# memcpy（C库函数）

## 描述

C 库函数 **void \*memcpy(void \*str1, const void \*str2, size\_t n)** 从存储区 **str2** 复制 **n** 个字节到存储区 **str1**。

## 声明

void \*memcpy(void \*str1, const void \*str2, size\_t n)

## 参数

**str1** -- 指向用于存储复制内容的目标数组，类型强制转换为 void\* 指针。

**str2** -- 指向要复制的数据源，类型强制转换为 void\* 指针。

**n** -- 要被复制的字节数。

## 返回值

该函数返回一个指向目标存储区 str1 的指针。

## 实例

const char src[50] = “zdwzd888”

char \*s = “zdwzd888”

char dest[90];

1. memcpy(dest, src, strlen(src) + 1) ; //加一将结尾0考进去，不会因为遇到\0结束，dest空间要够大，不然报错。重叠会覆盖原有数据。
2. memcpy(dest, src + 2 \* sizeof(char), 2\*sizeof(char)); //从第二个字符w开始复制，连续复制两个字符。
3. memcpy(&dest[2], &src[2], 2);
4. char \*p = (char \*)memcpy(dest, src, 2) //若要返回指针， 需要强制类型转换。

# memcmp（C库函数）

## 描述

C 库函数 **int memcmp(const void \*str1, const void \*str2, size\_t n)** 把存储区 **str1** 和存储区 str2 的前n个字节进行比较。

## 声明

int memcmp(const void \*str1, const void \*str2, size\_t n)

## 参数

**str1** -- 指向内存块的指针，类型强制转换为 void\* 指针。

**str2** -- 指向内存块的指针，类型强制转换为 void\* 指针。

**n** -- 要比较的字节数。

## 返回值

如果返回值 < 0, 则表示str1 小于 str2。

如果返回值 > 0, 则表示str1 大于 str2。

如果返回值 = 0, 则表示str1 等于 str2。

## 实例

Char str1[15];

Char str2[15];

Memcpy(str1, “abcdef”, 6);

Memcpy(str2, “ABCDEF”, 6);

Ret = memcmp(str1, str2, 5)

Ret >0

# memset（C库函数）

## 描述

C 库函数 void \*memset(void \*str, int c, size\_t n) 复制字符 c（一个无符号字符）到参数 str 所指向的字符串的前 n 个字符。

## 声明

void \*memset(void \*str, int c, size\_t n)

## 参数

str -- 指向要填充的内存块。

c -- 要被设置的值。该值以 int 形式传递，但是函数在填充内存块时是使用该值的无符号字符形式。

n -- 要被设置为该值的字符数。

## 返回值

该值返回一个指向存储区 str 的指针。

## 实例

#include <stdio.h>

#include <string.h>

int main ()

{

char str[50];

strcpy(str,"This is string.h library function");

puts(str);

memset(str,'$',7);

puts(str);

return(0);

}