ICS Seminar Week4 Prep

余文凯 康子熙 赵廷昊 许珈铭

2023.10.7

Rules

remainder <- ordinal number in WeChat Group % 4

for all questions do

    if question number % 4 == remainder then

        you should work on it

    end

end

func

9. `pushq %rbp` 的行为等价于以下（）中的两条指令。

A. `subq $8, %rsp  movq %rbp, (%rdx)`

B. `subq $8, %rsp  movq %rbp, (%rsp)`

C. `subq $8, %rsp  movq %rax, (%rsp)`

D. `subq $8, %rax  movq %rbp, (%rdx)`

B

5、已知函数 int x(int n) {　return　n*____; } 对应的汇编代码如下：

```
lea (%rdi, %rdi, 4), %rdi
lea (%rdi, %rdi, 1), %eax
retq
```

请问横线上的数字应该是（　　）

A. 4　　B. 5　　C. 2　　D. 10

D

12. 将下列汇编代码翻译成 C 代码

```
func:
    movq   %rsi, %rax
    testq  %rdi, %rdi
    jne .L7
    rep ret
.L7:
    subq   $8, %rsp
    imulq  %rdi, %rax
    movq   %rax, %rsi
    subq   $1, %rdi
    call   func
    addq   $8, %rsp
    ret
```

```
long func(long n, long m) {
    if (__________)
        return _________;
    return func (_____, _____);
}
```

n == 0
m
n – 1
m * n

6. 有如下代码段：

```
int func(int x, int y);
int (*p) (int a, int b);
p = func;
p(0,0);
```

对应的下列 x86-64 过程调用正确的是：

A. call *%rax　　B. call (%rax)

C. call *(%rax)　　D. call func

A/D

5. 已知函数 func 的参数超过 6 个。当 x86-64 机器执行完指令 call func 之后，%rsp 的值为 S。那么 func 的第 k(k > 6)个参数的存储地址是？

A. S + 8 * (k - 6)

B. S + 8 * (k - 7)

C. S - 8 * (k - 6)

D. S - 8 * (k - 7)

A

4、下列关于 x86-64 过程调用的叙述中，哪一个是不正确的？

A．每次递归调用都会生成一个新的栈帧，空间开销大

B．当传递给被调用函数的参数少于 6 个时，可以通过通用寄存器传递

C．被调用函数要为局部变量分配空间，返回时无需释放这些空间

D．过程调用返回时，向程序计数器中推送的地址是调用函数中调用指令的下一条指令的地址

C

pointer & list

7. 考虑以下 C 语言变量声明：

```
int *(*f[3])();
```

那么在一台 x86-64 机器上，sizeof(f)和 sizeof(*f)的值是多少？

A. 8 24

B. 24 8

C. 8 8

D. 8 不确定

B

7. 有A的定义：int A[3][2] = {{1,2}, {3,3}, {2,1}};
   那么A[2]的值为：

A. &A+16　　B. A+16　　C. *A+4　　D. *A+2

C

3、假定静态 int 型二维数组 a 和指针数组 pa 的声明如下：

```
static int a[4][4]={ {3, 8, -2, 6}, {2, 1, -5, 3 }, {1, 18,
4, 10},{4, -2, 0, 8}};
static int *pa[4]={a[0], a[1], a[2], a[3]};
```

若 a 的首地址为 0x601080，则&pa[0]和 pa[1]分别是：

A. 0x6010c0、0x601090

B. 0x6010e0、0x601090

C. 0x6010c0、0x6010a0

D. 0x6010e0、0x6010a0

A

7.阅读下列 C 代码和在 x86-64 机器上得到的汇编代码：

```
int a[__A__][__B__];
for (int i = 0; i < __C__; i++)
  a[i][__C__ - i] = 1;

  leaq 40(%rdi), %rax
  addq $440, %rdi
.L2:
  movl $1, (%rax)
  addq $40, %rax
  cmpq %rdi, %rax
  jne  .L2
```

假设 a 的地址初始时放在%rdi 中，假设程序正常运行且没有发生越界问题，则 C 代码中的 A、B、C 处应分别填：

A. 10、11、10

B. 9、11、9

C. 11、10、10

D. 11、11、11

A

struct & union

6、32 位 x86 计算机、Windows 操作系统下定义的一个 structure S 包含三个部分：double a, int b, char c，请问 S 在内存空间中最多和最少分别能占据多少个字节（32 位 Windows 系统按 1、4、8 的原则对齐 char、int、double）？答：（　　）

