Part 1: Theoretical Questions

1.
    a. Imperative paradigm - a series of commands, executed line by line. Uses statements that change a program's state, focuses on describing how a program operates.
    b. Procedural paradigm – a collection of procedures, contains functions that each of them contain a series of commands (also can return a value).
    c. Functional paradigm - programs are constructed by applying and composing functions, that can be bound to names, passed as arguments, and returned from other functions like other data types. All functions are deterministic mathematical or pure functions. Additionally, there are no side effects, this paradigm does not support variable assignment or state mutation.

    In contrast to imperative paradigm, in procedural paradigm there are local variables, sequence, selection, iteration, and modularization, so the code is more readable as we are able to avoid code repetitions, adapt easily to new values of the parameters, and the nature of the task is reflected in the structure of the code.

    In contrast to procedural paradigm, in functional paradigm we can apply concurrency so we don't have to implement step by step operations and get performance optimizations.
    Also we can create easily functional abstractions.

2.
    a. $<T> (x: T[\,], y: (a: T) => boolean) => boolean$
    b. $(x: number[\,]) => number$
    c. $<T> (x: boolean, y: T[\,]) => T$

3. Instead of doing complex operations inline, we can extract it out into a named function, and now we don\`t have to think about how it\`s calculated and we get a clear code.