<https://blog.csdn.net/hackerain/article/details/7954006>

真是有点惭愧，这些内容本应该很早就掌握的，以前只是糊里糊涂的用，不知道在内存中具体是怎么回事，现在才弄清楚。

realloc(void \*\_\_ptr, size\_t \_\_size)：更改已经配置的内存空间，即更改由malloc()函数分配的内存空间的大小。  
如果将分配的内存减少，realloc仅仅是改变索引的信息。  
如果是将分配的内存扩大，则有以下情况：  
1）如果当前内存段后面有需要的内存空间，则直接扩展这段内存空间，realloc()将返回原指针。  
2）如果当前内存段后面的空闲字节不够，那么就使用堆中的第一个能够满足这一要求的内存块，将目前的数据复制到新的位置，并将原来的数据块释放掉，返回新的内存块位置。  
3）如果申请失败，将返回NULL，此时，原来的指针仍然有效。  
  
注意：如果调用成功，不管当前内存段后面的空闲空间是否满足要求，都会释放掉原来的指针，重新返回一个指针，虽然返回的指针有可能和原来的指针一样，即不能再次释放掉原来的指针。

看一下示例代码

**[cpp]** [view plain](https://blog.csdn.net/hackerain/article/details/7954006) [copy](https://blog.csdn.net/hackerain/article/details/7954006)

1. #include <stdio.h>
2. #include <stdlib.h>
4. **int** main(**int** argc, **char**\* argv[], **char**\* envp[])
5. {
6. **int** input;
7. **int** n;
8. **int** \*numbers1;
9. **int** \*numbers2;
10. numbers1=NULL;
12. **if**((numbers2=(**int** \*)malloc(5\***sizeof**(**int**)))==NULL)//为numbers2在堆中分配内存空间
13. {
14. printf("malloc memory unsuccessful");
15. exit(1);
16. }
18. printf("numbers2 addr: %8X\n",(**int**)numbers2);
20. **for**(n=0;n<5;n++) //初始化
21. {
22. \*(numbers2+n)=n;
23. //printf("numbers2's data: %d\n",\*(numbers2+n));
24. }
26. printf("Enter new size: ");
27. scanf("%d",&input);
29. //重新分配内存空间，如果分配成功的话，就释放numbers2指针,
30. //但是并没有将numbers2指针赋为NULL,也就是说释放掉的是系统分配的堆空间，
31. //和该指针没有直接的关系，现在仍然可以用numbers2来访问这部分堆空间，但是
32. //现在的堆空间已经不属于该进程的了。
33. numbers1=(**int** \*)realloc(numbers2,(input+5)\***sizeof**(**int**));
35. **if**(numbers1==NULL)
36. {
37. printf("Error (re)allocating memory");
38. exit(1);
39. }
41. printf("numbers1 addr: %8X\n",(**int**)numbers1);
43. /\*for(n=0;n<5;n++) //输出从numbers2拷贝来的数据
44. {
45. printf("the numbers1's data copy from numbers2: %d\n",\*(numbers1+n));
46. }\*/
48. **for**(n=0;n<input;n++)//新数据初始化
49. {
50. \*(numbers1+5+n)=n+5;
51. //printf("numbers1' new data: %d\n",\*(numbers1+5+n));
52. }
54. printf("\n");
56. free(numbers1);//释放numbers1，此处不需要释放numbers1，因为在realloc()时已经释放
57. numbers1=NULL;
58. //free(numbers2);//不能再次释放
59. **return** 0;
60. }

如果当前内存段后有足够的空间，realloc()返回原来的指针：

**[plain]** [view plain](https://blog.csdn.net/hackerain/article/details/7954006) [copy](https://blog.csdn.net/hackerain/article/details/7954006)

1. yugsuo@ubuntu:~/linux/memange$ gcc -g -o realloc realloc\_example.c
2. yugsuo@ubuntu:~/linux/memange$ ./realloc
3. numbers2 addr:  8AFC008
4. Enter new size: 10
5. numbers1 addr:  8AFC008

如果当前内存段后没有足够的空间，realloc()返回一个新的内存段的指针：

**[plain]** [view plain](https://blog.csdn.net/hackerain/article/details/7954006) [copy](https://blog.csdn.net/hackerain/article/details/7954006)

1. yugsuo@ubuntu:~/linux/memange$ ./realloc
2. numbers2 addr:  9505008
3. Enter new size: 1000000
4. numbers1 addr: B716F008