

# CSAPP 往年期末题 参考

## 判断题

- 1-1 错误， $x = 0$ 时， $(x|-x)>>31$  为0。
- 1-2 正确，PPT 原话
- 1-3 错误，运算与比较中出现无符号数时有符号数转为无符号数，左端 -1 转为无符号数 ( $1u<<31$ )
- 1-4 错误，指令两端不能同时操作内存。
- 1-5 正确，这是 fork 函数的特征。
- 1-6 错误， $x>>3$  向下舍入， $x/8$  向0舍入。
- 1-7 正确
- 1-8 错误，load effective address 指令（lea）的目的操作数必须是一个寄存器（中文版P129）

## 单选题

- 2-1 B      错了？
- short 两字节，si的二进制表示为 `1000 0000`，转换为无符号数为128。
- 2-2 B
- 补充：上面那一段是a1.c，下面那一段时a2.c
- 编译时，a2.c中的变量 x 是弱符号，归入a1.c中的强符号 x 中，函数 f 对其产生的引用对全局的 x 变量产生影响。
- 2-3 A
- 注意，OF（Overflow Flag）针对的是有符号数溢出；这里的运算没有发生有符号数溢出，所以为0。值得注意的是 $-30 + 60 = (1110\ 0010) + (0011\ 1100)$ 发生无符号数溢出，CF（Carry Flag）为1。
- $\downarrow 1110\ 0010.$
- 2-4 D
- 2-5 A
- 小端法内存低位存数据低位，大端法内存地位存数据高位。
- 2-6 A
- 2-7 原问题补充： x 和 des 分别存储在寄存器 B
- 2-8 D
- 中断：设备IO，异步发生。
  - 陷阱： `syscall`，向内核进行请求，同步发生，总能返回原进程。
  - 故障：出现问题（例如缺页），可能出错，不一定能返回原进程，同步发生。
  - 终止：发生致命错误，同步发生，原进程停止。
- 2-9 D
- 2-10 D
- 动态链接库可以在多个阶段进行链接
- 2-11 C
- `gcc -o`：可执行文件
  - `gcc -S`：汇编文件
  - `gcc -c`：可重定位文件
  - `objdump -d`：二进制文件反汇编
- 2-12 C
- 2-13 D
- 书上没有显式指明 `%rax` 是调用者保存，但是 `%rax` 的功能是保存被调用者返回值，显然 `%rax` 的值在调用者调用前是需要调用者自己保存的。

2-14 B

$\text{ADDR}[\&T[i]] = x_t + \text{sizeof}(T[i]) * (i)$

2-15 C

指针类型强制转换不会改变其内部的值,  $\text{val}(\text{pointer} + \text{offset}) = \text{val}(\text{pointer}) + \text{sizeof}(\text{pointer指向的类型}) * \text{offset}$

2-16 A

2-17 A

movl 高32位清零。

2-18 D

2-19 C

标准IO库不适用于网络套字节的IO, 并且标准IO不能读取文件的元数据。

2-20 rectify:他的十进制是 A

$0xC0B00000 = (1\ 10000001\ 011), \text{sign} = -1, \text{Exponent} = 129 - \text{bias} = 2, M = 1 + f = 1.375$

填空题

4-1

表达式	十进制表示	二进制表示
零	0	000000
-	-10	<b>110110</b>
x	-1	<b>111111</b>
ux	<b>63</b>	<b>111111</b>
x>>1	<b>-1</b>	<b>111111</b>
TMin	<b>-32</b>	<b>100000</b>
TMin+TMin	<b>0</b>	<b>000000</b>

4-2

1. foo1 对应choice**5**
  - 他这里做的是从向下舍入除法变为向零舍入除法。
2. foo2 对应choice**2**
  - 注意 foo2 进行的是逻辑右移
3. foo3 对应choice**3**
  - 很典型的乘法二进制拆分

4-3

汇编进行一下简单翻译, 就是: `return *(mat1+3*i+j)+*(mat2+9*i+j)`

不难得知 $N = 3, M = 9$

4-4

在这里, 编译器用jump-to-middle法翻译for循环。

1. `result = 0`
2. `y < x`
3. `x--`
4. `y++`

4-5    **这个貌似也有问题**

- a[0] 被破坏
- a[1] 被破坏 (hint: 字符串末尾有00结束符)
- x 的值没有被破坏 (x的值被保存在了-0x24(%rbp), 没有被栈溢出攻击)

- `rbp` 中的值没有被破坏
- `a[0] = 0x33343536`
- `a[1] = 0x00bbccdd`
- `buf=01234567890123456`
- 当输入到第**32**个字符后无法从`bar`函数正常返回。