Question 1:

2. Because one instruction requires one ns for processing (assumed in the task) and the program for Bubble Sort in descending order for array[8] = {-3, 5, 11, 30, -7, 55, 19, 23} insists 5 functions:

- main: 1 time with 2 instructions
- bubble_sort: 8 times with 7 instructions
- loop: 35 times with 8 instructions
- swap: 20 times with 8 instructions
- end: 1 time with 2 instructions

Therefore, the expression to count instructions is 1*2 + 8*7 + 35*8 + 20*8 + 1*2 = 500 instructions (maybe approximately), so:

$$IPS = \frac{Clock\ Rate}{CPI} = 1\ (ns)$$

$$CPUTime = \frac{InstCount * CPI}{Clock\ Rate} = \frac{InstCount}{IPS} = \frac{500}{1} = 0.5\ (ms)$$

Question 2:

2. Because one instruction requires one ns (same Question 1) for processing (assumed in the task) and the program for Quick Sort in ascending order for array[12] = { 5, 30, -3, 11, -7, 55, 19, 23, -98, -78, 19, 27} insists 10 functions:

- main: 1 time with 7 instructions
- end: 1 time with 2 instructions
- quickSort: 23 times with 6 instructions
- execution: 11 times with 16 instructions
- return: 23 times with 3 instructions
- partition: 11 times with 8 instructions
- loop: 52 times with 11 instructions
- partition_swap: 29 times with 2 instructions
- swap: 29 times with 13 instructions
- swap_pivot: 11 times with 14 instructions

In execution function (where to call two recursions), actually there are only four instructions be called in every time (la $s0, elems, addi $s2, $t2, -1 to reload address of $s0 and restore value in high if first recursion and addi $s1, $t2, 1 if second recursion, and the last one is take return value $v1 to $t2). Furthermore, four instructions to load the value before recursive. Therefore, the expression to count instructions is 1*7 + 1*2 + 23*6 +

(11*4 + 4) + 23*3 + 11*8 + 52*11 + 29*2 + 29*13 + 11*14 = 1513 instructions (maybe approximately), so:

$$CPUTime = \frac{InstCount}{IPS} = \frac{1513}{1} = 1.513\ (ms)$$