Click to prove you're human

Java Programs - Java Programming Examples How to Read and Print an Integer Value in Java? Ways to Read Input from Console in Java Java Program to Multiply two Floating-Point Numbers Java Program to Swap Two Numbers Java Program to Add Two Binary Strings Java Program to Add two Complex Numbers Java Program to Check if a Given Integer is Odd or Even Java Program to Find the Largest of three Numbers Java Program to Find LCM of Two Numbers Java Program to Find GCD or HCF of Two Numbers Java Program to Display All Prime Numbers from 1 to N Java Program to Find if a Given Year is a Leap Year Java Program to Check Armstrong Number between Two Integers Java Program to Check If a Number is Neon Number or Not Java Program to Check Whether the Character is Vowel or Consonant Java Program for Factorial of a Number Java Program to Find Sum of Fibonacci Series Numbers of First N Even Indexes Java Program to Calculate Simple Interest Java Program for compound interest Java Program to Find the Perimeter of a Rectangle Java Program to Print Right Triangle Star Pattern Java Program to Print Left Triangle Star Pattern Java Program to Print Pyramid Number Pattern Java Program to Print Reverse Pyramid Star Pattern Java Program to Print Upper Star Triangle Pattern Java Program to Print Mirror Upper Star Triangle Pattern Java Program to Print Downward Triangle Star Pattern Java Program to Print Mirror Lower Star Triangle Pattern Java Program to Print Star Pascalâ€™s Triangle Java Program to Print Diamond Shape Star Pattern Java Program to Print Square Star Pattern Java Program to Print Pyramid Star Pattern Java Program to Print Spiral Pattern of Numbers Java Program to Convert Binary to Octal Java Program to Convert Octal to Decimal Java Program For Decimal to Octal Conversion Java Program for Hexadecimal to Decimal Conversion Java Program For Decimal to Hexadecimal Conversion Java Program for Decimal to Binary Conversion Java Program for Decimal to Binary Conversion Boolean toString() Method in Java Convert String to Double in Java Java Program to Convert Double to String Java Program to Convert String to Long Java Program For Int to Char Conversion Java Program to Convert Char to Int Classes and Objects in Java Abstract Class in Java Singleton Method Design Pattern in Java Java Interface Encapsulation in Java Inheritance in Java Abstraction in Java Difference Between Data Hiding and Abstraction in Java Polymorphism in Java Method Overloading in Java Overriding in Java Super Keyword in Java Java this Keyword static Keyword in Java Access Modifiers in Java Java main() Method - public static void main(String[] args) Difference between static and non-static method in Java HashTable forEach() method in Java with Examples StringBuilder toString() method in Java with Examples StringBuffer codePointAt() method in Java with Examples How compare() method works in Java Short equals() method in Java with Examples Difference Between next() and hasNext() Method in Java Collections What does start() function do in multithreading in Java? Java Thread.start() vs Thread.run() Method Java Program for Linear Search Binary Search in Java Java Program To Recursively Linearly Search An Element In An Array Check If a Value is Present in an Array in Java Java Program to Find Largest Element in an Array Arrays.sort() in Java Java Program to Sort the Elements of an Array in Descending Order Java Program to Sort the Elements of an Array in Ascending Order Remove duplicates from Sorted Array Java Program to Merge Two Arrays Java Program to Check if two Arrays are Equal or not Remove all Occurrences of an Element from Array in Java Java Program to Find Common Elements Between Two Arrays Array Copy in Java Java Program to Left Rotate the Elements of an Array Print a 2D Array or Matrix in Java Java Program to Add two Matrices Sorting a 2D Array according to values in any given column in Java Java Program to Check if two Arrays are Equal or not Java Program to Find Transpose of Matrix Java Program to Find the Determinant of a Matrix Java Program to Find the Normal and Trace of a Matrix Java Program to Print Boundary Elements of the Matrix Java Program to Rotate Matrix Elements Java Program to Compute the Sum of Diagonals of a Matrix Java Program to Interchange Elements of First and Last in a Matrix Across Rows Java Program to Interchange Elements of First and Last in a Matrix Across Columns Java Program to Get a Character from a String Replace a character at a specific index in a String in Java Reverse a String in Java Java Program to Reverse a String using Stack Sort a String in Java (2 