

题号	一	二	三	四	五	六	七	八	九	十	总分
得分											
评卷人											

一、单项选择题：(每小题 2 分，共 20 分，在以下每小题给出的 A、B、C、D 四个选项中，只有一个选项正确，将正确答案填写在试卷上的相应位置。)

1. 在编译系统中，szu.c 经过汇编阶段后生成文件为_____。

- A、szu.i B、szu.s C、szu.o D、szu.exe

2. 变量 x 的值为 0x01234567，地址 &x 为 0x100；则该变量的值在 Sun (大端机) 机器内存中的存储排列顺序正确的是_____。

选项	0x100	0x101	0x102	0x103
A	01	23	45	67
B	01	23	45	67
C	67	45	23	01
D	01	23	45	67

3. 在 IEEE754 浮点数运算中，NaN 的表示方式是_____。

- A、0 11111111 000000000000000000000000 B、1 11111111 000011111000000000000000
C、0 00000000 000000000000000000000000 D、1 00000000 000000000000000000000000

4. 假设寄存器 %eax 的值为 x，%ecx 的值为 y，那么汇编代码指令 leal (%eax, %ecx, 4), %edx，则存储在寄存器 %edx 中的值为_____。

- A、4x B、4y C、4x+y D、4y+x

5. 以下哪一个不在 ELF 可执行目标文件中_____?

- A、机器代码 B、全局变量 C、用户栈 D、符号表

6. 假设调用关系如下：func1.o → func2.o，func1.o → libx.a 中的函数，func2.o → libx.a 中的函数，libx.a → liby.a，同时 liby.a → libx.a，则以下正确的链接命令是_____。

- A、gcc static -o szu func1.o func2.o libx.a liby.a libx.a
B、gcc static -o szu func2.o func1.o liby.a libx.a liby.a
C、gcc -static -o szu libx.a liby.a libx.a func1.o func2.o

D、gcc -static -o szu liby.a libx.a liby.a func1.o func2.o

7. 下列程序运行的结果是_____?

<pre>/* main.c */ int i = 0; int main() { foo(); return 0; }</pre>	<pre>/* foo.c */ int i = 1; void foo() { printf("%d", i); }</pre>
--	---

A、编译错误

B、链接错误

C、段错误

D、有时打印输出 1，有时打印输出 0;

8. 下列说法中，错误的是_____?

A、函数名和已初始化的全局变量名是强符号

B、强符号只能被定义一次，否则产生链接错误

C、对弱符号的引用可以被解析为其强定义符号

D、未初始化的全局变量名和本地局部符号是弱符号

9. 下列几种存储器中，_____是易失性存储器。

A、cache

B、EPROM

C、Flash Memory

D、CD-ROM

10. 假定一个磁盘存储器有 2 个盘片，每个盘片有 2 个盘面，柱面数为 2000，每个磁道上有 3000 个扇区，每个扇区 512B，则该磁盘存储器的容量约为_____?

A、6TB

B、12TB

C、6GB

D、12GB

二、简答题：(2 小题，共 10 分)

1. (5 分) 对于 IEEE754 浮点数，如果减少 1 位指数位数，将其用于尾数部分会有怎样的效果?

2. (5 分) 请简单描述缓冲区溢出的过程。

三、(10 分)假定 CPU 从存储器中读出一个 4 字节信息 $D=BF400000H$, 请回答下列问题。

- (1) 若 D 是一个 32 位补码表示的带符号整数, 则其真值是多少?
- (2) 若 D 是一个 IEEE 754 单精度浮点数, 则其值是多少?

四、(10分) 考虑下面的结构声明、set_y 函数主体及对应的汇编代码。其中，A, B, C 未知

```
typedef struct {
    int x[B];
    int y;
    int z[C];
} struct_a;
typedef struct {
    struct_a data[A];
    int idx;
} struct_b;
void set_y(struct_b *bp, int val)
{
    int idx = bp->idx;
    bp->data[idx].y = val;
}
```

GCC 为 set_y 函数产生了如下的代码片段:

```
set_y:
    movslq 168(%rdi),%rax
    leaq (%rax,%rax,2), %rax
    movl %esi, 12(%rdi,%rax,8)
    ret
```

请回答如下问题:

- (1) 根据汇编代码, 推理出 A、B 和 C 的值各是多少? 要求有较详细的推理过程。
- (2) 这两个结构总共各需要多少个字节进行存储? 请画图说明每个结构体成员的存储位置, 偏移量分别是多少 (以字节为单位)。

五、(10分) 函数 Sum_N 的定义如下, 请补全该函数对应的汇编代码片段中缺失的部分。

```
int Sum_N(int *p, int N)
{
    int i, sum=0;
    for (i=0; i< N; i++) {
        sum = sum + *p;
        p++;
    }
    return sum;
}
```

```
movl    $0, %eax
movl    $0, %ecx
```

.L1

```
jl      .L1
ret
```

//参数 N 保存在哪个寄存器? _____

//%eax 保存的是哪个变量? _____

.L4

.L5

Ret

2. (15分) 有两个C程序编译链接而成的可执行文件 main-link
main-link.c

```
#include <stdio.h>
extern int var1;
int var2=100;
int localfun1()
{
    var1=exfun2(var2);
    return var1;
}
int main()
{
    int k;
    k=localfun1();
    return k;
}
```

exfun.c

```
int var1=100;
int exfun1(int a)
{
    int temp;
    temp=a+100;
    return temp;
}
int exfun2(int b)
{
    int temp;
    temp=100*b;
    return temp;
}
```

