

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šuráň** Jméno: **Jan** Osobní číslo: **515317**
Fakulta/ústav: **Fakulta informačních technologií**
Zadávající katedra/ústav: **Katedra teoretické informatiky**
Studijní program: **Informatika**
Specializace: **Teoretická informatika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Kompilace podmnožiny Clojure pomocí LLVM

Název bakalářské práce anglicky:

Compiling a subset of Clojure with LLVM

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Pierre Donat-Bouillud, Ph.D. katedra teoretické informatiky FIT

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **10.03.2026**

Termín odevzdání bakalářské práce: **15.05.2026**

podpis vedoucí(ho) ústavu/katedry

podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

Datum převzetí zadání

Šuráň Jan
Podpis studenta

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šuráň** Jméno: **Jan** Osobní číslo: **515317**
Fakulta/ústav: **Fakulta informačních technologií**
Zadávající katedra/ústav: **Katedra teoretické informatiky**
Studijní program: **Informatika**
Specializace: **Teoretická informatika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Kompilace podmnožiny Clojure pomocí LLVM

Název bakalářské práce anglicky:

Compiling a subset of Clojure with LLVM

Pokyny pro vypracování:

Clojure is a modern dialect of Lisp that runs on the JVM and supports dynamic typing, immutable data structures, first-class functions, and a macro system. While Clojure benefits from the JVM ecosystem, it is tightly coupled to its runtime and compilation model. LLVM is a modular compiler framework that enables native code generation for multiple architectures through its intermediate representation (LLVM IR). It is widely used to build optimizing compilers for modern languages.

The goal of this thesis is to define and implement a dynamically-typed subset of Clojure and compile it directly to LLVM IR. The subset should be small enough to be implementable within a bachelor thesis, yet expressive enough to support core Lisp features.

1. Research Clojure and LLVM
2. Define a subset of Clojure. The subset should include numbers, strings, lists, first-class functions, simple control flow (conditionals and recursion), and a simplified macro system
3. Implement a compiler from this subset to LLVM IR
4. Design a runtime system: implement memory management and runtime support for dynamic typing, lists, and closures.
5. Evaluate the system: assess correctness, expressiveness, and performance compared to JVM-based Clojure.

[1] Clojure documentation: <https://clojure.org>

[2] LLVM documentation: <https://llvm.org>

[3] Queinnec, C. Lisp in Small Pieces

[4] Appel, A. W. Modern Compiler Implementation

[5] <https://bernsteinbear.com/blog/compiling-a-lisp-0/>

Seznam doporučené literatury: