

华东师范大学期中考试试卷参考答案

2017—2018 学年第二学期

课程名称： 计算机系统

课程性质： 专业必修.

学生姓名： _____

学 号： _____

专 业： _____

年级/班级： _____

一、选择题 (20 分)

1. 32 位长的补码表示的 `integer` 数据的最小值是 ()。
(a) -2^{32} (b) $-2^{32}+1$ (c) -2^{31} (d) $-2^{31}+1$
2. `mov` 和 `lea` 指令的区别是 ()。
(a) `lea` 指令需要解引用 (dereferences) 地址, `mov` 指令不需要。
(b) `mov` 指令需要解引用 (dereferences) 地址, `lea` 指令不需要。
(c) `lea` 指令可以用于寄存器间数据传递, 而 `mov` 指令不能。
(d) `mov` 指令可以用于寄存器间数据传递, 而 `lea` 指令不能。
3. 下列代码执行后, 哪些变量的值为 0? ()

```
unsigned int a=0xffffffff;  
unsigned int b = 1;  
unsigned int c = a + b;  
unsigned long d = a + b;  
unsigned long e = (unsigned long)a + b;
```


(a) 都不为 0 (b) c (c) c and d (d) c, d, and e
4. X86-64 机型中关于 `%rbx` 和 `%ebx` 寄存器的区别, 下列说法正确的是 ()。
(a) `%rbx` 和 `%ebx` 是同一个寄存器。
(b) `%ebx` 引用的是 `%rbx` 寄存器的低 32 位数据。
(c) `%rbx` 和 `%ebx` 是完全不同的两个寄存器。
(d) `%ebx` 引用的是 `%rbx` 寄存器的高 32 位数据。
5. 下列说法不正确的是 () ?
(a) x86-64 能提供比 x86 更大的虚地址空间。
(b) x86 和 x86-64 的栈规则不同。
(c) x86 使用 `%ebp` 存储栈帧的基地址。

(d) x86-64 使用 %ebp 存储栈帧的基地址。

6. `test %eax, %eax`

`jne 3d<function+0x3d>`

当%eax 为 () 时, 会引起程序跳转。

(a) 1 (b) 0 (c) %eax 任意值 (d) %eax 没有合适值能满足指令跳转

7. 下列 C 语言代码中, 与汇编指令: `lea 0xffffffff(%esi), %eax` 功能相对应的是 ()。

(a) `*(esi-1) = eax`

(b) `esi = eax + 0xffffffff`

(c) `eax = esi - 1`

(d) `eax = *(esi - 1)`

8. 针对下列 C 代码片段, 其输出是 ()。

```
unsigned int x = 0xDEADBEEF;
```

```
unsigned short y = 0xFFFF;
```

```
signed int z = -1;
```

```
if (x > (signed short) y)    printf(“Hello”);
```

```
if (x > z)    printf(“World”);
```

(a) 没有输出 (b) “Hello” (c) “World” (d) “HelloWorld”

9. 假设一种 8 位浮点数定义为: 1 sign, 3 exponent, 4 fraction。下列哪种表示 NaN? ()

(a) 1 000 1111

(b) 0 111 1111

(c) 0 100 0000

(d) 1 111 0000

10. 假设%rsp 值为 0xdeadbeefdeadd0d0。执行完指令: `pushq %rbx` 后, %rsp 的值为 ()。

(a) 0xdeadbeefdeadd0d4

(b) 0xdeadbeefdeadd0d8

(c) 0xdeadbeefdeadd0cc

(d) 0xdeadbeefdeadd0c8

二、（8分）假设一种整数为6位字长，根据第一列的描述，填写其对应的值。

描述	值
U_{\max}	
T_{\min}	
(unsigned) ((int) 4)	
(unsigned) ((int) -7)	
((unsigned) 0x21) << 1 & 0x3F)	
(int) (20 + 12)	
12 && 4	
(! 0x15) > 16	

三、（8分）考虑基于 IEEE 浮点格式标准的 5 位浮点表示，这种格式没有符号位，只表示非负数（阶码 3 位，尾数 2 位）。

该 5 位浮点数数值为 $V=1.M \times 2^E$ ，其中 E 是偏置后的阶码值，M 是尾数值。

下面，给出以下十进制值，请写出其对应的浮点数二进制表示，并请给出其舍入值（采用 round-to-even 舍入规则）

值	浮点數位模式	舍入值
9/32	001 00	1/4
3	100 10	
9	110 00	
3/16	000 11	
15/2	110 00	

四、（10分）在X86-64机型上，产生整型变量x、y、z的随机数，试判断下列C表达式是否恒为真。

```
/* Create some arbitrary values */
int x = random();
int y = random();
int z = random();
```

```

/* Convert to other forms */
unsigned ux = (unsigned) x;
unsigned uy = (unsigned) y;
double dx = (double) x;
double dy = (double) y;
double dz = (double) z;