A. 16, 13
B. 16, 16
C. 24, 13
D. 24, 16

D

6、有如下定义的结构，在 x86-64 下，下述结论中错误的是？

```
struct {
    char c;
    union {
        char vc;
        double value;
        int vi;
    } u;
    int i;
} sa;
```

A. sizeof(sa) == 24

B. (&sa.i - &sa.u.vi) == 8

C. (&sa.u.vc - &sa.c) == 8

D. 优化成员变量的顺序，可以做到“sizeof(sa) == 16”

答：（ ）

B

5. 已知下面的数据结构，假设在 Linux/IA32 下要求对齐，这个结构的总的大小是多少个字节？如果重新排列其中的字段，最少可以达到多少个字节？

```
struct {
  char a;
  double *b;
  double c;
  short d;
  long long e;
  short f;
};
```

A. 32, 28　　B. 36, 32　　C. 28, 26　　D. 26, 26

A

5、假设结构体类型 student_info 的声明如下：

```
struct student_info {
        char id[8];
        char name[16];
        unsigned zip;
        char address[50];
        char phone[20];
}x;
```

若 x 的首地址在%rdx 中，则“unsigned xzip=x.zip;”所对应的汇编指令为：

A. movl 0x24(%rdx), %rax

B. movl 0x18(%rdx), %rax

C. leal 0x24(%rdx), %rax

D. leal 0x18(%rdx), %rax

B

14. 在 x86-64、Linux 操作系统下有如下 C 定义：

```
struct A {
    char CC1[6];
    int II1;
    long LL1;
    char CC2[10];
    long LL2;
    int II2;
};
```

(1) sizeof(A) = _____字节。

(2) 将 A 重排后，令结构体尽可能小，那么得到的新的结构体大小为_____字节。

56

40

7. 假设存储器按“大端法”存储数据对象，已知如下的C语言数据结构：union { char c[2]; int i; }; 当c的值为0x01, 0x23时，i的值为：

A. 0x0123　B. 0x2301　C. 0x01230000　D.不确定

D

8. 在 x86-64 架构下，有如下变量：

```
union {char c[8], int i;} x;
```

在 x.i=0x41424344 时，x.c[2]的值为多少（提示：'A'=0x41）：

A. 'A'　　B. 'B'　　C. 'C'　　D. 'D'

B

5. 以下代码的输出结果是

```
union {
    double d;
    struct {
        int i;
        char c[4];
    } s;
} u;
u.d = 1;
printf("%d\n", u.s.c[2]);
```

A) 0　　B) -16　　C) 240　　D) 191

B

1. 考虑如下代码在 x86-64 处理器上的运行情况：

```
union U{
  char x[12];
  char *p;
}u;

int main() {
  strcpy(u.x, "I love ICS!");
  printf("%p\n", u.p);
  return 0;
}
```

提示：

1) "I love ICS!"对应的字节序列是 49 20 6c 6f 76 65 20 49 43 53 21

2) %p 用于输出一个指针的值

程序运行的输出是：

A. 0x49206c6f76652049435321

B. 0x2153434920657666c2049

C. 0x49206c6f76652049

D. 0x492065766f6c2049

D

2. 函数 g 定义如下：

```
int g(float f){
  union {
    float f;
    int i;
  } x;
  x.f = f;
  return x.i ^ ((1u << (x.i < 0 ? 31 : 0)) - 1);
}
```

则对于两个 float 型变量 a,b，有 a<b 是 g(a)<g(b)的：

A. 充分必要条件

B. 充分不必要条件

C. 必要不充分条件

D. 不必要不充分条件

B

6. 下列关于C语言中的结构体(struct)以及联合(union)的说法中，正确的是：

A) 对于任意struct，将其成员按照其实际占用内存大小从小到大的顺序进行排列不一定会使之内存占用最小

B) 对于任意struct，将其成员按照其实际占用内存大小从小到大的顺序进行排列一定不会使之内存占用最大

C) 对于任意union，将其成员按照其实际占用内存大小从小到大的顺序进行排列不一定会使之内存占用最小

D) 对于任意union，将其成员按照其实际占用内存大小从小到大的顺序进行排列一定不会使之内存占用最大

A

1. (10 分) 在 x86-64、LINUX 操作系统下，考虑如下的 C 定义：

```
typedef union {
    char c[7];
    short h;
} union_e;
typedef struct {
    char d[3];
    union_e u;
    int i;
} struct_e;
struct_e s;
```

