考试中心填写：

___年___月___日
考试用

湖南大学课程考试试卷

课程名称：计算机系统　　；试卷编号：A；考试时间：120 分钟

专业班级：

学号：

姓名：

题号	一	二	三	四	五	六	七	八	九	十	总分
应得分	10	20	20	25	25						100
实得分											
评卷人											评分：

1.简答题（10 分）

小明设计了一款机器，整数和浮点数都占 10 个 bit，其中整数采用补码表示，浮点数采用 IEEE 754 标准。

（1）整数的表示范围是多少？请分别用十进制数和二进制数表示。（2 分）

（2）如果浮点数采用 1 位符号位，5 位阶码位，4 位尾数位，则在正数中，最大的非规格化数与最大的规格化数分别是多少？用二进制数表示。（3 分）

（3）如果要求浮点数能精确表示第一问中所有的整数，能否做到？如果能，请给出所有阶码位和尾数位的位数组合，并描述分析过程，如果不能，请分析原因。（5 分）

2.程序填空题（20 分，每空 4 分）

如下是一个 c 语言程序及其对应的汇编代码（32 位机，小端环境下编译），请参照汇编代码，完成 c 程序的空缺部分。

c 语言程序：

```
#include <stdio.h>
#define X     (1)
#define Y 23
int array1[X][Y];
int array2[X];

int test()
{
   int sum=  (2)  ;
   int i=0;
   do
   {
      if(  (3)  ) continue;
      sum+=  (4)  ;
   }while(i<X && i<Y);
   return  (5)  ;
```

```
}
int main()
{
    return 0;
}
```

汇编代码如下：

```
test:
    pushl  %ebp
    movl   %esp, %ebp
    subl   $16, %esp
    movl   $14, -8(%ebp)
    movl   $0, -4(%ebp)
.L5:
    movl   -4(%ebp), %eax
    movl   array2(,%eax,4), %ecx
    movl   -4(%ebp), %edx
    movl   %edx, %eax
    addl   %eax, %eax
    addl   %edx, %eax
    sall   $5, %eax
    addl   $array1, %eax
    movl   (%eax), %eax
    imull  %ecx, %eax
    cmpl   $6, %eax
    jle .L2
    movl   -4(%ebp), %eax
    movl   array2(,%eax,4), %ecx
    movl   -4(%ebp), %edx
    movl   %edx, %eax
    addl   %eax, %eax
    addl   %edx, %eax
    sall   $5, %eax
    addl   $array1, %eax
    movl   (%eax), %eax
    movl   %ecx, %edx
    subl   %eax, %edx
    movl   %edx, %eax
    cmpl   $7, %eax
    jle .L6
.L2:
    movl   -4(%ebp), %edx
    movl   %edx, %eax
    addl   %eax, %eax
    addl   %edx, %eax
```

```
   sall   $5, %eax
   addl   $array1, %eax
   movl   (%eax), %edx
   movl   -4(%ebp), %eax
   movl   array2(,%eax,4), %eax
   addl   %edx, %eax
   addl   -4(%ebp), %eax
   addl   %eax, -8(%ebp)
   addl   $1, -4(%ebp)
   jmp .L3
.L6:
   nop
.L3:
   cmpl   $18, -4(%ebp)
   jg  .L4
   cmpl   $22, -4(%ebp)
   jle .L5
.L4:
   movl   -8(%ebp), %eax
   subl   $5, %eax
   imull  -8(%ebp), %eax
   leave
   ret
```

