

# Programowanie funkcyjne

## Notatka z wykładu 1.

### 1 Wykład

1. **Wartość** - Niezmienny (niemutowalny) obiekt typu prostego (np. liczba, litera, ciąg znakowy) lub niemutowalna kompozycja (struktura danych) innych wartości (np. lista, krotka, słownik, instancja klasy).
2. **Identyfikator** - Ciąg znakowy (zazwyczaj alfanumeryczny) identyfikujący stałą lub zmienną.
3. **Stała** - Niemutowalna relacja pomiędzy wartością a identyfikatorem. Przypisanie wartości do nazwy. Posiada tylko jeden stan (wartość się nie zmienia).
4. **Zmienna** - Mutowalna relacja pomiędzy wartością a identyfikatorem.
5. **Mutowalność** - Możliwość zmiany danej wartości (w miejscu).
6. **Stan** - Wartość identyfikatora w danych momencie czasu.
7. **Stanowość** (jako cecha języka) (ang. *statefulness*) - Posiadanie zmiennych oraz mutowalnych obiektów. Posiadanie stanu w aplikacji.
8. **Efekt uboczny** - Zjawisko zmiany stanu poza lokalnym kontekstem, tj. obserwowalny efekt poza zwróceniem wartości.
9. **Przeźroczystość referencyjna** - Brak efektów ubocznych, determinizm w stosunku do argumentów.
10. **Imperatywna ewaluacja** - sekwencyjna ewaluacja instrukcji (ang. *statement*).
11. **Funkcyjna ewaluacja** - ewaluacja kompozycji wyrażeń.
12. **Lambda calculus** - matematyczny model obliczeniowy, prekursor programowania funkcyjnego (Lispa).
13. **Architektura von Neumanna/maszyna Turinga** - teoretyczna architektura komputera sekwencyjnego/sekwencyjny model obliczeniowy, prekursor programowania imperatywnego.

14. Przykłady języków funkcyjnych: Clojure, Lisp, Scheme, Common Lisp, Racket, Erlang, Scala, Haskell, ML, Idris, F#.

## 2 Materiały dodatkowe

1. Robert C. Martin - Functional programming; What? Why? When?  
<https://www.youtube.com/watch?v=7Zlp9rKHGD4>
2. Values and Change: Clojure's approach to Identity and State  
<https://clojure.org/about/state>
3. Structure and Interpretation of Computer Programs  
<https://mitpress.mit.edu/sites/default/files/sicp/index.html>
4. Why Functional Programming Matters  
<https://www.cs.kent.ac.uk/people/staff/dat/miranda/whyfp90.pdf>
5. Why Functional Programming Matters (talk)  
<https://www.youtube.com/watch?v=XrNdvWqxBvA>