主管领导审核签字

哈尔滨工业大学 2021 学年　春 季学期

计算机系统（A）　试　题

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

片纸鉴心 诚信不败

授课教师　姓名　学号　院系　密　封

一、 单项选择题（每小题 1 分，共 20 分）

1. printf 等函数的格式串在 elf 文件的（ C ）节
   A. data　B. bss　C. rodata　D. stack
2. 两个有符号数运算结果是否超出范围，可查看 CPU 的（ B ）
   A. CF　B.OF　C.SF　D.ZF
3. 在 Linux 下，采用 UTF-8 编码，汉字用（ D ）个字节编码。
   A. 1　B. 2　C. 3　D. >=2
4. 浮点数 1 的阶码是（ C ）
   A.0　B.1　C.127　D.128
5. 局部变量数组赋初值 0，通过（ D ）操作完成
   A.预处理或编译　B.请求二进制 0　C.OS 加载时　D.机器指令
6. 缓冲器溢出漏洞防范方法错误的是（ C ）
   A. 金丝雀　B.使用安全函数　C.加大局部变量占用空间　D. 编译加安全选项
7. 流水线 CPU 设计中数据转发时增加旁路路径是为了避免（ A ）
   A.数据冒险　B.加载/使用冒险　C.控制冒险　D.处理器异常
8. 计算机是 64 位是指（ B）
   A.数据总线 64 根　B.CPU 中通用寄存器是 64 位的
   C.安装的操作系统是 64 位的　D.CPU 中所有寄存器都是 64 位的
9. 函数（ D ）可以直接将进程的虚拟地址空间映射到磁盘文件。
   A. fork　B. execve　C.loader　D.mmap
10. 局部变量是（ D ）//static 局部变量是本地/局部符号，其他啥都不是
    A. 局部符号　B. 本地符号　C. 全局符号　D.以上都不对
11. CPU 访问 TLB、Cache 时使用的地址分别是（ B ）
    A. 虚拟地址、虚拟地址　B. 虚拟地址、物理地址
    C. 物理地址、虚拟地址　D. 物理地址、物理地址
12. 固态硬盘访问，修改数据时正确的是（ D ）
    A.修改原来位置的页面数据　B. 修改原来位置的整块数据
    C.写在已擦除的新页上　D. 写在已擦除的新块上
13. Intel X86-64 的现代处理器，采用（ B ）级页表
    A. 2　B.4　C.8　D.可在 OS 中配置
14. 如下数据在内存中地址最高的是（ B ）

A. 命令行参数 B.环境变量 C.main 的栈帧 D.当前子程序栈帧

15.库打桩不会发生在（ C ）
A.编译时 B.静态链接时
C.静态链接程序的运行时
D.动态链接程序的运行时

16.Linux 下，程序运行中按下 Ctrl-Z,会发生（ B ）
A.当前进程终止 B. 当前进程停止并进入后台
C.父进程停止 D. 父进程终止

17.调用 read()函数产生（ B ）
A.故障 B.陷阱 C.异步异常 D.进程切换

18.用 fopen 两次打开同一个文件，会产生两个（ B ）
A.描述符表 B.文件表项 C.v-node 表项 D.stat 结构

19.执行一条指令最不幸时需要访问（ D ）次各类存储器
A. 2 B.4 C.<=8 D.> 8

20.Hello World! 执行程序占 3186 字节，运行时占(D)页内存
A.1 B.2 C.4 D. > 4

二、填空题（ 每空 1 分，共 10 分 ）

21.float 数 0 的机器数有_____2_____个。

22.switch 语句的机器级实现中，采用的跳转表存在 elf 文件的_____.rodata_节。

23.C 语言 64 位系统中函数超过 6 个以上的参数采用_____(堆)栈_____传递。

24.缓冲器溢出漏洞中，是用黑客程序的地址覆盖了_____返回地址_____完成的

25.电脑主板上内存条中的每个二进制位信息采用___电容____来存储。

26.fork 后创建的子进程与父进程不同的信息是_____进程 ID_____。

27.隐式空闲链表的空闲块合并有___4_____种策略。

28.Intel I7 的 CPU 其 TLB 的每行的存储块 Block 是___8___字节。

29.运行一次，可返回多次的函数是_____setjmp_____。

30.Intel I7 CPU 的各级页表的元素个数为_____512_____。

三、判断对错（每小题 1 分，共 10 分，在题前打 √ X 符号）

31.（ X ）无符号 int 整数比有符号 int 整数多。
32.（ √ ）C 语言函数 scanf 是不安全的。
33.（ X ）浮点数计算后的舍入规则是四舍五入。
34.（ X ）流水线处理器的级数越多越好。
35.（ X ）Cache 的行数必须是 2 的 n 次幂。
36.（ X ）Ubuntu 中数据段、代码段的段基址（段描述符中保存的）是一样的。
37.（ X ）C 语言程序中的内存垃圾可以完全回收。
38.（ X ）Intel 64 位系统页表中 PTE 的物理页号 PPN 是 64 位的，占 8 个字节。
39.（ √ ）缺页异常处理子程序完成虚拟内存到物理内存的映射。
40.（ √ ）C 的标准 I/O 函数不能用在信号处理子程序中

四、简答题（每小题 5 分，共 20 分）

41. 针对有符号及无符号整数的加法运算，CPU、编译器、程序员是怎么配合完成不同类型整数的数据表示、数据运算，并如何判断其结果是否超出范围的？

答：C 程序员用 unsigned 或 signed（缺省，可不用）来区分数据类型，常数后加 U 表示无符号数。程序中可以自由进行比较、赋值、运算等。

CPU 并不知道数据类型，只是按位进行加法操作，并按照逻辑规定设置 CF、OF、ZF、SF、PF、AF 等标志位。

编译器会将数据转换成相应的二进制编码（无符号数）或补码（有符号数），如果类型不一致，都转换成无符号数再进行操作。

数据操作之后，编译器根据不同数据类型选择不同的分支转移指令，可按照如上标志位，无符号数用 JA/JB 等、有符号数用 JG/JL 等判断数据大小，并进行跳转。无符号溢出用 JC、有符号溢出用 JO 判断。

42. Unix I/O 函数与 C 标准 I/O 函数能否混合使用？为什么？并说明各自的适用范围。

答：不能。

因为 Unix I/O 函数是不带缓冲的，C 标准 I/O 函数是带缓冲的。如混合使用会导致数据输出顺序与程序中发送顺序不一致的情况，从而出错。

Unix I/O 函数是异步信号安全的函数，可用于信号处理程序中，并适用于一些实时性要求高（高性能）的 I/O 应用场合。

C 标准 I/O 函数带缓冲，能减少对 I/O 设备的访问次数，大大提高 I/O 的效率，如磁盘文件和终端文件等。

43. 以 Intel64 位现代处理器为例，简述加快页表 PTE 访问、大大降低页表占用空间的相关技术。

答：采用 TLB 加快页表 PTE 的访问：采用高速缓冲存储器作为页表的 Cache。TLB 中保存最近常用的虚拟页号对应的页表条目 PTE（含物理页号）。对于虚拟页数较少的进程，它的页表可以完全放在 TLB 中。

采用多级页表大大降低页表占用的空间：由于 VM 空间的页面有大量的连续页面都是未分配的，Intel64 位 CPU 采用 4 级页表后，一级页表的大量条目其内容为 NULL，他们对应的二、三、四级页表项就不用存储。相应的，二、三级页表也是如此，会有值为 NULL 的 PTE，他们的子页表也都不用存储。，这样只有已分配页表条目对应的 4、3、2、1 级页表项才需要存储，大大节省了页表空间。

44. 程序执行 int x=y/c 语句时，当 c=0 时程序执行结果是什么？并请结合异常、信号的概念及处理机制解释原因。

答：异常是指为响应某个事件将控制权转移到操作系统内核中的情况，每种异常都有一个异常号及对应的异常处理子程序（内核态）

信号是一条消息，它通知进程系统中发生了一个某种类型的事件，在进程进入运行态前由 OS 内核检查信号并执行其对应的信号处理子程序（用户态）。

程序执行到这条语句时会产生整数除法出错异常（异常号 0），执行 Divide_Eror( )异常处理子程序，在异常处理子程序中，向当前进程发送一个 SIGFPE 信号（8 号信号），而 SIGFPE（信号处理子程序）的默认行为就是显示“Floating point exception (core dumped)”，终止并转储内存。

