

# 北京大学信息科学技术学院考试试卷

考试科目： 计算机系统导论 姓名： \_\_\_\_\_ 学号： \_\_\_\_\_

考试时间： 2015 年 1 月 13 日 小班教师： \_\_\_\_\_

题号	一	二	三	四	五	六	七	八	九	总分
分数										
阅卷人										

## 北京大学考场纪律

1、考生进入考场后，按照监考老师安排隔位就座，将学生证放在桌面上。无学生证者不能参加考试；迟到超过 15 分钟不得入场。在考试开始 30 分钟后方可交卷出场。

2、除必要的文具和主考教师允许的工具书、参考书、计算器以外，其它所有物品（包括空白纸张、手机、或有存储、编程、查询功能的电子用品等）不得带入座位，已经带入考场的必须放在监考人员指定的位置。

3、考试使用的试题、答卷、草稿纸由监考人员统一发放，考试结束时收回，一律不准带出考场。若有试题印制问题请向监考教师提出，不得向其他考生询问。提前答完试卷，应举手示意请监考人员收卷后方可离开；交卷后不得在考场内逗留或在附近高声交谈。未交卷擅自离开考场，不得重新进入考场答卷。考试结束时间到，考生立即停止答卷，在座位上等待监考人员收卷清点后，方可离场。

4、考生要严格遵守考场规则，在规定时间内独立完成答卷。不准交头接耳，不准偷看、夹带、抄袭或者有意让他人抄袭答题内容，不准接传答案或者试卷等。凡有违纪作弊者，一经发现，当场取消其考试资格，并根据《北京大学本科考试工作与学术规范条例》及相关规定严肃处理。

5、考生须确认自己填写的个人信息真实、准确，并承担信息填写错误带来的一切责任与后果。

学校倡议所有考生以北京大学学生的荣誉与诚信答卷，共同维护北京大学的学术声誉。

以下为试题和答题纸，共 24 页。

得分

第一题 单项选择题（每小题 1 分，共 20 分）

1. 下面关于 IEEE 浮点数标准说法正确的是（ ）
  - A. 在位数一定的情况下，不论怎么分配 exponent bits 和 fraction bits，所能表示的数的个数是不变的
  - B. 如果甲类浮点数有 10 位，乙类浮点数有 11 位，那么甲所能表示的最大数一定比乙小
  - C. 如果甲类浮点数有 10 位，乙类浮点数有 11 位，那么甲所能表示的最小正数一定比乙小
  - D. "0111000"可能是 7 位浮点数的 NAN 表示

2. 假设有下面 x 和 y 的程序定义

```
int x = a >> 2;
```

```
int y = (x + a) / 4;
```

那么有多少个位于闭区间 $[-8, 8]$ 的整数 a 能使得 x 和 y 相等？（ ）

- A. 12
- B. 13
- C. 14
- D. 15

3. 左边的 C 函数中，在 x86\_64 服务器上采用 GCC 编译产生的汇编语言如右边所示。那么 (1) 和 (2) 的内容分别是：（ ）

```

                                <arith>:
                                lea    (%rsi,%rdi,1),%eax
                                mov     %esi,%edx
int arith(int x, int y) {      mov     %esi,%edx
    return (x < y) ? (1) : (2); sub     %edi,%edx
}                               cmp     %esi,%edi
                                cmovge %edx,%eax
                                retq

```

（提示：第一个参数放在 rdi 寄存器中，第二个参数放在 rsi 寄存器中）

- A. x-y, x+y    B. x+y, x-y    C. x+y, y-x    D. y-x, x+y

4. 假定 `struct P {int i; char c; int j; char d;};` 在 `x86_64` 服务器的 Linux 操作系统上, 下面哪个结构体的大小与其它三个不同: ( )
- A. `struct P1 {struct P a[3]};`
  - B. `struct P2 {int i[3]; char c[3]; int j[3]; char d[3]};`
  - C. `struct P3 {struct P *a[3]; char *c[3]};`
  - D. `struct P4 {struct P *a[3]; int *f[3]};`
5. 根据编译器安全优化的策略, 如下手工程序代码的优化, 哪个达不到优化效果? ( )
- A. 循环展开, 以减少循环的迭代次数
  - B. 将函数调用移到循环内, 以提高程序的模块性
  - C. 消除不必要的存储器引用, 减少访存开销
  - D. 分离多个累计变量, 以提高并行性
6. 通常情况下, 下面的哪些表述是正确的? ( )
- A. 在一次读操作中, 返回的内容由高速缓存中的信息块决定
  - B. 高速缓存利用了时间局部性
  - C. 大部分情况下, 缓存需要用户程序采取显式的管理行为
  - D. 一级高速缓存更看重命中率, 二级高速缓存更看重命中时间
7. 在代码中, 变量 `sum` 具有的特性是: ( )
- ```
int sumvec(int v[N])
{
    int i, sum = 0;

    for (i = 0; i < N; i++)
        sum += v[i];
    return sum;
}
```
- A. 良好的时间局部性
  - B. 良好的空间局部性
  - C. 同时具有良好的时间局部性和空间局部性
  - D. 都不具有
8. 下列关于静态库链接的描述中, 错误的是 ( )
- A. 链接时, 链接器只拷贝静态库中被程序引用的目标模块
  - B. 使用库的一般准则是将它们放在命令行的结尾

