C library function - memmove()

Description

The C library function **void *memmove(void *str1, const void *str2, size_t n)** copies **n** characters from **str2** to **str1**, but for overlapping memory blocks, memmove() is a safer approach than memcpy().

Declaration

Following is the declaration for memmove() function.

```
void *memmove(void *str1, const void *str2, size_t n)
```

Parameters

- **str1** − This is a pointer to the destination array where the content is to be copied, type-casted to a pointer of type void*.
- **str2** − This is a pointer to the source of data to be copied, type-casted to a pointer of type void*.
- **n** − This is the number of bytes to be copied.

Return Value

This function returns a pointer to the destination, which is str1.

Example

The following example shows the usage of memmove() function.

Live Demo

```
#include <stdio.h>
#include <string.h>

int main () {
   char dest[] = "oldstring";
   const char src[]  = "newstring";

   printf("Before memmove dest = %s, src = %s\n", dest, src);
   memmove(dest, src, 9);
   printf("After memmove dest = %s, src = %s\n", dest, src);
```

```
   return(0);
}
```

Let us compile and run the above program that will produce the following result −

```
Before memmove dest = oldstring, src = newstring
After memmove dest = newstring, src = newstring
```