String

1. Convert string to lowercase and uppercase string.

> Test Data

```
Enter your name: Parth
```

> Expected Output

```
Uppercase: PARTH
Lowercase: parth
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void uppercase(char *);
    void lowercase(char *);
    char str[30];

    printf("Enter your name: ");
    fgets(str, 30, stdin);
    // Uppercase
    uppercase(str);
    printf("\nUppercase: %s", str);
    // Lowercase
    lowercase(str);
    printf("\nLowercase: %s", str);

    return 0;
}
// Uppercase
void uppercase(char *str){
    int i, l;
    l = strlen(str);
    for(i = 0; i < l-1; i++){
        if(str[i] >= 'a' && str[i] <= 'z')
```

```
            str[i] = str[i] - 32;
    }
}
// Lowercase
void lowercase(char *str){
    int i, l;
    l = strlen(str);
    for(i = 0; i < l-1; i++){
        if(str[i] >= 'A' && str[i] <= 'Z')
            str[i] = str[i] + 32;
    }
}
```

2. Reverse a string

> Test Data

```
Enter a string: Reverse
```

> Expected Output

```
Reverse String: esreveR
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    void reverse(char *);
    char str[30];

    printf("Enter a String: ");
    fgets(str, 30, stdin);

    reverse(str);

    printf("Reverse String: %s", str);
    return 0;
}

void reverse(char *str){
    int i, l = strlen(str);
    for(i = 0; i < l/2; i++){
        int temp = str[i];
        str[i] = str[l-i-2];
        str[l-i-2] = temp;
    }
}
```

3. A String is Palindrome or Not

> Test Data

```
Enter a string: racecar
```

> Expected Output

```
Palindrome
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void reverse(char *);
    char str[30];
    char str1[30];
    int result;
    printf("Enter a String: ");
    fgets(str, 30, stdin);

    strcpy(str1, str);
    reverse(str);
    result = strcmp(str, str1);
    if(result == 0)
        printf("Palindrome");
    else
        printf("Not palindrome");
    return 0;
}

void reverse(char *str){
    int i, l = strlen(str);
    for(i = 0; i < l/2; i++){
        int temp = str[i];
        str[i] = str[l-i-2];
        str[l-i-2] = temp;
    }
}
```

4. Find the length of the string

> Test Data

```
Enter a string: Hello World!
```

> Expected Output

```
Using library function: 12
Using loop: 12
```

> Source Code

```c
#include <stdio.h>
#include <string.h>
```

```
int main(){
    char str[30];
    int length(char *);

    printf("Enter a string: ");
    fgets(str, 30, stdin);
    // using library function
    printf("\nUsing Library function: %d", strlen(str));
    // Using loop to calculate the length of string
    printf("\nUsing Loop: %d", length(str));
    return 0;
}

int length(char *str){
    int i;
    for(i = 0; str[i]; i++);
    return i;
}
```

5. A String Is an Anagram or Not

> Test Data

```
String 1: listen
String 2: silent
```

> Expected Output

```
'Anagram'
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    int isAnagram(char *, char *);
    char str1[30];
    char str2[30];

    printf("String 1: ");
    fgets(str1, 30, stdin);
    printf("String 2: ");
    fgets(str2, 30, stdin);

    if(isAnagram(str1, str2))
        printf("'Anagram'");
    else
        printf("'Not Anagram'");
    return 0;
}

int isAnagram(char *s1, char *s2){
    int i, j, flag;
    if(strlen(s1) != strlen(s2))
```

```
        return 0;
    for(i = 0; i < strlen(s1); i++){
        flag = 0;
        for(j = 0; j < strlen(s1); j++){
            if(s1[i] == s2[j]){
                flag = 1;
                break;
            }
        }
        if(flag == 0)
            return 0;
    }
    return flag;
}
```

6. Copy one string to another string.

> Test Data

```
Enter a string: Zero
```

> Expected Output

```
Copied string: Zero
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    void copyStr(char *, char *);
    char str1[30];
    char str2[30];
    char str3[30];

    printf("Enter a string: ");
    fgets(str1, 30, stdin);

    // Using library function
    strcpy(str2, str1);
    printf("Library Function: %s", str2);

    // Without using library function
    copyStr(str3, str1);
    printf("Using Loop: %s", str3);
    return 0;
}
void copyStr(char *str2, char *str1){
    int i;
    for(i = 0;str1[i]; i++){
        str2[i] = str1[i];
    }
    str2[i] = '\0';
}
```

