2018

哈尔滨工业大学　　学年　春　季学期

2019

计算机系统基础（A）　试　题

主管领导审核签字

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

片纸鉴心　诚信不败

授课教师　密　姓名　封　学号　线　院系

一、 单项选择题（每小题 2 分，共 20 分）

1. 计算机系统层次体系结构中（ C ）是软硬件系统的交界面。
   A.操作系统　B.汇编语言　C.指令集体系结构　D.微指令
2. 关于 float 浮点数，叙述正确的是（ B ）
   A. 与实数一一对应　B. 0 有两个表示　C. 无法表示无穷大　D. 以上皆错
3. 指令 xorl %eax,%eax 指令功能是（ A ）
   A. 将 eax 赋值为 0　B.同 nop 指令　C.比 movl $0,%eax 慢　D.将 eax 移 1 位
4. 下列存储设备最快的是（ B ）
   A. L1 cache　B. TLB　C.SSD　D.DRAM
5. 库打桩技术不包括是（ C/D ）
   A. 编译时打桩　B.连接时打桩　C.加载时打桩　D.运行时打桩
6. 关于程序运行的下面说法，正确的是（ B ）
   A.由 OS 全部加载到内存后运行　B.由缺页中断加载到物理内存后运行
   C.代码段、数据段是同时加载到内存的　D.运行到 main 时堆栈是空的。
7. 进程的状态不包括(B)
   A.运行　B.休眠　C.停止　D.终止
8. 信号处理子程序的描述错误的是（ D ）
   A.是异常处理子程序　B. 运行在核心态　C. 运行在用户态　D. 以上皆错
9. Linux 下对 IO 设备编程方法错误的是（ D ）
   A.Unix IO 方式　B.RIO 方式　C.标准 IO 方式　D.不同设备采用不同控制函数
10. 妨碍编译器优化的因素，错误的是（ C ）
    A.函数调用会产生副作用　B.多个变量指向同一内存
    C.编译器可对函数进行代码移动　D.编译器在函数附近进行弱优化

二、 填空题（每空 1 分，共 10 分）

1. int x=-4；则 x 在内存从低到高依次存放的数是__FC FF FF FF_(16 进制表示)
2. float y=0.2；则 x 在内存从低到高依次存放的数是__CD CC 4C 3E_(16 进制表示)
3. x=1<<20；其中移位运算由_____编译器_____系统完成。
4. 指令地址 804833d:　e8 _ae ff ff ff ___　　call　　80482f0
5. 现代 Intel I7 CPU，使用 SIMD 向量指令 vaddsd 一次完成__ 8 _个 float 数加法运算。
6. 若 b.o->lib2.a->lib1.a 且 lib1.a->lib2.a->b.o 则最小链接命令行_gcc b.o lib2.a lib1.a b.o_
7. 把进程组 32875 所有的进程终止的 Linux 命令为__kill -32875_____
8. 现代 Intel I7 CPU 中 TLB 采用_____组相联_____类型的高速缓冲器
9. 在 Ubuntu 下无特殊指示，gcc 连接的是_____动__ _____态库
10. IO 接口芯片中的寄存器，在用 IN/OUT 编程时采用_____端口/IO_____地址访问

三、 判断对错（每小题 1 分，共 10 分，正确打√、错误打×）

1. （ √ ）父进程没有回收子进程就终止了， 子进程也是可以被回收的，不再占用资源。
2. （ × ）浮点数进行舍入时采用四舍五入方法
3. （ × ）局部变量是弱符号
4. （ √ ）Linux 下 64 位函数可用 6 个不同的寄存器传递参数
5. （ √ ）不同什么类型的 C 函数，其返回值都在累加器中。
6. （ √ ）在进行 Socket 网络编程时，用 RIO 文件访问函数实现。
7. （ √ ）流水线 CPU 执行 ret 指令会产生分支预测错误，属于控制冒险。
8. （ √ ）Linux 下的程序的每个段的起始地址都是 0
9. （ × ）movl 0x8060020,0x9080010
10. （ √ ）Intel I7 CPU 的三级 Cache 的每一路/行/块都是一样的，占 64 个字节。

四、 简答题（每小题 5 分，共 20 分）

1. C 语言 main 函数调用了 sum(int a[],int n)，函数体中用 int val 存储累加和，int i 作为循环变量。请画出调用 sum 前后的堆栈框架/栈帧，标识堆栈各区及寄存器参数的内容。

答：

32 位系统

	esi	现场寄存器	子程序栈帧
	edi		
	ebx		
	i	局部变量	
	val		
ebp->	ebp	主程序EBP	
	返回地址	返回地址	主程序栈帧
	a	参数	
	n		