查看 exfun.o 目标文件的反汇编信息如下

0000000000000000 <exfun1>

0: 55	push %rbp
1: 48 89 e5	mov %rsp,%rbp
4: 89 7d ec	mov %edi,-0x14(%rbp)
7: 8b 45 ec	mov -0x14(%rbp),%eax
a: 83 c0 64	add \$0x64,%eax
d: 89 45 fc	mov %eax,-0x4(%rbp)
10: 8b 45 fc	mov -0x4(%rbp),%eax
13: 5d	pop %rbp
14: c3	retq

0000000000000015 <exfun2>:

15: 55	push %rbp
16: 48 89 e5	mov %rsp,%rbp
19: 89 7d ec	mov %edi,-0x14(%rbp)
1c: 8b 45 ec	mov -0x14(%rbp),%eax
1f: 6b c0 64	imul \$0x64,%eax,%eax
22: 89 45 fc	mov %eax,-0x4(%rbp)
25: 8b 45 fc	mov -0x4(%rbp),%eax
28: 5d	pop %rbp
29: c3	retq

可执行文件 main-link 的反汇编部分内容如下。

00000000004004ed <localfun1>:

六、(10分) 考虑如下 echo 函数源代码和其对应的 x86-64 位汇编代码：画出 subq 语句执行前和 addq 语句执行前的栈帧结构，标出对应寄存器前后变化，并回答以下的问题。

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

```
echo:
    subq $24,%rsp
    movq %rsp+0x10,%rdi
    call gets
    movq %rsp+0x10,%rdi
    call puts
    addq $24,%rsp
    ret
```

- (1) 若输入的字符串为 0123456 并回车，函数 echo 的输出为何？
- (2) 若输入的字符串为 0123456789 并回车，程序能否正常输出 012345678901234567？为什么？
- (3) 如果我们想利用函数 echo 的漏洞来执行一个起始地址为 0x3030303030303132 的函数，请给出一个恰当的输入字符串。(字符 0-9 的 ASCII 码为 0x30-0x39)

七、(10分) 考虑如下一个 C 函数和对应的 x86-64 汇编代码。C 代码有缺失, 请在下划线处补全 (注意: 0x400498 是 C 标准库函数 malloc 的地址)。

```
typedef struct node
{
    void *data;
    struct node *next;
} node_t;

node_t *lmao (node_t *n, int f (node_t *))
{
    node_t *a, *b;

    if (_____)
        return NULL;

    a = _____;

    if (_____)
    {
        b = _____;
        b->data = n->data;

        b->next = _____;
        return b;
    }

    return _____;
}
```

```
0x4005d0: mov    %rbx, -0x18(%rsp)
0x4005d5: mov    %rbp, -0x10(%rsp)
0x4005da: xor    %eax, %eax
0x4005dc: mov    %r12, -0x8(%rsp)
0x4005e1: sub    $0x18, %rsp
0x4005e5: test   %rdi, %rdi
0x4005e8: mov    %rdi, %rbx
0x4005eb: mov    %rsi, %rbp
0x4005ee: je     0x40061e<lmao+78>
0x4005f0: mov    0x8(%rdi), %rdi
0x4005f4: callq  0x4005d0 <lmao>
0x4005f9: mov    %rbx, %rdi
0x4005fc: mov    %rax, %r12
0x4005ff: callq  *%rbp
0x400601: mov    %eax, %edx
0x400603: mov    %r12, %rax
0x400606: test   %edx, %edx
0x400608: jle    0x40061e <lmao+78>
0x40060a: mov    $0x10, %edi
0x40060f: callq  0x400498 <malloc>
0x400614: mov    (%rbx), %rdx
0x400617: mov    %r12, 0x8(%rax)
0x40061b: mov    %rdx, (%rax)
0x40061e: mov    (%rsp), %rbx
0x400622: mov    0x8(%rsp), %rbp
0x400627: mov    0x10(%rsp), %r12
0x40062c: add    $0x18, %rsp
0x400630: retq
```


八、(10分) 已知一个C语言程序有两个源文件 fact1.c 和 fact2.c, 其内容如下:

```
/*fact1.c*/
int table[12];
int fact(int n);
int main (int argc, char **argv) {
    int n;
    table[0] = 0; table[1] = 1;
    if (argc == 2) {
        argv++;
        sscanf(argv, "%d", &n);
    }
    printf("fact(%d) = %d\n", n, fact(n));
}
```

```
/*fact2.c*/
int* table;
int fact(n) {
    static int num = 2;
    if (n >= num) {
        int i = num;
        while (i <= n) {
            table[i] = table[i-1] * i;
            i++;
        }
        num = i;
    }
    return table[n];
}
```

(1) 对于每个程序中的相应符号, 给出它的符号类型(局部变量、强定义或弱定义), 它在链接后位于 ELF 文件中的什么位置? (提示: 如果某表项内容无法确定, 请画 X)。

fact1.c

变量	是否链接符号	在哪个模块定义	符号类型	所在节
table				
fact				
num				

fact2.c

符号	是否链接符号	在哪个模块定义	符号类型	所在节
table				
fact				
num				

(2) 对上述两个文件进行链接之后, 会对每个符号进行解析。请给出链接后下列符号被定义的模块 (fact1 or fact2)。

符号	定义模块
table	
fact	
num	

(3) 使用 gcc (命令: gcc -o fact fact1.c fact2.c) 来编译之后得到的可执行文件是否能够正确执行? 为什么?