7. Concatenate two strings.

> Test Data

```
Enter first string: Hello
Enter second string: World
```

> Expected Output

```
Concatenation string: Hello World
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void concat(char *, char *);
    char str1[60];
    char str2[30];

    printf("Enter first string: ");
    gets(str1);
    printf("Enter second string: ");
    gets(str2);

    concat(str1, str2);
    printf("Concatenation string: %s", str1);
    return 0;
}
void concat(char *s1, char *s2){
    int i, l1, l2, j=0;
    l1 = strlen(s1);
    l2 = strlen(s2);
    for(i = l1;i < l1+l2+1; i++){
        s1[l1] = ' ';
        s1[i+1] = s2[j];
        j++;
    }
    s1[i] = '\0';
}
```

8. Compare two string

> Test Data

```
Enter first string: Hello
Enter second string: Hello
```

> Expected Output

```
Same string
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    int comStr(char *, char *);
    char str1[30];
    char str2[30];

    printf("Enter first string: ");
    gets(str1);
    printf("Enter second string: ");
    gets(str2);

    if(comStr(str1, str2))
        printf("Same String");
    else
        printf("Not Same");
    return 0;
}
int comStr(char *s1, char *s2){
    int i, l1, l2;
    l1 = strlen(s1);
    l2 = strlen(s2);
    if(l1 != l2)
        return 0;
    for(i = 0; i < l1; i++){
        if(s1[i] != s2[i])
            return 0;
    }
    return 1;
}
```

9. Toggle case of each character of a string.

> Test Data

```
Enter the string: Hello World
```

> Expected Output

```
String in toggle case: hELLO wORLD
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void changeCase(char *);
    char str[30];
```

```
    printf("Enter the string: ");
    gets(str);

    changeCase(str);

    printf("String in toggle case: %s", str);
    return 0;
}

void changeCase(char *str){
    int i, l;
    l = strlen(str);
    for(i = 0; i < l;i++){
        if(str[i] >= 'A' && str[i] <= 'Z')
            str[i] = str[i] + 32;
        else if(str[i] >= 'a' && str[i] <= 'z')
            str[i] = str[i] - 32;
        printf("%d ", str[i]);
    }
}
```

10. Find a total number of alphabets, digits or special character in a string.

> Test Data

```
Enter the string: '5' has the int value 53
```

> Expected Output

```
Alphabet = 14 digits = 3 Special character = 7
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    void findADS(char *);
    char str[30];

    printf("Enter the string: ");
    gets(str);

    findADS(str);
    return 0;
}

void findADS(char *str){
    int i, l, a=0, d=0, s=0;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] >= '0' && str[i] <= '9')
            d++;
        else if(str[i] >= 'a' && str[i] <= 'z' || str[i]>= 'A' && str[i] <= 'Z')
            a++;
```

```
        else
            s++;
    }
    printf("Alphabet = %d digits = %d Special Character = %d", a, d, s);
}
```

11. Count the total number of vowels and consonants in a string.

> Test Data

```
Enter the string: Hello World
```

> Expected Output

```
Vowels = 3 Consonant = 8
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    void countVC(char *);
    char str[30];

    printf("Enter the string: ");
    gets(str);

    countVC(str);
    return 0;
}

void countVC(char *str){
    int i, l, c=0, v=0;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] ==
            v++;
        else
            c++;
    }
    printf("Vowels = %d Consonant = %d", v, c);
}
```

12. Count the total number of words in a string.

> Test Data

```
Enter the string: Beauty lies in the eyes of beholder
```

> Expected Output

```
Words = 7
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void countWords(char *);
    char str[30];

    printf("Enter the string: ");
    gets(str);

    countWords(str);
    return 0;
}

void countWords(char *str){
    int i, l, cw = 1;
    l = strlen(str);
    if(str[0] == ' ')
        cw = 1;
    for(i = 1; i < l; i++){
        if(str[i] == ' ')
            cw++;
    }
    printf("Words = %d", cw);
}
```

