Assembly Code	Corresponding C statement

`slli t0, s1, 2`
`add t1, s0, t0`
`lw t2, 4(t1)`
`sw t2, 0(t1)`
	`V[i] = V[i+1];`

`slli t0, s1, 2`
`add t1, s0, t0`
`lw t2, 32(t1)`
`sw t2, 12(t1)`
	`V[i+3] = V[i+8];`

`slli t0, s1, 2`
`add t1, s0, t0`
`lw t2, 0(t1)`
`slli t3, t2, 2`
`add t4, s0, t3`
`lw t5, 0(t4)`
`sw t5, -8(t1)`
	`V[i-2] = V[V[i]];`

Question 1 (14 points):

In the C programming language, an array is represented by the address of the memory location that contains the first element of the array. Assume that register `s0` contains the address of the first element of an array `V` of 32-bit integers and that the register `s1` contains an unsigned integer value `i` that is used as the index for an element of `V`.

a. (**10 points**) On top of the page you are given three independent segments of assembly code. Each of these code segments was generated for an statement in the C program described above. Using array notation, write the C-language statement that originated this assembly code.

For the third assembly segment:

```
slli t0, s1, 2   # t0 <- 4*i
add  t1, s0, t0  # t1 <- &V[i]
lw   t2, 0(t1)   # t2 <- V[i]
slli t3, t2, 2   # t3 <- 4*V[i]
add  t4, s0, t3  # t4 <- &(V[V[i]])
lw   t5, 0(t4)   # t5 <- V[V[i]]
sw   t5, -8(t1)  # V[i-2] = V[V[i]]
```

b. (**4 points**) Using a minimum number of instructions, write the assembly code that implements the following C language statement:

```
V[V[i-2]] = V[i-1];
```

```
slli t0, s1, 2    # t0 <- 4*i
add  t1, s0, t0   # t1 <- &V[i]
lw   t2, -8(t1)   # t2 <- V[i-2]
lw   t3, -4(t1)   # t3 <- V[i-1]
slli t4, t2, 2    # t4 <- 4*(V[i-2])
add  t5, t4, s0   # t5 <- &(V[V[i-2]])
sw   t3, 0(t5)    # V[V[i-2]] = V[i-1]
```