INFDEV02-2
Homework 5

Dev Team

Abstraction, functions, recursion, methods, and data structures.

1 Exercise 1 - warm-up

- Design a class `Player` with methods: `Heal` and `Damage`. `Heal` adds one life point to the player, and `Damage` removes one,
- Design a class `Game` with two attributes `Player1` and `Player2`, and a method `Cheat` that calls `Heal` on `Player1` and `Damage` on `Player2`.

2 Exercise 2 - isEmpty/length/sum

- Improve the data structures `Node` and `Empty` to support the following new methods: `IsEmpty`, `Length`, and `Sum`, so that for a given list `l`:
  - `IsEmpty` returns whether `l` is `Node` or `Empty`,
  - `Length` recursively computes the length of `l`,
  - `Sum` recursively returns the sum of all values of `l`.
- Test the new methods with the following list: `Node(5, Node(9, Node(-1, Empty)))`

3 Exercise 3 - map/filter

- Improve the data structures `Node` and `Empty` to support the following new methods: `Map`, `Filter`.

- Test the new methods with the following list: `Node(5, Node(9, Node(-1, Empty)))`. When testing as transformation function for `Map` implement a lambda that increases its input by 2, and as predicate for `Filter` implement a lambda that checks whether its input is dividable by 3.

4 Exercise 4 - fold

- Improve the data structures `Node` and `Empty` to support the following new method: `Fold`.
- Use `Fold` to implement the `Map`, `Filter`, `Length`, and `Sum` methods defined previously.