**Java String class methods**

The java.lang.String class provides many useful methods to perform operations on sequence of char values.

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| 1 | [char charAt(int index)](https://www.javatpoint.com/java-string-charat) | It returns char value for the particular index |
| 2 | [int length()](https://www.javatpoint.com/java-string-length) | It returns string length |
|  |  |  |
| 5 | [String substring(int beginIndex)](https://www.javatpoint.com/java-string-substring) | It returns substring for given begin index. |
| 6 | [String substring(int beginIndex, int endIndex)](https://www.javatpoint.com/java-string-substring) | It returns substring for given begin index and end index. |
| 7 | [boolean contains(CharSequence s)](https://www.javatpoint.com/java-string-contains) | It returns true or false after matching the sequence of char value. |
| 8 | [static String join(CharSequence delimiter, CharSequence... elements)](https://www.javatpoint.com/java-string-join) | It returns a joined string. |
|  |  |  |
| 10 | [boolean equals(Object another)](https://www.javatpoint.com/java-string-equals) | It checks the equality of string with the given object. |
| 11 | [boolean isEmpty()](https://www.javatpoint.com/java-string-isempty) | It checks if string is empty. |
| 12 | [String concat(String str)](https://www.javatpoint.com/java-string-concat) | It concatenates the specified string. |
| 13 | [String replace(char old, char new)](https://www.javatpoint.com/java-string-replace) | It replaces all occurrences of the specified char value. |
| 14 | [String replace(CharSequence old, CharSequence new)](https://www.javatpoint.com/java-string-replace) | It replaces all occurrences of the specified CharSequence. |
| 15 | [static String equalsIgnoreCase(String another)](https://www.javatpoint.com/java-string-equalsignorecase) | It compares another string. It doesn't check case. |
|  |  |  |
|  |  |  |
|  |  |  |
| 19 | [int indexOf(int ch)](https://www.javatpoint.com/java-string-indexof) | It returns the specified char value index. |
| 20 | [int indexOf(int ch, int fromIndex)](https://www.javatpoint.com/java-string-indexof) | It returns the specified char value index starting with given index. |
| 21 | [int indexOf(String substring)](https://www.javatpoint.com/java-string-indexof) | It returns the specified substring index. |
| 22 | [int indexOf(String substring, int fromIndex)](https://www.javatpoint.com/java-string-indexof) | It returns the specified substring index starting with given index. |
| 23 | [String toLowerCase()](https://www.javatpoint.com/java-string-tolowercase) | It returns a string in lowercase. |
| 24 | [String toLowerCase(Locale l)](https://www.javatpoint.com/java-string-tolowercase) | It returns a string in lowercase using specified locale. |
| 25 | [String toUpperCase()](https://www.javatpoint.com/java-string-touppercase) | It returns a string in uppercase. |
| 26 | [String toUpperCase(Locale l)](https://www.javatpoint.com/java-string-touppercase) | It returns a string in uppercase using specified locale. |
| 27 | [String trim()](https://www.javatpoint.com/java-string-trim) | It removes beginning and ending spaces of this string. |
| 28 | [static String valueOf(int value)](https://www.javatpoint.com/java-string-valueof) | It converts given type into string. It is an overloaded method. |

1. //Java Program to illustrate how to declare, instantiate, initialize
2. //and traverse the Java array.
3. class Testarray{
4. public static void main(String args[]){
5. int a[]=new int[5];//declaration and instantiation
6. a[0]=10;//initialization
7. a[1]=20;
8. a[2]=70;
9. a[3]=40;
10. a[4]=50;
11. //traversing array
12. for(int i=0;i<a.length;i++)//length is the property of array
13. System.out.println(a[i]);
14. }}
15. //Java Program to illustrate the use of declaration, instantiation
16. //and initialization of Java array in a single line
17. class Testarray1{
18. public static void main(String args[]){
19. int a[]={33,3,4,5};//declaration, instantiation and initialization
20. //printing array
21. for(int i=0;i<a.length;i++)//length is the property of array
22. System.out.println(a[i]);
23. }}
24. //Java Program to print the array elements using for-each loop
25. class Testarray1{
26. public static void main(String args[]){
27. int arr[]={33,3,4,5};
28. //printing array using for-each loop
29. for(int i:arr)
30. System.out.println(i);
31. }}
32. //Java Program to demonstrate the way of passing an array
33. //to method.
34. class Testarray2{
35. //creating a method which receives an array as a parameter
36. static void min(int arr[]){
37. int min=arr[0];
38. for(int i=1;i<arr.length;i++)
39. if(min>arr[i])
40. min=arr[i];
42. System.out.println(min);
43. }
45. public static void main(String args[]){
46. int a[]={33,3,4,5};//declaring and initializing an array
47. min(a);//passing array to method
48. }}
49. //Java Program to demonstrate the way of passing an anonymous array
50. //to method.
51. public class TestAnonymousArray{
52. //creating a method which receives an array as a parameter
53. static void printArray(int arr[]){
54. for(int i=0;i<arr.length;i++)
55. System.out.println(arr[i]);
56. }
58. public static void main(String args[]){
59. printArray(new int[]{10,22,44,66});//passing anonymous array to method
60. }}
61. public class MatrixAdditionExample{
62. public static void main(String args[]){
63. //creating two matrices
64. int a[][]={{1,3,4},{2,4,3},{3,4,5}};
65. int b[][]={{1,3,4},{2,4,3},{1,2,4}};
67. //creating another matrix to store the sum of two matrices
68. int c[][]=new int[3][3];  //3 rows and 3 columns
70. //adding and printing addition of 2 matrices
71. for(int i=0;i<3;i++){
72. for(int j=0;j<3;j++){
73. c[i][j]=a[i][j]+b[i][j];    //use - for subtraction
74. System.out.print(c[i][j]+" ");
75. }
76. System.out.println();//new line
77. }
78. }}
79. public class MatrixTransposeExample{
80. public static void main(String args[]){
81. //creating a matrix
82. int original[][]={{1,3,4},{2,4,3},{3,4,5}};
84. //creating another matrix to store transpose of a matrix
85. int transpose[][]=new int[3][3];  //3 rows and 3 columns
87. //Code to transpose a matrix
88. for(int i=0;i<3;i++){
89. for(int j=0;j<3;j++){
90. transpose[i][j]=original[j][i];
91. }
92. }
94. System.out.println("Printing Matrix without transpose:");
95. for(int i=0;i<3;i++){
96. for(int j=0;j<3;j++){
97. System.out.print(original[i][j]+" ");
98. }
99. System.out.println();//new line
100. }
101. System.out.println("Printing Matrix After Transpose:");
102. for(int i=0;i<3;i++){
103. for(int j=0;j<3;j++){
104. System.out.print(transpose[i][j]+" ");
105. }
106. System.out.println();//new line
107. }
108. }}