五、系统分析题（每小题 5 分，共 20 分）

45. 请分析如下 C 程序，指出各符号对应的运行时内存区域

```
int i=100;                                  .data _____i_____
char *s=“Hello World!\n”;
short a[1<<30];                             .rodata_____s_____
void main(int argc,char *argv[])
{   ……                                      .bss _____a,count_____
static int count=0;
int *p1,*p2;                                .stack____c,argc,argv、p1、p2_____
char c[100]=“1234567890ABCDEF”;
p1=malloc(4096);                            .heap(brk 堆)____*p1、*p2_____
p2=malloc(256*1024);
        ……                                  注（有 1 个就算对）
```

46. 某 C 函数的反汇编结果分别如下：

```
（1） 401160:   83 ee 01                sub    $0x1,%esi
（2） 401163:   b8 00 00 00 00          mov    $0x0,%eax
（3） 401168:   85 f6                   test   %esi,%esi
（4） 40116a:   78 0b                   js     401177 <f+0x17>
（5） 40116c:   48 63 d6                movslq %esi,%rdx
（6） 40116f:   03 04 97                add    (%rdi,%rdx,4),%eax
（7） 401172:   83 ee 01                sub    $0x1,%esi
（8） 401175:   eb f1                   jmp    401168 <f+0x8>
（9） 401177:   c3                      retq
```

请写出 函数 f 的返回值类型是_____int____,参数的类型__int * 或 int[]____,__int__

第 3、4 条指令的功能 _____ 若 esi<0 则程序结束并返回

_____

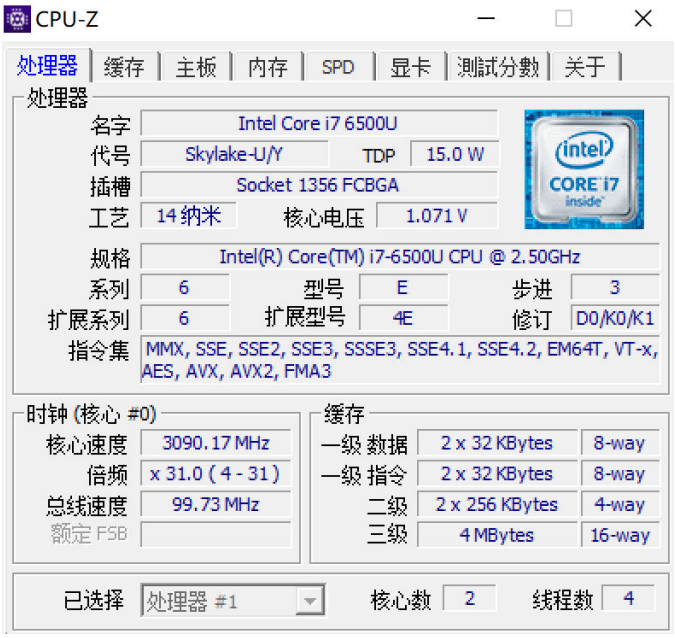
第 5、6 条指令的功能_____参数 1 的第 esi 号元素内容加到累加器 eax 上_____

47. 写出 46 题函数的 C 语言实现

答：

```
int sum(int a[],int n)          1 分
{
   int res=0;
   int i;                       1 分
   for(i=n-1;i>=0;i- -)         2 分
        res+=a[i];              1 分
   return res;
}
```

