2.3

```
sub  $t0,  $s3,  $s4              #   t0 = i - j
sll $t0, $t0, 2                   #     t0 = (i - j) * 4
add $t0, $s6, $t0                 #     t0 = &A[i-j]
lw  $t1, 0($t0)                   #      t1 = A[i-j]
sw $t1, 32($s7)                   #      B[8] = t1 = A[i-j]
```

2.4

```
sll $t0, $s2, $2                  #  t0 = f * 4
add $t0, $s6, $t0                 #  t0 = &A[f]
sll $t1, $t2, $s1, 2              #  t1 = g * 4
add $t1, $s7, $t1                 #   t1 = &B[g]
lw $s0, 0($t0)                    #  s0 = A[f]
addi $t2, $t0, 4                  #  t2 = &A[f+1]
lw $t0, 0($t2)                    #  t0 = A[f+1]
add $t0, $t0, $s0                 #  t0 = A[f+1] + A[f]
sw $t0, 0($t1)                    # B[g] = A[f+1] + A[f]
```

2.6.1

```
tmp = Array[0]              // 2 4 3 6 1  tmp = 2
Array[0] = Array[4]         //  1 4 3 6 1  tmp = 2
Array[4] = Array[1]         //  1 4 3 6 4  tmp = 2
Array[1] = tmp              //  1 2 3 6 4 tmp = 2
tmp = Array[3]              //  1 2 3 6 4 tmp = 6
Array[3] = Array[4]         //  1 2 3 4 4 tmp = 6
Array[4] = tmp              //  1 2 3 4 6 tmp = 6
```

2.7

假设用一个int存储这个数(4字节)

大端:

地址	数据
3	12
2	ef
1	cd
0	ab

小端：

地址	数据
0	12
1	ef
2	cd
3	ab

2.10

addi $t0, $s6, 4 # t0 = &A[1]

add $t1, $s6, $s0 # t1 = &A[0]

sw $t1, 0($t0) # A[1] = &A[0]

add $s0, $t1, $t2 # s0 = &A[0] + &A[0]

表示 **f = 2 * (&A)**

13.1 由于此寄存器最多只能存储范围为-2^{31}——$2^{31}-1$, 则取值x满足：
$128+x>2^{31}-1\ or\ 128+x<-2^{31}$

解得 $x<-2^{31}-128\ or\ x>2^{31}-129$

14.

r - format 表示： add $s0, $s0, $s0

16.

r - format 表示：sub $v1, $v1, $v0

二进制表示： 00000 00011 00010 00011 00000 100010

17.

I -format 表示：lw $v0, 4($at)

二进制表示为：010111 00001 00010 0000000000000100

18.1 opcode变大4倍，rs,rt,rd变大4倍， 分别变成8， 7， 7， 7

18.2 opcode变大4倍，rs, rt变大4倍， 分别变成8， 7， 7

18.3 更多的寄存器会增加rs,rt,rd的长度，使得代码增加

但也会使寄存器更少的溢出，使得代码减少

更多的指令会使得操作码变多，使得代码增加

但也会产生更多有效合适的指令，使代码减少