

**Trimester: I/II/III Subject: Programming and Problem Solving**

**Name**: Krishnaraj Thadesar **Division:** 9

**Roll No.:** 109054 **Batch:** I3

**Experiment No.:** 5

**Name of the Experiment:** Write a Menu driven C program to perform all String operations using User Defined functions.

**Performed on:** 10rd February 2022

**Submitted on:** 15th February 2022

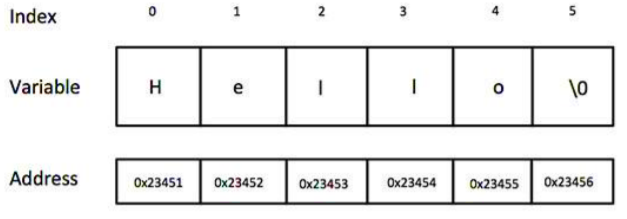
**AIM**: Write a Menu driven C program to perform all String operations using User Defined functions.

**OBJECTIVE:**

To learn string Operations in C.

**THEORY:**

Strings are actually one-dimensional array of characters terminated by a null character '\0'



**String Library Functions:**

1. **strcpy(s1, s2);**

Copies string s2 into string s1.

1. **strcat(s1, s2);**

Concatenates string s2 onto the end of string s1.

1. **strlen(s1);**

Returns the length of string s1.

**strcmp(s1, s2);**

Returns 0 if s1 and s2 are the same; less than 0 if s1s2.

# PLATFORM: *64-Bit ArchLinux x86 with gdb/g++ compiler.*

# ALGORITHM:

Step 1: Start

Step 1: Start

Step 2: Declare/create user defined string functions to compute the **length** of a string, strlength(), to **concatenate** two strings, strconcatenate(), to **compare** two strings, strcompare() and to **copy** a string, strcopy()

Step 3: Declare two arrays of type character, str1[], str2[] and variable c

Step 4: Read strings, str1[], str2[],c

Step 5: Declare switch case statement

switch(c)

case 1: Compute the length of a string with user defined function strlength()

break;

case 2: Concatenate a two strings with user defined function strconcatenate()

break;

case 3: Compare a two strings with user defined function strcompare()

break;

case 4: Copy a one string from another string with user defined function strcopy()

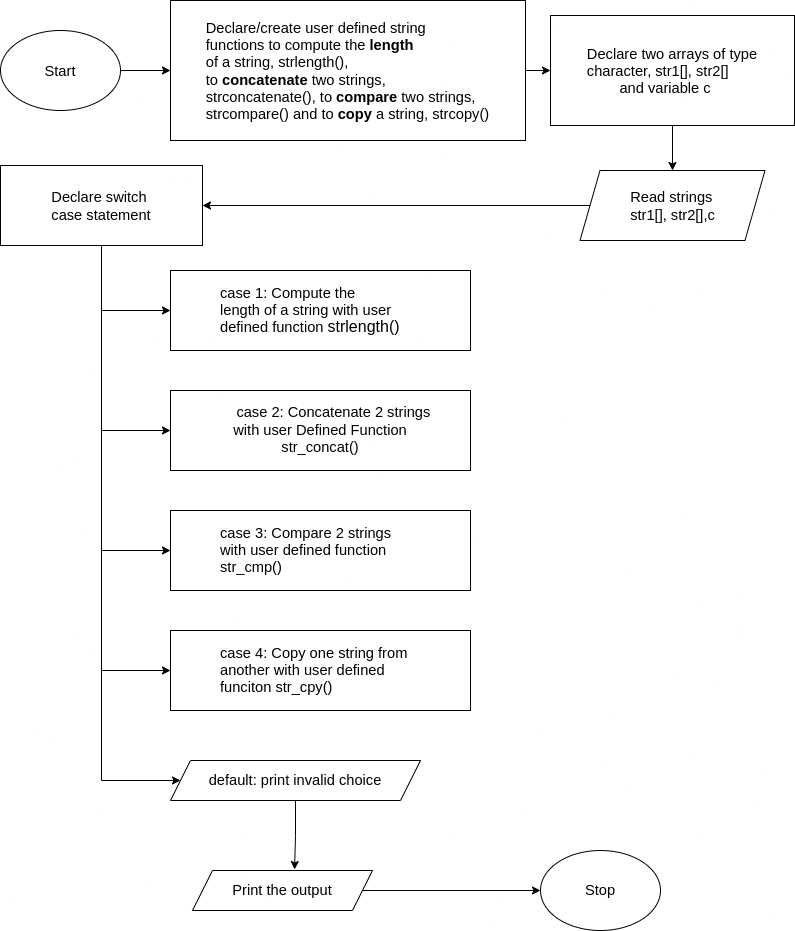
break;

default:

Invalid choice

Step 6: Stop

# Flowchart:



# CODE:

//     strlen() computes string's length strcpy() copies a string to another

//     strcat() concatenates(joins) two strings

//     strcmp() compares two strings

//     strlwr() converts string to lowercase

//     strupr() converts string to uppercase

// Write a menu driven program to perform all string operations (user defined functions)

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

// User Defined Functions

int str\_length(char \*ptr)

{

    /\*

        Function: Returns the number of characters in the given character.

        Input: char \* pointing to the character array.

        Returns: Integer.

     \*/

    int count = 0;

    for (int i = 0; ptr[i] != '\0'; i++)

    {

        count++;

    }

    return count;

}

char \*str\_concat(char \*user\_string\_1, char \*user\_string\_2)

{

    /\*

        Function: Returns a character pointer pointing to an array of characters that is made by concatenating 2 strings.

        Input: char \* pointing to the 2 strings.

        Returns: char \*.

     \*/

    // allocating on the heap coz otherwise it would be a local variable

    // that you cant pass outside the scope of this function as a pointer, as memory would be invalid.

    char \*concat\_string = malloc(1000);

    strcpy(concat\_string, user\_string\_1);

    for (int i = 0; i <= str\_length(user\_string\_2); i++)

    {

        concat\_string[str\_length(user\_string\_1) + i] = user\_string\_2[i];

    }

    return concat\_string;

}

int str\_compare(char \*user\_string\_1, char \*user\_string\_2)

{

    /\*

        Compares the C string str1 to the C string str2.

        This function starts comparing the first character of each string.

        If they are equal to each other, it continues with the following pairs until the characters

        differ or until a terminating null-character is reached.

        Returns:

        <0  the first character that does not match has a lower value in ptr1 than in ptr2

        0   the contents of both strings are equal

        >0  the first character that does not match has a greater value in ptr1 than in ptr2

    \*/

    int result = 0;

    for (int i = 0; user\_string\_1[i] != '\0' || user\_string\_2[i] != '\0'; i++)

    {

        if (user\_string\_1[i] == user\_string\_2[i])

        {

            if (user\_string\_1[i + 1] == '\0' && user\_string\_2[i + 1] != '\0')

            {

                result = 0;

                continue;

            }

            else if (user\_string\_1[i + 1] == '\0' && user\_string\_2[i + 1] != '\0')

            {

                result = -1;

            }

            else if (user\_string\_1[i + 1] == '\0' && user\_string\_2[i + 1] == '\0')

            {

                result = 0;

            }

            if (user\_string\_1[i + 1] != '\0' && user\_string\_2[i + 1] == '\0')

            {

                result = 1;

            }

        }

        else if (user\_string\_1[i] < user\_string\_2[i] || user\_string\_1[i] > user\_string\_2[i])

        {

            result = (user\_string\_1[i] - user\_string\_2[i]) / abs(user\_string\_1[i] - user\_string\_2[i]);

            break;

        }

    }

}

void str\_cpy(char \*str\_to\_copy, char \*user\_string\_2)

{

/\*

Function: Copies string 2 to string 1.

Input: char \* pointing to the 2 strings.

