注意：不要用 memcmp 比较 结构体

例如：

```
1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4.
5. typedef struct padding_type {
6.     short m1;
7.     int m2;
8. } padding_type_t;
9.
10. int main()
11. {
12.     padding_type_t a = {
13.         .m1 = 0,
14.         .m2 = 0,
15.     };
16.     padding_type_t b;
17.
18.     memset(&b, 0, sizeof(b));
19.
20.     if (0 == memcmp(&a, &b, sizeof(a))) {
21.         printf("Equal!\n");
22.     }
23.     else {
24.         printf("No equal!\n");
25.     }
26.
27.     return 0;
28.
29. }
```

```
1. laptop:~/works/test$ qcc -q test.c
2. laptop:~/works/test$ ./a.out
3. No equal!
```

结果是不相等的，
因为struct padding_type->m1的类型是short型，而m2的类型是int型，根据自然对齐的原则。padding_type的每个成员需要对齐到4字节。因此编译器会在m1后面插入2个padding字节，而padding的字节的值是随机的。也就是说a中的padding 字节的值是随机的，而b中的padding则被清零。所以当使用memcmpy去比较这两个结构体时，返回值是不等。