- C. 如果库不是相互独立的，那么它们必须排序
- D. 每个库在命令行只须出现一次即可

9. 在 `foo.c` 文件中的函数外，如果添加如下一条语句：

```
static int count = 0xdeadbeef;
```

那么它在编译为 `foo.o` 后，会影响到 ELF 可重定位目标文件中的除 `.text` 以外的哪些字段？ ( )

- A. `.rodata`
- B. `.data`, `.symtab`,
- C. `.data`, `.symtab`, `.rel.data`
- D. `.rodata`, `.symtab`, `.rel.data`

10. 在系统调用成功的情况下，下列代码会输出几个 `hello`? ( )

```
void doit()
{
    if ( fork() == 0 ) {
        printf("hello\n");
        fork();
    }
    return ;
}

int main()
{
    doit();
    printf("hello\n");
    exit(0) ;
}
```

- A. 3
- B. 4
- C. 5
- D. 6

11. 下列说法中哪一个是错误的？ ( )

- A. 中断一定是异步发生的
- B. 异常处理程序一定运行在内核模式下
- C. 故障处理一定返回到当前指令
- D. 陷阱一定是同步发生的

12. 下列这段代码的输出不可能是 ( )

```
void handler()
{
    printf("h");
}

int main()
{
    signal(SIGCHLD, handler) ;

    if ( fork() == 0 ) {
        printf("a") ;
    } else {
        printf("b") ;
    }
    printf("c") ;
    exit(0) ;
}
```

- A. abcc      B. abch      C. bcach      D. bchac

13. 对于虚拟存储系统, 一次访存过程中, 下列命中组合不可能发生的是 ( )

- A. TLB 未命中, Cache 未命中, Page 未命中  
B. TLB 未命中, Cache 命中, Page 命中  
C. TLB 命中, Cache 未命中, Page 命中  
D. TLB 命中, Cache 命中, Page 未命中

14. 有程序段如下:

```
int foo( ) {
    char str1[20], *str2;
    str2 = (char*)malloc(20*sizeof(char));
    free(str2);
}
```

下列说法中正确的是 ( )

- A. str1 和 str2 指向的内存都是分配在栈空间内的  
B. str1 和 str2 指向的内存都是分配在堆空间内的  
C. str1 指向的内存是分配在栈空间内的, str2 指向的内存是分配在堆空间内的  
D. str1 指向的内存是分配在堆空间内的, str2 指向的内存是分配在栈空间内的

15. ICS.txt 中包含 3000 个字符，考虑如下代码段：

```
int main(int argc, char** argv) {
    int fd = open("ICS.txt", O_CREAT | O_RDWR, S_IRUSR |
S_IWUSR);
    write(fd, "ICS", 3);

    char buf[128];
    int i;
    for (i = 0; i < 10; i++) {
        int fd1 = open("ICS.txt", O_RDWR);
        int fd2 = dup(fd1);

        int cnt = read(fd1, buf, 128);
        write(fd2, buf, cnt);
    }
    return 0;
}
```

上述代码执行完后，ICS.txt 中包含多少个字符（假设所有系统调用都成功）？  
( )

- A. 3                      B. 256                      C. 3000                      D. 3072

16. 下列系统 I/O 的说法中，正确的是 ( )

- A. C 语言中的标准 I/O 函数在不同操作系统中的实现代码一样
- B. 对于同一个文件描述符，混用 RIO 包中的 rio\_readnb 和 rio\_readn 两个函数不会造成问题
- C. C 语言中的标准 I/O 函数是异步线程安全的
- D. 使用 I/O 缓冲区可以减少系统调用的次数，从而加快 I/O 的速度

17. 如果两个局域网高层分别采用 TCP/IP 协议和 SPX/IPX 协议，那么可以选择的互连设备应是 ( )

- A. 网桥
- B. 集线器
- C. 路由器
- D. 交换机

18. 下面说法正确的是 ( )

