

Inhalt

1.	Einleitung	1
1.1.	Algorithmen und Programme	1
1.2.	Notwendigkeit der Formalisierung der Beschreibung von Programmiersprachen	4
1.3.	Verschiedene Vorgehensweisen der Semantikdefinition. Ein Beispiel	6
2.	Methoden der Syntaxdefinition	10
2.1.	Sprachen und generative Grammatiken	10
2.2.	Kontextfreie Grammatiken	12
2.3.	Anwendung kontextfreier Grammatiken zur Definition von Programmiersprachen	16
2.3.1.	Backus-Naur-Form (BNF)	16
2.3.2.	Die Beispielprogrammiersprache BPS	19
2.3.3.	Vorteile und Grenzen kontextfreier Grammatiken als Mittel zur Definition von Programmiersprachen	21
2.4.	Zweistufiggrammatiken	23
2.4.1.	Der Begriff der Zweistufiggrammatik	23
2.4.2.	Beschreibung der Untermenge U der Sprache BPS durch eine Zweistufiggrammatik	26
2.4.3.	Vor- und Nachteile von Zweistufiggrammatiken als Mittel zur Definition von Programmiersprachen	30
2.5.	Die abstrakte Syntax von Programmiersprachen	31
2.5.1.	Konkrete und abstrakte Syntax	31
2.5.2.	Die Wiener Methode der Definition der abstrakten Syntax	33
2.6.	Bibliographie	37
3.	Methoden der Semantikdefinition	39
3.1.	Maschine, Sprache und Implementation	41
3.1.1.	Der Begriff der Maschine	41
3.1.2.	Direkte Implementation einer Programmiersprache: Interpretation und Kompilation	48
3.1.3.	Die Computerumgebung: Variablen, Werte und Zustände	57
3.2.	Die operationale Methode der Semantikdefinition	58
3.2.1.	Operationale Methode und direkte Implementation	95
3.2.2.	Die Wiener Methode der Semantikdefinition als Beispiel der operationalen Methode	63

3.2.3.	Systeme semantischer Regeln	68
3.2.4.	Verallgemeinerte Systeme semantischer Regeln	71
3.3.	Semantikdefinition durch Fixpunkte von Gleichungssystemen	75
3.3.1.	Gleichungssysteme und ihre Lösung	77
3.3.2.	Fixpunktmethode und operationale Methode	82
3.4.	Die denotationale Methode der Semantikdefinition und Übereinstimmung verschiedenartig definierter Semantiken	84
3.4.1.	Die denotationale Methode	84
3.4.2.	Korrektheit und Äquivalenz semantischer Definitionen	87
3.5.	Erweiterung: Semantik ausgewählter Programmkonstruktionen	92
3.5.1.	Sprünge und Fortsetzung	92
3.5.2.	Programme mit Blockstruktur	97
3.5.3.	Prozeduren (Funktionen)	100
3.5.4.	Rekursive Prozeduren	103
3.5.5.	Prozeduren und Fortsetzungen	109
3.5.6.	Ein- und Ausgabeanweisungen	111
3.5.7.	Datentypen	113
3.6.	Die axiomatische Methode der Semantikdefinition	114
3.6.1.	Formalisierte Theorien und formalisierte Theorien erster Stufe für Program- miersprachen	116
3.6.2.	Das Hoaresche System von Schlußregeln für iterative Programme	123
3.6.3.	Ein System von Schlußregeln für Programme mit Blockstruktur und nicht- rekursiven Prozeduren	130
3.6.4.	Ein System von Schlußregeln für rekursive Prozeduren	133
3.7.	Bibliographie	137
4.	Attributierte Grammatiken	139
4.1.	Arten attributierter Grammatiken	141
4.1.1.	Knuthsche attributierte Grammatiken	141
4.1.2.	Grammatiken syntaktischer Funktionen	145
4.2.	Anwendung attributierter Grammatiken zur Semantikdefinition	158
4.3.	Bibliographie	169
5.	Algebraische Modelle von Programmiersprachen	170
5.1.	Algebraisches Modell für kontextfreie Programmiersprachen	171
5.1.1.	Repräsentationalgebra	172
5.1.2.	Abstrakte Syntax	175
5.1.3.	Semantische Algebren	179
5.2.	Darstellung von Kontextbedingungen	187
5.3.	Bibliographie	190
	Anhang	191
	Literatur	194
	Sachwortverzeichnis	201