\*/

for (int i = 0; i < str\_length(user\_string\_2); i++)

{

str\_to\_copy[i] = user\_string\_2[i];

}

}

char \*str\_lower(char \*user\_string)

{

    /\*

    Returns a new char \* to an array that contains the converted lowercase of the user\_string

    \*/

    char \*lower\_string = malloc(1000);

    strcpy(lower\_string, user\_string);

    for (int i = 0; i < str\_length(lower\_string); i++)

    {

        if (lower\_string[i] >= 'A' && lower\_string[i] <= 'Z')

        {

            int AASCI\_val = lower\_string[i] + 32;

            lower\_string[i] = AASCI\_val;

        }

    }

    return lower\_string;

}

char \*str\_upper(char \*user\_string)

{

    /\*

    Returns a new char \* to an array that contains the converted uppercase of the user\_string

    \*/

    char \*upper\_string = malloc(1000);

    strcpy(upper\_string, user\_string);

    for (int i = 0; i < str\_length(upper\_string); i++)

    {

        if (upper\_string[i] >= 'A' && upper\_string[i] <= 'Z')

        {

            int AASCI\_val = upper\_string[i] - 32;

            upper\_string[i] = AASCI\_val;

        }

    }

    return upper\_string;

}

char \*str\_reverse(char \*user\_string)

{

    /\*

    Returns a new char \* to an array that contains the reversed user\_string

    \*/

    // allocating on the heap coz otherwise it would be a local variable

    // that you cant pass outside the scope of this function as a pointer, as memory would be invalid.

    char \*rev\_string = malloc(1000);

    strcpy(rev\_string, user\_string);

    for (int i = 0; i < str\_length(user\_string); i++)

    {

        rev\_string[i] = user\_string[str\_length(user\_string) - i - 1];

    }

    rev\_string[str\_length(user\_string)] = '\0';

    return rev\_string;

}

int main()

{

    int choice = 0;

    char user\_string[500];

    char user\_string\_1[500], user\_string\_2[500];

    printf("Enter What operation you want to perform [1, 2, 3, 4, 5]: \n\

        1. Find the length of the String\n\

        2. Concatenate 2 Strings\n\

        3. Compare 2 Strings\n\

        4. Convert a String to lowercase\n\

        5. Convert a String to Uppercase\n\

        6. Reverse a string\n\

        ");

    scanf("%d", &choice);

    switch (choice)

    {

    case 1:

        printf("Enter the String that you want to find the length of: ");

        scanf("%s", &user\_string);

        printf("The Length is: %d", str\_length(user\_string));

        break;

    case 2:

        printf("Enter the First String: ");

        scanf("%s", &user\_string\_1);

        printf("Enter the First String: ");

        scanf("%s", &user\_string\_2);

        printf("The Concatenated is: %s", str\_concat(user\_string\_1, user\_string\_2));

        break;

    case 3:

        printf("Enter the First String: ");

        scanf("%s", &user\_string\_1);

        printf("Enter the First String: ");

        scanf("%s", &user\_string\_2);

        printf("The Comparison of the Strings is: %d", str\_compare(user\_string\_1, user\_string\_2));

        break;

    case 4:

        printf("Enter the String that you want to convert to lowercase to: ");

        scanf("%s", &user\_string);

        printf("The converted String is: %s", str\_lower(user\_string));

        break;

    case 5:

        printf("Enter the String that you want to convert to Uppercase to: ");

        scanf("%s", &user\_string);

        printf("The converted String is: %s", str\_upper(user\_string));

        break;

    case 6:

        printf("Enter the String that you want to reverse: ");

        scanf("%s", &user\_string);

        printf("The converted String is: %s", str\_reverse(user\_string));

        break;

case 7:

printf(" the String that you want to copy: ");

scanf("%s", &user\_string\_2);

str\_cpy(user\_string\_1, user\_string\_2);

printf("The copied strings are: %s and %s", user\_string\_1, user\_string\_1);

break;

    default:

        printf("Incorrect Choice, Please try again.");

    }

    return 0;

}

**OUTPUT**

What operation you want to perform [1, 2, 3, 4, 5]:

        1. Find the length of the String

        2. Concatenate 2 Strings

        3. Compare 2 Strings

        4. Convert a String to lowercase

        5. Convert a String to Uppercase

        6. Reverse a string

        1

Enter the String that you want to find the length of: example

The Length is: 7

2

Enter the First String: example

Enter the First String: String

The Concatenated is: exampleString

3

Enter the First String: First

Enter the First String: Second

The Comparison of the Strings is: -1

3

Enter the First String: First

Enter the First String: First

The Comparison of the Strings is: 0

4

Enter the String that you want to convert to lowercase to: LOWERcase

The converted String is: lowercase

5

Enter the String that you want to convert to Uppercase to: upperCASE

The converted String is: UPPERCASE

6

Enter the String that you want to reverse: reverse

The converted String is: esrever

7

the String that you want to copy: hello\_world

The copied strings are: hello\_world and hello\_world

**CONCLUSION:**

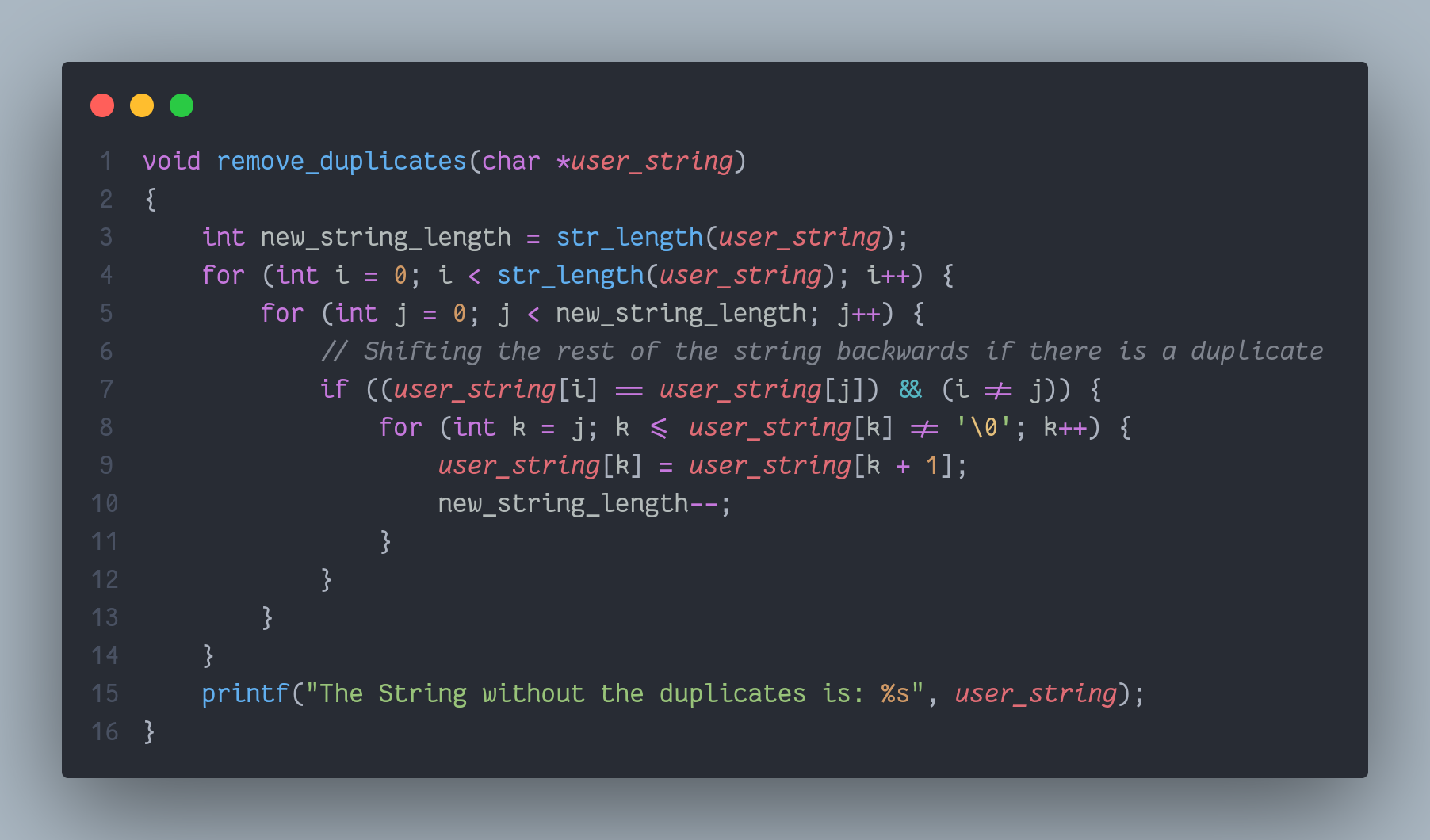
Thus we have learned various string operations in C