- A. TCP 是一种可靠的无连接协议
- B. UDP 是一种不可靠的无连接协议
- C. Web 浏览器与 web 服务器通信采用的协议是 HTML
- D. 数字数据只能通过数字信号传输

19. 对于如下 C 语言程序:

```
#include "csapp.h"
void *thread (void * arg)
{
    printf("Hello World") ;
    Pthread_detach(pthread_self()) ;
}
int main(void)
{
    pthread_t tid;
    int sta ;
    sta = Pthread_create(&tid, NULL, thread, NULL);
    if (sta==0)
        printf("Oops, I can not create thread\n");
    exit(NULL);
}
```

在上述程序中, Pthread\_detach 函数的作用是 ( )

- A. 使主线程阻塞以等待线程 thread 结束
- B. 线程 thread 运行结束后会自动释放所有资源
- C. 线程 thread 运行后主动释放 CPU 给其他线程
- D. 线程 thread 运行后成为僵尸线程

20. 两个线程中共享如下一段 C 代码:

```
for (j = 0; j < N; j++)
    count + = 2;
```

假设其对应的汇编代码如下:

```
movq (%rdi), %rcx
testq %rcx,%rcx
jle .L2
movl $0, %eax
.L3:
movq count(%rip),%rdx
addq $2, %rdx
movq %rdx, count(%rip)

addq $1, %rax
cmpq %rcx, %rax
jne .L3
.L2:
```

} H<sub>i</sub>

L<sub>i</sub>  
U<sub>i</sub>  
S<sub>i</sub>

} T<sub>i</sub>

请问在下列指令顺序对应的轨迹线中, 哪一个安全轨迹线? ( )

- A. H<sub>1</sub>, H<sub>2</sub>, L<sub>2</sub>, L<sub>1</sub>, U<sub>2</sub>, U<sub>1</sub>, S<sub>1</sub>, S<sub>2</sub>, T<sub>1</sub>, T<sub>2</sub>
- B. H<sub>1</sub>, L<sub>1</sub>, U<sub>1</sub>, H<sub>2</sub>, L<sub>2</sub>, S<sub>1</sub>, T<sub>1</sub>, U<sub>2</sub>, S<sub>2</sub>, T<sub>2</sub>
- C. H<sub>2</sub>, L<sub>2</sub>, U<sub>2</sub>, H<sub>1</sub>, S<sub>2</sub>, L<sub>1</sub>, T<sub>2</sub>, U<sub>1</sub>, S<sub>1</sub>, T<sub>1</sub>
- D. H<sub>2</sub>, L<sub>2</sub>, H<sub>1</sub>, L<sub>1</sub>, U<sub>1</sub>, U<sub>2</sub>, S<sub>2</sub>, T<sub>2</sub>, S<sub>1</sub>, T<sub>1</sub>