13. Reverse order of words in a given string.

> Test Data

```
Enter the string: I Love Programming very much
```

> Expected Output

```
much very Programming Love I
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

void reverseWord(char *);
void reverse(char *, char *);

int main(){

    char str[50];
```

```
    printf("Enter the string: ");
    gets(str);

    reverseWord(str);
    printf("%s", str);
    return 0;
}

void reverseWord(char *str){
    char *begin = str;
    char *temp = str;
    while(*temp){
        temp++;
        if(*temp == ' '){
            reverse(begin, temp-1);
            begin = temp + 1;
        } else if(*temp == '\0'){
            reverse(begin, temp-1);
        }
    }
    reverse(str, temp-1);
}
void reverse(char *begin, char *end){
    while (begin < end) {
        char temp = *begin;
        *begin++ = *end;
        *end-- = temp;
    }
}
```

14. Find the first occurrence of a character in a given string.

> Test Data

```
Enter the string: Be yourself; everyone else is already taken
Enter the character to be searched: s
```

> Expected Output

```
Character 's' is first occurrence at location: 8
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    int find(char *, char);
    char str[50];
    char ch;
    int f;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be searched: ");
```

```
    scanf("%c", &ch);

    f = find(str, ch);

    printf("Character '%c' is first occurrence at location: %d", ch, f);
    return 0;
}
int find(char *str, char ch){
    int i, l;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            return i + 1;
    }
    return -1;
}
```

15. Find the last occurrence of a character in a given string.

> Test Data

```
Enter the string: Be yourself; everyone else is already taken
Enter the character to be searched: s
```

> Expected Output

```
Character 's' is first occurrence at location: 29
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main(){
    int find(char *, char);
    char str[50];
    char ch;
    int f;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be searched: ");
    scanf("%c", &ch);

    f = find(str, ch);

    printf("Character '%c' is first occurrence at location: %d", ch, f);
    return 0;
}
int find(char *str, char ch){
    int i, l;
    l = strlen(str);
    for(i = l-1; i >= 0; i--){
        if(str[i] == ch)
            return i + 1;
```

```
    }
    return -1;
}
```

16. Enter the string: Keep your eyes on the stars and your feet on the ground

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter the character to be searched: e
```

> Expected Output

```
character 'e' found at location: 2
character 'e' found at location: 3
character 'e' found at location: 11
character 'e' found at location: 13
character 'e' found at location: 21
character 'e' found at location: 39
character 'e' found at location: 40
character 'e' found at location: 48
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    void find(char *, char);
    char str[100];
    char ch;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be searched: ");
    scanf("%c", &ch);

    find(str, ch);
    return 0;
}
void find(char *str, char ch){
    int i, l;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            printf("\ncharacter '%c' found at location: %d", ch, i+1);
    }
}
```

17. Count occurrences of a character in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

> Expected Output

```
number of occurrence of 'e' = 8
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main(){
    int countChar(char *, char);
    char str[100];
    char ch;
    int count;
    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be searched: ");
    scanf("%c", &ch);

    count = countChar(str, ch);

    printf("number of occurrence of '%c' = %d", ch, count);
    return 0;
}
int countChar(char *str, char ch){
    int i, l, c = 0;;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            c++;
    }
    return c;
}
```

18. C Program to Find Highest Frequency Character in a String

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

> Expected Output

```
Max repeated character in the string 'e' It occurs 8 times
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int countCH(char *, char);
```

```
int main(){
    void frequency(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    frequency(str);
    return 0;
}
void frequency(char *str){
    int i, max = 0, o, l;
    char ch;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] != ' ')
            o = countCH(str, str[i]);
        if(max < o){
            max = o;
            ch = str[i];
        }
    }
    printf("Max repeated character in the string '%c' It occurs %d times", ch, max);
}

int countCH(char *str, char ch){
    int i, l, c = 0;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            c++;
    }
    return c;
}
```

19. Find the lowest frequency character in a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

> Expected Output

