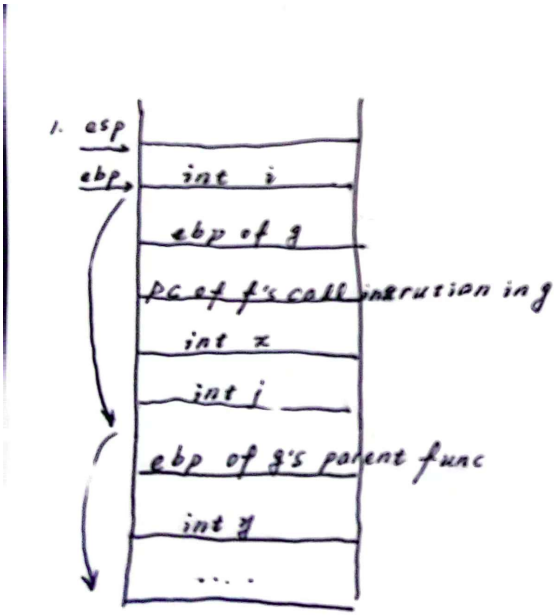
H13

1



2

2. (1) 0 sizeof(a) = 0 × 4 × 8 = 0

(2) **a 的值，会有变动

传入 printf("%ld, %d, %d, %d, %d, %d\n", sizeof(a), a[0][0], **a, a, &i, &j);

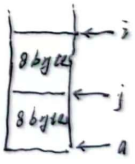
然后 a[0][0] 的输出与 **a 相同

a - &i = 16

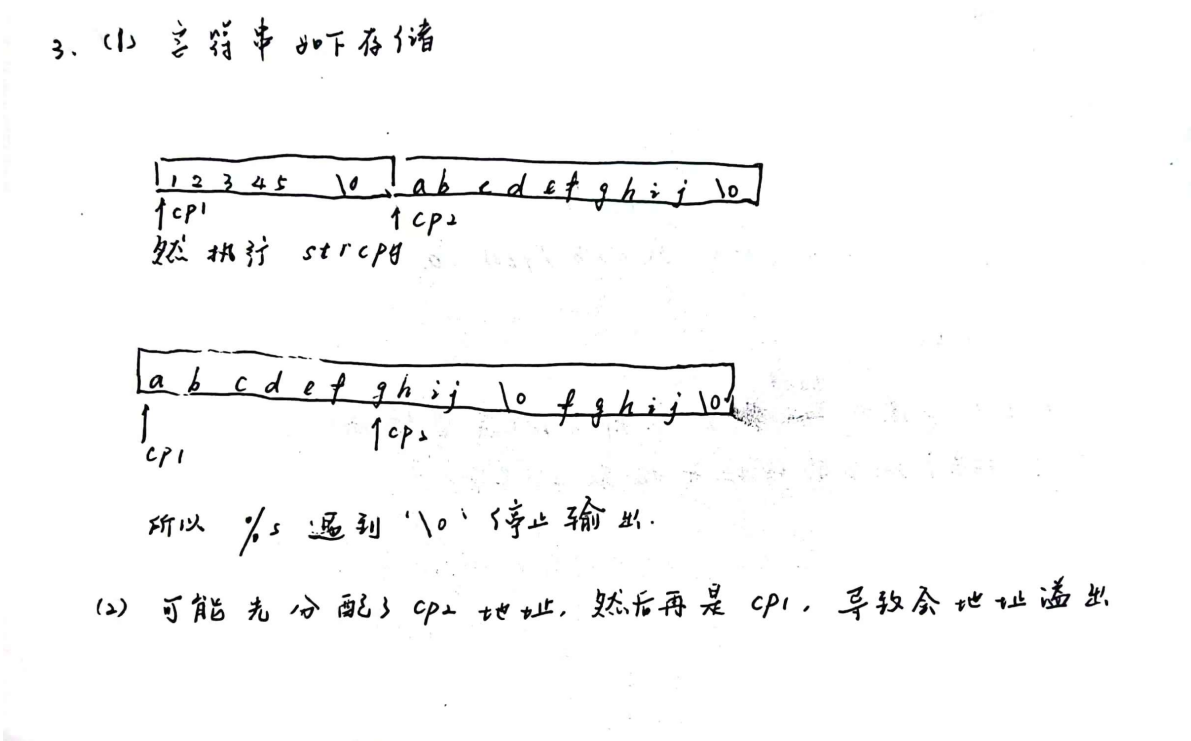
&j - &i = -8

说明 先分配 i，然后 j，然后 a

所以 a[0][0] 应该是 a 向下取 4 个字节的值



3



4

导出的是64位下的.s文件

```
    .file   "1.c"
    .text
    .globl  f
    .type   f, @function
f:
.LFB0:
    .cfi_startproc
    endbr64
    pushq   %rbp #存放rbp的值
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp #更新现在的rbp的值
    .cfi_def_cfa_register 6
    movl    %edi, %eax
    movss   %xmm0, -8(%rbp) #将浮点数f放到rbp -8的地方
    movw    %ax, -4(%rbp) #将整数a放到rbp -4的地方
    cmpw    $2, -4(%rbp) #将a与2进行比较
    jne .L2 #如果a != 2就跳转到return a;
    movss   -8(%rbp), %xmm0 #就以f作为返回值，就先将它存在xmm0里面待转化
    cvttss2sil  %xmm0, %eax #转换一个双精度浮点数(xmm)到带符号整数(r)
    jmp .L3
.L2:
    movzwl  -4(%rbp), %eax #以a作为返回值，存在eax寄存器里面
.L3:
    popq    %rbp #恢复rbp的值
    .cfi_def_cfa 7, 8
    ret #返回
    .cfi_endproc
```

```
.LFE0:
    .size   f, .-f
    .ident  "GCC: (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0"
    .section    .note.GNU-stack,"",@progbits
    .section    .note.gnu.property,"a"
    .align 8 #控制八字节对齐
    .long    1f - 0f #这里2个字节存储short类型的a
    .long    4f - 1f #这里8个字节存储float类型的f
    .long    5
0:
    .string  "GNU"
1:
    .align 8 #控制八字节对齐
    .long    0xc0000002
    .long    3f - 2f #这里2个字节存储short类型的返回值
2:
    .long    0x3
3:
    .align 8
4:
```