|    |
|----|
| 得分 |
|    |

## 第二题（10 分）汇编语言

阅读下面的 c 代码：

```
unsigned char d[256] = {
    0x00, 0x80, 0x40, 0xc0, 0x20, 0xa0, 0x60, 0xe0,
    0x10, 0x90, 0x50, 0xd0, 0x30, 0xb0, 0x70, 0xf0,
    0x08, 0x88, 0x48, 0xc8, 0x28, 0xa8, 0x68, 0xe8,
    0x18, 0x98, 0x58, 0xd8, 0x38, 0xb8, 0x78, 0xf8,
    0x04, 0x84, 0x44, 0xc4, 0x24, 0xa4, 0x64, 0xe4,
    0x14, 0x94, 0x54, 0xd4, 0x34, 0xb4, 0x74, 0xf4,
    0x0c, 0x8c, 0x4c, 0xcc, 0x2c, 0xac, 0x6c, 0xec,
    0x1c, 0x9c, 0x5c, 0xdc, 0x3c, 0xbc, 0x7c, 0xfc,
    0x02, 0x82, 0x42, 0xc2, 0x22, 0xa2, 0x62, 0xe2,
    0x12, 0x92, 0x52, 0xd2, 0x32, 0xb2, 0x72, 0xf2,
    0x0a, 0x8a, 0x4a, 0xca, 0x2a, 0xaa, 0x6a, 0xea,
    0x1a, 0x9a, 0x5a, 0xda, 0x3a, 0xba, 0x7a, 0xfa,
    0x06, 0x86, 0x46, 0xc6, 0x26, 0xa6, 0x66, 0xe6,
    0x16, 0x96, 0x56, 0xd6, 0x36, 0xb6, 0x76, 0xf6,
    0x0e, 0x8e, 0x4e, 0xce, 0x2e, 0xae, 0x6e, 0xee,
    0x1e, 0x9e, 0x5e, 0xde, 0x3e, 0xbe, 0x7e, 0xfe,
    0x01, 0x81, 0x41, 0xc1, 0x21, 0xa1, 0x61, 0xe1,
    0x11, 0x91, 0x51, 0xd1, 0x31, 0xb1, 0x71, 0xf1,
    0x09, 0x89, 0x49, 0xc9, 0x29, 0xa9, 0x69, 0xe9,
    0x19, 0x99, 0x59, 0xd9, 0x39, 0xb9, 0x79, 0xf9,
    0x05, 0x85, 0x45, 0xc5, 0x25, 0xa5, 0x65, 0xe5,
    0x15, 0x95, 0x55, 0xd5, 0x35, 0xb5, 0x75, 0xf5,
    0x0d, 0x8d, 0x4d, 0xcd, 0x2d, 0xad, 0x6d, 0xed,
    0x1d, 0x9d, 0x5d, 0xdd, 0x3d, 0xbd, 0x7d, 0xfd,
    0x03, 0x83, 0x43, 0xc3, 0x23, 0xa3, 0x63, 0xe3,
    0x13, 0x93, 0x53, 0xd3, 0x33, 0xb3, 0x73, 0xf3,
    0x0b, 0x8b, 0x4b, 0xcb, 0x2b, 0xab, 0x6b, 0xeb,
    0x1b, 0x9b, 0x5b, 0xdb, 0x3b, 0xbb, 0x7b, 0xfb,
    0x07, 0x87, 0x47, 0xc7, 0x27, 0xa7, 0x67, 0xe7,
    0x17, 0x97, 0x57, 0xd7, 0x37, 0xb7, 0x77, 0xf7,
    0x0f, 0x8f, 0x4f, 0xcf, 0x2f, 0xaf, 0x6f, 0xef,
```



```
    0x1f, 0x9f, 0x5f, 0xdf, 0x3f, 0xbf, 0x7f, 0xff,
};
```

```
static inline unsigned char a(unsigned char x)
{
    return d[x];
}
```

```
unsigned short b(unsigned short x)
{
    return (a(x & 0xff) << 8) | a(x >> 8);
}
```

```
unsigned int c(unsigned int x)
{
    return (b(x & 0xffff) << 16) | b(x >> 16);
}
```

1、根据程序逻辑，下面的结果是：

```
c(b(a(1))) = _____ (1)
a(b(c(1))) = _____ (2)
```

2、填写下面反汇编中的缺失的内容：（数组 d 的地址为 0x6009a0）  
（提示：注意反汇编格式与汇编格式有所区别）

00000000004004d0 <b>:

```
4004d0:    mov    %edi,%eax
4004d2:    movzbl %dil,%edx
4004d6:    movzbl 0x6009a0(%rdx),%edx
4004dd:    movzbl %ah,_____(1)_____
4004e0:    movzbl 0x6009a0(%rax),%eax
4004e7:    shl    _____(2)_____,%edx
4004ea:    or     %edx,%eax
4004ec:    retq
```

00000000004004f0 <c>:

```
4004f0:    mov    %edi,%eax
```

```

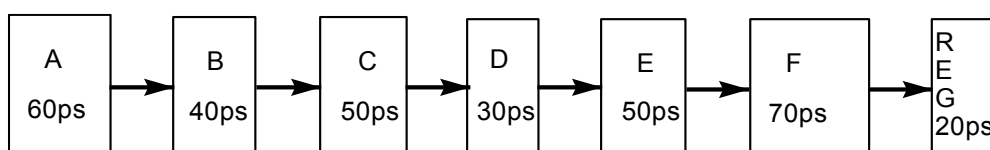
4004f2:      push      _____(3)_____
4004f3:      mov       %edi,%ebx
4004f5:      shr       _____(4)_____,%eax
4004f8:      movzbl    %bh,%ebx
4004fb:      movzbl    %al,_____(5)_____
4004fe:      movzbl    %ah,_____(6)_____
400501:      movzbl    0x6009a0(%rdx),%edx
400508:      movzbl    0x6009a0(%rax),%eax
40050f:      shl       _____(7)_____,%edx
400512:      or        %edx,%eax
400514:      movzbl    %dil,%edx
400518:      movzbl    0x6009a0(%rdx),%ecx
40051f:      movzbl    0x6009a0(%rbx),%edx
400526:      movzwl    %ax,%eax
400529:      pop        _____(3)_____
40052a:      shl       $0x8,%ecx
40052d:      or        %ecx,%edx
40052f:      shl       $0x10,%edx
400532:      or        %edx,_____(8)_____
400534:      retq

```

|    |
|----|
| 得分 |
|    |

### 第三题（10 分）处理器

如图所示，每个模块表示一个单独的组合逻辑单元，每个单元的延迟已在图中标出。通过在两个单元间添加寄存器的方式，可以对该数据通路进行流水化改造。假设每个寄存器的延迟为  $20\text{ps}$ 。注意，由于电路互联特点 A 与 B 之间如果插入寄存器，B 本身的延迟将增加到  $50\text{ps}$ 。



1) 如果改造为一个二级流水线（只插入一个寄存器），为获得最大的吞吐率，该寄存器应在哪里插入？请计算该流水线的吞吐率，并说明计算过程。结果可以是分数形式也可以是小数形式。

2) 如果改造为一个三级流水线（插入两个寄存器），为获得最大的吞吐率，寄存器应在哪里插入？请计算该流水线的吞吐率，并说明计算过程。结果可以是分数形式也可以是小数形式。

3) 如果改造为一个四级流水线(插入三个寄存器), 为获得最大的吞吐率, 寄存器应在哪里插入? 请计算该流水线的吞吐率, 并说明计算过程。结果可以是分数形式也可以是小数形式。

4) 不改变单元划分, 为获得最大性能, 该设计至少需要划分成几级? 请计算对应的吞吐率, 并说明计算过程。结果可以是分数形式也可以是小数形式。

|    |
|----|
| 得分 |
|    |

#### 第四题（10 分）链接

考虑如下3个文件：main.c, fib.c和bignat.c:

```
/* main.c */
void fib (int n);
int main (int argc, char** argv) {
    int n = 0;
    sscanf(argv[1], "%d", &n);
    fib(n);
}

/* fib.c */
#define N 16

static unsigned int ring[3][N];

static void print_bignat(unsigned int* a) {
    int i;
    for (i = N-1; i >= 0; i--)
        printf("%u ", a[i]); /* print a[i] as unsigned int
        */
    printf("\n");
}

void fib (int n) {
    int i, carry;
    from_int(N, 0, ring[0]); /* fib(0) = 0 */
    from_int(N, 1, ring[1]); /* fib(1) = 1 */
    for (i = 0; i <= n-2; i++) {
        carry = plus(N, ring[i%3], ring[(i+1)%3],
        ring[(i+2)%3]);
        if (carry)
            { printf("Overflow at fib(%d)\n", i+2);
            exit(0); }
    }
    print_bignat(ring[n%3]);
}
```

另外，假设在文件 bignat.c 中定义了如下两个函数 plus 和 from\_int (具体定义略):

```
int plus (int n, unsigned int* a, unsigned int* b, unsigned
int* c);
void from_int (int n, unsigned int k, unsigned int* a);
```

1. (5 分) 对于每个程序中的相应符号, 给出它的属性 (局部或全局, 强符号或弱符号) (提示: 如果某表项中的内容无法确定, 请画 x。)

main.c

|      | 局部或全局? | 强或弱? |
|------|--------|------|
| fib  |        |      |
| main |        |      |

fib.c

|      | 局部或全局? | 强或弱? |
|------|--------|------|
| ring |        |      |
| fib  |        |      |
| plus |        |      |

2. (3 分) 假设文件 bignat.c 被编译为一个静态库 bignat.a, 对于如下的 gcc 调用, 会得到什么样的结果 (请选择)?

- (A) 编译和链接都正确  
(B) 链接失败 (原因是包含未定义的引用)  
(C) 链接失败 (原因是包含重复定义)

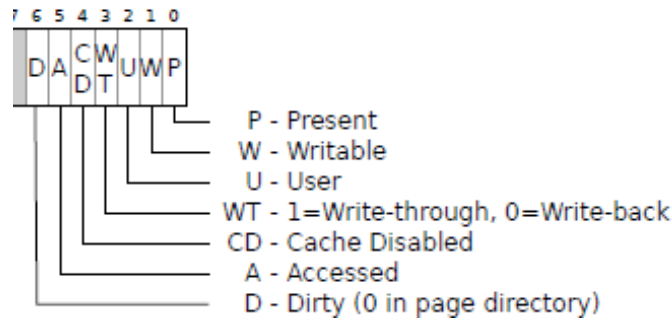
| 命令                               | 结果 (A, B 或 C) |
|----------------------------------|---------------|
| gcc -o fib main.c fib.c bignat.a |               |
| gcc -o fib bignat.a main.c fib.c |               |
| gcc -o fib fib.c main.c bignat.a |               |

3. (2 分) 如果在文件 fib.c 中, 程序员在声明变量 ring 时, 不小心把它写成了: static int ring[3][N];  
会不会影响这些文件的编译、链接和运行结果? 为什么?

|    |
|----|
| 得分 |
|    |

第五题（10 分）虚拟存储

Intel 的 IA32 体系结构采用小端法和二级页表。其中两级页表的大小相同，页大小为 4KB。一级页表和二级页表的表项结构相同，其中页表项后六位的含义如下。



已知一级页表的地址为 0x0c23b000，物理内存中的部分内容如下图所示。

| 地址       | 内容 | 地址       | 内容 | 地址       | 内容 | 地址       | 内容 |
|----------|----|----------|----|----------|----|----------|----|
| 00023000 | E0 | 00023001 | BE | 00023002 | EF | 00023003 | BE |
| 00023120 | 83 | 00023121 | C8 | 00023122 | FD | 00023123 | 12 |
| 00023200 | 23 | 00023201 | FD | 00023202 | BC | 00023203 | DE |
| 00023320 | 33 | 00023321 | 29 | 00023322 | E5 | 00023323 | D2 |
| 00023FF8 | 29 | 00023FF9 | FF | 00023FFA | DE | 00023FFB | BC |
| 00055004 | 03 | 00055005 | D0 | 00055006 | 74 | 00055007 | 89 |
| 0005545C | 97 | 0005545D | C2 | 0005545E | 7B | 0005545F | 45 |
| 00055460 | 97 | 00055461 | D2 | 00055462 | 7B | 00055463 | 45 |
| 00055464 | 97 | 00055465 | E2 | 00055466 | 7B | 00055467 | 45 |
| 0C23B020 | 55 | 0C23B021 | EB | 0C23B022 | AE | 0C23B023 | 24 |
| 0C23B040 | 55 | 0C23B041 | AB | 0C23B042 | 2A | 0C23B043 | 01 |
| 0C23B080 | 05 | 0C23B081 | 5D | 0C23B082 | 05 | 0C23B083 | 00 |
| 0C23B09D | 05 | 0C23B09E | D3 | 0C23B09F | F2 | 0C23B0A0 | 0F |
| 0C23B274 | 05 | 0C23B275 | 3D | 0C23B276 | 02 | 0C23B277 | 00 |
| 0C23B9FC | 25 | 0C23B9FD | D2 | 0C23B9FE | 14 | 0C23B9FF | 23 |
| 2314D200 | 23 | 2314D201 | 12 | 2314D202 | DC | 2314D203 | 0F |
| 2314D220 | A9 | 2314D221 | 45 | 2314D222 | 13 | 2314D223 | D2 |
| 2314D4A0 | BD | 2314D4A1 | BC | 2314D4A2 | 88 | 2314D4A3 | D3 |

|          |    |          |    |          |    |          |    |
|----------|----|----------|----|----------|----|----------|----|
| 2314D890 | 00 | 2314D891 | 2D | 2314D892 | B3 | 2314D893 | 00 |
| 24AEE001 | 07 | 24AEE002 | A0 | 24AEE003 | 37 | 24AEE004 | C2 |
| 24AEE520 | D1 | 24AEE521 | DA | 24AEE522 | 8C | 24AEE523 | B5 |
| 29DE2504 | 02 | 29DE2505 | AD | 29DE2506 | FF | 29DE2507 | 56 |
| 29DE4400 | D0 | 29DE4401 | 5C | 29DE4402 | B4 | 29DE4403 | 2A |
| 29DE9402 | 00 | 29DE9403 | 20 | 29DE9404 | 73 | 29DE9405 | D4 |
| 29DEE500 | B0 | 29DEE501 | CD | 29DEE502 | 23 | 29DEE503 | 1A |

TLB 采用直接映射，TLB 的内容如下所示。

| 索引 | TLB 标记  | 内容       | 有效位 |
|----|---------|----------|-----|
| 0  | 0x08001 | 2314d220 | 1   |
| 1  | 0x01000 | 24aee520 | 0   |
| 2  | 0x005AE | 00055004 | 0   |
| 3  | 0x016BA | 0c23b09d | 1   |
| 4  | 0x0AA00 | 0005545c | 1   |
| 5  | 0x0000A | 29dee500 | 0   |
| 6  | 0x5AE82 | 00023320 | 1   |
| 7  | 0x28DFC | 00023000 | 1   |

1. （2 分）某用户态进程试图写入虚拟地址：0x080016ba。该访问的最后结果是\_\_\_\_\_。

- (a) 该进程成功写入，未触发异常
- (b) 该进程触发了一个缺页异常
- (c) 该进程触发了一个非法访问异常

2. 下面描述了具体的访问过程，请填空。如果某个空在访问过程中已不可用，请填入“--”