3.综合应用题（20 分）

编译一个 C 语言程序（未使用优化选项），再对形成的可执行文件进行反汇编，得到如下汇编代码：

```
0804841d <main>:
 804841d: 55                    push   %ebp
 804841e: 89 e5                 mov    %esp,%ebp
 8048420: 83 e4 f0              and    $0xfffffff0,%esp
 8048423: 83 ec 20              sub    $0x20,%esp
 8048426: c7 44 24 18 0a 00 00    movl   $0xa,0x18(%esp)
 804842d: 00
 804842e: c7 44 24 1c 00 00 00    movl   $0x0,0x1c(%esp)
 8048435: 00
 8048436: 8b 44 24 18           mov    0x18(%esp),%eax
 804843a: 89 04 24              mov    %eax,(%esp)
 804843d: e8 1a 00 00 00        call   804845c <function>
 8048442: 89 44 24 1c           mov    %eax,0x1c(%esp)
 8048446: 8b 44 24 1c           mov    0x1c(%esp),%eax
 804844a: 89 44 24 04           mov    %eax,0x4(%esp)
 804844e: c7 04 24 50 85 04 08    movl   $0x8048550,(%esp)
```

```
 8048455: e8 96 fe ff ff      call   80482f0 <printf@plt>
 804845a: c9                  leave
 804845b: c3                  ret

0804845c <function>:
 804845c: 55                  push   %ebp
 804845d: 89 e5               mov    %esp,%ebp
 804845f: 53                  push   %ebx
 8048460: 83 ec 24            sub    $0x24,%esp
 8048463: c7 45 f4 00 00 00 00    movl   $0x0,-0xc(%ebp)
 804846a: 83 7d 08 00         cmpl   $0x0,0x8(%ebp)
 804846e: 75 09               jne    8048479 <function+0x1d>
 8048470: c7 45 f4 00 00 00 00    movl   $0x0,-0xc(%ebp)
 8048477: eb 38               jmp    80484b1 <function+0x55>
 8048479: 83 7d 08 01         cmpl   $0x1,0x8(%ebp)
 804847d: 74 06               je     8048485 <function+0x29>
 804847f: 83 7d 08 02         cmpl   $0x2,0x8(%ebp)
 8048483: 75 09               jne    804848e <function+0x32>
 8048485: c7 45 f4 01 00 00 00    movl   $0x1,-0xc(%ebp)
 804848c: eb 23               jmp    80484b1 <function+0x55>
 804848e: 8b 45 08            mov    0x8(%ebp),%eax
 8048491: 83 e8 01            sub    $0x1,%eax
 8048494: 89 04 24            mov    %eax,(%esp)
 8048497: e8 c0 ff ff ff      call   804845c <function>
 804849c: 89 c3               mov    %eax,%ebx
 804849e: 8b 45 08            mov    0x8(%ebp),%eax
 80484a1: 83 e8 02            sub    $0x2,%eax
 80484a4: 89 04 24            mov    %eax,(%esp)
 80484a7: e8 b0 ff ff ff      call   804845c <function>
 80484ac: 01 d8               add    %ebx,%eax
 80484ae: 89 45 f4            mov    %eax,-0xc(%ebp)
 80484b1: 8b 45 f4            mov    -0xc(%ebp),%eax
 80484b4: 83 c4 24            add    $0x24,%esp
 80484b7: 5b                  pop    %ebx
 80484b8: 5d                  pop    %ebp
 80484b9: c3                  ret
```

该 C 程序代码如下（部分缺失）：

```
#include "stdio.h"
void main()
{
   int v=10,result=0;
   result=function(v);
```

```
    printf("%d\n",result);
}
int function(int f)
{
    int e=0;
    if(f==_①_)   e=0;
    else if(f==_②_||f==_③_)  e=1;
    else   e=function(_④_)+function(_⑤_);
    return e;
}
```

（1）补充上述 C 代码中缺失的部分（每空 2 分，共 10 分）

（2）已知在主函数调用 function 前，%ESP = 0XBFFFF0D0。请将下列信息填写到栈帧图中的正确位置（每一格 4 字节，且此时栈帧图为 function 第一次调用时的状态，所有变量写初始值即可，**答案请写在答题纸上，**每空 2 分，共 10 分）：