48. 某电脑使用 CPUZ 查看结果如下图所示，虚拟地址 48 位，物理地址 47 位，每页 4KB，每块 64B，分析并填写如下内容



1. 此 CPU 的 3 级 Cache 都采用了__组相联__类型的高速缓冲器。

2. L3Cache 有__4096/4K__组

3. L2Cache 的 Tag 有__31___位

4. L1Cache 的组索引位数__6____

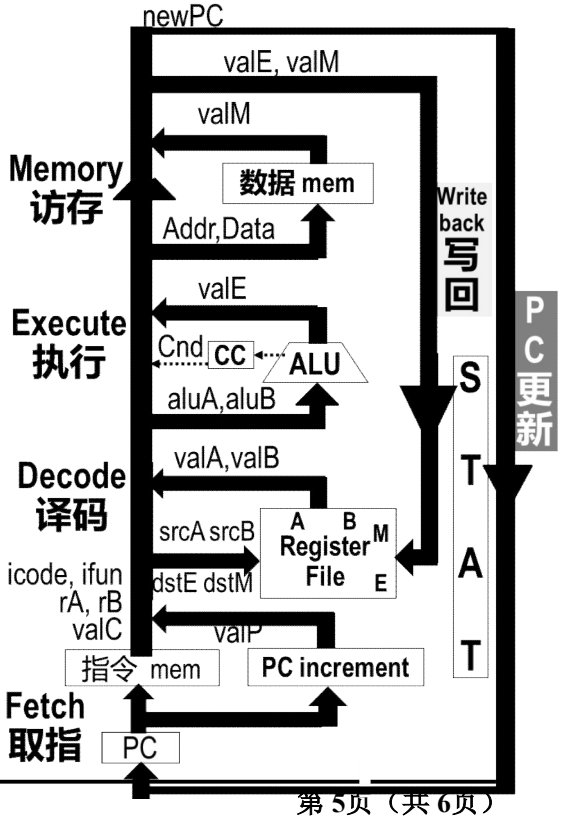
5. L1Cache 的行数为_8(一组的行数)/512(i 或 d cache 的总行数)/1024（L1 的总行数）

六、综合设计题（每小题 10 分，共 20 分）

49. Y86-64 顺序结构的 CPU，请按顺序写出 ret（机器语言 0x90）指令在各阶段的微操作

答：

	ret	
取指	icode:ifun ← M_1[PC]	读指令字节
译码	valA ← R[%rsp] valB ← R[%rsp]	读操作数栈指针 读操作数栈指针
执行	valE ← valB + 8	栈指针增加
访存	valM ← M_8[valA]	读返回地址
写回	R[%rsp] ← valE	更新栈指针
更新PC	PC ← valM	PC指向返回地址



取指：1 分
更新 PC：1 分
其他每个阶段 2 分

50. 一个图像处理程序实现图像的平滑，其图像分辨率为 1920*1080，每一点颜色值为 long 类型，用 long img[1920][1080]存储屏幕上的所有点颜色值。（long 为 64 位）
平滑算法为：任一点的颜色值为其上下左右 4 个点颜色的平均值，即：

val[i][j]=(img[i-1][j]+img[i+1][j]+img[i][j-1]+img[i][j+1])/4。

请面向 48 题的 CPU，利用本课程学过的优化技术，编写程序，并说明你所采用的优化方法。

答：

（1）一般有用的优化，共享共用子表达式、复杂指令简化

```
up =      img[i-1][j]  = img[(i-1)*n + j ];
down =    img[i+1][j]  = img[(i+1)*n + j ];
left =    img[i][j-1]  = img[i*n   + j-1 ];
right =   img[i][j+1]  = img[i*n   + j+1];
valij = (up + down + left + right)/4
```

优化为：

```
long *inj = img+i*n + j;
up =       *( inj  -  n);
down =     *( inj +   n);
left =     *( inj  -  1);
right =    *( inj +   1);
valij = (up + down + left + right) >> 2
```

（2）面向编译器的优化：用局部变量

（3）面向标量 CPU 优化：带分离的累加器的循环展开。通过比较不同展开因子 L 时的最小 CPE，从而确定最优的 L 展开因子。

面向向量 CPU 优化：采用 vaddpd 及 YMMi 寄存器编程

（4）面向 Cache 优化：

空间局部性：重新排列（局部变量、循环变量顺序重排）提高空间局部性

时间局部性：分块，考虑到 Cache 32K。

注意：如上四类优化方法，任选 2 种即可满分。第 3 种、第 4 种又可各自有两个方法任选。有程序实现有瑕疵，若能多种方法集成在一起形成一个程序可加分直至本题目满分。边缘点的外邻点颜色值为 0，如没有考虑，不扣分。