```
Minimum occurring character: 'K' 'p' 'f' 'g' = 1 times
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int countCH(char *, char);

int main(){
    void frequency(char *);
    char str[100];
```

```
    printf("Enter the string: ");
    gets(str);

    frequency(str);
    return 0;
}
void frequency(char *str){
    int i, min, o, l;
    char ch;
    l = strlen(str);
    min = countCH(str, str[0]);
    ch = str[0];
    for(i = 0; i < l; i++){
        if(str[i] != ' ')
            o = countCH(str, str[i]);
        if(min > o){
            min = o;
            ch = str[i];
        }
    }
    printf("Minimum occurring character: ");
    for(i = 0; i < l; i++){
        o = countCH(str, str[i]);
        if(o == min){
            printf("'%c' ", str[i]);
        }
    }
    printf("= %d times", min);
}

int countCH(char *str, char ch){
    int i, l, c = 0;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            c++;
    }
    return c;
}
```

20. Count the frequency of each character in a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

> Expected Output

```
Frequency of 'K' = 1
Frequency of 'e' = 8
Frequency of 'p' = 1
Frequency of ' ' = 11
Frequency of 'y' = 3
Frequency of 'o' = 5
Frequency of 'u' = 3
Frequency of 'r' = 4
Frequency of 's' = 3
```

```
Frequency of 'n' = 4
Frequency of 't' = 4
Frequency of 'h' = 2
Frequency of 'a' = 2
Frequency of 'd' = 2
Frequency of 'f' = 1
Frequency of 'g' = 1
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int countCH(char *, char);

int main(){
    void frequency(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    frequency(str);
    return 0;
}
void frequency(char *str){
    int i, l, o, f;
    l = strlen(str);
    for(i = 0; i < l; i++){
        o = countCH(str, str[i]);
        for(int j = 0;j < i;j++){
            f = 1;
            if(str[j] == str[i]){
                f = 0;
                break;
            }
        }
        if(f == 1 || i == 0)
            printf("\nFrequency of '%c' = %d", str[i], o);
    }
}

int countCH(char *str, char ch){
    int i, l, c = 0;
    l = strlen(str);
    for(i = 0; i < l; i++){
        if(str[i] == ch)
            c++;
    }
    return c;
}
```

21. Remove the first occurrence of a character from a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

```
Enter the character to be removed: e
```

> Expected Output

```
Kep your eyes on the stars and your feet on the ground
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccur(char *, char);
    char str[100];
    char c;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be removed: ");
    scanf("%c", &c);

    removeOccur(str, c);

    printf("%s", str);
    return 0;
}
void removeOccur(char *str, char c)
{
    int i, j, l, t = 0;
    l = strlen(str);
    for (i = 0; i < l; i++)
    {
        if (str[i] == c && t == 0)
        {
            for (j = i; j < l - 1; j++)
            {
                str[j] = str[j + 1];
            }
            l = l - 1;
            str[l] = '\0';
            t = 1;
        }
    }
}
```

22. Remove the last occurrence of a character from a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter the character to be removed: e
```

> Expected Output

```
Keep your eyes on the stars and your feet on th ground
```

Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccur(char *, char);
    char str[100];
    char c;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be removed: ");
    scanf("%c", &c);

    removeOccur(str, c);

    printf("%s", str);
    return 0;
}
void removeOccur(char *str, char c)
{
    int i, j, l, t = 0;
    l = strlen(str);
    for (i = l - 1; i >= 0; i--)
    {
        if (str[i] == c && t == 0)
        {
            for (j = i; j < l - 1; j++)
            {
                str[j] = str[j + 1];
            }
            l = l - 1;
            str[l] = '\0';
            t = 1;
        }
    }
}
```

23. Delete all occurrences of a character from a string.

Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter the character to be removed: e
```

Expected Output

```
Kp your ys on th stars and your ft on th ground
```

Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccur(char *, char);
    char str[100];
    char c;

    printf("Enter the string: ");
    gets(str);

    printf("Enter the character to be removed: ");
    scanf("%c", &c);

    removeOccur(str, c);

    printf("%s", str);
    return 0;
}
void removeOccur(char *str, char c)
{
    int i, j, l;
    l = strlen(str);
    for (i = 0; i < l; i++)
    {
        if (str[i] == c)
        {
            for (j = i; j < l - 1; j++)
            {
                str[j] = str[j + 1];
            }
            l = l - 1;
            str[l] = '\0';
            i = i - 1;
        }
    }
}
```

24. Remove all repeated characters from a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
```