64 位系统

寄存器参数：　rdi---------->a

寄存器参数：　esi---------->n

val:　eax

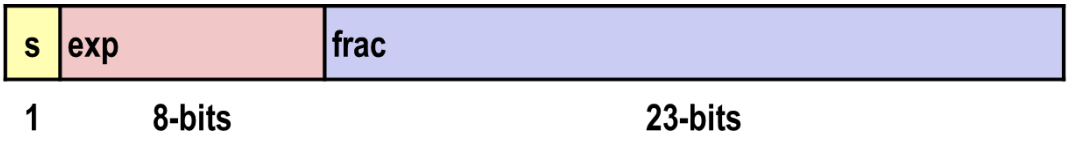
i:　　ecx

i:	ecx->	rbp-8
val:	eax->	rbp-4
		rbp
		返回地址

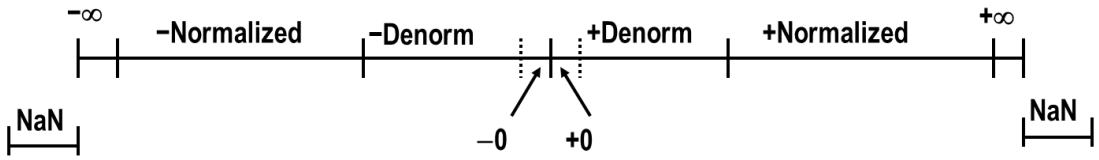
根据编译选项不同，优化结果不同会有变化

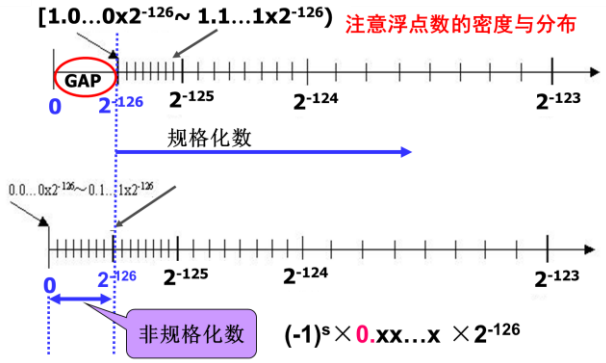
2.　请结合 IEEE754 编码分析 float 数据在数轴上的密度分布

答：浮点数在机器内的表示如下图。



其中 s 为符号位，exp 为解码的移码（+127），frac 为 1.xxxx 的规格化表示的尾数



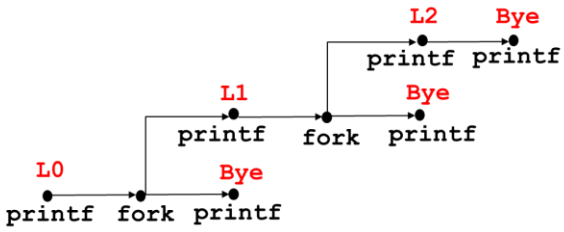


由图可见：原点 O 邻近区域的非规格数(exp=0)其密度为 2E-126 / 2E23=2E-149；而紧邻的规格化数密度依次为 2E-149、2E-148、2E-147……2E102、2E103。而最大规格化数到+∞的密度为 0。数轴负向类似。越靠近数轴密度越大，反之越小。

3.　画出下列程序的进程刻画图，写出一个可能以及一个不可能的输出序列

```
void fork5()
{
    printf("L0\n");
    if (fork() == 0) {
        printf("L1\n");
        if (fork() == 0) {
            printf("L2\n");
        }
    }
    printf("Bye\n");
}
```

答：　输出只要符合要求即可，不要求与答案一致



可能的输出:
L0
Bye
L1
L2
Bye
Bye

不可能的输出:
L0
Bye
L1
Bye
Bye
L2

4. 简述 hello world 程序在 shell 下 fork 及 execve 的工作过程

答：fork：(1)父进程 shell 调用 fork 创建一个新的运行态的子进程 (2)调用 1 次，返回 2 次，父进程返回子进程的 PID，子进程返回 0。（3）新创建的子进程得到与父进程虚拟地址空间相同的但是独立的一份副本（代码、数据段、堆、共享库以及用户栈）;（4）子进程获得与父进程任何打开文件描述符相同的副本；（5）子进程有不同于父进程的 PID（PCB 拷贝了一份但进程 id 不同）。

execve：(1) 调用一次并从不返回（除非文件出错）；（2）加载器函数 loader 在当前进程中载入并运行程序；（3）Loader 删除子进程现有的虚拟内存段，创建一组新的段（栈与堆初始化为 0),并将虚拟地址空间中的页映射到可执行文件的页大小的片 chunk，新的代码与数据段被初始化为可执行文件的内容，然后跳到_start…… 除了一些头部信息实际没读文件，直到缺页中断加载程序后，开始执行代码。