```

表达式	是否恒为 “真”	表达式	是否恒为 “真”
$(x < y) == (-x > -y)$		$((x \gg 1) \ll 1) \leq x$	
$((x+y) \ll 4) + y - x == 17*y + 15*x$		$(\text{double})(\text{float}) x == (\text{double}) x$	
$\sim x + \sim y + 1 == \sim (x+y)$		$dx + dy == (\text{double}) (y+x)$	
$ux - uy == -(y - x)$		$dx + dy + dz == dz + dy + dx$	
$(x \geq 0) \mid\mid (x < ux)$		$dx * dy * dz == dz * dy * dx$	

五、(8分)考虑如下C函数, 请根据对应的汇编代码, 填写C代码中缺失的表达式。

```

long mystery1(unsigned long x) {
    if(_____)
        return ____;
    unsigned long nx = ____;
    long rv = mystery1(nx);
    return ____;
}

```

其对应的汇编代码如下:

```

long mystery1(unsigned long x)
x in %rdi
mystery1:
    pushq %rbx
    movq %rdi, %rbx
    movl $0, %eax
    testq %rdi, %rdi
    je .L2
    shrq $2, %rdi
    call mystery1
    addq %rbx, %rax
.L2:
    popq %rbx
    ret

```

六、(8分)考虑如下C程序。

```
#include <stdio.h>

/* Read a string from stdin into buf */

int evil_read_string() {

int buf[2];

scanf("%s",buf); return buf[1];

}

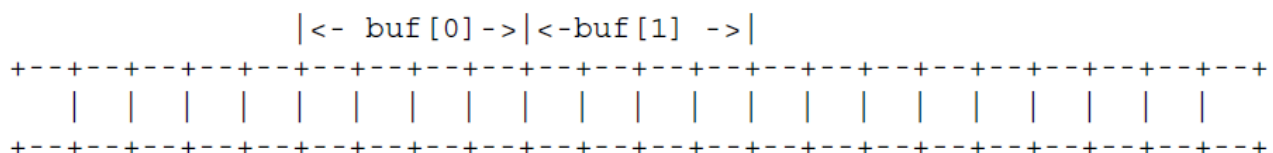
int main() {

printf("0x%x\n", evil_read_string()); }
```

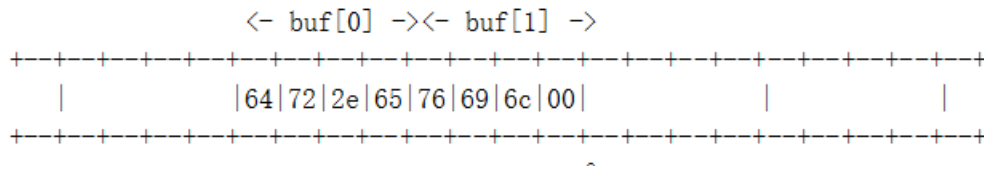
- scanf("%s", buf)：从标准输入流（stdin）中读入字符串，并存储在 buf 开始的存储器中（字符串以 ' \0' 结束，不检查 buf 存储器的长度）。
- printf("0x%x", i)：以 "0x" 开始打印输出整型变量 i 的十六进制值。
- Linux/x86 机器采用小端法。
- 部分字符的 ASCII 码如下：

字符	ASCII	字符	ASCII	字符	ASCII
'd'	0x64	'i'	0x69	'e'	0x65
'v'	0x76	'.'	0x2e	's'	0x73
'r'	0x72	'l'	0x6c	'\0'	0x00

A. 上述程序运行在 Linux/x86 机器，输入字符串 dr.evil，针对如下所示的栈结构，请填写 buf[0]和 buf[1]中存储的数据（以十六进制表示）



B. 程序的输出结果是什么? 0x_____



七、 (18 分) 考虑 x86-64 机型下的如下结构:

```
struct my_struct {
    char a;
    long long b;
    short c;
    float *d[2];
    unsigned char e[3];
    float f;
};
```

