DSA SQUAD BY SOURCIFY IN

DAY-4

Strings

Strings are used for storing text/characters.

For example, "Hello World" is a string of characters.

Unlike many other programming languages, C does not have a String type to easily create string variables. Instead, you must use the char type and create an array of characters to make a string in C:

char greetings [] = "Hello World!";

Remainder: For string we have to use double quotes and not single quotes

Now, lets about how to take input and output in c.

Input

Using gets:

```
#include <stdio.h>
int main()
{
    char str[20];
    gets(str);
```

```
    return 0;
}
```

Using fgets(it is used latest)

```
#include <stdio.h>
#define MAX_LIMIT 20
int main()
{
    char str[MAX_LIMIT];
    fgets(str, MAX_LIMIT, stdin);
    return 0;
}
```

Output

```
#include <stdio.h>
void main()
{
    char greetings [] = "Hello World!";
    printf ("%s", greetings);
    return 0;
}
```


Access Strings

Since strings are actually arrays in C, you can access a string by referring to its index number inside square brackets [].

This example prints the first character (0) in greetings:

Example

```
char greetings[] = "Hello World!";
printf("%c", greetings[0]);
```

Note: The %c format specifier to print a single character.

Modify Strings

To change the value of a specific character in a string, refer to the index number, and use **single quotes**:

Example

```
char greetings[] = "Hello World!";
greetings[0] = 'J';
printf("%s", greetings);
```


Traversing through a string using a loop

You can also loop through the characters of a string, using a for loop:

Example

```
char carName[] = "Volvo";
int i;

for (i = 0; i < 5; ++i) {
  printf("%c\n", carName[i]);
}
```

Now, we can also specify the size of the string using **sizeof** function

Example:

```
char carName[] = "Volvo";
int length = sizeof(carName) / sizeof(carName[0]);
int i;

for (i = 0; i < length; ++i) {
  printf("%c\n", carName[i]);
```

Another way of creating a string is by using a set of characters

```
char greetings[] = {'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd', '!', '\0'};
printf("%s", greetings);
```

Note: '\0' is known as the **"null terminating character"**, and must be included when creating strings using this method. It tells C that this is the end of the string. It is mandatory to write this .

Strings - Special Characters

Because strings must be written within quotes, C will misunderstand this string, and generate an error:

char txt[] = "We are the so-called "Vikings" from the north.";

Escape character	Result	Description
\'	'	Single quote
\"	"	Double quote
\\	\	Backslash

String functions

To use, string function we must include string.h in our c program.

String Length

For example, to get the length of a string, you can use the strlen() function:

Example

```
char alphabet [] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
printf ("%d", strlen(alphabet));
```

Concatenate Strings

To concatenate (combine) two strings, you can use the strcat() function:

Example

```
char str1[20] = "Hello ";
char str2[] = "World!";
```

```
// Concatenate str2 to str1 (result is stored in str1)
strcat(str1, str2);

// Print str1
printf("%s", str1);
```

Copy Strings

To copy the value of one string to another, you can use the strcpy() function:

Example

```
char str1[20] = "Hello World!";
char str2[20];

// Copy str1 to str2
strcpy(str2, str1);

// Print str2
printf("%s", str2);
```


Compare Strings

To compare two strings, you can use the strcmp() function.

It returns 0 if the two strings are equal, otherwise a value that is not 0:

Example

```
char str1[] = "Hello";
char str2[] = "Hello";
char str3[] = "Hi";

// Compare str1 and str2, and print the result
printf("%d\n", strcmp(str1, str2));  // Returns 0 (the strings are equal)

// Compare str1 and str3, and print the result
printf("%d\n", strcmp(str1, str3));  // Returns -4 (the strings are not equal)
```