2.1 TLB 的索引为\_\_\_\_\_，访问为\_\_\_\_\_ (a) 命中 (b) 不命中（2 分）

2.2 一级页表表项地址为\_\_\_\_\_。（2 分）

2.3 二级页表表项地址为\_\_\_\_\_。（2 分）

2.4 最后物理地址为\_\_\_\_\_。（2 分）



|    |
|----|
| 得分 |
|    |

第六题（10 分）异常

1.（5 分）以下程序运行时系统调用全部正确执行，buffer.txt 文件的内容为 pekinguniv。请给出代码运行后打印输出的结果，并给出程序运行结束后 buffer.txt 文件的内容。

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    char c;
    int file1 = open("buffer.txt", O_RDWR);
    int file2;

    read(file1, &c, 1);
    file2 = dup(file1);
    write(file2, &c, 1);
    printf("1 = %c\n", c);

    int pid = fork() ;
    if (pid == 0) {
        read(file1, &c, 1);
        write(file2, &c, 1);
        printf("2 = %c\n", c);
        read(file1, &c, 1);
        printf("3 = %c\n", c);
        close(file1);
        exit(0);
    } else {
        waitpid(pid, NULL, 0);
        close(file2);
        dup2(file1, file2);
        read(file2, &c, 1);
        write(file2, &c, 1);
        printf("4 = %c\n", c);
    }
    return 0;
}
```

答:

2.(5分)某程序员实现了一个课程实验用的操作系统 ICSNIX,其系统函数 sleep 用以下代码实现。请分析该代码存在哪些问题。

```
1 #include    <signal.h>
2 #include    <unistd.h>
3 static void sig_alrm(int signo)
4 {
5     /* nothing to do, just return to wake up the pause */
6 }
7
8 unsigned int sleep(unsigned int seconds)
9 {
10     if (signal(SIGALRM, sig_alrm) == SIG_ERR)
11         return(seconds);
12
13     alarm(seconds); /* start the timer */
14     pause(); /* next caught signal wakes us up */
15     return(alarm(0)); /* turn off timer, return unslept
time */
16 }
```

答:

|    |
|----|
| 得分 |
|    |

第七题（10 分）系统 I/O

请阅读下面的代码：