主函数返回地址“地址：Rtn Addr”；

主函数局部变量 v

（请填：地址：主函数局部变量 v = ***）；

主函数局部变量 result

（请填：地址：result = ***）；

主函数传递参数 v

（请填：地址：主函数传递参数 v = ***）；

function 的局部变量 e

（请填：地址：e = ***）

注：先声明的局部变量存放小地址。

地址	内容
0XBFFFF0F8	OLD EBP
……	……
0EC	
0E8	
0E4	
……	……
0D0	
0CC	
(EBP) → 0C8	OLD EBP
0C4	
0C0	
0BC	
0B8	
……	……
0A4	
(ESP) → 0A0	

4.综合应用题（25 分）

现有一个 project 里包括两个.c 文件，如下图所示。

```
// 这是 main.c 文件
   #include <stdio.h>
    int d=50;
    int x=100;
    void p1( );
    int main()    {
        p1();
        printf (“d=%d\n”, d);
        printf (“x=%d\n”, x);
     return 0;       }
```

```
//这是 p.c 文件
   double d;
   void p1()    {
       d=1.0;
 }
```

请回答如下问题：

（1）从链接需要出发，请指出这个 project 里有什么符号？分别属于什么类型？

（2）如果要编译执行这个 project，应该在 Linux shell（即你的终端窗口）中输入：$ gcc_____，请将该命令行填充完整，可以获得一个可执行程序 p1（如果你觉得一行 gcc 命令不够，也可以使用多行 gcc 命令，使得如上图所示的两个.c 文件最终能形成可执行文件）；

（3）运行 p1 后，屏幕输出是什么？结合链接器解析多重定义的全局符号的规则，分析为什么会有这样的输出？

（4）请根据进程与加载运行程序的相关知识，描述运行该可执行程序 p1 需要进行哪些系统调用，分别有什么作用？

（5）当 p1 开始运行时，其用户栈的典型组织结构是什么样的？

5.综合应用题（25 分）

假定一个具有以下属性的计算机系统：

- 系统有 1MB 的虚拟内存
- 系统有 256KB 的物理内存
- 页面大小为 4KB
- TLB 是 2 路组相联，总共有 8 个条目。

下面给出了 TLB 的内容和页表前 32 个条目。**所有数字均为十六进制**。

TLB			
Index	Tag	PPN	Valid
0	05	13	1
	3F	15	1
1	10	0F	1
	0F	1E	0
2	1F	01	1
	11	1F	0
3	03	2B	1
	1D	23	0

Page Table					
VPN	PPN	Valid	VPN	PPN	Valid
00	17	1	10	26	0
01	28	1	11	17	0
02	14	1	12	0E	1
03	0B	0	13	10	1
04	26	0	14	13	1
05	13	0	15	1B	1
06	0F	1	16	31	1
07	10	1	17	12	0
08	1C	0	18	23	1
09	25	1	19	04	0
0A	31	0	1A	0C	1
0B	16	1	1B	2B	0
0C	01	0	1C	1E	0
0D	15	0	1D	3E	1
0E	0C	0	1E	27	1
0F	2B	1	1F	15	1

（1）热身问题

(a) 需要多少位来表示虚拟地址空间？

(b) 需要多少位来表示物理地址空间？

(c) 需要多少位来表示页表偏移量？

（2）.虚拟地址转换一

请完成以下虚拟地址到物理地址的转换。如果缺页，物理地址栏输入“-”。

虚拟地址：**0x1F557**

使用下面的布局作为虚拟地址位的暂存空间。请填写与 VPN、TLB 索引 (TLBI)和 TLB 标签 (TLBT)对应的位。

19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

参数	值	参数	值
VPN	0x	TLB 命中？（是/否）	
TLBI	0x	缺页？（是/否）	
TLBT	0x	物理地址	0x

（3）.虚拟地址转换二

请完成以下虚拟地址到物理地址的转换。如果缺页，物理地址栏输入“-”。

虚拟地址：**0x14557**

使用下面的布局作为虚拟地址位的暂存空间。请填写与 VPN、TLB 索引 (TLBI)和 TLB 标签 (TLBT)对应的位。

19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

参数	值	参数	值
VPN	0x	TLB 命中？（是/否）	
TLBI	0x	缺页？（是/否）	
TLBT	0x	物理地址	0x