在 x86-64、LINUX 操作系统下，考虑如下的 C 定义:

```
typedef union{
    char c[7];
    short h;
}union_e;
typedef struct f{
    char d[3];
    union e u;
    int i;
}struct_e;
struct_e s;
```

回答如下问题:

(1) s.u.c 的首地址相对 s 的首地址的偏移量是_____字节。

(2) sizeof(union_e)=_____字节。

(3) s.i 的首地址相对于 s 的首地址的偏移量是_____字节。

(4) sizeof(struct_e)=_____字节。

(5) 若只将 i 的类型改成 short，那么 sizeof(struct_e)=_____字节。

(6) 若只将 h 的类型改成 int，那么 sizeof(union_e) =_____字节。

(7) 若将 i 的类型改成 short、将 h 的类型改成 int，那么 sizeof(union_e)=_____字节，sizeof(struct_e) =_____字节。

(8) 若只将 short h 的定义删除，那么(1)~(4)问的答案分别是_____，_____，_____，_____。

请分析下面的 C 语言程序和对应的 x86-64 汇编代码。

1.其中，有一部分缺失的代码（用标号标出），请在标号对应的横线上填写缺失的内容。注:汇编与机器码中的数字用 16 进制数填写。(1-11 每空 1 分，共 11 分)

C 语言代码如下：

```
typedef struct _parameters {
    int n;
    int product;
} parameters;
    int bar(parameters *params, int x) {
    params->product *= x;
}
void foo (parameters *params) {
    if (params->n <= 1)
    ___（1）___                                          （1）_____
    bar(params, ___（2）___ );                           （2）_____
    params->n--;
    foo(params);
}
```

x86-64 汇编代码如下（为简单起见，函数内指令地址只给出后四位，需要时可补全)：(

```
0x0000555555555189 <bar>:
5189: f3 0f 1e fa endbr64
518d: 55 push %rbp
518e: 48 89 e5 mov %rsp,%rbp
5191: 48 89 7d f8 mov _（3）_,-0x8(%rbp)                 （3）_____
5195: 89 75 f4 mov %esi,-0xc(%rbp)
5198: 48 8b 45 f8 mov -0x8(%rbp),%rax
519c: 8b 40 04 mov 0x4(%rax),%eax
519f: 0f af 45 f4 imul _（4）_(%rbp),%eax                （4）_____
51a3: 89 c2 mov %eax,%edx
51a5: 48 8b 45 f8 mov -0x8(%rbp),%rax
51a9: 89 50 04 mov %edx,0x4(%rax)
51ac: 90 nop
51ad: 5d pop _（5）_                                     （5）_____
51ae: c3 retq
00005555555551af <foo>:
51af: f3 0f 1e fa endbr64
51b3: 55 push %rbp
51b4: 48 89 e5 mov %rsp,%rbp
51b7: 48 83 ec 10 _（6）_ $0x10,%rsp                     （6）_____
51bb: 48 89 7d f8 mov %rdi,-0x8(%rbp)
51bf: 48 8b 45 f8 mov -0x8(%rbp),%rax
51c3: 8b 00 mov (%rax),%eax
```

```
51c5: 83 f8 01 cmp $0x1,%eax
51c8: 7e 31 _（7）_ 51fb <foo+0x4c>                    （7）_____
51ca: 48 8b 45 f8 mov -0x8(%rbp),%rax
51ce: 8b 10 mov (%rax),%edx
51d0: 48 8b 45 f8 mov -0x8(%rbp),%rax
51d4: 89 d6 mov %edx,%esi
51d6: 48 89 c7 mov %rax,%rdi
51d9: e8 ab ff ff ff callq 0x0000555555555189 <bar>
51de: 48 8b 45 f8 mov -0x8(%rbp),%rax
51e2: 8b 00 mov (%rax),%eax
51e4: 8d 50 ff lea -0x1(_（8）_),%edx                （8）_____
51e7: 48 8b 45 f8 mov -0x8(%rbp),%rax
51eb: 89 10 mov _（9）_,(%rax)                       （9）_____
51ed: 48 8b 45 f8 mov _（10）_ ,%rax                 （10）_____
51f1: 48 89 c7 mov %rax,%rdi
51f4: e8 b6 ff ff ff callq _（11）_                  （11）_____
51f9: eb 01 jmp 51fc <foo+0x4d>
51fb: 90 nop
51fc: c9 leaveq
51fd: c3 retq
```

