Functional Programming - Exercise 1 - FP on familiar terrain (25 Points)

The idea of this exercise is to reflect on the topics of the first lectures. The examples are toy examples but motivate to think about functional programming in general and functional style in mainstream programming languages. Also please use a language of your choice - it is interesting to see different approaches.

- **Exercise 1: Thinking imperatively.** (8 Points), Define a function which takes the list/array and compute the sum of all odd numbers and return it as an integer. Use imperative constructs like loops and variable assignments.
- **Exercise 2: Thinking functionally.** (8 Points), Implement the same function as in exercise 1 but embrace a functional style (no assignments etc).
- **Exercise 3: Abstraction and changed requirements.** (3 Points), The requirements have changed. Rewrite both the imperative as well as the functional solution and compute the average of all odd numbers instead of the sum. The change seems minimal - how much code did you need to touch?
- **Exercise 4: Reflections on the topic** (6 Points). Answer those questions (choose your favorit format, it can also be submitted as part of the code as a large comment):
    - Given Exercise 1 and Exercise 2, in comparison, how does the code compare in terms of succinctness, potential code re-use and testability (3 Points).
    - Analyze Exercise 1 and 2 in terms of runtime performance. Are there performance differences, and if so, why? Either use a simple benchmark or argue in terms of algorithmic complexity, justify your method briefly. (3 Points).

- **Sumbission.** Submit your as condensed as possible - e.g. a single file with all the code. Please don't submit a whole git folder :),

  (*Choose whatever programming language you like. If you think the task is too simple, you are right. The main point is in reflecting on styles of programming. *)