STRING

String Basics

What are Strings?

A string is an array of characters. Data of the same type are stored in an array, for example, integers can be stored in an integer array, similarly, a group of characters can be stored in a character array. This character array is also known as strings. A string is a one-dimensional array of characters that is terminated by a null ('\0').

Declaration of Strings:

Declaring a string is very simple, the same as declaring a one-dimensional array. It's just that we are considering it as an array of characters.

Below is the syntax for declaring a string.

```
char string_name[string_size];
```

In the above syntax, string_name is any name given to the string variable, and string_size is used to define the length of the string, i.e., the number of characters that the strings will store. Keep in mind that there is an extra terminating character which is the null character ('\0') that is used to indicate the termination of the string.

Example of string:

```
#include <stdio.h>

int main()

{

  // declare and initialise string

  char str[] = "CodeWithHarry";

  printf("%s", str);

  return 0;

}
```

Output:

```
CodeWithHarry
```

String Functions

We can use C's string handling library functions to handle strings. The string.h library is used to perform string operations. It provides several functions for manipulating strings.

Following are some commonly used string handling functions:

1. strcat():

This function is used to concatenate the source string to the end of the target string. This function expects two parameters, first, the base address of the source string and then the base address of the target string. For example, “Hello” and “World” on concatenation would result in a string “HelloWorld”.

Here is how we can use the strcat() function:

```
#include <stdio.h>

#include <string.h>

int main()

{

   char s[] = "Hello";

   char t[] = "World";

   strcat(s, t);

   printf("String = %s", s);

}
```

Output:

```
String = HelloWorld
```

2. strlen():

This function is used to count the number of characters present in a string.

Here is how we can use the strlen() function:

```
#include <stdio.h>

#include <string.h>

int main()

{

   char s[] = "Hello";

   int len = strlen(s);
```

```
    printf("Length = %d", len);
}
```

Output:

```
Length = 5
```

3. strcpy():

This function is used to copy the contents of one string into the other. This function expects two parameters, first, the base address of the source string and then the base address of the target string.

Here is how we can use the strcpy() function:

```
#include <stdio.h>
#include <string.h>
int main()
{
    char s[] = "CodeWithHarry";
    char t[50];
    strcpy(t, s);
    printf("Source string = %s\n", s);
    printf("Target string = %s", t);
}
```

Output:

```
Source string = CodeWithHarry
Target string = CodeWithHarry
```

4. strcmp():

The strcmp() function is used to compare two strings to find out whether they are the same or different. It takes two strings as two of its parameters. It will compare two strings, character by character until there is a mismatch or the iterator reaches the end of one of the strings.

If both of the strings are identical, strcmp() returns a value of zero. If they are not identical, it will return a value less than zero, considering the ASCII value of the mismatched character in the first string is less than the mismatched character in the second string. Else, it will return a value greater than 0.

Here is how we can use the strcmp() function:

```
#include <stdio.h>

#include <string.h>

int main()

{

   char s[] = "Hello";

   char t[] = "World";

   int cmp = strcmp(s, t);

   printf("Comparison result = %d", cmp);

}
```

Output:

```
Comparison result = -1
```

5. strrev():

This function is used to return the reverse of the string.

Here is how we can use the strrev() function:

```
#include <stdio.h>

#include <string.h>

int main()

{

   char s[] = "Hello";

   printf("Reversed string = %s", strrev(s));

}
```

Output:

```
Reversed string = olleH
```