean endsWith(String another)    // Searching & Indexing  int indexOf(String search)  int indexOf(String search, int fromIndex)  int indexOf(int character)  int indexOf(int character, int fromIndex) // search forward starting at fromIndex  int lastIndexOf(String search)  int lastIndexOf(String search, int fromIndex) // search backward starting at fromIndex  int lastIndexOf(int character)  int lastIndexOf(int character, int fromIndex)    // Extracting a char or part of the String (substring)  char charAt(int index) // index from 0 to String's length - 1  String substring(int fromIndex)  String substring(int fromIndex, int endIndex) // exclude endIndex    // Creating a new String or char[] from the original (Strings are immutable!)  String toLowerCase()  String toUpperCase()  String trim() // create a new String removing white spaces from front and back  String replace(char oldChar, char newChar) // create a new String with oldChar replaced by newChar  String concat(String another) // same as *thisString* + another  char[] toCharArray() // create a char[] from this string  void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) // copy into dst char[]    // Static methods for converting primitives to String  static String ValueOf(*type arg*) // *type* can be primitives or char[]  // Static method resulted in a formatted String using format specifiers  static String format(String formattingString, Object... args) // same as printf() | |
| **Program & output** | **All Functions**  import java.util.\*;  class StrAllFunc  {  public static void main(String []args)  {  Scanner s =new Scanner (System.in);    String a = args[0] ;  String b = args[1] ;  System.out.println("Length : "+a.length());  System.out.println("Char At 0 : "+a.charAt(0));  System.out.println("Equals : "+a.equals(b));  System.out.println("Equals Ignore Case : "+a.equalsIgnoreCase(b));  System.out.println("Compare : "+a.compareTo(b));  System.out.println("Compare Ignore Case : "+a.compareToIgnoreCase(b));  System.out.println("Upper Case : "+a.toUpperCase());  System.out.println("Lower Case : "+a.toLowerCase());  System.out.print("Enter Char to Search in a and b : ");  char ch = s.next().charAt(0);  System.out.println(a.indexOf(ch));  System.out.println(b.indexOf(ch));  System.out.print("Enter String to Search : ");  String c = s.next();  System.out.println(a.indexOf(c));  System.out.println(b.indexOf(c));  System.out.print("Enter String to check Start : ");  String d = s.next();  System.out.println(a.startsWith(d));  System.out.println(b.startsWith(d));  System.out.print("Enter String to search and starting index : ");  String e = s.next();  int i = s.nextInt();  System.out.println(a.startsWith(e,i));  System.out.println(b.startsWith(e,i));  System.out.print("Enter String to Check End : ");  String f = s.next();  System.out.println(a.endsWith(f));  System.out.println(b.endsWith(f));  System.out.print("Enter starting and ending index of Substring of First String : ");  int j = s.nextInt();  int k = s.nextInt();  System.out.println(a.substring(j,k));  System.out.print("Enter the starting index of substring for b : ");  int l = s.nextInt();  System.out.println(b.substring(l));  System.out.print("Enter a char or String to check last occurance : ");  String g = s.next();  System.out.println(a.lastIndexOf(g));  System.out.println(b.lastIndexOf(g));  System.out.println("Concat : "+a.concat(b));  }  }  C:\Users\Ashu\Desktop\Coding\Codes\Java Programs\Screen Shots\StrAllFunc.png  **Palindrome**  import java.util.\* ;  class StrPal  {  public static void main(String[]args)  { int i,n ;  n=args[0].length() ;  char[]c =new char[n] ;  c=args[0].toCharArray() ;    for(i=0;i<n;i++)  {  if(c[i]!=c[n-1-i])  {  System.out.print("\nNot A Palindrome\n") ;  System.exit(1) ;  }  }    System.out.print("\nPalindrome\n") ;  }  }  C:\Users\Ashu\Desktop\Coding\Codes\Java Programs\Screen Shots\StrPal.png | |
| **Conclusion** | Hence String class can be used to perform various operations. | |