different ways) Swapping Pairs of Characters in a String in Java Check if a given string is Pangram in Java Print first letter of each word in a string using regex Java Program to Determine the Unicode Code Point at Given Index in String Remove Leading Zeros From String in Java Compare two Strings in Java Compare two strings lexicographically in Java Java program to print Even length words in a String Insert a String into another String in Java Split a String into a Number of Substrings in Java Learn how to reverse a String in Python. There is no built-in function to reverse a String in Python. The fastest (and easiest?) way is to use a slice that steps backwards, -1. Reverse the string "Hello World": txt = "Hello World"[::-1]print(txt) Try it Yourself » Example Explained We have a string, "Hello World", which we want to reverse: txt = "Hello World"[::-1] print(txt) Create a slice that starts at the end of the string, and moves backwards. In this particular example, the slice statement [::-1] means start at the end of the string and end at position 0, move with the step -1, negative one, which means one step backwards. txt = "Hello World"[::-1] print(txt) Now we have a string txt that reads "Hello World" backwards. Print the String to demonstrate the result txt = "Hello World"[::-1] print(txt) If you like to have a function where you can send your strings, and return them backwards, you can create a function and insert the code from the example above. def my_function(x): return x[::-1]mytxt = my_function("I wonder how this text looks like backwards") print(mytxt) Try it Yourself » Example Explained Create a function that takes a String as an argument. def my_function(x): return x[::-1]mytxt = my_function("I wonder how this text looks like backwards") print(mytxt) Slice the string starting at the end of the string and move backwards. def my_function(x): return x[::-1]mytxt = my_function("I wonder how this text looks like backwards") print(mytxt) Return the backward String def my_function(x): return x[::-1]mytxt = my_function("I wonder how this text looks like backwards") print(mytxt ) Call the function, with a string as a parameter: def my_function(x): return x[::-1]mytxt = my_function("I wonder how this text looks like backwards") print(mytxt) Print the result: def my_function(x): return x[::-1] mytxt = my_function("I wonder how this text looks like backwards") print(mytxt) Today we will learn C program to reverse a string and also how to write a c program without using string function. Reversing a string means changing the positions of the characters in such a way that the last character comes in the first position, second last on the second position and so on. There are so many ways to reverse a string we will see it one by one. In this program, we will use inbuilt library function called as strrev() which reverses the string and print that string on the screen. #include #include int main() { char s[100]; printf("Enter a string to reverse"); gets(s); strrev(s); printf("Reverse of the string: %s", s); return 0; } Output: The first program was very simple because we use library function for reversing a string. But how to reverse a string without using the function. In this program, you will learn how to reverse a string without using a function. #include #include int main() { char Str[100], RevStr[100]; int i, j, len; printf(" Please Enter any String : "); gets(Str); j = 0; len = strlen(Str); for (i = len - 1; i >= 0; i--) { RevStr[j++] = Str[i]; } RevStr[i] = '\0'; printf(" String after Reversing = %s", RevStr); return 0; } Output: Explanation: In the above output, you can observe that str[]=Hello and at the initial stage the value of i is 0 i.e i=0. So, len=strlen(Str)=5.Here strlen is the inbuilt function to find the length of the string i.e 5. 1st Iteration: for(i=len-1;i>=0;i–) i.e for(i=5-1;4>=0;4–).Here the condition (4>=0) is true.Therefore, RevStr[j++]=Str[i] i.e RevStr[0]=Str[4]=o. 2nd Iteration: for(i=len-1;i>=0;i–) i.e for(i=4-1;3>=0;3–).Here the condition (3>=0) is true.Therefore, RevStr[j++]=Str[i] i.e RevStr[1]=Str[3]=l. 3rd Iteration: for(i=len-1;i>=0;i–) i.e for(i=3-1;2>=0;2–).Here the condition (2>=0) is true.Therefore, RevStr[j++]=Str[i] i.e RevStr[2]=Str[2]=l. 4th Iteration: for(i=len-1;i>=0;i–) i.e for(i=2-1;1>=0;1–).Here the condition (1>=0) is true.Therefore, RevStr[j++]=Str[i] i.e RevStr[3]=Str[1]=e. 5th Iteration: for(i=len-1;i>=0;i–) i.e for(i=1-1;0>=0;0–).Here the condition (0>=0) is true.Therefore, RevStr[j++]=Str[i] i.e RevStr[4]=Str[0]=H. 6th Iteration: now the for loop will stop executing and the program will exit. In the above