10

2.在程序执行到 0x000055555555518e 时（该指令还未执行），此时的栈帧如下，请填写空格中对应的值。（每空 2 分，共 6 分）

地址	值	
0x7fffffffe308	0xffffe340	
0x7fffffffe304	0x00000000	
0x7fffffffe300	0x00000000	
0x7fffffffe2fc	0x00005555	
0x7fffffffe2f8		（12）_____
0x7fffffffe2f4	0x00007fff	
0x7fffffffe2f0	0xffffe310	
0x7fffffffe2ec	0x00007fff	
0x7fffffffe2e8	0xffffe340	
0x7fffffffe2e4	0x00000004	
0x7fffffffe2e0	0xffffe350	
0x7fffffffe2dc	0x00005555	
0x7fffffffe2d8		（13）_____
0x7fffffffe2d4	0x00007fff	
0x7fffffffe2d0		（14）_____

3.当 params={n,1}时，foo(&params)函数的功能是什么？（3 分）

阅读下面的汇编代码：

```
<f>:
  4004c4:       push   %rbp
  4004c5:       mov    %rsp,%rbp
  4004c8:       sub    $0x10,%rsp
  4004cc:       mov    %edi,-0x4(%rbp)
  4004cf:       cmpl   $0x1,-0x4(%rbp)
  4004d3:       ja     4004dc <f+0x18>
  4004d5:       mov    $0x1,%eax
  4004da:       jmp    40052d <f+0x69>
  4004dc:       mov    -0x4(%rbp),%eax
  4004df:       and    $0x1,%eax
  4004e2:       test   %eax,%eax
  4004e4:       jne    4004f5 <f+0x31>
  4004e6:       mov    0x200440(%rip),%eax        # 60092c <x.1604>
  4004ec:       add    $0x1,%eax
  4004ef:       mov    %eax,0x200437(%rip)        # 60092c <x.1604>
  4004f5:       mov    -0x4(%rbp),%eax
  4004f8:       and    $0x1,%eax
  4004fb:       test   %al,%al
  4004fd:       je     40050e <f+0x4a>
  4004ff:       mov    0x20042b(%rip),%eax        # 600930 <y.1605>
  400505:       add    $0x1,%eax
  400508:       mov    %eax,0x200422(%rip)        # 600930 <y.1605>
  40050e:       mov    -0x4(%rbp),%eax
  400511:       sub    $0x1,%eax
  400514:       mov    %eax,%edi
  400516:       callq  4004c4 <f>
  40051b:       mov    0x20040f(%rip),%edx        # 600930 <y.1605>
  400521:       lea    (%rax,%rdx,1),%edx
  400524:       mov    0x200402(%rip),%eax        # 60092c <x.1604>
  40052a:       lea    (%rdx,%rax,1),%eax
  40052d:       leaveq
  40052e:       retq
```

1）程序

```
main()
{
    unsigned int n;
    for (n=1; n< 4; n++) {
        printf("f(%d) = %x\n", n, f(n));
    }
```

```
    }
```

的运行结果为：f(1)=1，f(2)=4e，f(3)=9f，请填写 f 函数所需要的内容（每空 1 分）：

```
#define N   (1)
#define M   (2)

struct P1 {char c[N]; char *d[N]; char e[N]; } P1;
struct P2 {int i[M]; char j[M]; short k[M]; } P2;

unsigned int f(unsigned int n)
{
    (3)                 unsigned int x = sizeof(P1);
    (4)                 unsigned int y = sizeof(P2);

   if (  (5)                    )
       return 1;

   if (  (6)                    )
       x++;

   if (  (7)                    )
       y++;

   return  (8)                  ;
}
```

2、程序

```
    main()
    {
        printf("%x, %x\n", f(2), f(2));
    }
```

的运行结果为：（2 分）