***FAQs:***

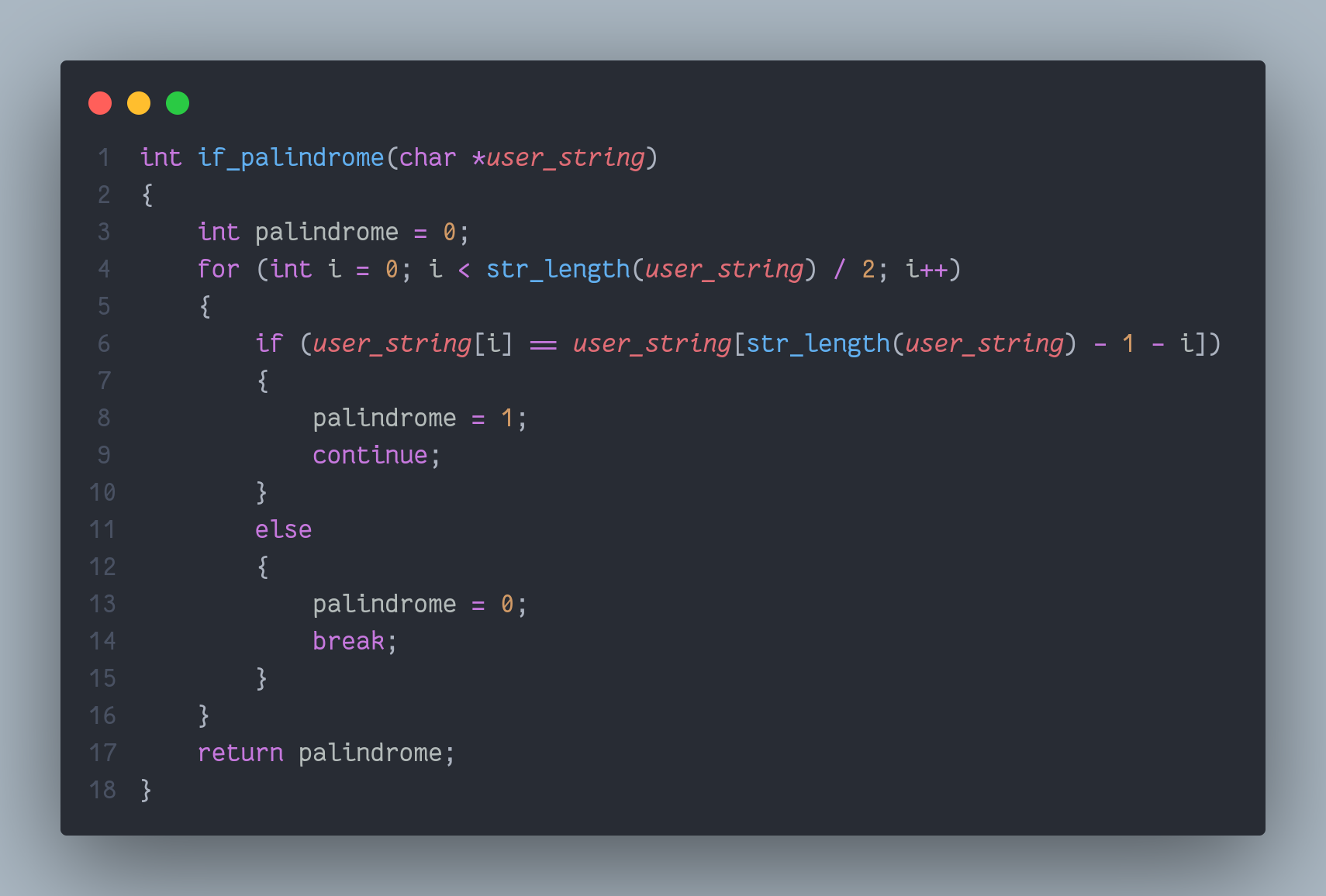
*Q1. How to find the maximum occurring character in given String?*

**

*Q2. How to remove all duplicates from a given string?*

**

*Q3. How do you check if a given String is Palindrome or not?*

**