第二章相关内容：

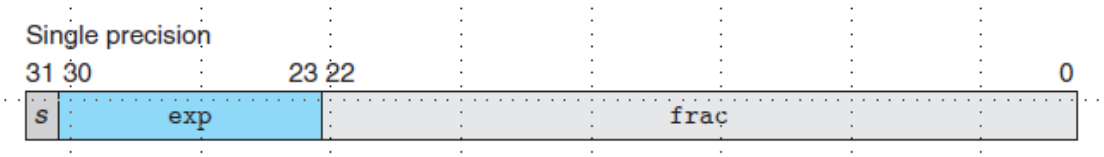
1. 请写出 int 类型最大值、最小值、-1 和 0 值的十六进制表示，unsigned short 类型的最大值、最小值的二进制表示。

答：

- Int 类型：
  - 最大值：0x7FFFFFFF
  - 最小值：0x80000000
  - -1：0xFFFFFFFF
  - 0：0x00000000
- Unsigned short 类型：
  - 最大值：0xFFFF
  - 最小值：0x0000

2. 请写出单精度浮点数的“非负值最小规格化数”的小数表示 和 “最小非规格化数”的二进制表示(单精度浮点数的阶码字段占用 8 位)。

答：



- 单精度浮点数的“非负值最小规格化数”的小数表示：1.0×2^{-126}
  - 因为是非负值，所以符号位 s=0;
  - 规格化数要求阶码 exp 不为 0 和 255，所以最小规格化数的阶码 exp 为 0x01;
  - 尾数 frac 最小可为 0;
  - 则小数表示为$1.0 \times 2^{1-127} = 1.0 \times 2^{-126}$。
- 单精度浮点数的“最小非规格化数”的二进制表示：
  - 因为没有要求为非负值，则符号位 s 可为 1;
  - 非规格化数，所以 exp 为 0x00000000;
  - 由于符号位为 1，即此浮点数为负数，因此尾数要最大，才能使得该浮点数最大，则 frac 有 23 个 1 构成;
  - 因此二进制表示为：1 00000000 11111111111111111111111。

3. 写出 8 位浮点数（阶码采用 4 位，小数位采用 3 位）“0 0110 110”所表示的数值。

答：

- 符号位为 0，则该浮点数为非负值;
- 阶码为 0110，则阶数为 0110 – 0111 = 1111,即-1
- 尾数为 110，则“0 0110 110”所表示的数值为$1.110 \times 2^{-1} = 0.1110$(二进制表示)，其 10 进制表示为 0.875

4. 现有代码： int i=0xab cd ef 01;
short si=i;
请问代码执行后，变量 si 的数值表示为（十六进制）？

答：直接截断取低 16 位，因此 si 的数值表示为 0xef 01。

5. 如果 int i=0x86 23 11 32，&i=0x400320，请问地址 0x400322 所对应内存上的那个字节存储的数值是？

这道题可能会有歧义，会被理解成两条语句指向完后，地址 0x400322 所对应内存上的那个字节存储的数值是？如果这样理解的话，&i=0x400320;是不能编译过的。建议将题目改成：
"如果 int i=0x86 23 11 32，i 的地址为 0x400320，请问地址 0x400322 所对应内存上的那个字节存储的数值是？**"**

答：

- 大端：

地址	0x400320	0x400321	0x400322	0x400323
值	0x86	0x23	0x11	0x32

地址 0x400322 所对应内存上的那个字节存储的数值是为 0x11。

- 小端：

地址	0x400320	0x400321	0x400322	0x400323
值	0x32	0x11	0x23	0x86

地址 0x400322 所对应内存上的那个字节存储的数值是为 0x23。

第三章相关内容：

1. （控制）写出下面函数 Func1 汇编代码对应的 C 程序，其中参数 1 为 x，参数 2 为 y：

```
Func1:
    cmpq    %rsi, %rdi
    jge  .L2
    leaq    3(%rsi), %rdi
    jmp .L3
.L2:
    leaq    (%rdi,%rdi,4), %rsi
    addq    %rsi, %rsi
.L3:
    leaq    (%rdi,%rsi), %rax
    ret
```

答：

```
long func(long x, long y) {
```

```
    if (x < y) {
        x = y + 3;
    } else {
        // y = 10*x;
        y = 5*x;
        y += y;
    }
    return x + y;
}
```

2 （多重数组+lea 指令）对于数组 int B[8][5],需要将 B[i][j]保存到 eax 中，数组起始地址保存在 rdi，i 保存在 rsi，j 保存在 rdx 中，请完成以下代码中的空缺

```
leaq    (    ,%rsi,    ), %rax
leaq    (    ,    ,    ), %rax
movl    (    ,    ,    ), %eax
```

答:

```
B[i][j] 等价于 *(B + 5*i + j)
leaq    (  %rsi  ,%rsi,  4 ), %rax          // %rax = 5*i
leaq    ( %rax , %rdx ,  1 ), %rax          // %rax += j
movl    (  %rdi ,  %rax ,  4 ), %eax        // %eax = *(B + %rax)
```

3. （数组+函数+乘法的移位实现）已知 int P[M][N]和 int Q[N][M], 有以下函数:

```
int addfun(  int i,int j){
    return P[i][j]+Q[j][i];
}
```

对应有汇编代码如下，请问 M 和 N 分别是多少?

```
addfun:
    movl    %edi, %edx
    shl     $2,%edx
    addl    %esi,%edx
    movl    %esi,%eax
    shll    $2,%eax
    addl    %eax,%edi
    movl    Q(,%rdi,4),%eax
    addl    P(,%rdx,4), %eax
    ret
```

答:

```
addfun:
```

```
movl    %edi, %edx        # %edx = i
shl     $2,%edx           # %edx = 4*i
addl    %esi,%edx         # %edx = 4*i + j
movl    %esi,%eax         # %eax = j
shll    $2,%eax           # %eax = 4*j
addl    %eax,%edi         # %edi = 4*j + i
movl    Q(,%rdi,4),%eax   # %eax = *(Q + 4*j + i)
addl    P(,%rdx,4), %eax # %eax += *(P + 4*I + j)
ret
```

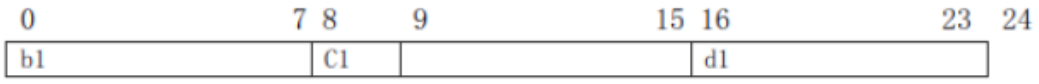
由上可知 M 和 N 均为 4

4. （union+结构体）

```
union a1{
    struct {int * b1; char c1; long d1 } str1;
    double data[3];
    }
```

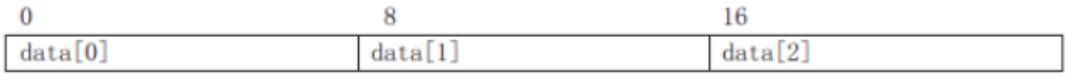
请问按照默认的对齐方式，上述 a1.str1 占用多少字节空间？ a1 占用多少字节空间？

a1.str1:



共 24 字节。

a1.data:



同样是共 24 字节。

即 union 中需要最大的空间为 24 字节，则 a1 占用 24 字节。

5. （结构体+函数+控制）已知 node 结构体定义如下

```
struct node{
        long a;
```

```
        struct node *next;
    }
```

请对以下 init 函数进行逆向分析，写出其 C 代码

```
Init:
    movl    $12,%eax
    jmp     .TestExprStat
.Loop:
    addq    (%rdi),%rax
    movq    8(%rdi),%rdi
.TestExprStat:
    testq   %rdi,%rdi
    jne     .Loop
    ret
```

答:

```
long init(struct node *p) {
    long res = 12;
    while(p) {
        res += p->a;
        p = p->next;
    }
    return res;
}
```

6. （结构体）已知结构体定义如下

```
struct{
    char a;
    char *b;
    short c;
    int d;
}
```

请问在紧凑布局和对齐布局中 a/b/c/d 字段的偏移量各是多少?