> Expected Output

```
Kep yoursnthadfg
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
```

```
{
    void removeOccur(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    removeOccur(str);

    printf("%s", str);
    return 0;
}
void removeOccur(char *str)
{
    int i, l, j, k;
    l = strlen(str);
    for (i = 0; i < l - 1; i++)
    {
        for (j = i + 1; j < l; j++)
        {
            if (str[i] == str[j])
            {
                for (k = j; k < l - 1; k++)
                {
                    str[k] = str[k + 1];
                }
                l -= 1;
                str[l] = '\0';
                j = j - 1;
            }
        }
    }
}
```

25. Replace the first occurrence of a character with another in a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Find: e
Replace: _
```

> Expected Output

```
K_ep your eyes on the stars and your feet on the ground
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    void replace(char *, char, char);
    char str[100];
    char fi, re;
```

```
    printf("Enter the string: ");
    gets(str);

    printf("Find: ");
    scanf("%c", &fi);

    fflush(stdin);

    printf("Replace: ");
    scanf("%c", &re);

    replace(str, fi, re);

    printf("%s", str);
    return 0;
}

void replace(char *str, char fi, char re)
{
    int i, l, t = 0;
    l = strlen(str);
    for (i = 0; i < l; i++)
    {
        if (str[i] == fi && t == 0)
        {
            str[i] = re;
            t = 1;
        }
    }
}
```

26. Replace the last occurrence of a character with another in a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Find: e
Replace: _
```

> Expected Output

```
Keep your eyes on the stars and your feet on th_ ground
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    void replace(char *, char, char);
    char str[100];
    char fi, re;

    printf("Enter the string: ");
```

```
    gets(str);

    printf("Find: ");
    scanf("%c", &fi);

    fflush(stdin);

    printf("Replace: ");
    scanf("%c", &re);

    replace(str, fi, re);

    printf("%s", str);
    return 0;
}

void replace(char *str, char fi, char re)
{
    int i, l, t = 0;
    l = strlen(str);
    for (i = l; i >= 0; i--)
    {
        if (str[i] == fi && t == 0)
        {
            str[i] = re;
            t = 1;
        }
    }
}
```

27. Put all occurrences of a character with another in a string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Find: e
Replace: _
```

> Expected Output

```
K__p your _y_s on th_ stars and your f__t on th_ ground
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    void replace(char *, char, char);
    char str[100];
    char fi, re;

    printf("Enter the string: ");
    gets(str);
```

```
    printf("Find: ");
    scanf("%c", &fi);

    fflush(stdin);

    printf("Replace: ");
    scanf("%c", &re);

    replace(str, fi, re);

    printf("%s", str);
    return 0;
}

void replace(char *str, char fi, char re)
{
    int i, l;
    l = strlen(str);
    for (i = l; i >= 0; i--)
    {
        if (str[i] == fi)
        {
            str[i] = re;
        }
    }
}
```

28. Find the first occurrence of a word in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to be searched: eyes
```

> Expected Output

```
Word 'eyes' is first occurrence at location: 10
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    int findOccurrence(char *, char *);
    char str[100];
    char fw[50];
    int idx;

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to be searched: ");
    gets(fw);
```

```
    idx = findOccurrence(str, fw);

    printf("Word '%s' is first occurrence at location: %d", fw, idx);
    return 0;
}

int findOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, temp;
    int findAt = -1;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = 0; i < sl; i++)
    {
        j = 0;
        temp = i;
        while (j < wl)
        {
            if (str[temp] == fw[j])
            {
                j++;
                temp++;
            }
            else
                break;
        }
        if (j == wl)
        {
            findAt = i;
            break;
        }
    }
    return findAt;
}
```

28. Find the last occurrence of a word in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to be searched: eyes
```

> Expected Output

```
Word 'on' is last occurrence at location: 42
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    int findOccurrence(char *, char *);
    char str[100];
    char fw[50];
```

```
    int idx;

