2.4)

f, g, h, i, j A B

$s0, $s1, $s2, $s3, $s4 $s6, $s7

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

C code

B[g] = A[f+1] + A[f]

2.5) To calculate A[f+1] we don't need to add 4 to $t0 get address instead it can be done as follows

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

The addi and lw lines can be replaced by [lw $t0, 4($t0)]

2.6) i)

```
int n = sizeof(Array)/sizeof(Array[0]);
for( int c=0; c<n-1; <c++) {
    for (int d=0; d<n-c-1; d++) {
        if (Array[d] > Array[d+1]) {
            int swap = Array[d];
            Array[d] = Array[d+1];
            Array[d+1] = swap;
        }
    }
}
```

a) ii)

```
main:
        la  $t0, 24             #base address of  Array into $t1
        add $t0, $t0, 20        # 4 bytes per int * 10 ints = 20 bytes

Loop1:
        add $t1, $0, $0         # $t1 holds a flag to determine when the list is sorted
        la  $a0, 24             # Set $a0 to the base address of the Array

Loop2:
        lw  $t2, 0($a0)         # sets $t0 to the current element in array
        lw  $t3, 4($a0)         # sets $t1 to the next element in array
        slt $t5, $t2, $t3       # $t5 = 1 if $t0 < $t1
        beq $t5, $0, CONTINUE   # if $t5 = 1, then swap them
        add $t1, $0, 1          # if we need to swap, we need to check the list again
        sw  $t2, 4($a0)         # store the greater number in the higher position in array (swap)
        sw  $t3, 0($a0)         # store the lesser number in the lower position in array (swap)

CONTINUE:
        addi $a0, $a0, 4        # advance the array to start at the next location from last time
        bne  $a0, $t0, Loop2    # If $a0 != the end of Array, jump back to Loop2
        bne  $t1, $0, Loop1     # $t1 = 1, another pass is needed, jump back to Loop1
```