回答如下问题：

(1) `s.u.c` 的首地址相对于 s 的首地址的偏移量是_____字节。

(2) `sizeof(union_e)` = _____字节。

(3) s.i 的首地址相对于 s 的首地址的偏移量是_____字节。

(4) 若将`i` 的类型改成`short`、将`h` 的类型改成`int`，那么`sizeof(union_e)` = _____字节，`sizeof(struct_e)` = _____字节。

blanks

第三题（20 分）

1.考虑下面的union的声明，回答后面的问题。

```
union ELE {
    struct {
        int x;
        int *p;
    }e1;
    struct {
        union ELE * next;
        int y;
    }e2;
};
```

(注：32位机器)

这个union总共大小为多少____字节。

2.假设编译器为process的主体产生了如下了代码，请补充完整下面的过程：（只有一个不需要任何强制类型转换且不违反任何类型限制的答案）

```
    movl 8(%ebp),%eax
    movl (%eax),%ecx
    movl 4(%ecx),%edx
    movl (%edx),%edx
    subl 4(%eax),%edx
    movl %edx,(%ecx)
```

```
void process(union ELE * up)
{
    up->_____=_____- _____;
}
```

第三题（20 分）

1.考虑下面的union的声明，回答后面的问题。

```
union ELE {
    struct {
        int x;
        int *p;
    }e1;
    struct {
        union ELE * next;
        int y;
    }e2;
};
```

(注：32位机器)

这个union总共大小为多少____字节。

2.假设编译器为process的主体产生了如下了代码，请补充完整下面的过程：（只有一个不需要任何强制类型转换且不违反任何类型限制的答案）

```
    movl 8(%ebp),%eax
    movl (%eax),%ecx
    movl 4(%ecx),%edx
    movl (%edx),%edx
    subl 4(%eax),%edx
    movl %edx,(%ecx)

void process(union ELE * up)
{
    up->_____=_____- _____;
}
```

8

up->next->x=*(up->next->p) - (up->y)

第三题（20 分）

1.考虑下面的union的声明，回答后面的问题。

```
union ELE {
    struct {
        int x;
        int *p;
    }e1;
    struct {
        union ELE * next;
        int y;
    }e2;
};
```

(注：32位机器)

这个union总共大小为多少____字节。

3.请查看下文完成如下功能的汇编代码，定位错误语句并进行更正：

给出 n(在%ebp+8 位置,n>=1),up(在%ebp+12 位置,ELE* 类型),假设以*up 为头元素(设*up 为第 0 个),由声明中的 next 连接形成了一个链表,请将第 n 个元素(假设链表足够长)的 x 的值放入%eax 中

X86 代码

```
xorl %ecx,%ecx
movl 8(%edx),%ebp
movl 12(%ebp),%eax

LOOP:
movl (%eax),%eax
add $1,%ecx
test %ecx,%edx
jne LOOP

movl (%eax),%eax
```

第三题（20 分）

1.考虑下面的union的声明，回答后面的问题。

```
union ELE {
    struct {
        int x;
        int *p;
    }e1;
    struct {
        union ELE * next;
        int y;
    }e2;
};
```

(注：32位机器)

这个union总共大小为多少____字节。

3.请查看下文完成如下功能的汇编代码，定位错误语句并进行更正：

给出 n(在%ebp+8 位置,n>=1),up(在%ebp+12 位置,ELE* 类型),假设以*up 为头元素(设*up 为第 0 个),由声明中的 next 连接形成了一个链表,请将第 n 个元素(假设链表足够长)的 x 的值放入%eax 中

X86 代码

```
xorl %ecx,%ecx
movl 8(%edx),%ebp
movl 12(%ebp),%eax

LOOP:
movl (%eax),%eax
add $1,%ecx
test %ecx,%edx
jne LOOP

movl (%eax),%eax
```

8

movl 8 (%edx), %ebp1应为movl 8 (%ebp), %edx

4.阅读下列代码，回答后面的问题

```
typedef struct {
    short x[A][B];
    int y;
}str1;

typedef struct {
    char array[B];
    int t;
    short s[B];
    int u;
}str2;
```

```
void setVal(str1 *p,str2 *q) {
    int v1=q->t;
    int v2=q->u;
    p->y=v1+v2;
}
```

