**A21 - WAP to replace each string of one or more blanks by a single blank**

/\*

Name : Nestin Gregorios Sunny

Date : 26.05.2025

Description :

Read a string , remove unwanted spaces and print the edited string.

Sample Input :

Pointers are sharp knives\

Sample Output :

Pointers are sharp knives

\*/

#include <stdio.h>

void replace\_blank(char []);

int main()

{

char str[100];

printf("Enter the string with more spaces in between two words\n");

scanf("%[^\n]", str);

replace\_blank(str);

printf("%s\n", str);

}

//function definition

void replace\_blank(char \*str)

{

int i = 0;

while(str[i] != '\0')

{

if(str[i] == ' ' || str[i] == '\t')

{

str[i] = ' ';

if(str[i + 1] == ' ' || str[i + 1] == '\t')

{

int j = i + 1;

while(str[j] != '\0')

{

str[j] = str[j + 1]; //overwriting to previous element

j++;

}

str[j] = '\0';

i--;

}

}

i++;

}

}

**A22 - WAP to implement getword function**

/\*

Name : Nestin Gregorios Sunny

Date : 27.06.2025

Description :

Write a function to get the length of 1st word in a string.

Sample Input :

Enter the string : Hello World

Sample Output :

You entered Hello and the length is 5

\*/

#include <stdio.h>

int getword(char str[]);

int main()

{

int len = 0;

char str[100];

printf("Enter the string : ");

scanf(" %[^\n]", str);

len = getword(str);

printf("You entered %s and the length is %d\n", str, len);

}

//function declaration

int getword(char \*str)

{

int i, length = 0;

while(str[i] != '\0')

{

if(str[i] == ' ') //if space it will overwrite it to '\0'

{

str[i] = '\0';

break;

}

i++;

length++; //counting the char upto null

}

return length;

}

**A25 - WAP to reverse the given string using iterative method**

/\*

Name : Nestin Gregorios Sunny

Date : 27.05.2025

Description :

Read a string and print the reversed string.

Sample Input :

Enter a string : EMERTXE

Sample Output :

Reversed string is : EXTREME

\*/

#include <stdio.h>

void reverse\_iterative(char str[]);

int my\_strlen(char \*); //creating our own function for finding length of string

int main()

{

char str[30];

printf("Enter any string : ");

scanf("%[^\n]", str);

reverse\_iterative(str); //function call

printf("Reversed string is %s\n", str);

}

//Function definition

int my\_strlen(char \*str)

{

int i = 0, count = 0;

while(str[i] != '\0')

{

count++;

i++;

}

return count;

}

void reverse\_iterative(char \*str)

{

int len = my\_strlen(str);

int begin = 0;

int end = len -1;

char temp = str[0];

while(begin < end) //condition for reversing a string

{

temp = str[end];

str[end] = str[begin];

str[begin] = temp;

begin++;

end--;

}

}

**A26 - WAP to reverse the given string using recursive method**

/\*

Name : Nestin Gregorios Sunny

Date : 27.05.2025

Description :

Read a string and use recursive function to print reversed string.

Sample Input :

Enter a string : Hello World

Sample Output :

Reverse string is : dlroW olleH

\*/

#include <stdio.h>

//function declaration

void reverse\_recursive(char str[], int ind, int len);

int my\_strlen(char \*str);

int main()

{

char str[30];

printf("Enter any string : ");

scanf("%[^\n]", str);

int len = my\_strlen(str); //call to get string length

int ind = 0;

reverse\_recursive(str, ind, len - 1); //call for reverse function

printf("Reversed string is %s\n", str);

}

//function definition

void reverse\_recursive(char \*str, int ind, int len)

{

if(ind >= len)

{

return ;

}

char temp = str[ind];

str[ind] = str[len];

str[len] = temp;

reverse\_recursive(str, ind + 1, len - 1);

}

int my\_strlen(char \*str)

{

int i = 0, count = 0;

while(str[i] != '\0')

{

count++;

i++;

}

return count;

}

**A30 - WAP to check given string is Pangram or not**

/\*

Name : Nestin Gregorios Sunny

Date : 28.05.2025

Description :

Write a C program to check given string is PANGRAM or not

Sample Input :

Enter the string: The quick brown fox jumps over the lazy dog

Sample Output :

The Entered String is a Pangram String

\*/

#include <stdio.h>

int pangram(char []);

int main()

{

char str[100];

int res;

printf("Enter the string: ");

scanf("%[^\n]",str);

res = pangram(str); //call

if(res == 26)

{

printf("The Entered String is a Pangram String");

}

else

{

printf("The Entered String is not a Pangram String");

}

return 0;

}

int pangram(char \*str)

{

int i, count = 0;

int arr[26] = {0}; //array to flag alphabets

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

{

if(str[i] >= 'a' && str[i] <= 'z')

{

arr[str[i] - 97] = 1;

}

else if(str[i] >= 'A' && str[i] <= 'Z')

{

arr[str[i] - 65] = 1;

}

}

for(i = 0; i < 26; i++)

{

if(arr[i] == 1) //checking whether every alphabets are flagged to 1 or not

{

count++;

}

}

return count;

}

**A23 - WAP to implement atoi function**

/\*

Name : Nestin Gregorios Sunny

Date : 31.05.2025

Description :

Read a string and implement atoi function [my\_atoi(const char \*s)]

Sample Input :

Enter a numeric string: -123abc

Sample Output :

String to integer is -123

\*/

#include <stdio.h>

int my\_atoi(const char []);

int main()

{

char str[20];

printf("Enter a numeric string : ");

scanf("%s", str);

printf("String to integer is %d\n", my\_atoi(str));

}

int my\_atoi(const char \*str)

{

int i = 0, digit;

int flag = 1, num = 0;

//condition to check if given string is negative or positive

if(str[i] == '-')

{

flag = -1;

i++;

}

else if(str[i] == '+')

{

i++;

}

//loop to get only integer and store that value to num

while(str[i] != '\0')

{

// check if initial values are alphabets or punctuation marks

if(str[i] >= 33 && str[i] <= 47 || str[i] >= 65 && str[i] <=90 || str[i] >= 97 && str[i] <= 122)

{

break;

}

digit = str[i] - 48;

num \*= 10;

num += digit;

i++;

}

return flag \* num;

}

**A24 - WAP to implement itoa function**

/\*

Name : Nestin Gregorios Sunny

Date : 31.05.2025

Description :

Read a integer and convert it to string using function "void itoa(int n, char \*s)"

Sample Input :

Enter the number : 1234

Sample Output :

Integer to string is 1234

\*/

#include <stdio.h>

void itoa(int num, char str[]);

int main()

{

int num;

char str[10];

printf("Enter the number:");

if(scanf("%d", &num) != 1)

{

printf("Integer to string is 0");

return 1;

}

itoa(num, str);

printf("Integer to string is %s", str);

}

void itoa(int num, char \*str)

{

int i = 0;

int flag = (num < 0) ? 1 : 0;

if(flag) //if negative integer change it to positive

{

num = -num;

}

if(num == 0)

{

str[i++] = '0';

}

else

{

//getting remainder to store character by character

while(num)

{

str[i++] = num % 10 + '0';

num = num/10;

}

}

if(flag) //if given integer was negative add - at end

{

str[i++] = '-';

}

str[i] = '\0';

int begin = 0, end = i - 1;

//reverse the string

while(begin < end)

{

char temp = str[begin];

str[begin] = str[end];

str[end] = temp;

begin++;

end--;

}

}

**A29 - WAP to print all possible combinations of given string.**

/\*

Name : Nestin Gregorios Sunny

Date : 30.05.2025

Description :

Read a string from the user, find and print all the possible combinations of given string.

Sample Input :

Enter a string: abc

Sample Output :

All possible combinations of given string :abc

acb

bca

bac

cab

cba

\*/

#include<stdio.h>

//function declaration

void combination(char [],int ,int );

int my\_strlen(char []);

int main()

{

char str[100];

int n;

int res;

printf("Enter a string: ");

scanf("%100[^\n]",str);

n = my\_strlen(str);

printf("All possible combinations of given string : ");

combination(str,0,n-1);

return 0;

}

//function definition

int my\_strlen(char \*str)

{

int i = 0, count = 0;

while(str[i] != '\0')

{

count++;

i++;

}

return count;

}

void swap(char \*i, char \*start) //function to swap character in a string

{

char temp = \*i;

\*i = \*start;

\*start = temp;

}

void combination(char \*str, int start, int end)

{

int i;

if(start == end)

{

printf("%s\n",str);

}

else

{

for(i = start; i <= end; i++)

{

swap(&str[i], &str[start]);

combination(str, start + 1, end);

swap(&str[i], &str[start]);

}

}

}

**A28 - Squeeze the character in s1 that matches any character in the string s2**

/\*

Name : Nestin Gregorios Sunny

Date : 30.05.2025

Description :

Read 2 strings and remove the characters in s1 that matches with s2.

Sample Input :

Enter s1 : Dennis Ritchie

Enter s2 : Linux

Sample Output :

Afteer squeeze s1 : Des Rtche

\*/

#include <stdio.h>

void squeeze(char [], char []);

int main()

{

char str1[30], str2[30];

printf("Enter string1:");

scanf("%[^\n]", str1);

printf("Enter string2:");

scanf(" %[^\n]", str2);

squeeze(str1, str2);

printf("After squeeze s1 : %s\n", str1);

}

void squeeze(char \*s1, char \*s2)

{

int i, j, k;

k = 0;

for(i = 0; s1[i]; i++) //loop to check and and update string1

{

int found = 0; //initializing a flag variable to point out a matching character if available in both string

for(j = 0; s2[j]; j++)

{

/\*

if(s1[i] == s2[j]) //condition to check matching characters

{

found = 1;

break;

}

\*/

char c1 = s1[i];

char c2 = s2[j];

if(c1 == c2)

{

found = 1;

break;

}

}

if(!found) //after the above loop execution if no matching character are found, this if condition will update/squeeze the 1st string

{

s1[k++] = s1[i];

}

}

s1[k] = '\0'; //inserting NULL character to end of updated string

}

**A29 - WAP to print all possible combinations of given string.**

/\*

Name : Nestin Gregorios Sunny

Date : 30.05.2025

Description :

Read a string from the user, find and print all the possible combinations of given string.

Sample Input :

Enter a string: abc

Sample Output :

All possible combinations of given string :abc

acb

bca

bac

cab

cba

\*/

#include<stdio.h>

//function declaration

void combination(char [],int ,int );

int my\_strlen(char []);

int main()

{

char str[100];

int n;

int res;

printf("Enter a string: ");

scanf("%100[^\n]",str);

n = my\_strlen(str);

printf("All possible combinations of given string : ");

combination(str,0,n-1);

return 0;

}

//function definition

int my\_strlen(char \*str)

{

int i = 0, count = 0;

while(str[i] != '\0')

{

count++;

i++;

}

return count;

}

void swap(char \*i, char \*start) //function to swap character in a string

{

char temp = \*i;

\*i = \*start;

\*start = temp;

}

void combination(char \*str, int start, int end)

{

int i;

if(start == end)

{

printf("%s\n",str);

}

else

{

for(i = start; i <= end; i++)

{

swap(&str[i], &str[start]);

combination(str, start + 1, end);

swap(&str[i], &str[start]);

}

}

}

**A27 - WAP to implement strtok function**

/\*

Name : Nestin Gregorios Sunny

Date : 05\_06.2025

Description :

Implement our own strtok function.

Sample Input :

Enter the string : hello;;;how;--are...you

Enter the delimiter : ;-.

Sample Output :

Tokens:

hello

how

are

you

\*/

#include <stdio.h>

#include <string.h>

#include <stdio\_ext.h>

char \*my\_strtok(char str[], const char delim[]);

int main()

{

char str[50], delim[50];

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

scanf("%s", str);

\_\_fpurge(stdout);

printf("Enter the delimeter : ");

scanf("\n%s", delim);

\_\_fpurge(stdout);

char \*token = my\_strtok(str, delim);

printf("Tokens :\n");

while (token)

{

printf("%s\n", token);

token = my\_strtok(NULL, delim);

}

}

char \*my\_strtok(char \*str, const char \*delim)

{

static int i = 0;

int j = 0;

static char \*duplicate;

int start\_index;

if (str != NULL)

{

duplicate = str;

}

start\_index = i;

while (duplicate[i] != '\0')

{

j = 0;

while (delim[j] != '\0')

{

if (duplicate[i] == delim[j])

{

duplicate[i] = '\0';

i++;

if(duplicate[start\_index] == '\0')

{

start\_index = i;

i--;

break;

}

else

{

return &duplicate[start\_index];

}

}

j++;

}

i++;

}

if (duplicate[start\_index] != '\0')

{

return &duplicate[start\_index];

}

else

{

return NULL;

}

}