# Literatur

**allgemeine Literatur zur Vorlesung:**

[1] A.V. Aho, M.S. Lam, R. Sethi und J.D. Ullman, *Compilers - Principles, Techniques, and Tools*, Pearson/Addison Wesley 2007.

[2] H. Alblas und A. Nymeyer, *Practice and Principles of Compiler Buildung with C*, Prentice Hall 1996.

[3] A.W. Appel *Modern Compiler Implementation in Java (2nd Edition)*, Cambridge University Press, 2002.

[4] B. Bauer und R. Höllerer, *Übersetzung objektorientierter Sprachen* Springer 1998.

[5] J.P. Bennett, *Introduction to Compiling Techniques (second edition), A first course using ANSI C, LEX, and YACC*, MacGraw-Hill 1996.

[6] C. Fraser und D. Hanson, *A retargetable C Compiler: Design and Implementation*, The Benjamin/Cummings Publishing Company, 1995.

[7] D. Grune, H.E. Bal, C.J.H. Jacobs und K.G. Langendoen *Modern Compiler Design*, Wiley 2000.

[8] R.H. Güting und M. Erwig, *Übersetzerbau*, Springer 1999.

[9] J. Levine, T. Mason und D. Brown, *Lex & Yacc (2nd Edition)*, O¥Reilly & Associates, Inc. 1992.

[10] S.S. Muchnik, *Advanced Compiler Design and Implementation*, Morgan Kaufmann 1997.

[11] T.W. Parsons, *Introduction to Compiler Construction*, Computer Science Press, 1992.

[12] B.C. Pierce, *Types and Programming Languages*, MIT-Press, 2002.

[13] T. Pittman und J. Peters, *The Art of Compiler Design*, Prentice Hall 1992.

[14] T.W. Pratt und M.V. Zelkowitz, *Programming Languages - Design and Implementation*, Prentice-Hall 2001.

[15] F.J. Schmitt, *Praxis des Compilerbaus*, Hanser 1992.

[16] L. Schmitz, *Syntaxbasierte Programmierwerkzeuge*, Teubner 1995.

[17] R. Wilhelm und D. Maurer, *Übersetzerbau - Theorie, Konstruktion, Generierung*, Springer 1992.

[18] N. Wirth, *Compiler Construction*, Addison-Wesley 1996.

**spezielle Literatur**

[19] A.V, Aho und M.J. Corasick, *Efficient string matching: An aid to bibliographic search*, Commun. ACM 18 (1975), 333-340.

[20] A.V. Aho, S.C. Johnson, *Optimal code generation for expression trees*, J. ACM 23 (1976), 488-501.

[21] A.V. Aho, S.C. Johnson und J.D. Ullman, *Code Generation for expressions with common subexpressions*, J. ACM 24 (1977), 146-160.

[22] A.V. Aho, M. Ganapathi, S.W.K. Tijang, *Code Generation Using Tree Pattern Matching and Dynamic Programming*, ACM TOPLAS 11 (1989), 491-516.

[23] D.F. Bacon, S.L. Graham und O.J. Sharp, *Compiler Transformations for High Performance Computing*, ACM Comp. Surveys 26 (1994) 345-420.

[24] J. Bruno, R. Sethi, *Code generation for a one-register machine*, J. ACM 23 (1976), 502-510.

[25] C. Becker und P. Hagemann, *GRAM - Ein Drei-Adresscode Interpreter*, Studienarbeit (2002)

[26] G.V. Bochmann, *Semantic Evaluation from Left to Right*, Comm. ACM 19 (1976), 55-62.

[27] r.G.G. Cattell, *Automatic derivation of code generators from machine descriptions*, TOPLAS 2 (1980), 173-190

[28] M. Ganapathi, C.N. Fischer, *Description-Driven Code Generation Using Attribute Grammars*, 9. ACM Symposium on Principles of Programming Languages (1982), 107-119.

[29] R.S. Glanville, S.L. Graham, *A New Method for Compiler Code Generation* 5. ACM Symposium on Principles of Programming Languages (1978), 231-240.

[30] A. Goldberg, D. Robson, *Smalltalk-80, The Language and its Implementation*, Addison-Wesley 1983.

[31] C.W. Hoffman und M.J. O´Donnel, *Pattern matching in trees*, J. ACM 29 (1982), 68-95.

[32] E.T. Irons, *A syntax directed compiler for Algol 60*, Comm. ACM 4 (1961), 51-55. Springer, 1987.

[33] M. Jazayeri, W.F. Ogden und W.C. Rounds, *The intrinsic exponential complexity of the circularity problem for attribute grammars*, Comm. ACM 18 (1975), 697-706. Springer, 1983.

[34] D.E. Knuth, *Semantics of context-free languages*, Math. System Theory 2 (1968), p 127-145, Errata Math. System Theory 5 (1971), 95-96.

[35] P.M. Lewis, D.J. Rosenkrantz und E.E. Stearns, *Attributed Translations*, JCSS 9 (1969), 524-549.

[36] R. Sethi, J.D. Ullman, *The generation of optimal code for arithmetic expressions*, J. ACM 17 (1970), 715-728.

[37] L. Sterling und E. Shapiro, *The Art of Prolog*, MIT Press, Cambridge 1986.

[38] T. Wichers, *Optimale Code-Generierung mit Hilfe von Tree-Pattern Matching*, Masterarbeit, Universität Hannover, (2005).