五、 系统分析题（20 分）

1. 请阅读如下程序，回答问题，并给出修改后正确的程序。（5 分）

```
int sum(int a[], unsigned len)
{
  int i,sum = 0;
  for (i = 0; i <= len-1; i++)
   sum += a[i];
  return sum;
}
```

- **当用len=0调用sum时，运行结果是什么？**

答： 运行结果为： Segmentation fault！ 即段故障。

分析：len 为无符号数，len-1 也是无符号数，i<len-1 会转换成 2 个无符号数比较。len=0 时,len-1 为 2E32-1;故循环次数为 2E32。数组 a 的元素个数是有限的，当 i 超出数组范围，元素地址超出其在的数据段大小时，就会发生段超越故障。

修改：在 for 前增加语句 if(len==0) return 0;

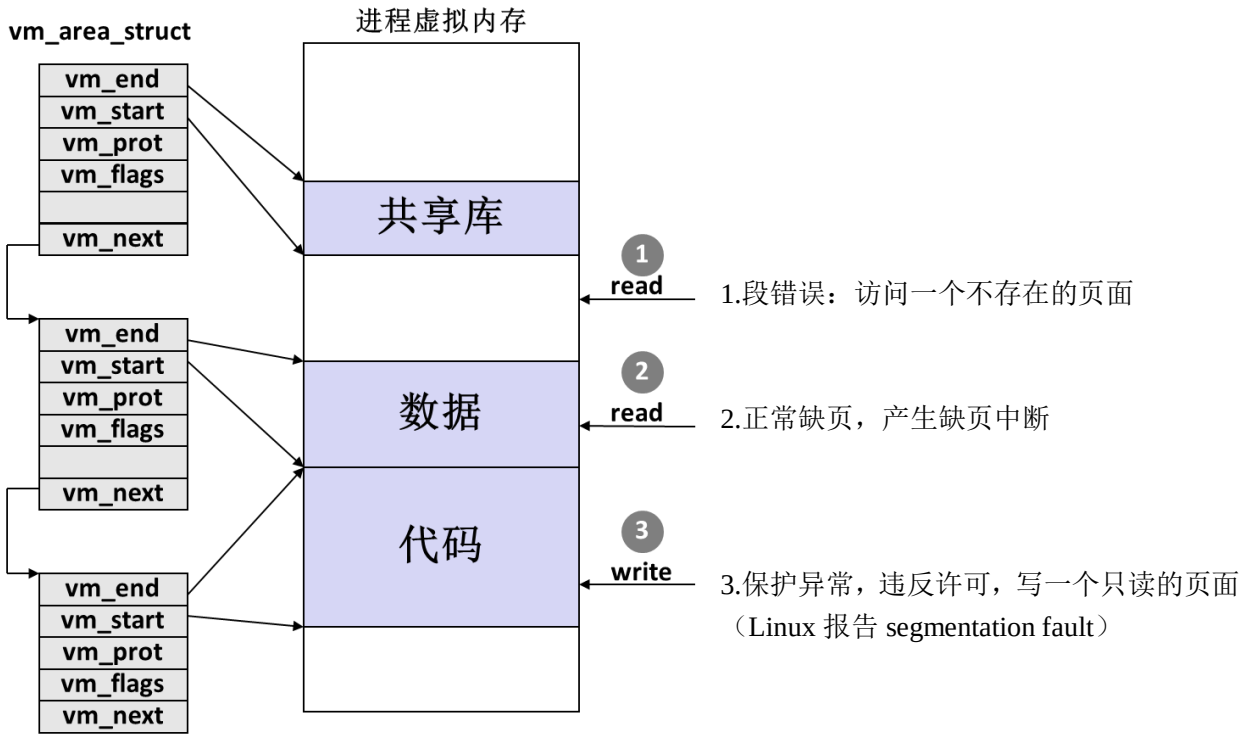
2. 请阅读如下 getval 子程序，说明程序每条语句功能，给出对应 C 语言程序（5）

```
400517:83 c2 02                 add     $0x2,%edx                条件码+2➔数组元素下标
40051a:83 fa 04                 cmp     $0x4,%edx                超出数组下标范围吗
40051d:77 32                    ja      400551 <getval+0x3a>     超出则到 default
40051f: 89 d2                   mov     %edx,%edx                下标扩展为 64 位
400521:ff 24 d5 68 06 40 00     jmpq    *0x400668(,%rdx,8)       跳转到本元素的跳转表内容-地址
400528:83 ee 01                 sub     $0x1,%esi                i--
40052b:48 63 f6                 movslq  %esi,%rsi                扩展为 64 位
40052e:8b 04 b7                 mov     (%rdi,%rsi,4),%eax       返回 a[--i]
400531:c3                       retq
400532:48 63 f6                 movslq  %esi,%rsi
400535:8b 04 b7                 mov     (%rdi,%rsi,4),%eax       返回 a[i--] 或者 a[i]
400538:c3                       retq
400539:48 63 f6                 movslq  %esi,%rsi
40053c:8b 04 b7                 mov     (%rdi,%rsi,4),%eax       返回 a[i]
40053f: c3                      retq
400540:83 c6 01                 add     $0x1,%esi
400543:48 63 f6                 movslq  %esi,%rsi
400546:8b 04 b7                 mov     (%rdi,%rsi,4),%eax       返回 a[++i]
400549:c3                       retq
40054a:48 63 f6                 movslq  %esi,%rsi
40054d:8b 04 b7                 mov     (%rdi,%rsi,4),%eax       返回 a[i++] 或者 a[i]
400550:c3                       retq
400551:b8 00 00 00 00           mov     $0x0,%eax                default：返回 0
400556:c3                       retq
```

