Teaching Functional Programming

Zoltán Horváth
hz@inf.elte.hu
lecturers: István Bozó, Zoltán Csörnyei , Andrea Kovács,
Gábor Páli, Máté Tejfel, Melinda Tóth

Faculty of Informatics, Eötvös Loránd University, Budapest

15th Workshop on Software Engineering Education and Reverse Engineering
Bohinj, Slovenia, Aug 24, 2015

Why Functional Programming?

- emphasis on *what* computation should be performed and not *how* to compute it, helps to understand programming methodology with abstractions (problem specification, problem refinement, pre- and postconditions, types, invariants, programming theorems)

- close to the specification due to their inherited mathematical nature.

```
[ x*x  \\ x <- [1..] | odd(x) ]
```

  $\{x^2 \mid x \in \mathcal{N} \wedge 2 \nmid x\}$

- referential transparency, equational reasoning for verification

```
take n xs ++ drop n xs == xs
```

Why Functional Programming?

- clean, simple, and composable abstractions (e.g. use of higher-order functions).

```
and = foldr (&&) True
```

- parallel evaluation, Paraphrase - refactoring for multicore parallel patterns

```
skel:do( [ { farm, [ { pipe, [ { seq, fun read/1 },
                                { seq, fun scan/1 },
                                { seq, fun parse/1 }
                              ] }
             ], 10 } ], Modules ).
```

- 5-10 times less SLOC, fewer errors, easier to maintain (improved distributed software robustness and productivity)

```
queens 0 = [[]]
queens n = [[q:b] \\ b <- queens (n - 1),q <- [0..7] | safe q b]
safe q b = and [not (checks q b i)  \\ i <- [0..(length b) - 1]]
checks q b i =   q ==  b !! i  ||  abs (q - b !! i)  ==  i + 1
```

Who uses Functional Programming - Erlang, Haskell, Caml, F#, Clean ..

- Ericsson – telecommunication (AXD301 ATM switch), simulation, testing, 3G, GPRS
- Amazon – Simple DB (DBMS)
- Yahoo – Online bookmarks service
- Facebook – chat server
- T-Mobile – SMS gateway
- Motorola – call processing
- CouchDb – document database server (multicore, multiserver clusters)
- YAWS – Yet Another Web Server
- Wings3D – 3D modeling
- Alcatel-Lucent- Haskell to prototype narrowband software radio systems, running in (soft) real-time
- Galois in application areas of security, information assurance and cryptography

Curricula - 2015

Level	Course title	Semester
BSc	Functional Programming	2nd
	Advanced Haskell	optional
	Erlang	optional
	λ - calculus	optional
	Web and mobile application programming in F#	optional
	Verification Tools of Functional Programming	optional
	Agda	optional
MSc	Concepts of Functional Programming Languages	1st
	Type systems	1st
	Advanced Functional Programming	2nd
	Software Technology Lab - RefactorErl, DSL	4 semesters
PhD	Distributed Functional Programming	freely chosen
	Verification of Functional Programs	freely chosen

Topics, concepts

- Higher order functions
- Algebraic types, type classes
- Higher-order types, existential types
- Uniqueness typing
- Dynamics, generic programming
- Purely functional data structures
- Parallel and distributed programming
- Combinators, combinator libraries
- Monadic programming
- Dependent types
- Interactive programs (functional reactive programming)
- Embedded domain-specific languages

Methods and tools in eductaion

- lecture, lab, consultation
- E-learning - ActiveHS (developed in Haskell)
- weekly test questions on concepts
- weekly home assignments to solve and upload via BE-AD
- complex home assignment
- lab programming test
- exam - ActiveHS

- lecture notes, textbooks, slides
- svn repository for assignments

ActiveHS – https://hackage.haskell.org/package/activehs

http://pnyf.inf.elte.hu/fp/Index_en.xml

Exercise: 10, 9, 8, ..., -9, -10

List the numbers from 10 to -10.

Solution>

Exercise: 10, 9, 8, ..., -9, -10

List the numbers from 10 to -10.

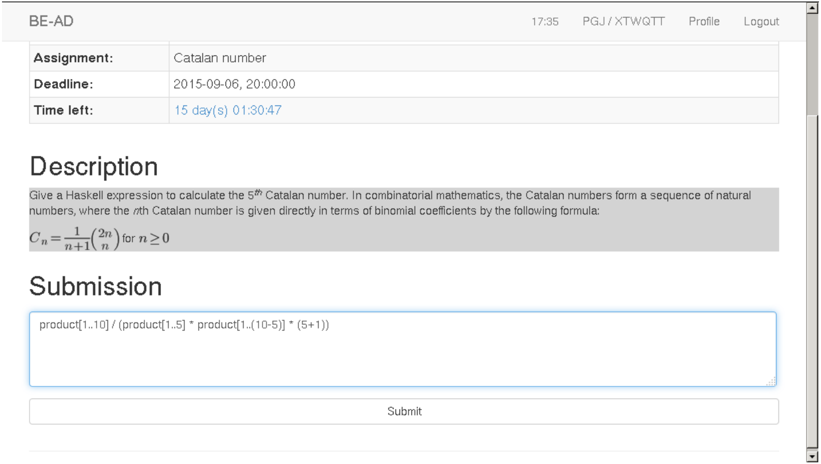
Solution> [10,9 .. -10]

```
[10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, -1, -2, -3, -4, -5, -6, -7, -8,
 -9, -10]
:: [Integer]
```

Jó megoldás! Szintén helyes megoldás:

```
[10,9.. -10] :: [Int]
```

BE-AD – https://github.com/andorp/bead

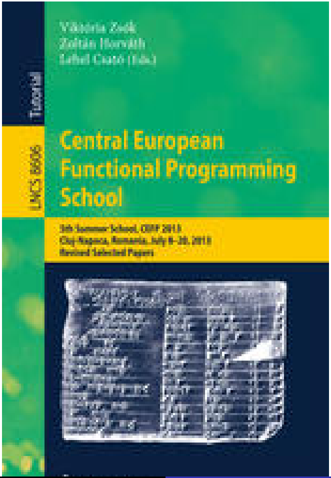


developed in Haskell, used by 14 courses, 40 groups of students, 208 assignments, 12407 submitted solutions

Recent results

- BSc: 128 of 193 students completed all assignments, 125 passed the exam
- MSc: 66 of 97 completed the course successfully

Central European Functional Programming Summer School
CEFP 2005, 2007, 2009, 2011, 2013, 2015 - LNCS volumes
supproted by CEEPUS, Erasmus IP and industry



Conclusions

- The precise formal definition of abstract concepts and experience in formal specification form a solid basis for better understanding on issues of functional programming and vice versa
- specification by pre- and postconditions, function compositions given in postcondition correspond to the structure of solution
- concepts with high expressive power (like higher order functions and types) need outstanding abilities, however not all of the students possess them.
- Industry needs a growing number of new staff members with expertise in functional programming.
- joint industry-university R&D labs are running in functional programming (static analysis, refactoring, defining FP embedded DSL-s for telecommunication)
- dozen of the PhD students have chosen functional programming as their main research domain.