# Compilerbau

**Winter**

(engl. Compiler Construction )

## Modulnummer: FIN-INF-100382

| Kürzel | CP | Semester | Fachsem. | Dauer | Sprache | Niveau |
|---|---|---|---|---|---|---|
| CB | 5 | Winter | ab 3. | 1 Semester | deutsch | Bachelor |

| | |
|---|---|
| **Link zum LSF:** | LSF |
| **Zugang beschränkt:** | |
| **Verantwortung:** | Prof. Dr. Michael Kuhn |
| **Dozent:in:** | Prof. Dr. Michael Kuhn |
| **Lehrveranstaltungen:** | • Vorlesung *Compilerbau*<br>• Übung *Compilerbau* |
| **Verwendbarkeit:** | - B.Sc. INF: Informatik - Wahlpflicht<br>- B.Sc. INF: Informatik Wahlpflicht (SPO 2027)<br>- B.Sc. CV: Informatik - Wahlpflicht<br>- B.Sc. CV: Computervisualistik Wahlpflicht (SPO 2027)<br>- B.Sc. INGINF: Informatik - Wahlpflicht<br>- B.Sc. INGINF: Informatik Wahlpflicht (SPO 2027)<br>- B.Sc. WIF: Gestalten und Anwenden - Wahlpflicht<br>- B.Sc. WIF: Informatik Wahlpflicht (SPO 2027)<br>- B.Sc. INF (bilingual): Informatik - Wahlpflicht<br>- B.Sc. INF (bilingual): Informatik Wahlpflicht (SPO 2027) |

**Angestrebte Lernergebnisse:**
Die Teilnehmenden...

- verstehen theoretische Grundlagen von formalen Sprachen
- wenden Algorithmen zur Optimierung von regulären Ausdrücken, Grammatiken und Code-Repräsentationen an
- verstehen wichtige Konzepte von Compilern
- erstellen Komponenten eines Compilers in kleineren Teams
- präsentieren Lösungsansätze und Ergebnisse anderen Teilnehmenden

**Inhalt:**
Compiler übersetzen Programme aus einer Ausgangssprache in eine Zielsprache, üblicherweise aus einer höheren Programmiersprache in eine Maschinensprache. Dies erlaubt es unter anderem, die gleiche Anwendung mit relativ geringem Aufwand für unterschiedliche Prozessorarchitekturen zu übersetzen und darauf auszuführen. Ein tiefergehendes Verständnis von Compilern hilft dabei, effizientere und sicherere Programme zu schreiben. Die zu übersetzenden Programme werden außerdem durch Compiler überprüft und verbessert. Wie viele Fehler dabei gefunden werden können, hängt von der Striktheit der Sprache ab. Einige Sprachen, wie z. B. Rust, machen es schwieriger, Programme ohne Fehler zu schreiben, bieten dafür aber gewisse Korrektheitsgarantien. Darüber hinaus führen Compiler eine Vielzahl an Optimierungen durch, um die gegebene Rechnerarchitektur möglichst effizient zu nutzen. Die Vorlesung findet nach dem Prinzip des Flipped Classroom statt. Dabei sollen selbstständig die entsprechenden Kapitel des Buches „Introduction to Compilers and Language Design“ gelesen werden, die dann in der Vorlesung vertieft besprochen werden. In der Vorlesung werden die Konzepte vermittelt, die Übungen dienen der praktischen Anwendung und Umsetzung der erworbenen Kenntnisse.

**Arbeitsaufwand:**

- 56h Präsenz
- 94h selbstständiges Arbeiten

| **Prüfungsvorleistungen:** | **Studien-/Prüfungsleistungen:** | **Lehrform / SWS:** |
|---|---|---|
| • Aktive und erfolgreiche Teilnahme an den Übungen | Mündliche Prüfung | • Vorlesung (2 SWS)<br>• Übung (2 SWS) |

| **Voraussetzungen nach Prüfungsordnung:** | **Empfohlene Voraussetzungen:** |
|---|---|
| keine | • Praktische Kenntnis einer Systemprogrammiersprache wie C oder C++<br>• Grundkenntnisse in Datenstrukturen und Rechnerarchitekturen (z. B. Algorithmen und Datenstrukturen, Technische Informatik) |

| **Medienformen:** | **Literatur:** |
|---|---|

| | |
|---|---|
| | • Introduction to Compilers and Language Design (Douglas Thain) |

**Hinweise:**

---

Revision #1
Created 2026-10-31 09:01:55 UTC by Fakultät für Informatik
Updated 2026-10-31 09:01:55 UTC by Fakultät für Informatik