1.

(1)

函数 g

old ebp
%gs(14)
a[9]
a[8]
a[7]
a[6]
a[5]
a[4]
a[3]
a[2]
a[1]
a[0]

函数 f

old ebp
%gs(14)
b[1]
b[0]

(2)

b[0] = 6　　b[1] = -48

b[0]的位置是原来 a[8]的位置，为 6.

2.

(1)　&A[i][j][k] = addr(A) + i*S*T +j*T + k

(2)

	%eax	%ecx	%edx
3	7	6	10
4	7	52	10
5	7	52	7
6	14	52	7
7	14	52	14
8	112	52	14
9	98	52	14
10	98	52	9464
11	98	52	9562
12	6	52	9562
13	6	52	9568
14	161220076	52	9568
15	161220076	52	9568
16	378560	52	9568

(3)

```
1  push    %ebp
2  mov     %esp,%ebp
```

```
3   mov     0xc(%ebp),%eax          // %eax -> j
4   mov     0x8(%ebp),%ecx          // %ecx -> i
5   mov     %eax,%edx               // %edx -> j
6   lea     (%edx,%edx,1),%eax      // %eax -> 2j
7   mov     %eax,%edx               // %edx -> 2j
8   lea     0x0(,%edx,8),%eax       // %eax -> 16j
9   sub     %edx,%eax               // %eax ->14j
10  imul    $0xb6,%ecx,%edx         // %edx -> 182i
11  add     %eax,%edx               // %edx -> 14j + 182i
12  mov     0x10(%ebp),%eax         // %eax -> k
13  add     %eax,%edx               // %edx -> 182i + 14j + k
14  mov     0x14(%ebp),%eax         // %eax ->  dest
15  mov     %eax,0x804a060(,%edx,4)
16  mov     $0x5c6c0,%eax
17  pop     %ebp
18  ret
```

3~14 行在行末标注了寄存器中的值的变化，15 行可以得出 A[i][j][k]相对于数组 A 的起始地址的偏置，由此可以得到一个方程：

$$\mathbf{4} * (\mathbf{182i} + \mathbf{14j} + \mathbf{k}) = \mathbf{sizeof}(\mathbf{int}) * (\mathbf{iST} + \mathbf{jT} + \mathbf{k})$$

其中 sizeof(int) = 4。解得 S = 13，T = 14。

又由第 16 行得到 sizeof (A) = $0x5c6c0_{16}$ = 378560_{10}。R = sizeof (A) / (S * T) = 520。

故 R = 520，S = 13, T=14。

4.

(1) 确定下列字节的偏移量。

e1.p	0
e1.x	4
y	0
y[0]	**0**
y[1]	**4**
y[2]	**8**
next	**12**

(2) 数组在声明时申请的是连续的存储空间，所以数组的元素占用的是一块连续的空间，然而链表一个节点包含当前节点的信息和下一个节点的地址，不需要一块连续的存储空间。