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to be searched: ");
    gets(fw);

    idx = findOccurrence(str, fw);

    printf("Word '%s' is last occurrence at location %d", fw, idx);
    return 0;
}

int findOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, temp;
    int findAt = -1;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = sl - 1; i >= 0; i--)
    {
        j = 0;
        temp = i;
        while (j < wl)
        {
            if (str[temp] == fw[j])
            {
                j++;
                temp++;
            }
            else
                break;
        }
        if (j == wl)
        {
            findAt = i;
            break;
        }
    }
    return findAt;
}
```

24. Search all occurrences of a word in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter the word to search: on
```

> Expected Output

```
'on' is found at index: 15
'on' is found at index: 42
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void findOccurrence(char *, char *);
    char str[100];
    char fw[30];

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to search: ");
    gets(fw);

    findOccurrence(str, fw);

    return 0;
}

void findOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, t, idx;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = 0; i < sl; i++)
    {
        t = i;
        j = 0;
        while (j < wl)
        {
            if (str[t] == fw[j])
            {
                j++;
                t++;
            }
            else
            {
                break;
            }
        }
        if (j == wl)
        {
            printf("\n'%s' is found at index: %d", fw, i);
        }
    }
}
```

25. Count occurrences of a word in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to search: on
```

> Expected Output

```
Occurrences of word 'on' = 2 Time
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    int countOccurrence(char *, char *);
    char str[100];
    char fw[30];
    int count;

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to search: ");
    gets(fw);

    count = countOccurrence(str, fw);

    printf("Occurrences of word '%s' = %d Time", fw, count);
    return 0;
}

int countOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, t, idx, count = 0;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = 0; i < sl; i++)
    {
        t = i;
        j = 0;
        while (j < wl)
        {
            if (str[t] == fw[j])
            {
                j++;
                t++;
            }
            else
            {
                break;
            }
        }
        if (j == wl)
        {
            count++;
        }
    }
    return count;
}
```

26. Remove the first occurrence of a word from the string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to be remove: on
```

Expected Output

```
Keep your eyes the stars and your feet on the ground
```

Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccurrence(char *, char *);
    char str[100];
    char fw[50];

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to be remove: ");
    gets(fw);

    removeOccurrence(str, fw);

    printf("%s", str);
    return 0;
}

void removeOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, temp, f = 0;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = 0; i < sl; i++)
    {
        j = 0;
        temp = i;
        while (j < wl)
        {
            if (str[temp] == fw[j] && f == 0)
            {
                j++;
                temp++;
            }
            else
                break;
        }
        if (j == wl)
        {
            for (int k = i; k < sl; k++)
                str[k] = str[k + wl + 1];
            f = 1;
        }
    }
}
```

26. Remove the last occurrence of a word from the string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to be remove: on
```

> Expected Output

```
Keep your eyes on the stars and your feet the ground
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccurrence(char *, char *);
    char str[100];
    char fw[50];

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to be remove: ");
    gets(fw);

    removeOccurrence(str, fw);

    printf("%s", str);
    return 0;
}

void removeOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, temp, f = 0;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = sl - 1; i >= 0; i--)
    {
        j = 0;
        temp = i;
        while (j < wl)
        {
            if (str[temp] == fw[j] && f == 0)
            {
                j++;
                temp++;
            }
            else
                break;
        }
        if (j == wl)
        {
            for (int k = i; k < sl; k++)
                str[k] = str[k + wl + 1];
            f = 1;
        }
```

```
    }
}
```

27. Delete all occurrence of a word in a given string.

> Test Data

```
Enter the string: Keep your eyes on the stars and your feet on the ground
Enter word to be remove: on
```

> Expected Output

```
Keep your eyes the stars and your feet the ground
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void removeOccurrence(char *, char *);
    char str[100];
    char fw[50];

    printf("Enter the string: ");
    gets(str);

    printf("Enter word to be remove: ");
    gets(fw);

    removeOccurrence(str, fw);

    printf("%s", str);
    return 0;
}

void removeOccurrence(char *str, char *fw)
{
    int i, j, sl, wl, temp, k;
    sl = strlen(str);
    wl = strlen(fw);
    for (i = 0; i < sl; i++)
    {
        j = 0;
        temp = i;
        while (j < wl)
        {
            if (str[temp] == fw[j])
            {
                j++;
                temp++;
            }
            else
                break;
        }
```

```
        if (j == wl)
        {
            for (k = i; k < sl; k++)
                str[k] = str[k + wl + 1];
            str[k] = '\0';
            i = i - 1;
        }
    }
}
```