A. (8 分) 请在下面的存储器中完成结构 `my_struct` 的布局。利用 `XX` 表示填充的字节, 并使用 “`end`” 指示结构的结束。

地址								
0x0								
0x8								
0x10								
0x18								
0x20								
0x28								
0x30								
0x38								

B. (6 分) 考虑如下 C 语言函数, 请填写汇编代码中缺失部分。

```
void foo(struct my_struct *st) {
    st->a = ' e' ;
    st->d[0] = NULL;
    st->c = 0x213;

    printf("%lld %p %hhu\n", st->b, &st->f,
st->e[1]);
}
```

```
(gdb) disassemble foo
Dump of assembler code for function foo:
0x00000000004004e4 <+0>:      sub     $0x8,%rsp
0x00000000004004e8 <+4>:      movb    $0x65,_____(%rdi)
0x00000000004004eb <+7>:      movq    $0x0,_____(%rdi)
0x00000000004004f3 <+15>:     movw    $0x213,_____(%rdi)
0x00000000004004f9 <+21>:     movzbl  _____(%rdi),%ecx
0x00000000004004fd <+25>:     lea     _____(%rdi),%rdx
0x0000000000400501 <+29>:     mov     _____(%rdi),%rsi
0x0000000000400505 <+33>:     mov     $0x40062c,%edi
0x000000000040050a <+38>:     mov     $0x0,%eax
0x000000000040050f <+43>:     callq   0x4003e0 <printf@plt>
0x0000000000400514 <+48>:     add     $0x8,%rsp
0x0000000000400518 <+52>:     retq
End of assembler dump.
```

C. (4 分) 对 my_struct 结构中元素重新排序, 该结构最少应包含多少字节?

- (a) 48
- (b) 36
- (c) 40
- (d) 以上都不是

八、(10 points): 某程序中包含 switch 语句, 采用跳转表实现。其跳转表如下:

```
0x4004b7:      jmpq    *0x400600(,%rax,8)
```

Using GDB, we extract the 8-entry jump table:

```
0x400600: 0x00000000004004d1 0x00000000004004c8
0x400610: 0x00000000004004c8 0x00000000004004be
0x400620: 0x00000000004004c1 0x00000000004004d7
0x400630: 0x00000000004004c8 0x00000000004004be
```

包含 switch 语句的 C 语言程序的反汇编代码如下：

```
# on entry: %rdi = x, %rsi = y, %rdx = z
0x4004b0:    cmp     $0x7,%edx
0x4004b3:    ja      0x4004c8
0x4004b5:    mov     %edx,%eax
0x4004b7:    jmpq    *0x400600(,%rax,8)
0x4004be:    mov     %edi,%eax
0x4004c0:    retq
0x4004c1:    mov     $0x3,%eax
0x4004c6:    jmp     0x4004da
0x4004c8:    mov     %esi,%eax
0x4004ca:    nopw    0x(%rax,%rax,1)
0x4004d0:    retq
0x4004d1:    mov     %edi,%eax
0x4004d3:    and     $0x19,%eax
0x4004d6:    retq
0x4004d7:    lea     (%rdi,%rdi,1),%eax
0x4004da:    add     %esi,%eax
0x4004dc:    retq
```

请分析反汇编程序以及跳转表，填写下列 C 程序中的缺失部分：

```
int test(int x, int y, int z)
{
    int result = 3; switch(z)
    {
        case   (1)  :
                      (2)          ;

        case   (3)  :

        case   (4)  :
            result =           (5)          ;

            break;

        case   (6)  :
            result =           (7)          ;

        case   (8)  :
            result =           (9)          ;

            break;

        default:
```



```

    result = _____(10)_____; }
return result; }

```

九、(10 分):考虑如下 C 语言代码, 其中 M 和 N 是由#define 声明的常量。

```
int array1[M][N]; int array2[N][M];
```

```
int copy(int i, int j) {
    array1[i][j] = array2[j][i]; }

```

对应的汇编代码如下:

```

# on entry: %rdi = i, %rsi = j
copy:

movl %edi,%ecx

movl %esi,%ebx

leal (%ecx,%ecx,8),%edx

sall $2,%edx

movl %ebx,%eax

sall $4,%eax

subl %ebx,%eax

sall $2,%eax

movl array2(%eax,%ecx,4),%eax    //60*j+4*i

movl %eax,array1(%edx,%ebx,4)    //36*i+4*j

ret

```

M 和 N 的值是多少?