Lab session (22nd Dec)

Easy:

1. For $i \leftarrow 0$, to 19, do the following, while incrementing i by 1 each time:

   $arr[i] \leftarrow 3 * i * i * i + 2 * i * i + 42$.

2. Trivial. Just use the scanf and IF condition syntax correctly.

3. Pseudocode:

   - Let A be an array of positive integers of length n.
   - Let *count_even* and *count_odd* be integer variables.
   - Initialize count_even and count_odd to 0.
   - For $i \leftarrow 0$ to $n - 1$, while incrementing i at each step, do the following:
   - if 2 divides $A[i]$ with remainder 0, then increment count_even.
   - Else increment count_odd.
   - Print count_even and count_odd.

4. Pseudocode:

   - Let str be a char array with value "Hello World\n".
   - Let len be an integer that denotes the length of str. Initialize len to 12.
   - For $i \leftarrow (len - 1)$ to 0, do the following while decrementing i at each step:
     - print $str[i]$

Part 2:

1. Pseudocode:

   - Let n be the integer that was taken as input.
   - For loop with init $i \leftarrow 0$, as long as $(i * i) < n$, do the following while incrementing i at each step:
     - print the value of $i * i$.

2. Pseudocode:

   - Let n be the integer that was taken as input.
   - For $i \leftarrow 4$, as long as $i < n$, do the following while incrementing i at each step:

     We check if i is composite as follows:

     - For $j \leftarrow 2$, as long as (j does not divide i) **and** ($j < i/2$), iterate while incrementing j.
     - If j divides i, then print i.

Remark 2: Loop starts at 4 because 4 is the smallest composite number. The loop that checks if i is composite has an empty body! The if condition that follows is *outside* the loop.

3. Let n_1 and n_2 be the input numbers. Pseudocode is exactly like the solution to 2.2 from Lab 2 exercises. Replace the following line:

   > Print the value of `next` followed by space

   with:

If ($next > n_1$ **and** $next < n_2$) **or** ($next > n_2$ **and** $next < n_1$) then print the value of $next$.

Remark: Alternatively, we could first check if $n_1 < n_2$. If this is not true, then we swap the values of n_1 and n_2 so that $n_1 < n_2$ after swap. In mathematics, we typically skip all these trivialities by saying: "Without loss of generality, assume that $n_1 < n_2$".

4. Trivial. Use the correct syntax, and index bounds when handling the array.