28. C Program To Trim (front) Leading & Trailing White Space Characters From String

> Test Data

```
Enter the string:      Removing leading and trailing white spaces
```

> Expected Output

```
Before: '     Removing leading and trailing white spaces'
After: 'Removing leading and trailing white spaces'
```

> Source Code

```
#include <stdio.h>
#include <string.h>

int main()
{
    void trimSpaces(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    printf("\nBefore: '%s'", str);
    trimSpaces(str);
    printf("\nAfter: '%s'", str);
    return 0;
}

void trimSpaces(char *str)
{
    int i, l, j;
    l = strlen(str);
    while (str[0] == ' ' || str[0] == '\t')
    {
        for (j = 0; j < l - 1; j++)
        {
            str[j] = str[j + 1];
        }
        str[j] = '\0';
        l--;
    }
}
```

29. Trim trailing (end) white space characters from a given string.

> Test Data

```
Enter the string: Removing leading and trailing white spaces
```

> Expected Output

```
Before: 'Removing leading and trailing white spaces     '
After: 'Removing leading and trailing white spaces'
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void trimSpaces(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    printf("\nBefore: '%s'", str);
    trimSpaces(str);
    printf("\nAfter: '%s'", str);
    return 0;
}

void trimSpaces(char *str)
{
    int i, l, j;
    l = strlen(str);
    while (str[l - 1] == ' ' || str[l - 1] == '\t')
    {
        str[l - 1] = '\0';
        l--;
    }
}
```

30. Trim both leading and trailing white space characters from a given string

> Test Data

```
Enter the string:      Removing leading and trailing white spaces
```

> Expected Output

```
Before: '     Removing leading and trailing white spaces      '
After: 'Removing leading and trailing white spaces'
```

> Source Code

```c
#include <stdio.h>
#include <string.h>

int main()
{
    void trimSpaces(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    printf("\nBefore: '%s'", str);
    trimSpaces(str);
    printf("\nAfter: '%s'", str);
    return 0;
}

void trimSpaces(char *str)
{
    int i, l, j;
    l = strlen(str);
    while (str[0] == ' ' || str[0] == '\t')
    {
        for (j = 0; j < l - 1; j++)
        {
            str[j] = str[j + 1];
        }
        str[j] = '\0';
        l--;
    }
    while (str[l - 1] == ' ' || str[l - 1] == '\t')
    {
        str[l - 1] = '\0';
        l--;
    }
}
```

31. Remove all extra blank spaces from the given string.

> Test Data

```
Enter the string:   Keep  your eyes  on  the stars and your feet  on the   ground.
```

> Expected Output

```
Before: '  Keep  your eyes  on  the stars and your feet  on the   ground.        '

After: 'Keep your eyes on the stars and your feet on the ground.'
```

> Source Code

```c
#include <stdio.h>
#include <string.h>
```

```c
int main()
{
    void trimSpaces(char *);
    char str[100];

    printf("Enter the string: ");
    gets(str);

    printf("\nBefore: '%s'", str);
    trimSpaces(str);
    printf("\nAfter: '%s'", str);
    return 0;
}

void trimSpaces(char *str)
{
    int i, l, j;
    l = strlen(str);
    while (str[0] == ' ' || str[0] == '\t')
    {
        for (j = 0; j < l - 1; j++)
        {
            str[j] = str[j + 1];
        }
        str[j] = '\0';
        l--;
    }
    while (str[l - 1] == ' ' || str[l - 1] == '\t')
    {
        str[l - 1] = '\0';
        l--;
    }
    for (i = 0; i < l; i++)
    {
        if ((str[i] == ' ' || str[i] == '\t') &&
        (str[i + 1] == ' ' || str[i + 1] == '\t'))
        {
            for (j = i; j < l - 1; j++)
            {
                str[j] = str[j + 1];
                str[i] = ' ';
            }
            str[j] = '\0';
            l--;
            i--;
        }
    }
}
```

Hi 👋, I'm Rajiv Kumar

A passionate frontend developer from India

🌐 Connect with me:

GITHUB TWITTER LINKEDIN

💻 Tech Stack: