3.2 Character Arrays and Strings:

A **character array** is a collection of characters stored in contiguous memory locations.
Used to store text in older C++ programs or for low-level string manipulation.

Declaring a string in c-style

String can be declared like one-dimensional arrays

```
char str[30];
char text[80];
```

String initialization

char name[20] = “Hello";

Internally, it looks like this in memory:
‘H’ ‘e’ ‘l’ ‘l’ ‘o’ ‘\0’

C-style string is simply an **array of characters** that ends with a **null character** ('\0').

1. Compile Time Initialization

(iv) Array initialization with a string: - Consider the declaration with string initialization.

Ex:-

char b[]="COMPUTER";

The array b is initialized as shown in figure.

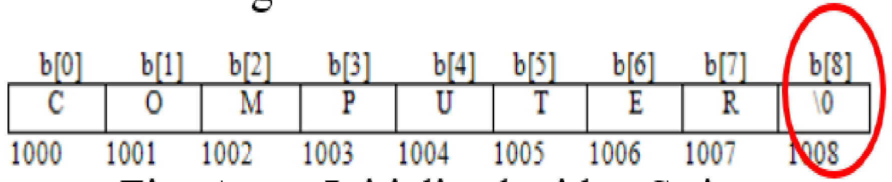


Fig: Array Initialized with a String

Eventhough the string "COMPUTER" contains 8 characters, because it is a string. It always ends with null character. So, the array size is 9 bytes (i.e., string length 1 byte for null character).

Ex:-

char b[9]="COMPUTER"; // correct

char b[8]="COMPUTER"; // wrong

- If we initialize char s[2] = "ab";
- Only 2 bytes of space — the null terminator doesn’t fit.
- Compiler warning: initializer-string for char array is too long

Correct way:

- char s[3] = "ab"; // or char s[] = "ab";
- Compiler adds '\0' automatically.

```
int main() {
    char name[6] = {'A','B','c','d','e','\0'};
    cout << "Name: " << name << endl;
    return 0;
}
```

Must include \0 at the end to indicate the string's termination.
Array size = number of characters + 1 (for \0).

Reading strings

There are two common ways to read strings in C++:

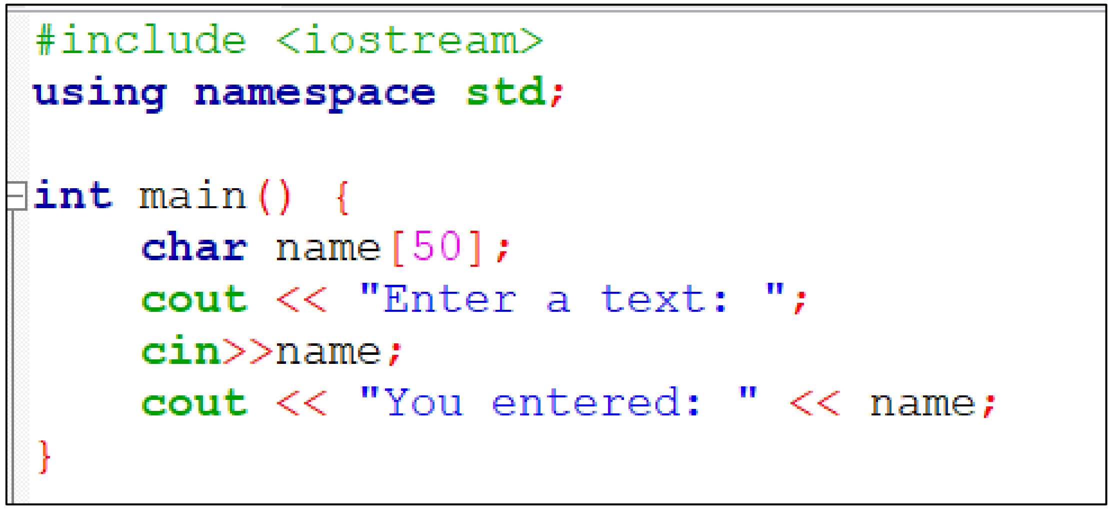
1. Using cin — reads a single word (stops at space).

```
char str[20];
cin >> str;  // Reads a word (stops at whitespace)
```

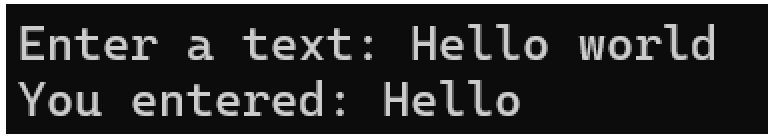
2. Using getline() — reads a full line including spaces.

```
cin.getline(str, 20); // Reads a full line including spaces
```

Reading a string using cin

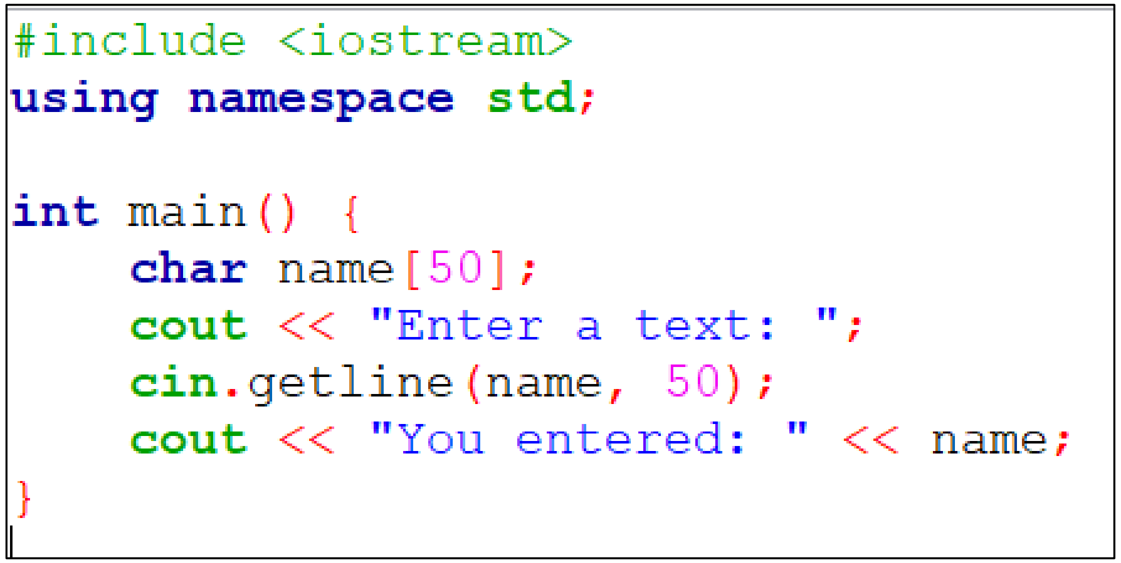


```
#include <iostream>
using namespace std;

int main() {
      char name[50];
      cout << "Enter a text: ";
      cin>>name;
      cout << "You entered: " << name;
}
```



```
Enter a text: Hello world
You entered: Hello
```

reading using getline



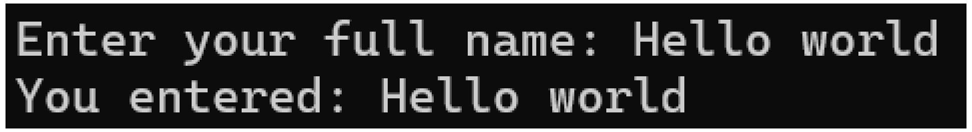
```
#include <iostream>
using namespace std;

int main() {
    char name[50];
    cout << "Enter a text: ";
    cin.getline(name, 50);
    cout << "You entered: " << name;
}
```

getline reads the entire line, including spaces, until:

- a newline ('\n') is found,

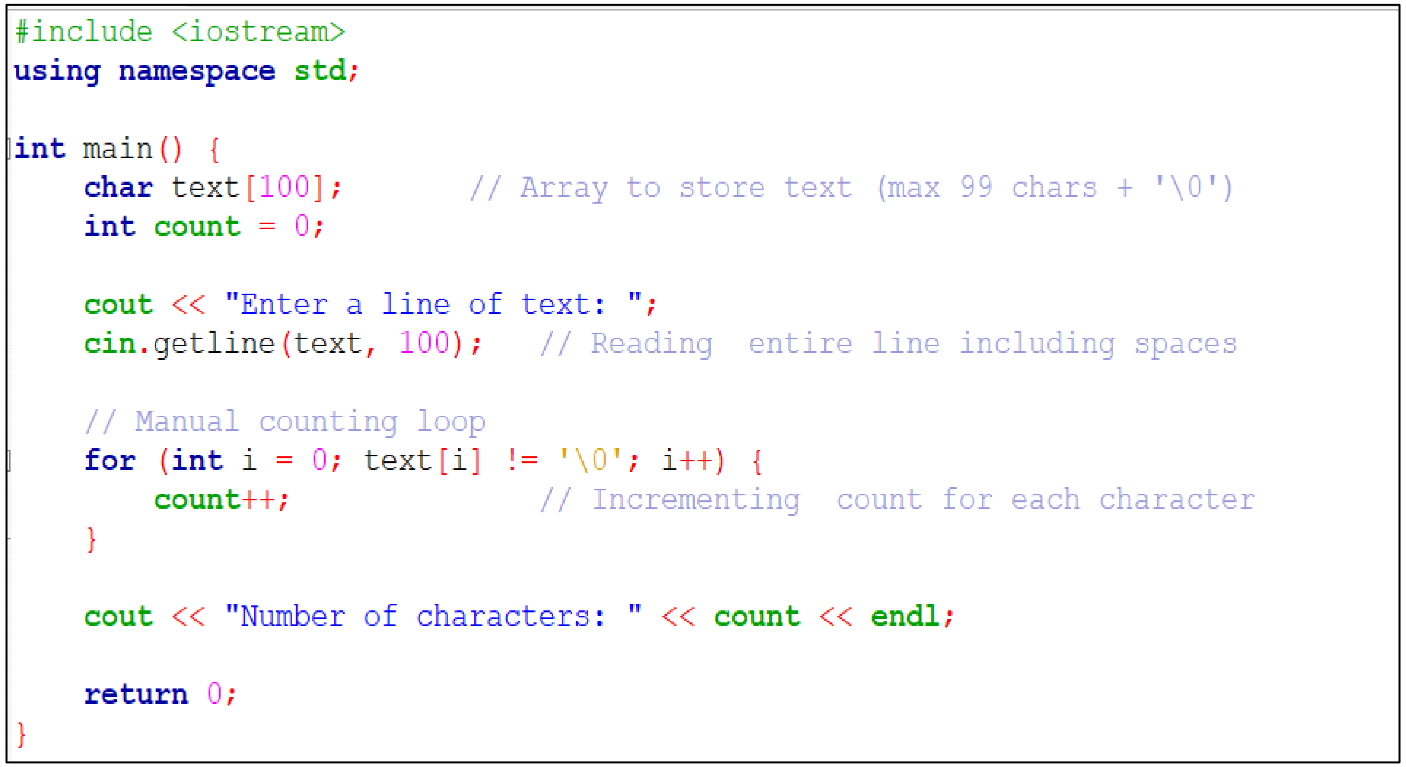
Or

- the specified size limit is reached (here 50)

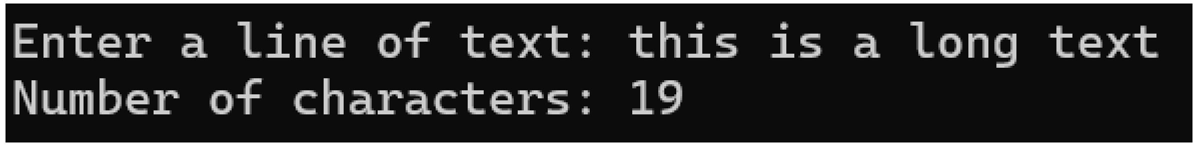


```
Enter your full name: Hello world
You entered: Hello world
```

Program to count characters in text

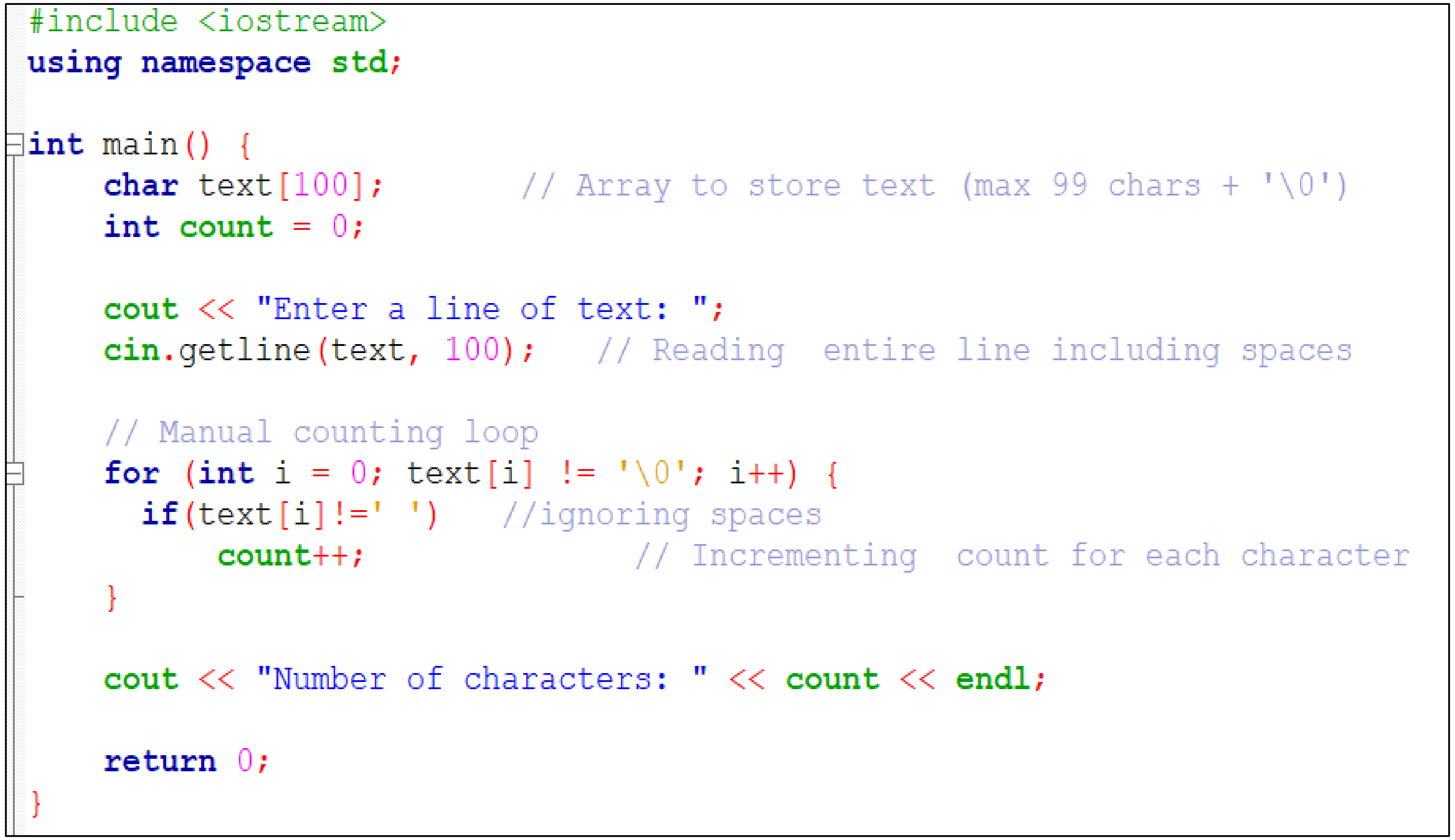


```cpp
#include <iostream>
using namespace std;

int main() {
    char text[100];       // Array to store text (max 99 chars + '\0')
    int count = 0;

    cout << "Enter a line of text: ";
    cin.getline(text, 100);   // Reading  entire line including spaces

    // Manual counting loop
    for (int i = 0; text[i] != '\0'; i++) {
        count++;                  // Incrementing  count for each character
    }

    cout << "Number of characters: " << count << endl;

    return 0;
}
```

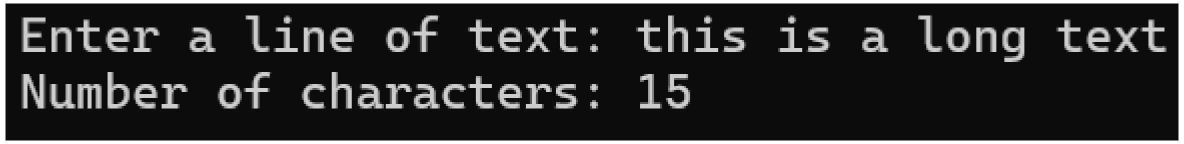


```
Enter a line of text: this is a long text
Number of characters: 19
```

Modified program to exclude spaces

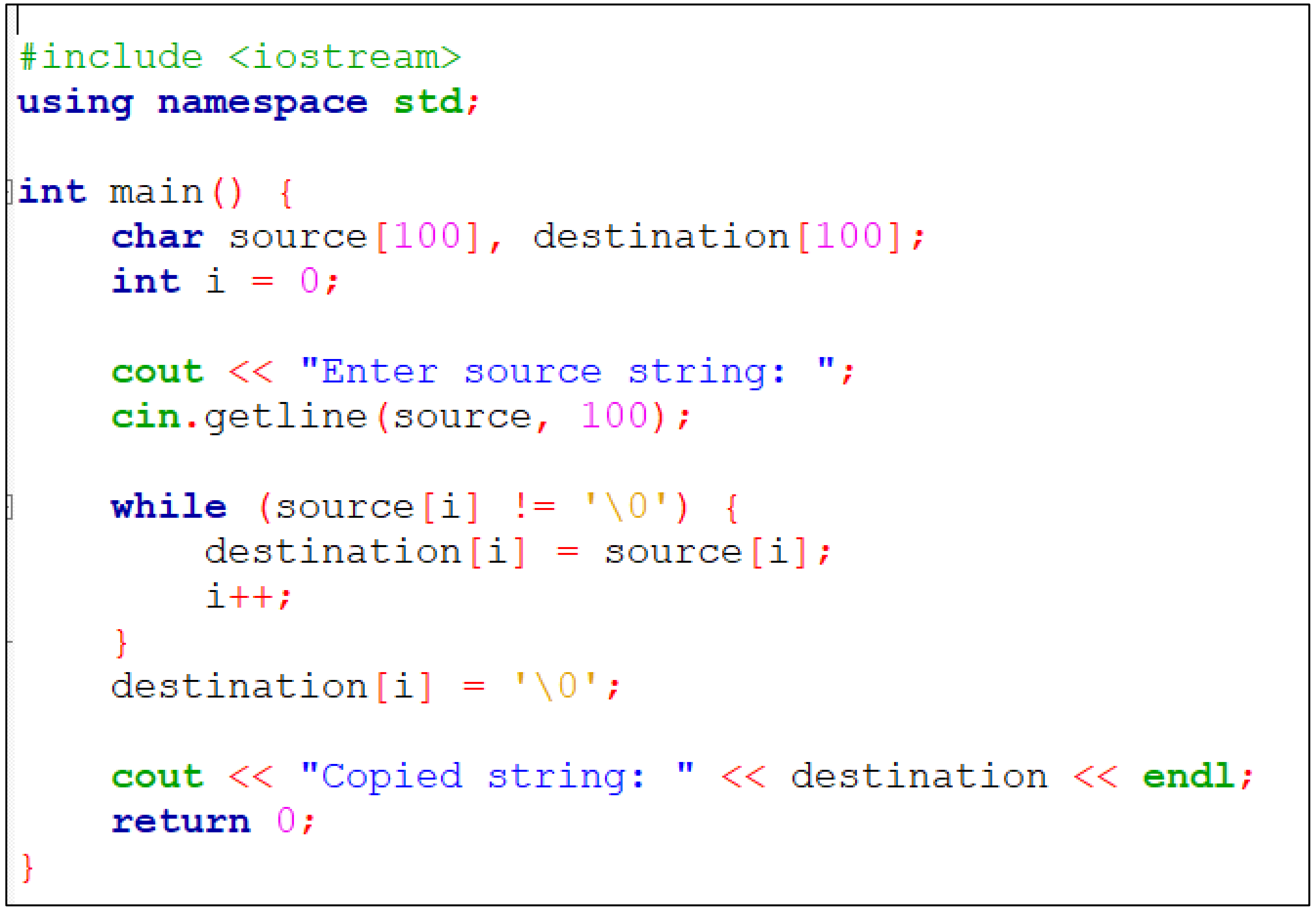


```cpp
#include <iostream>
using namespace std;

int main() {
    char text[100];       // Array to store text (max 99 chars + '\0')
    int count = 0;

    cout << "Enter a line of text: ";
    cin.getline(text, 100);    // Reading  entire line including spaces

    // Manual counting loop
    for (int i = 0; text[i] != '\0'; i++) {
      if(text[i]!=' ')    //ignoring spaces
          count++;              // Incrementing  count for each character
    }

    cout << "Number of characters: " << count << endl;

    return 0;
}
```

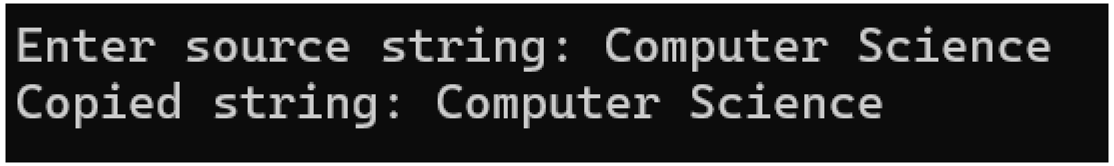


```
Enter a line of text: this is a long text
Number of characters: 15
```

WAP to read a string, copy it to another string, and display the copied string.

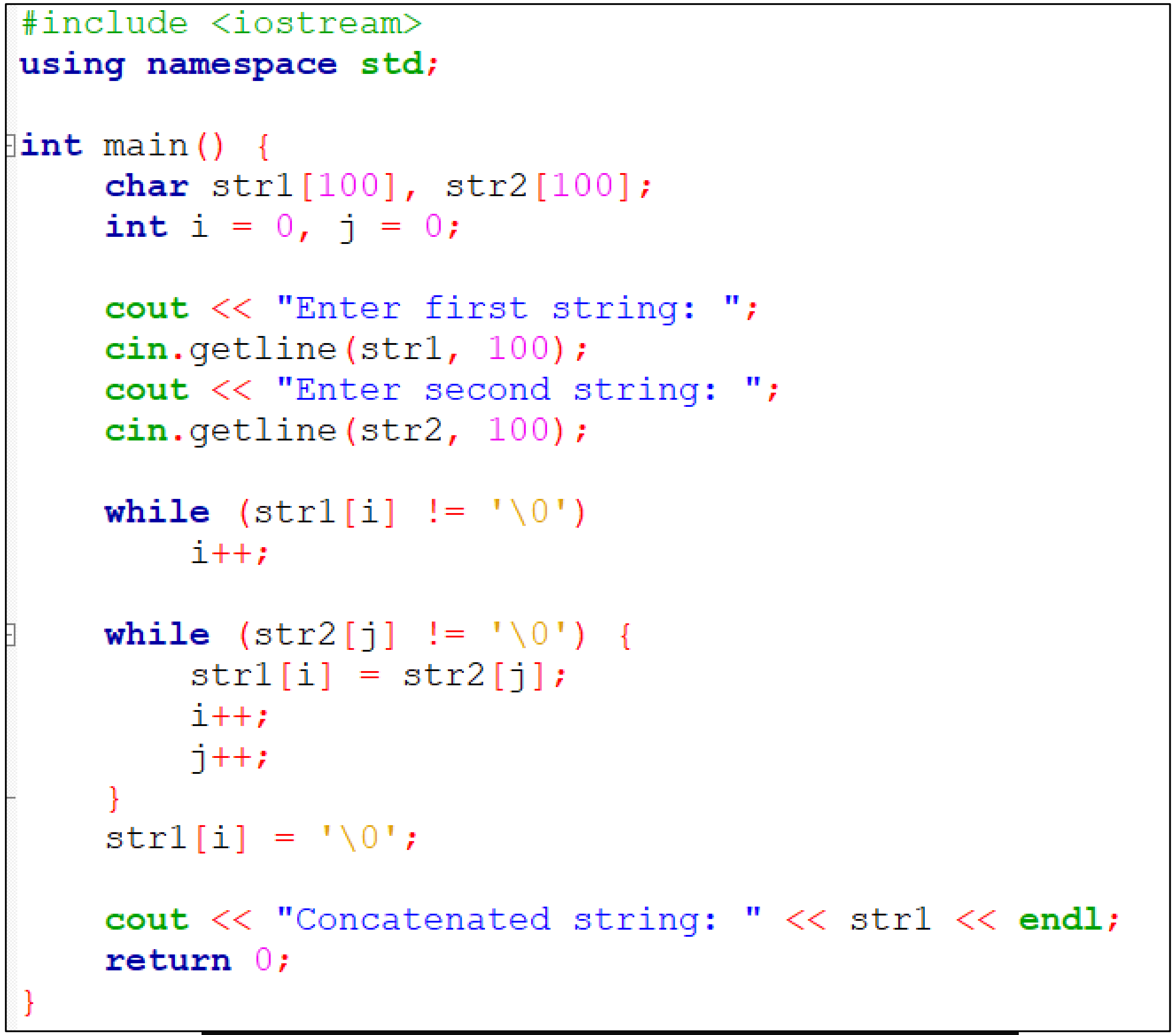


```cpp
#include <iostream>
using namespace std;

int main() {
    char source[100], destination[100];
    int i = 0;

    cout << "Enter source string: ";
    cin.getline(source, 100);

    while (source[i] != '\0') {
        destination[i] = source[i];
        i++;
    }
    destination[i] = '\0';

    cout << "Copied string: " << destination << endl;
    return 0;
}
```

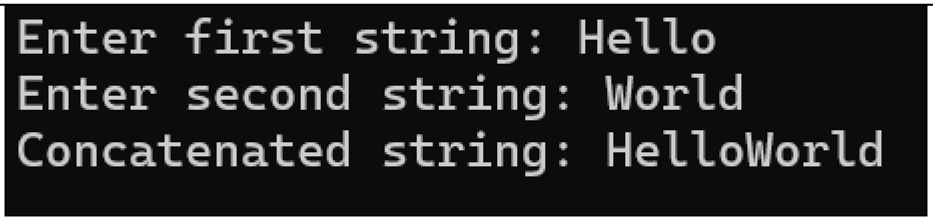


```
Enter source string: Computer Science
Copied string: Computer Science
```

- Write a program to accept two strings and display the result after appending the second string to the end of the first string.

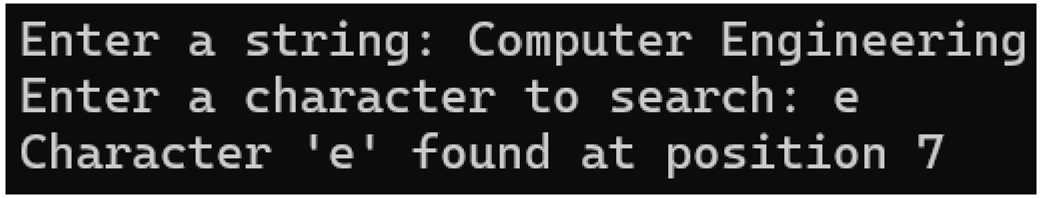


```cpp
#include <iostream>
using namespace std;

int main() {
    char str1[100], str2[100];
    int i = 0, j = 0;

    cout << "Enter first string: ";
    cin.getline(str1, 100);
    cout << "Enter second string: ";
    cin.getline(str2, 100);

    while (str1[i] != '\0')
        i++;

    while (str2[j] != '\0') {
        str1[i] = str2[j];
        i++;
        j++;
    }
    str1[i] = '\0';

    cout << "Concatenated string: " << str1 << endl;
    return 0;
}
```

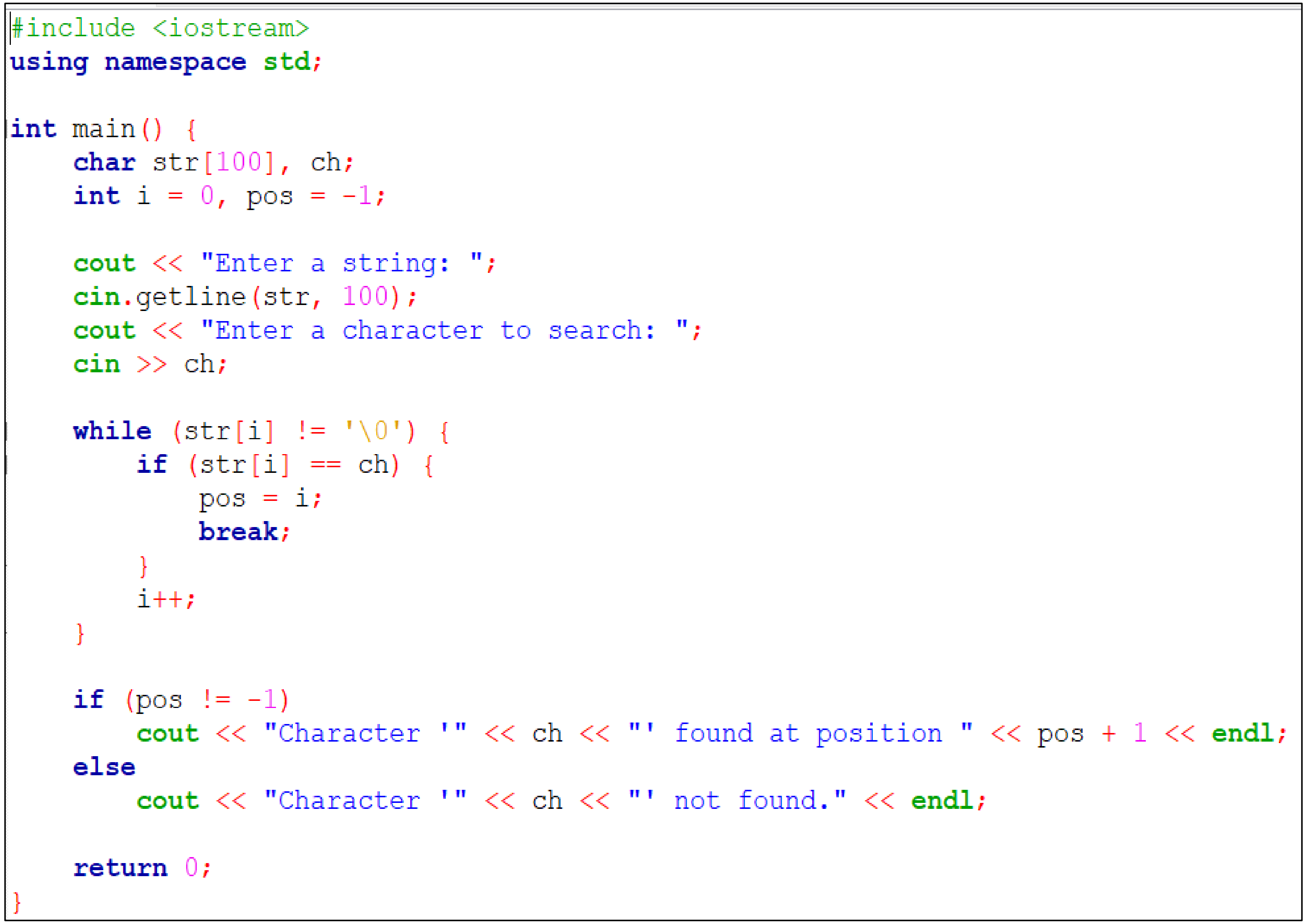


```
Enter first string: Hello
Enter second string: World
Concatenated string: HelloWorld
```

WAP to find and display the position of the first occurrence of a given character in a string.



```
Enter a string: Computer Engineering
Enter a character to search: e
Character 'e' found at position 7
```



```cpp
#include <iostream>
using namespace std;

int main() {
    char str[100], ch;
    int i = 0, pos = -1;

    cout << "Enter a string: ";
    cin.getline(str, 100);
    cout << "Enter a character to search: ";
    cin >> ch;

    while (str[i] != '\0') {
        if (str[i] == ch) {
            pos = i;
            break;
        }
        i++;
    }

    if (pos != -1)
        cout << "Character '" << ch << "' found at position " << pos + 1 << endl;
    else
        cout << "Character '" << ch << "' not found." << endl;

    return 0;
}
```

WAP to compare two strings and display 0 if they are equal, a negative value if the first string appears earlier, and a positive value if it appears later in dictionary order

Sample Input-output:

Case 1:

```
Enter first string: apple
Enter second string: apple
Comparison result: 0
-> Both strings are equal.
```

```
Case 2:
Enter first string: apple
Enter second string: banana
Comparison result: -1
→ First string appears earlier in dictionary order.
Case 3:
Enter first string: mango
Enter second string: grapes
Comparison result: 6
→ First string appears later in dictionary order.
```

```
cout << "Enter first string: ";
cin.getline(str1, 100);

cout << "Enter second string: ";
cin.getline(str2, 100);

// Compare character by character
while (str1[i] != '\0' && str2[i] != '\0') {
    if (str1[i] != str2[i]) {
        result = str1[i] - str2[i];  // difference of ASCII values
        break;
    }
    i++;
}
// If one string ended but other didn't
if (result == 0)
    result = str1[i] - str2[i];

cout << "Comparison result: " << result << endl;

if (result == 0)
    cout << "Both strings are equal." << endl;
else if (result < 0)
    cout << "First string appears earlier in dictionary order." << endl;
else
    cout << "First string appears later in dictionary order." << endl;
```

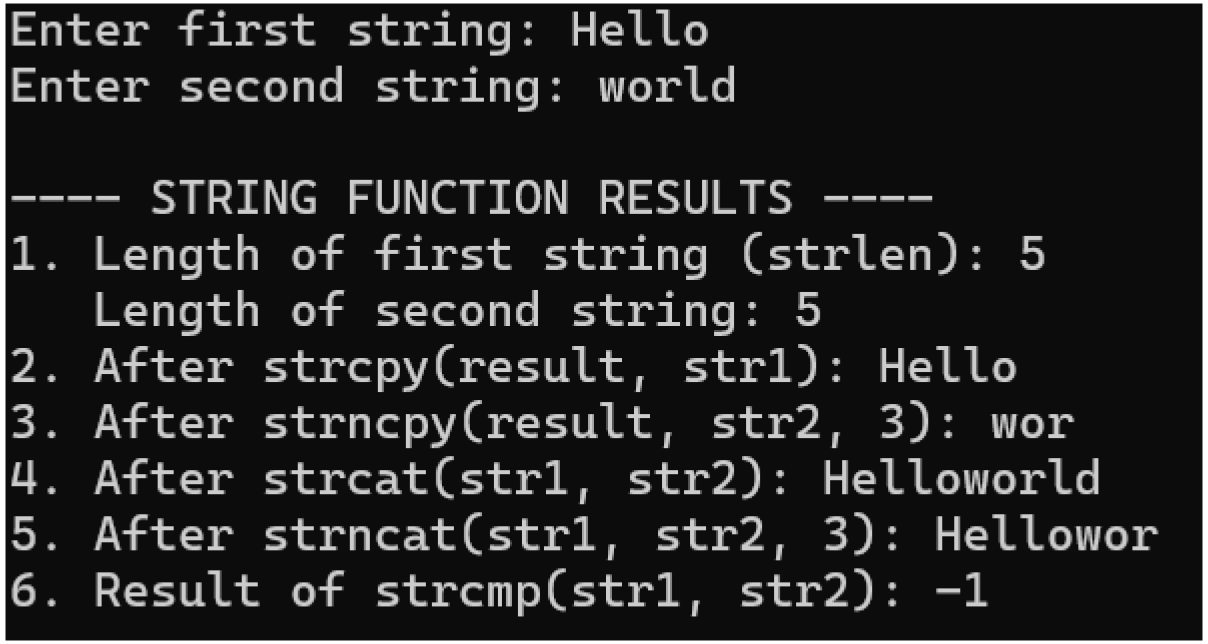
<cstring> header file

- **<cstring>** is a **C++ standard header file** that provides a set of **functions for manipulating C-style strings**

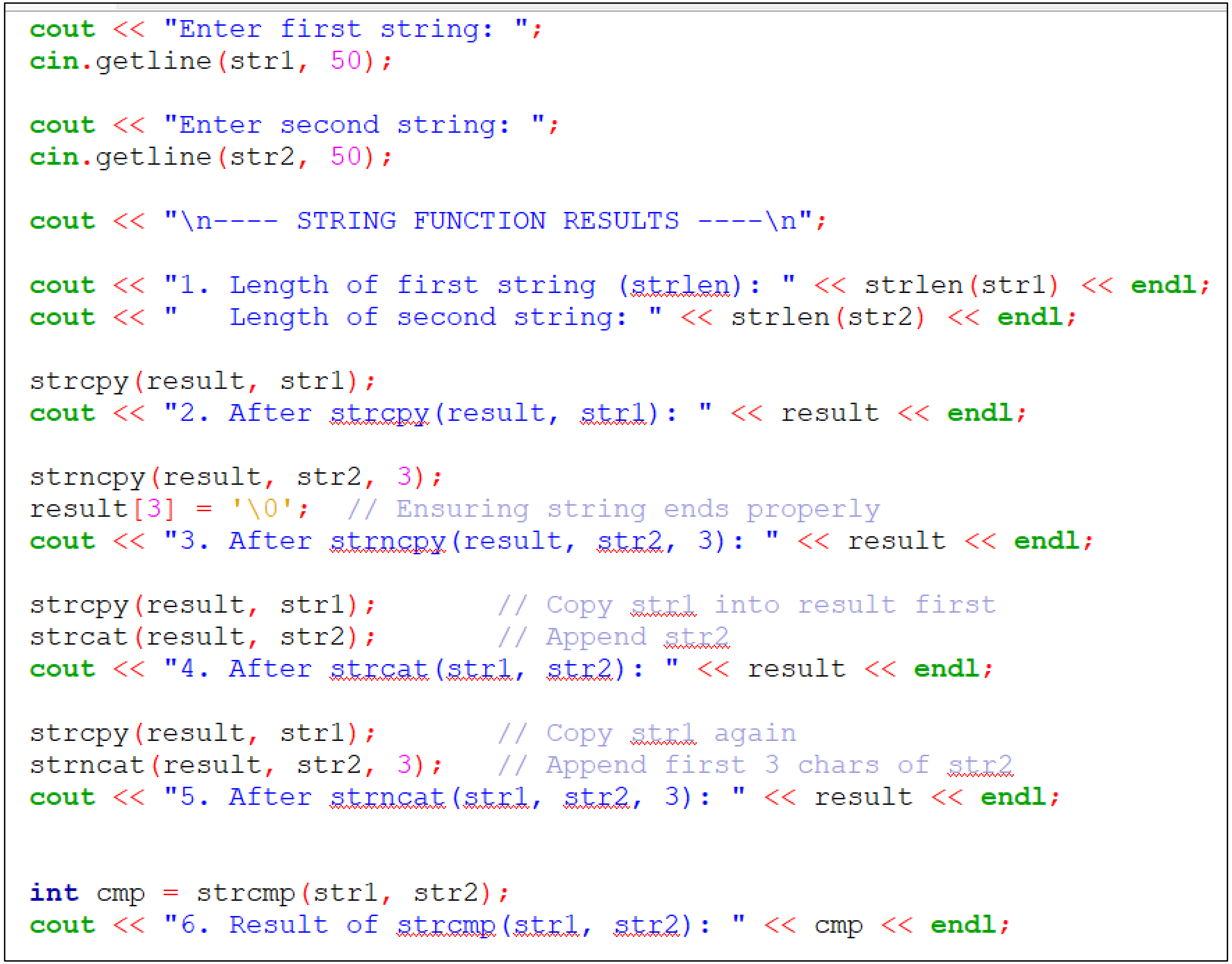
Few string function of <cstring> header file

Function	Purpose / Description	Example	Sample Output / Meaning
strlen(str)	Returns number of characters in the string (excluding '\0')	strlen("Hello")	5
strcpy(dest, src)	Copies one string into another	strcpy(b, a)	b = "Hello"
strncpy(dest, src, n)	Copies first n characters only	strncpy(b, a, 3)	b = "Hel"
strcat(dest, src)	Appends one string at the end of another	strcat(a, b)	a = "HelloWorld"
strncat(dest, src, n)	Appends first n characters of one string to another	strncat(a, b, 3)	a = "HelloWor"
strcmp(s1, s2)	Compares two strings lexicographically	strcmp("abc", "abd")	< 0 (since "abc" < "abd")
strchr(str, ch)	Finds the first occurrence of a character in a string	strchr("Hello", "l")	Returns "llo" (address of first l)
strstr(str1, str2)	Finds the first occurrence of a substring in another string	strstr("HelloWorld", "World")	Returns "World" (pointer to substring)

Demonstration of string functions

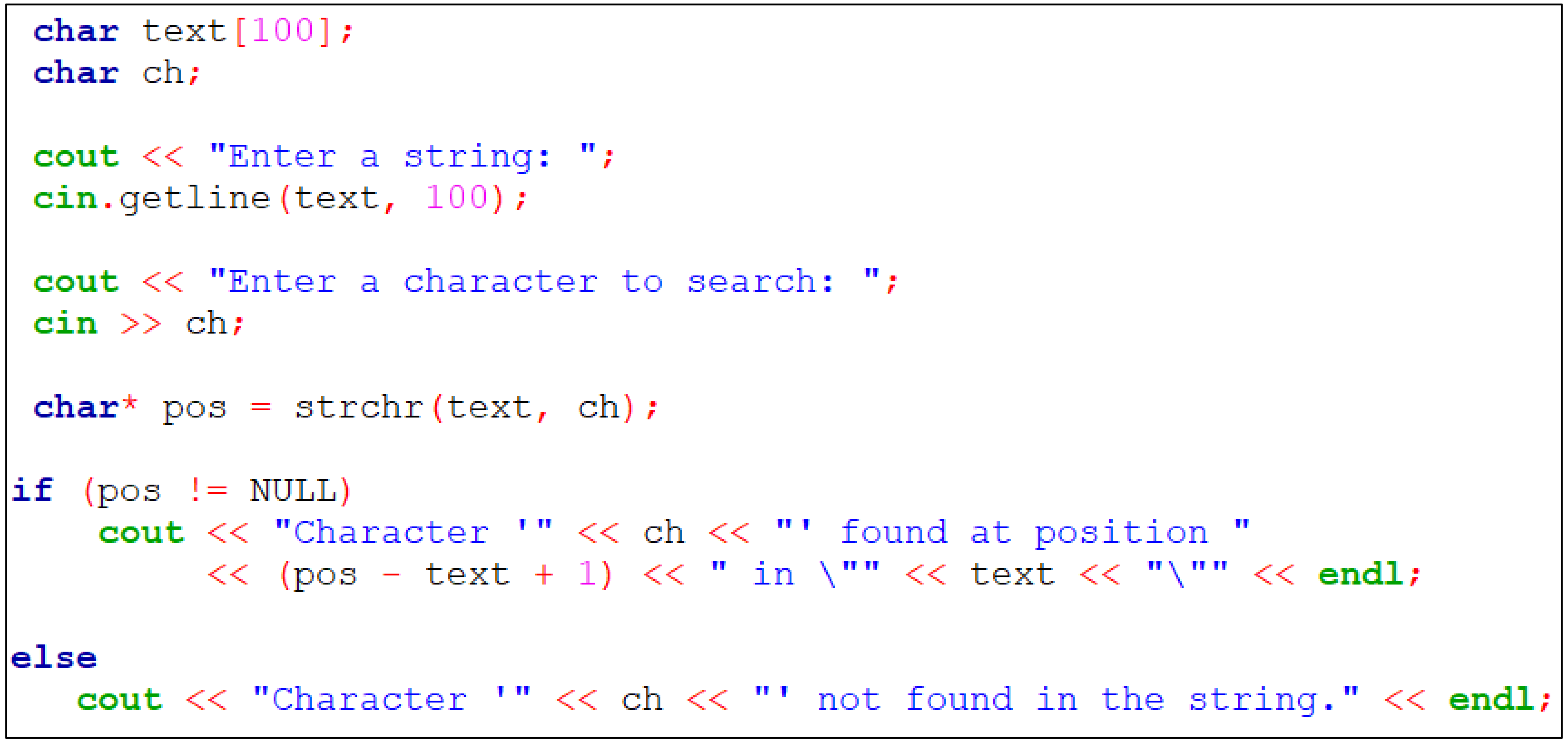


```
Enter first string: Hello
Enter second string: world

---- STRING FUNCTION RESULTS ----
1. Length of first string (strlen): 5
   Length of second string: 5
2. After strcpy(result, str1): Hello
3. After strncpy(result, str2, 3): wor
4. After strcat(str1, str2): Helloworld
5. After strncat(str1, str2, 3): Hellowor
6. Result of strcmp(str1, str2): -1
```

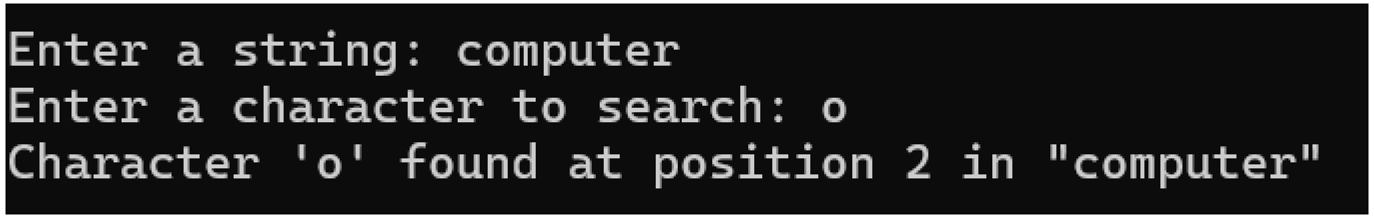


```cpp
cout << "Enter first string: ";
cin.getline(str1, 50);

cout << "Enter second string: ";
cin.getline(str2, 50);

cout << "\n---- STRING FUNCTION RESULTS ----\n";

cout << "1. Length of first string (strlen): " << strlen(str1) << endl;
cout << "   Length of second string: " << strlen(str2) << endl;

strcpy(result, str1);
cout << "2. After strcpy(result, str1): " << result << endl;

strncpy(result, str2, 3);
result[3] = '\0';  // Ensuring string ends properly
cout << "3. After strncpy(result, str2, 3): " << result << endl;

strcpy(result, str1);       // Copy str1 into result first
strcat(result, str2);       // Append str2
cout << "4. After strcat(str1, str2): " << result << endl;

strcpy(result, str1);       // Copy str1 again
strncat(result, str2, 3);   // Append first 3 chars of str2
cout << "5. After strncat(str1, str2, 3): " << result << endl;


int cmp = strcmp(str1, str2);
cout << "6. Result of strcmp(str1, str2): " << cmp << endl;
```

Demo of strchr function

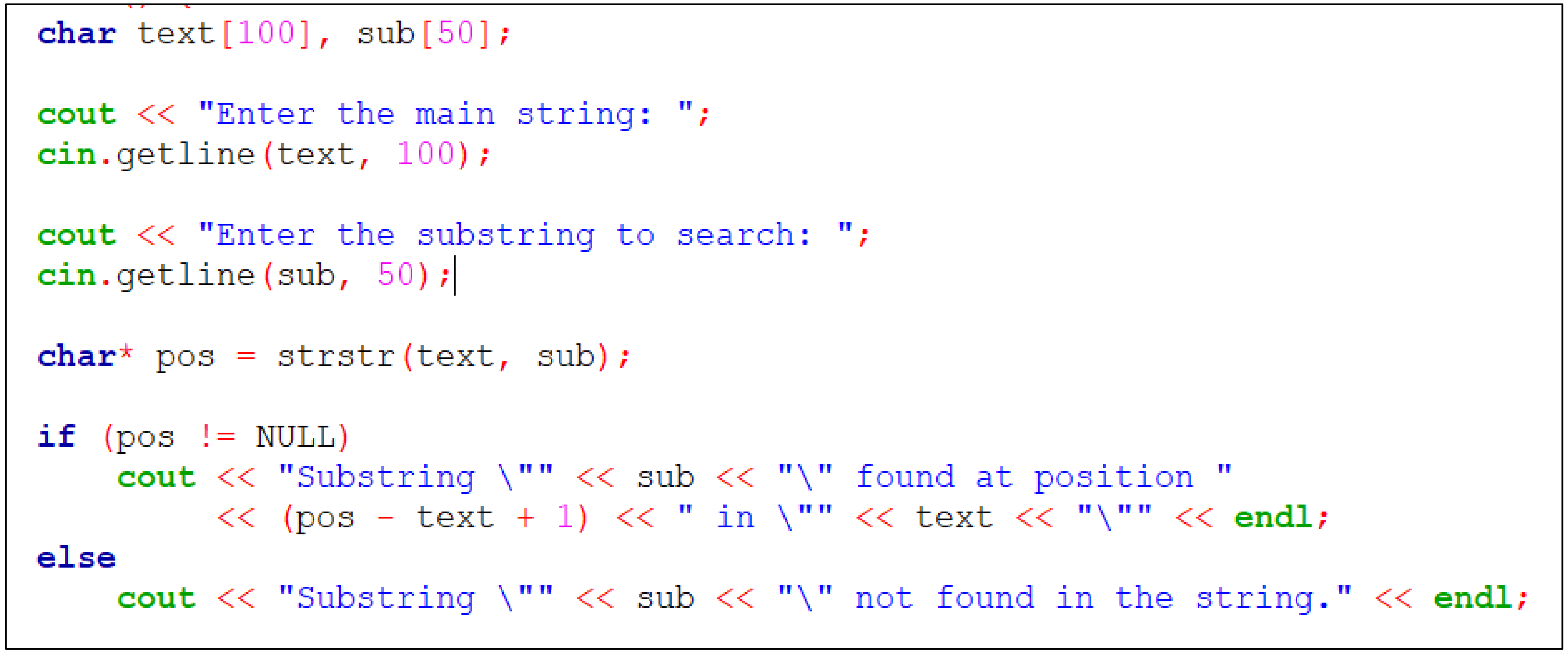


```
 char text[100];
 char ch;

 cout << "Enter a string: ";
 cin.getline(text, 100);

 cout << "Enter a character to search: ";
 cin >> ch;

 char* pos = strchr(text, ch);

if (pos != NULL)
    cout << "Character '" << ch << "' found at position "
         << (pos - text + 1) << " in \"" << text << "\"" << endl;

else
   cout << "Character '" << ch << "' not found in the string." << endl;
```

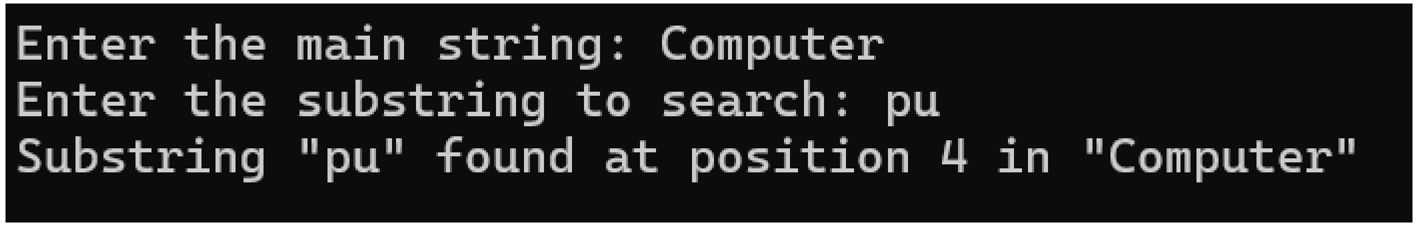


```
Enter a string: computer
Enter a character to search: o
Character 'o' found at position 2 in "computer"
```

Demo of strstr function



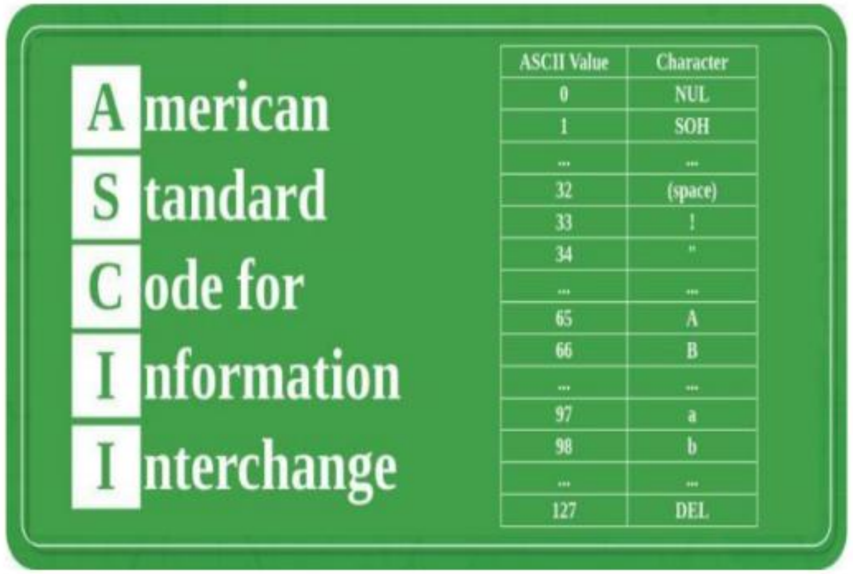
```cpp
char text[100], sub[50];

cout << "Enter the main string: ";
cin.getline(text, 100);

cout << "Enter the substring to search: ";
cin.getline(sub, 50);

char* pos = strstr(text, sub);

if (pos != NULL)
    cout << "Substring \"" << sub << "\" found at position "
         << (pos - text + 1) << " in \"" << text << "\"" << endl;
else
    cout << "Substring \"" << sub << "\" not found in the string." << endl;
```



```
Enter the main string: Computer
Enter the substring to search: pu
Substring "pu" found at position 4 in "Computer"
```

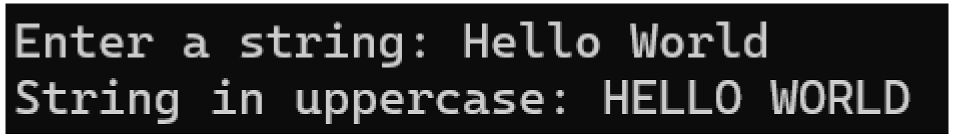
WAP to capitalize all the letters of the string

ASCII codes


American Standard Code for Information Interchange

ASCII Value	Character
0	NUL
1	SOH
...	...
32	(space)
33	!
34	"
...	...
65	A
66	B
...	...
97	a
98	b
...	...
127	DEL

```cpp
char str[100];
cout << "Enter a string: ";
cin.getline(str, 100);
// Convert each character to uppercase
for (int i = 0; i < strlen(str); i++) {
    if (str[i] >= 'a' && str[i] <= 'z')   // check if lowercase
        str[i] = str[i] - 32;             // convert to uppercase
}

cout << "String in uppercase: " << str << endl;
```



```
Enter a string: Hello World
String in uppercase: HELLO WORLD
```

<cctype> header file

<cctype> header provides predefined functions like isalpha(), isdigit(), toupper(), etc.,to check type of character and to convert characters between cases.

<cctype> header file

Function	Purpose / Description	Example	Output / Meaning
isalpha(ch)	Checks if character is an alphabet (A–Z or a–z)	isalpha('A')	1 (true)
isdigit(ch)	Checks if character is a digit (0–9)	isdigit('9')	1 (true)
isalnum(ch)	Checks if character is alphanumeric (A–Z, a–z, 0–9)	isalnum('#')	0 (false)
isspace(ch)	Checks if character is a whitespace (space, tab, newline)	isspace(' ')	1 (true)
islower(ch)	Checks if character is lowercase	islower('g')	1 (true)
isupper(ch)	Checks if character is uppercase	isupper('G')	1 (true)
toupper(ch)	Converts lowercase to uppercase	toupper('b')	'B'
tolower(ch)	Converts uppercase to lowercase	tolower('H')	'h'
ispunct(ch)	Checks if character is punctuation (!, ?, etc.)	ispunct('!')	1 (true)
isprint(ch)	Checks if character is printable	isprint('\n')	0 (false)
iscntrl(ch)	Checks if character is a control character	iscntrl('\n')	1 (true)
isxdigit(ch)	Checks if character is a hexadecimal digit (0–9, A–F, a–f)	isxdigit('F')	1 (true)

WAP to reverse the string and check whether it a palindrome (use strlen, strcmp)

WAP to Count Digits, Alphabets, and Special Characters (use isalpha(), isdigit() function)

Input: Hello123@C++
Output:

```
Alphabets: 6

Digits: 3

Special characters: 3
```