答:

（这里的答案是指 64 位环境下的，因此指针大小为 64bit，即 8 字节）

字段名	类型	类型大小(字节)	紧凑布局偏移	对齐布局偏移
a	char	1	0	0
b	char *	8	1	8
c	short	2	9	16
d	int	4	11	20

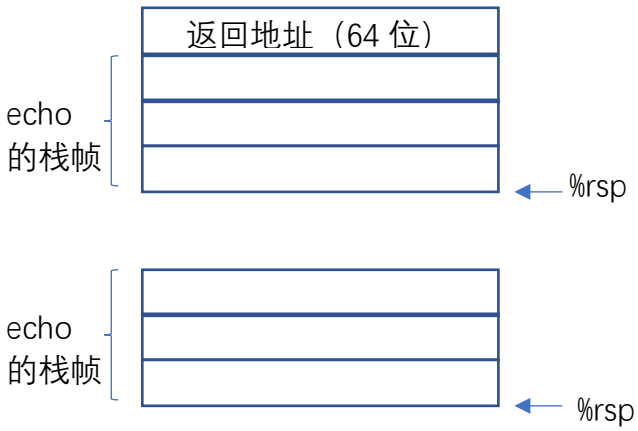
7. （堆栈破坏问题）函数 echo 定义如下:

```
    void echo(){
```

```
    char buf[8];
    gets(buf);
    puts(buf);
}
```

对应的汇编代码如下：

```
echo:
        subq    $24,%rsp
        movq    %fs:40,%rax
        movq    %rax,8(%rsp)
        xorl    %eax,%eax
        movq    %rsp,%rdi
        call    gets
        movq    %rsp,%rdi
        call    puts
        movq    8(%rsp),%rax
        xorq    %fs:40,%rax
        je      .L9
        call    __stack_chk_fail
        addq    $24,%rsp
        ret
```



观察代码，判定该函数是否具有堆栈破坏的检测能力？如果%fs:38 地址开始存放了 0x00/01/02/03/04/05/06/07/08/09/0a/0b/0c/0d/0e/0f。请问刚进入 echo 函数时，echo 栈帧中%rsp+8 位置存放的 8 字节数值是？如果此时输入按键 abcdefg 并回车，程序将如何执行？如果此时输入按键 123456789 并回车，程序能否正常返回？如果不能将执行什么处理？

答：

- 具有堆栈破坏的检测能力，因为在调用 gets 前，在传入的 buf 后面设置了一个金丝雀值，并在 echo 函数退出前，检查了该值是否被修改；
- 刚进入 echo 函数时，echo 栈帧中%rsp+8 位置存放的 8 字节数值是 echo 的返回地址；
- 因为“abcdefg”只有 7 个字符，加上“\0”刚好 8 个字符，buf 刚好为大小为 8 的字符数组，因此正常执行，输出 abcdefg；
- 因为“123456789”含有 8 个字符，因此在调用 gets 时会出现缓冲区溢出的现象，因此修改了金丝雀值，故会调用“__stack_chk_fail”，不能正常返回。

8. （函数参数+浮点）对于一下汇编代码，请写出对应的 C 函数代码（整数参数请使用 a/b，浮点参数请使用 c）

```
myfun:
    movsbl      %dil, %edi
    imull       $30, %edi, %edi
    addl        (%rsi), %edi
    movl        %edi, (%rsi)
    cvtsi2ss    %edi, %xmm1
    addss       %xmm1, %xmm0
    ret
```

答:

- movsbl：进行符号扩展，将 1Byte 符号扩展成 4Byte
- cvtsi2ss：将一个有符号整数转换为一个单精度浮点数
- addss：浮点数加法
- %xmm0：为第一个浮点参数和单精度浮点数类型的返回值

```c
float myfunc(char a, int *b, float c) {
    *b = *b + 30 * a;
    return c + *b;
}
```