(short 以 2 字节计算)

GCC 为 setVal 的主体产生下面的代码：

```
    movl 12(%ebp),%eax
    movl 28(%eax),%edx
    addl 8(%eax),%edx
    movl 8(%ebp),%eax
    movl %edx,44(%eax)
```

请直接写出A和B的值各是多少？

4.阅读下列代码，回答后面的问题

```
typedef struct {
    short x[A][B];
    int y;
}str1;

typedef struct {
    char array[B];
    int t;
    short s[B];
    int u;
}str2;
```

```
void setVal(str1 *p,str2 *q) {
    int v1=q->t;
    int v2=q->u;
    p->y=v1+v2;
}
```

(short 以 2 字节计算)

GCC 为 setVal 的主体产生下面的代码：

```
    movl 12(%ebp),%eax
    movl 28(%eax),%edx
    addl 8(%eax),%edx
    movl 8(%ebp),%eax
    movl %edx,44(%eax)
```

请直接写出A和B的值各是多少？

A=3, B=7

第三题（15 分）

数据结构定义如下：（测试用机：64 位 Linux 机器）

```
typedef struct s1 {
        char cc[N];
        int ii[N];
        int *ip;
} S1;

S1 t1[N];
```

1、计算该数据结构的空间：

当 N=3 时，sizeof(S1) = (1)_____，sizeof(t1) = (2)_____

当 N=4 时，sizeof(S1) = (3)_____，sizeof(t1) = (4)_____

当 N=5 时，sizeof(S1) = (5)_____，sizeof(t1) = (6)_____

2、当 N=4 时，该数据结构初始化代码如下：

```
void init(int n)
{
        int i;
        for (i=0; i<n; i++) {
                t1[i].ip = &(t1[i].ii[i]);
        }
}
```

根据上述代码，填写下面汇编中缺失的内容：

```
init:
        movl    $0, %ecx
        jmp     .L2
.L3:
                movslq  %ecx, %rax
                leaq    ((1)_____,%rax,8), %rsi
        leaq    0(,%rsi,4), %rdx
        addq    $t1+4, %rdx
        salq    (2)_____, %rax
        movq    (3)_____, ((4)_____)
        addl    $1, %ecx
.L2:
        cmpl    (5)_____, %ecx
        jl      .L3
        rep ret
```

第三题（15 分）

数据结构定义如下：（测试用机：64 位 Linux 机器）

```
typedef struct s1 {
        char cc[N];
        int ii[N];
        int *ip;
} S1;

S1 t1[N];
```

1、计算该数据结构的空间：

当 N=3 时，sizeof(S1) = (1)_____，sizeof(t1) = (2)_____

当 N=4 时，sizeof(S1) = (3)_____，sizeof(t1) = (4)_____

当 N=5 时，sizeof(S1) = (5)_____，sizeof(t1) = (6)_____

2、当 N=4 时，该数据结构初始化代码如下：

```
void init(int n)
{
        int i;
        for (i=0; i<n; i++) {
                t1[i].ip = &(t1[i].ii[i]);
        }
}
```

根据上述代码，填写下面汇编中缺失的内容：

```
init:
        movl    $0, %ecx
        jmp     .L2
.L3:
                movslq  %ecx, %rax
                leaq    ((1)_____,%rax,8), %rsi
        leaq    0(,%rsi,4), %rdx
        addq    $t1+4, %rdx
        salq    (2)_____, %rax
        movq    (3)_____, ((4)_____)
        addl    $1, %ecx
.L2:
        cmpl    (5)_____, %ecx
        jl      .L3
        rep ret
```

24 72
32 128
40 200
%rax
$5
%rdx
t1+24(%rax)
%edi

第三题（15 分）

数据结构定义如下：（测试用机：64 位 Linux 机器）

```
typedef struct s1 {
        char cc[N];
        int ii[N];
        int *ip;
} S1;

S1 t1[N];
```