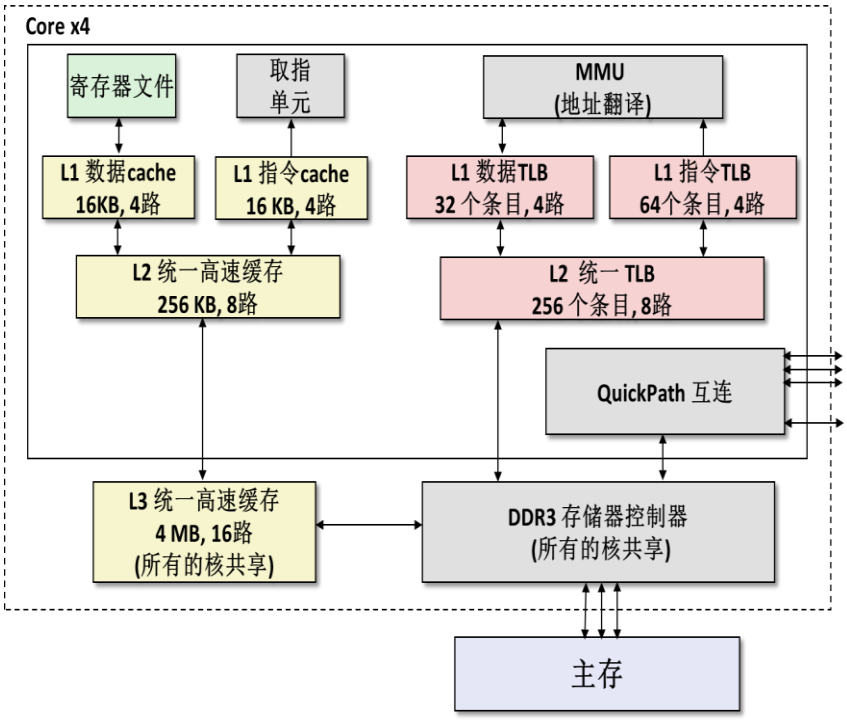
答：

```
int getval(int a[],int i,int v)
{
    int val;

    switch(v)
    {
      case -2:  val=a[--i];     break;
      case -1:  val=a[i--];     break;          此处可用 i
      case 0:   val=a[i];       break;
      case 1:   val=a[++i];     break;
      case 2:   val=a[i++];     break;          此处可用 i
      default:  val=0;     break;
    }

    return val;
}
```

3. 下面是某进程执行过程中的 3 次访存（每一区域都是第一次访问），请分析每次访存的执行结果，说明原因（5 分）。



4. Intel I5 CPU 的虚拟地址 48 位，物理地址 52 位。其内部结构如下图所示，依据此结构，分析并填空（5 分）



每一页面 4KB，分析如下项目：

虚拟地址中的 VPN 占__36__位；

其一级页表为__512__项。

L1 数据 TLB 的组索引位数 TLBI 为__3__位。

L1 数据 Cache 共__64__组。

用物理地址访问 L1 数据 Cache 时， Cache 标记 CT 占__40__位

院系＿＿＿＿＿＿　学号＿＿＿＿＿＿　姓名＿＿＿＿＿＿　授课教师＿＿＿＿＿＿

密 封 线

六、 综合设计题（20 分）

```
void trans(int M, int N, int A[M][N], int B[N][M]) {
    for (int i = 0; i < M; i++)
        for (int j = 0; j < N; j++)
            B[j][i] =A[i][j];
}
```

1. 现代超标量 CPU X86-64，请对如上程序进行优化，给出面向编译器以及 CPU 友好的程序（设 M=N 为 2 的 n 次幂）。

答：根据要求，可灵活选择优化方法，只要是面向编译器以及 CPU 友好的方法都算正确。

2.设 CPU 的 L1 Cache 的参数 s=5，E=1，b=5，若 M=N=32，请编写面向 L1 Cache 友好的程序

答：根据要求，可灵活选择优化方法，只要是面向 Cache 友好的方法都算正确。