program, we were storing the reverse string in a separate array. In this program, we will not store the reverse string in a separate array. #include #include int main() { char Str[100]; int i, len; printf(" Please Enter String : "); gets(Str); len = strlen(Str); printf(" Reverse string is : "); for (i = len - 1; i >= 0; i--) { printf("%c", Str[i]); } return 0; } Output: In this program, we will use the concept of swapping. The temporary variable temp is declared in the program. #include #include int main() { char Str[100], temp; int i, j, len; printf(" Please Enter any String : "); gets(Str); len = strlen(Str) - 1; for (i = 0; i < strlen(Str)/2; i++) { temp = Str[i]; Str[i] = Str[len]; Str[len--] = temp; } printf(" String after Reversing = %s", Str); return 0; } Output: In this program, we will declare the user-defined function to reverse a string. #include #include void reverse_String(char [], int, int); int main() { char Str[100], temp; int i, j, len; printf(" Please Enter any String : "); gets(Str); len = strlen(Str); reverse_String(Str, 0, len -1); printf(" String after Reversing = %s", Str); return 0; } void reverse_String(char Str[], int i, int len) { char temp; temp = Str[i]; Str[i] = Str[len - i]; Str[len - i] = temp; if (i == len/2) { return; } reverse_String(Str, i + 1, len); } Output: In the above program the user-defined function is reverse_String. Recursion means calling a function again and again till the condition get false. #include char* reverse(char* str); // declaring recursive function void main() { int i, j, k; char str[100]; char *rev; printf("Enter the string:"); gets(str); printf("The original string is:%s ",str); rev = reverse(str); printf(" The reversed string is: "); puts(rev); } char* reverse(char *str) { // defining the function static int i = 0; static char rev[100]; if(*str) { reverse(str+1); rev[i++] = *str; } return rev; } Output: In this program, we will reverse the string with the help of pointers. #include #include char* reverse_String(char *Str) { static int i = 0; static char RevStr[10]; if(*Str) { reverse_String(Str + 1); RevStr[i++] = *Str; } return RevStr; } int main() { char Str[100], temp; int i, j, len; printf(" Please Enter any String : "); gets(Str); printf(" String after Reversing = %s", reverse_String(Str)); return 0; } Output: Also Read: HomeCodeReverse a String in C, C++, Java & Python – Code with Explanation & Examples in Short and Simple MSBTE Solutions #include #include int main() { char s[100]; gets(s); for(int i=strlen(s)-1; i>=0; i--) printf("%c", s[i]); }Input: hello Output: olleh#include #include using namespace std; int main() { string s; getline(cin, s); for(int i=s.size()-1; i>=0; i--) cout =0; i--) System.out.print(s.charAt(i)); } } s = input() print(s[::-1])Input: Python Output: nohtyPIn-Depth Learning – Complete Concept in ParagraphsWhat Does "Reversing a String" Mean?Reversing a string means flipping the characters so that the first character becomes the last, and the last becomes the first. For example, the reverse of "code" is "edoc". This is one of the simplest and most commonly asked programming tasks, and it tests your understanding of loops, string indexing, and basic data handling in various programming languages.In C/C++, we loop from the end index of the string (length - 1) to index 0 by using a for loop and print each character individually.In Java, we utilize charAt(i) within a reverse loop.Python provides the most shorthand method: s[::-1]. This is string slicing, with [::-1] informing Python to take the whole string but in a reversed step.Original index: c o d i n g → Index 0 to 5Reversed index: g n i d o c → Printed from 5 to 0This reverse method is non-destructive (i.e., the original string is not changed unless it is stored in reverse).Imagine reading a word in a mirror. If you type "MIRROR" and read it in a mirror, it looks reversed: "RORRIM". Reversing a string is analogous to reading it through that mirror.Second analogy — reading names on sport jerseys via a rear-view camera!Where and When Is It Used?Reversing a string finds most of its uses in:Palindrome checks (checking a string against its reverse)Text processing (e.g., word reversal, reading messages back)Interview questions (simple demonstration of logic)Practice recursion (reverse a string recursively)Also used in encryption, compression, or pretty-printing output displays.Time and Space ComplexityTime Complexity: O(n) → you have to go through each character onceO(1) if done in-place (such as printing in reverse)O(n) if kept in a fresh variable (as in slicing or constructing a reversed string)Python's slicing is secretly making