1、计算该数据结构的空间：

当 N=3 时，sizeof(S1) = (1)_____，sizeof(t1) = (2)_____

当 N=4 时，sizeof(S1) = (3)_____，sizeof(t1) = (4)_____

当 N=5 时，sizeof(S1) = (5)_____，sizeof(t1) = (6)_____

3、当 N=3 时，函数 fun 的汇编代码如下：

```
fun:
        movslq  %esi, %rax
        movslq  %edi, %rdi
        leaq    (%rdi,%rdi), %rdx
        leaq    (%rdx,%rdi), %r8
        leaq    (%r8,%r8), %rcx
        addq    %rcx, %rax
        movl    %esi, t1+4(,%rax,4)
        addq    %rdx, %rdi
        leaq    0(,%rdi,8), %rax
        movq    t1+16(%rax), %rax
        movl    %esi, (%rax)
        ret
```

根据上述代码，填写函数 fun 的 C 语言代码：

```
void fun(int x, int y)
{
        (1)_____ = (2)_____;
        (3)_____ = (4)_____;
}
```

第三题（15 分）

数据结构定义如下：（测试用机：64 位 Linux 机器）

```
typedef struct s1 {
        char cc[N];
        int ii[N];
        int *ip;
} S1;

S1 t1[N];
```

1、计算该数据结构的空间：

当 N=3 时，sizeof(S1) = (1)_____，sizeof(t1) = (2)_____

当 N=4 时，sizeof(S1) = (3)_____，sizeof(t1) = (4)_____

当 N=5 时，sizeof(S1) = (5)_____，sizeof(t1) = (6)_____

3、当 N=3 时，函数 fun 的汇编代码如下：

```
fun:
        movslq  %esi, %rax
        movslq  %edi, %rdi
        leaq    (%rdi,%rdi), %rdx
        leaq    (%rdx,%rdi), %r8
        leaq    (%r8,%r8), %rcx
        addq    %rcx, %rax
        movl    %esi, t1+4(,%rax,4)
        addq    %rdx, %rdi
        leaq    0(,%rdi,8), %rax
        movq    t1+16(%rax), %rax
        movl    %esi, (%rax)
        ret
```

根据上述代码，填写函数 fun 的 C 语言代码：

```
void fun(int x, int y)
{
        (1)_____ = (2)_____;
        (3)_____ = (4)_____;
}
```

24 72

32 128

40 200

t1[x].ii[y] = y

*(t1[x].ip) = y

第一段代码：

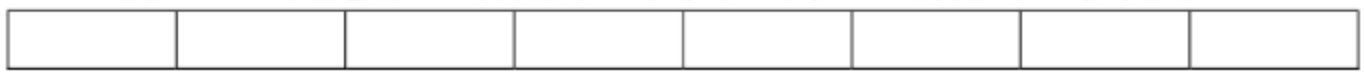
```
#include<stdio.h>

typedef struct
{
    int x;
    int y;
}struct_e;

typedef union
{
    struct_e s;
    double d;
}union_e;

int main () {
    union_e test;
    test.s.x = 0;
    test.s.y = -1;
    printf("%lf\n", test.d);

    test.s.y = 0x3ff00000;
    printf("%lf\n", test.d);
}
```

（1）以上程序的两次输出分别是 _____， _____。

（2） sizeof(union_e) = _____。

（3）如果某次程序输出的 test.d 是最大的负非规格化数，则

此时从 test 的起始地址开始的 8 个字节（用 16 进制表示）依次是：



此时

test.s.x = _____

test.s.y = _____

第一段代码：

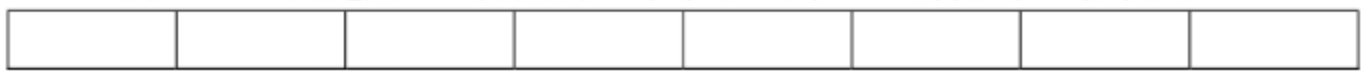
```
#include<stdio.h>

typedef struct
{
    int x;
    int y;
}struct_e;

typedef union
{
    struct_e s;
    double d;
}union_e;

int main () {
    union_e test;
    test.s.x = 0;
    test.s.y = -1;
    printf("%lf\n", test.d);

    test.s.y = 0x3ff00000;
    printf("%lf\n", test.d);
}
```

（1）以上程序的两次输出分别是 _____， _____。

（2） sizeof(union_e) = _____。

（3）如果某次程序输出的 test.d 是最大的负非规格化数，则

此时从 test 的起始地址开始的 8 个字节（用 16 进制表示）依次是：



此时

test.s.x = _____

test.s.y = _____

NAN 1

8

01 00 00 00 00 00 00 80

1

-2147483647-1 (-2^31)