```
1: int main(int argc, char** argv) {
2:     int fd1 = open("ICS.txt", O_CREAT|O_RDWR,
3: S_IRUSR|S_IWUSR);
4:     write(fd1, "abc", 3);
5:
6:     int fd2 = fd1;
7:     int fd3 = dup(fd2);
8:     int fd4 = open("ICS.txt", O_APPEND|O_RDWR);
9:     write(fd2, "defghi", 6);
10:    write(fd4, "xyz", 3);
11:
12:    int fd5 = fd4;
13:    dup2(fd3, fd5);
14:    write(fd4, "pqr", 3);
15:
16:    close(fd1);
17:
18:    return 0;
19: }
```

1. 请填写在第 16 行代码刚刚执行完之后，下面的打开文件表和 v-node 表中表项的部分值，并画出表项之间的指向关系。（6 分）

初始时，ICS.txt 文件不存在。程序执行时，所有的系统调用均会成功，所有表项均会从上到下依次分配，描述符表一开始被占用掉前 3 个表项。对于已经释放的打开文件表表项，请填写释放前那一刻的值和指向的 v-node 表表项。

对于多余的表项，请直接忽略。

| 描述符表             | 打开文件表           |        |     | v-node 表     |
|------------------|-----------------|--------|-----|--------------|
| Descriptor table | Open file table |        |     | v-node table |
| ...              | pos             | refcnt | 释放? | 文件名          |
| 3                |                 |        |     |              |
| 4                |                 |        |     |              |
| 5                |                 |        |     |              |
| 6                |                 |        |     |              |
| 7                |                 |        |     |              |

2. 请填写在第 16 行代码刚刚执行完之后，下列变量的值（2 分）

| fd1 | fd2 | fd3 | fd4 | fd5 |
|-----|-----|-----|-----|-----|
|     |     |     |     |     |

3. 请写出程序执行完之后，ICS.txt 文件中的内容（2 分）

|    |
|----|
| 得分 |
|    |

第八题（10 分）网络

1.（1 分）以下问题默认为 IPv4 协议。一个服务器拥有四个独立的固定 IP 地址，那么它在 web 应用端口 80，理论上可以最多再监听\_\_\_\_\_个来自一个客户端独立的 socket 连接（客户端只有一个固定 IP 地址）。

2.（2 分）在 client-server 模型中，一个连接（connection）可以由 IP 地址，端口号的组合来表示。假设客户端 IP 地址为 162.105.192.178，内网 IP 为 192.168.100.121。HTTP 服务器端 IP 地址为 208.216.181.15。

服务器使用的是默认监听端口号。

指出下面这个网页浏览器应用的 Connection socket pair 有什么错误，并简要说明原因？

| 客户端 IP：端口号            | 服务器端 IP：端口号       |
|-----------------------|-------------------|
| 192.168.100.121:15321 | 208.216.181.15:25 |

答：

3.（4 分）在 Echo Server 程序中，客户端（Client）与服务器端（Server）通过 socket 进行一系列的命令和数据交互。

注意：客户端 Connect 命令包含在其 Open\_clientfd 命令中。

请在下图中用单向箭头标出这些交互步骤。例如，当 Client 给 Server 端发送某个命令或者数据时，则需要在 Client 端相应代码行，朝向 Server 端相应代码行画一条单向箭头。

| Echo client code                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Echo Server code                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> int main(int argc, char **argv){     int clientfd, port;     char *host, buf[MAXLINE];     rio_t rio;     host = argv[1];     port = atoi(argv[2]);      clientfd = Open_clientfd(host, port);      Rio_readinitb(&amp;rio, clientfd);      printf("type:");     fflush(stdout);      while(Fgets(buf, MAXLINE, stdin) != NULL) {          Rio_writen(clientfd, buf, strlen(buf));          Rio_readlineb(&amp;rio, buf, MAXLINE);         printf("echo:");         Fputs(buf, stdout);         printf("type:"); fflush(stdout);     }      Close(clientfd);     exit(0); } </pre> | <pre> int main(int argc, char **argv) {     int listenfd, connfd, port, clientlen;     struct sockaddr_in clientaddr;     struct hostent *hp;     char *haddrp;     unsigned short client_port;     port = atoi(argv[1]);      listenfd = open_listenfd(port);     while (1) {         clientlen = sizeof(clientaddr);          connfd = Accept(listenfd, (SA*)&amp;clientaddr, &amp;clientlen);          hp = Gethostbyaddr((const char*)&amp;clientaddr.sin_addr.s_addr,             sizeof(clientaddr.sin_addr.s_addr), AF_INET);          haddrp = inet_ntoa(clientaddr.sin_addr);          client_port = ntohs(clientaddr.sin_port);         printf("server connected");          size_t n; char buf[MAXLINE]; rio_t rio;          Rio_readinitb(&amp;rio, connfd);          while((n = Rio_readlineb(&amp;rio, buf, MAXLINE)) != 0) {             upper_case(buf);              Rio_writen(connfd, buf, n);         }          Close(connfd);     } } </pre> |

4. 关于 Tiny Server 程序，请回答下列问题。

a. (1 分) 下面这段服务器代码用来生成内容的文件是哪个参数？

b. (1 分) 所生成的内容是静态还是动态？请简述原因。

c. (1 分) 如果支持多个客户端请求，下面程序需要添加一个什么功能？

```

/* Return first part of HTTP response */
    sprintf(buf, "HTTP/1.0 200 OK\r\n");
    Rio_writen(fd, buf, strlen(buf));
    sprintf(buf, "Server: Tiny Web Server\r\n");
    Rio_writen(fd, buf, strlen(buf));

    /* Real server would set all CGI vars here */
    setenv("QUERY_STRING", cgiargs, 1);
    Dup2(fd, STDOUT_FILENO); /*Redirect stdout to socket and
client */
    Execve(filename, emptylist, environ);/* Run CGI prog */

```

|    |
|----|
| 得分 |
|    |

第九题（10 分）并发

桌子上有一个水果盘，能容纳一个水果。一家四口人：爸爸、妈妈、儿子、女儿。爸爸专门往盘子里放苹果，妈妈专门往盘子里放桔子；儿子专等盘子里的苹果吃，女儿专等盘子里的桔子吃。

```
dad() {
    while(1) {
        准备好一个苹果;
        ①
        往果盘中放苹果;
        ②
    }
}
mom() {
    while(1) {
        准备好一个桔子;
        ③
        往果盘中放桔子;
        ④
    }
}
boy() {
    while(1) {
        ⑤
        从果盘中拿走苹果;
        ⑥
        吃苹果;
    }
}
girl() {
    while(1) {
        ⑦
        从果盘中拿走桔子;
        ⑧
        吃桔子;
    }
}
```

1. (2 分) 请设计若干信号量，给出每一个信号量的作用和初值。
2. (8 分) 请将信号量上对应的 PV 操作填写在代码中适当位置。