Chapter 8 - Strings

A string is a 1-D character array terminated by a null ('\0')
↳ This is null character

null character is used to denote string termination
characters are stored in contiguous memory locations

Initializing Strings

Since string is an array of characters, it can be initialized as follows:

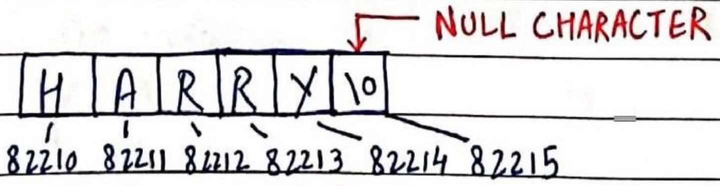
```
char S[] = {'H', 'A', 'R', 'R', 'Y', '\0'};
```

There is another shortcut for initializing strings in C language:

```
char S[] = "HARRY";
```
⇒ In this case C adds a null charater automatically.

Strings In Memory

A string is stored just like an array in the memory as shown below



Contiguous blocks in memory

Quick Quiz → Create a string using " " and print its content using a loop.

Printing Strings
A string can be printed character by character using printf and %c
But there is another convenient way to print strings in C.

```
Char St[] = " HARRY";
printf(" %S", St);
```
⟹ prints the entire string.

Taking string input from the user
We can use %s with scanf to take string input from the user:

```
Char St[50];
Scanf("%S", [redacted]St);
```

Scanf automatically adds the null character when the the enter key is pressed.

Note:
1. The string should be short enough to fit into the array
2. Scanf cannot be used to input multi-word strings with spaces.

gets() and puts()
gets() is a function which can be used to receive a multi-word string.

```
Char st[30];
gets(st);  => The entered String is Stored in st!
```

Multiple gets() calls will be needed for multiple strings

Likewise, puts can be used to output a string.

```
puts(st);  => prints the string
              places the cursor on the next line
```

Declaring a string using pointers
We can declare strings using pointers

```
Char *ptr = "Harry";
```

This tells the compiler to store the string in memory and assigned address is stored in a char pointer

Note:
1> Once a string is defined using char st[] = "Harry", it cannot be reinitialized to something else.
2> A string defined using pointers can be reinitialized
    ptr = "Rohan";

Standard library functions for Strings

C provides a set of standard library functions for string manipulation.

Some of the most commonly used string functions are:

Strlen()

This function is used to count the number of characters in the string excluding the null ('\0') character.

```
int length = Strlen(st);
```

These functions are declared under <string.h> header file

Strcpy()

This function is used to copy the content of second string into first string passed to it.

```
Char source[] = "Harry";
char target[30];
Strcpy(target, Source);  => target now
                            contains "Harry"
```

Target string should have enough capacity to store the source string.

Strcat()
This function is used to concatenate two strings

Char S_1[11] = "Hello";
Char S_2[ ] = "Harry";

Strcat (S_1, S_2); ⇒ S_1 now contains "HelloHarry"
<No space in between>

Strcmp()
This function is used to compare two Strings.
It returns: 0 if Strings are equal
Negative value if first string's mismatching character's ASCII value is not greater than second String's corresponding mismatching character. It returns positive values otherwise.

Strcmp ("Far", "Joke"); ⇒ Negative value
Strcmp ("Joke", "Far"); ⇒ Positive value