a copy, but it's fast and memory efficient.Python's slicing is very, very powerful:s[::-1] # reverses the entire strings[::2] # selects every second chars[1::2] # skips first, then every secondMastering this unlocks powerful text-processing powers with minimal code.SEO-Optimized Natural Paragraph for RankingNeed to reverse a string in C, C++, Java, or Python? This tutorial provides the shortest, most efficient code examples for reversing a string with and without loops. String reversal is among the fundamental programming problems and finds extensive use in text processing, palindrome checking, encryption, and competitive programming. Find out how to reverse a string with basic loops in Java/C/C++ and how Python just makes it easy with slicing. This tutorial contains live examples, analogies, and performance-enhanced code for mastering this basic string operation. Try it on GfG Practice Given a string s, the task is to reverse the string. Reversing a string means rearranging the characters such that the first character becomes the last, the second character becomes second last and so on.Examples:Input: s = "GeeksforGeeks"Output: "skeeGrofskeeG"Explanation : The first character G moves to last position, the second character e moves to second-last and so on.Input: s = "abdcfe"Output: "efcdba"Explanation: The first character a moves to last position, the second character b moves to second-last and so on. Using backward traversal – O(n) Time and O(n) SpaceThe idea is to start at the last character of the string and move backward, appending each character to a new string res. This new string res will contain the characters of the original string in reverse order. C++ // C++ program to reverse a string using backward traversal #include #include using namespace std; string reverseString(string& s) { string res; // Traverse on s in backward direction // and add each charecter to a new string for (int i = s.size() - 1; i >= 0; i--) { res += s[i]; } return res; } int main() { string s = "abdcfe"; string res = reverseString(s); cout = 0; i--) { res[j] = s[i]; j++; } // Null-terminate the result string res[n] = '\0'; return res; } int main() { char s[] = "abdcfe"; char *res = reverseString(s); printf("%s", res); return 0; } Java // Java program to reverse a string using backward traversal class GfG { static String reverseString(String s) { StringBuilder res = new StringBuilder(); // Traverse on s in backward direction // and add each character to a new string for (int i = s.length() - 1; i >= 0; i--) { res.append(s.charAt(i)); } return res.toString(); } public static void main(String[] args) { String s = "abdcfe"; String res = reverseString(s); System.out.print(res); } } Python # Python program to reverse a string using backward traversal def reverseString(s): res = [] # Traverse on s in backward direction # and add each character to the list for i in range(len(s) - 1, -1, -1): res.append(s[i]) # Convert list back to string return ''.join(res) if __name__ == "__main__": s = "abdcfe" print(reverseString(s)) C# // C# program to reverse a string using backward traversal using System; using System.Text; class GfG { static string reverseString(string s) { StringBuilder res = new StringBuilder(); // Traverse on s in backward direction // and add each character to a new string for (int i = s.Length - 1; i >= 0; i--) { res.Append(s[i]); } // Convert StringBuilder to string return res.ToString(); } static void Main(string[] args) { string s = "abdcfe"; string res = reverseString(s); Console.WriteLine(res); } } JavaScript // JavaScript program to reverse a string using backward traversal function reverseString(s) { let res = []; // Traverse on s in backward direction // and add each character to the array for (let i = s.length - 1; i >= 0; i--) { res.push(s[i]); } return res.join(''); } const s = "abdcfe"; console.log(reverseString(s)); Time Complexity: O(n) for backward traversalAuxiliary Space: O(n) for storing the reversed string.Using Two Pointers - O(n) Time and O(1) SpaceThe idea is to maintain two pointers: left and right, such that left points to the beginning of the string and right points to the end of the string. While left pointer is less than the right pointer, swap the characters at these two positions. After each swap, increment the left pointer and decrement the right pointer to move towards the center of the string. This will swap all the characters in the first half with their corresponding character in the second half.Illustration: C++ // C++ program to reverse a string using two pointers #include using namespace std; string reverseString(string &s) { int left = 0, right = s.length() - 1; // Swap characters from both ends till we reach // the middle of the string while (left < right) { swap(s[left], s[right]); left++; right--; } return s; } int main() { string s = "abdcfe"; cout = r) return; swap(s[l], s[r]); // Recur for the remaining string reverseStringRec(s, l + 1, r - 1); } // function to reverse a string string reverseString(string &s) { int n = s.length(); reverseStringRec(s, 0, n - 1); return s; } int main() { string s = "abdcfe"; cout = r) return; // Swap the characters at the ends char temp = s[l]; s[l] = s[r]; s[r] = temp; // Recur for the remaining string reverseStringRec(s, l + 1, r - 1); } // Function to reverse a string static String reverseString(String s) { char[] arr = s.toCharArray(); reverseStringRec(arr, 0, arr.length - 1); return new String(arr); } public static void main(String[] args) { String s = "abdcfe"; System.out.println(reverseString(s)); } } Python # Python program to reverse a string using Recursion # Recursive Function to reverse a string def reverseStringRec(arr, l, r): if l >= r: return # Swap the characters at the ends arr[l], arr[r] = arr[r], arr[l] # Recur for the remaining string reverseStringRec(arr, l + 1, r - 1) def reverseString(s): # Convert string to list of characters arr = list(s) reverseStringRec(arr, 0, len(arr) - 1) # Convert list back to string return ''.join(arr) if __name__ == "__main__": s = "abdcfe" print(reverseString(s)) C# // C# program to reverse a string using Recursion using System; using System.Text; class GfG { // recursive function to reverse a string from l to r static void reverseStringRec(StringBuilder s, int l, int r) { if (l >= r) return; char temp = s[l]; s[l] = s[r]; s[r] = temp; // Recur for the remaining string reverseStringRec(s, l + 1, r - 1); } // function to reverse a string static string reverseString(string input) { StringBuilder s = new StringBuilder(input); int n = s.Length; reverseStringRec(s, 0, n - 1); return s.ToString(); } static void Main() { string s = "abdcfe"; Console.WriteLine(reverseString(s)); } } JavaScript // JavaScript program to reverse a string using Recursion // Recursive Function to reverse a string function reverseStringRec(res, l, r) { if (l >= r) return; // Swap the characters at the ends [res[l], res[r]] = [res[r], res[l]]; // Recur for the remaining string reverseStringRec(res, l + 1, r - 1); } function reverseString(s) { // Convert string to array of characters let res = s.split(''); reverseStringRec(res, 0, res.length - 1); // Convert array back to string return res.join(''); } let s = "abdcfe"; console.log(reverseString(s)); Time Complexity: O(n) where n is length of stringAuxiliary Space: O(n)Using Stack - O(n) Time and O(n) SpaceThe idea is to use stack for reversing a string because Stack follows Last In First Out (LIFO) principle. This means the last character you add is the first one you'll take out. So, when we push all the characters of a string into the stack, the last character becomes the first one to pop. Illustration: C++ // C++ program to reverse a string using stack #include using namespace std; string reverseString(string &s) { stack st; // Push the charcters into stack for (int i = 0; i < s.size(); i++) { st.push(s[i]); } // Pop the characters of stack into the original string for (int i = 0; i < s.size(); i++) { s[i] = st.top(); st.pop(); } return s; } int main() { string s = "abdcfe"; cout 0) res.Append(st.Pop()); // Convert the StringBuilder back to a string return res.ToString(); } static void Main() { string s = "abdcfe"; Console.WriteLine(reverseString(s)); } } JavaScript // JavaScript program to reverse a string using stack function reverseString(s) { // To store the characters of the original string. const stack = []; // Push the characters into the stack. for (let i = 0; i < s.length; i++) { stack.push(s[i]); } // Create an array to hold the reversed characters const res = new Array(s.length); // Pop the characters of the stack and store in the array for (let i = 0; i < s.length; i++) { res[i] = stack.pop(); } // Join the array to form the reversed string return res.join(''); } let str = "abdcfe"; console.log(reverseString(str)); Time Complexity: O(n) Auxiliary Space: O(n)Using Inbuilt methods - O(n) Time and O(1) SpaceThe idea is to use built-in reverse method to reverse the string. If built-in method for string reversal does not exist, then convert string to array or list and use their built-in method for reverse. Then convert it back to string. C++ #include using namespace std; string reverseString(string &s) { reverse(s.begin(), s.end()); return s; } int main() { string s = "abdcfe"; cout