

Inhaltsverzeichnis

1 EINFÜHRUNG IN DIE PROGRAMMIERSPRACHE C	1
1.1 Das erste Programm	2
1.2 Ursprung und Ziele von C	5
1.3 Standardisierung von C	6
1.3.1 Der C99-Standard	7
1.3.2 Der C11-Standard	8
1.4 Eigenschaften von C	8
1.5 C und C++	10
1.6 Stammbaum imperativer Programmiersprachen	11
1.7 Abstraktionsgrad von Programmiersprachen	12
1.7.1 Abstraktion bei Ausdrücken	13
1.7.2 Abstraktion bei Kontrollstrukturen	14
1.7.3 Datenabstraktion	15
1.8 Zusammenfassung	16
2 EINFACHE BEISPIELPROGRAMME	19
2.1 Ausgabe auf dem Bildschirm	20
2.1.1 Kommentarzeile	21
2.1.2 Include-Anweisung	21
2.1.3 Leerzeilen	21
2.1.4 Die Funktion main()	21
2.1.5 Geschweifte Klammern	22
2.1.6 Strichpunkt	23
2.1.7 Definitionen und Anweisungen	23
2.1.8 Die Funktion printf()	23
2.1.9 for-Schleife	24
2.1.10 Inkludieren von Bibliotheksfunktionen	24
2.2 Lokale Variablen, Ausdrücke und Schleifen	25
2.2.1 Variante mit symbolischen Konstanten und int-Variablen	25
2.2.2 Variante ohne symbolische Konstanten	27
2.2.3 Variante mit einer double-Variablen	28
2.3 Zahlen von der Tastatur einlesen	29
2.4 Formatierung bei der Ausgabe	30
2.5 Zusammenfassung	32

3 ENTWURF VON PROGRAMMEN	35
3.1 Vom Problem zum Programm	36
3.1.1 Der euklidische Algorithmus als Beispiel für Algorithmen.....	37
3.1.2 Beschreibung sequenzieller Abläufe	38
3.1.3 Programmierung ohne Sprünge	40
3.1.4 Variablen und Zuweisungen.....	41
3.2 Entwurf mit Nassi-Shneiderman-Diagrammen	43
3.2.1 Diagramme für die Sequenz.....	44
3.2.2 Diagramme für die Selektion	45
3.2.3 Diagramme für die Iteration.....	47
3.2.4 Vom Struktogramm zum Programm	50
3.3 Pseudocode	53
3.3.1 Natürliche und formale Sprachen	53
3.3.2 Freier und formaler Pseudocode	53
3.4 Zusammenfassung	54
3.5 Übungsaufgaben	57
4 DATEN UND FUNKTIONEN	59
4.1 Daten in prozeduralen Programmen.....	60
4.1.1 Zeichen	60
4.1.2 Variablen	63
4.1.3 Datentypen.....	63
4.2 Funktionen in prozeduralen Programmen	64
4.2.1 Unterprogramme und Bibliotheken.....	65
4.2.2 Aufrufhierarchie von Unterprogrammen	66
4.2.3 Vorteile von Unterprogrammen	66
4.2.4 Unterprogramme in Nassi-Shneiderman-Diagrammen	66
4.2.5 Schrittweise Verfeinerung beim Top-Down-Design	68
4.2.6 Das EVA-Prinzip	69
4.2.7 Exemplarische Durchführung eines Top-Down-Design.....	69
4.3 Daten in C	72
4.3.1 Der Datentyp int in der Programmiersprache C	73
4.3.2 Der Datentyp float in C	74
4.3.3 Operationen auf einfachen Datentypen in C	75
4.3.4 Selbst definierte Datentypen in C	75
4.4 Funktionen in C	76
4.4.1 Definition von Funktionen.....	76
4.4.2 Aufruf von Funktionen	77
4.4.3 Beispielprogramm zur Berechnung einer Summe	78
4.5 Struktur einer Quelldatei in C	79

4.5.1	Kommunikation zwischen Funktionen	80
4.5.2	Interne und externe Variablen	80
4.5.3	Beispiel für eine Aufrufhierarchie	81
4.5.4	Reihenfolge der externen Variablen und Funktionen	82
4.5.5	Funktionsprototypen	82
4.5.6	Bestandteile eines Programms	83
4.5.7	Aufbau einer Quelldatei	83
4.5.8	Beispielprogramm für den Aufbau einer Quelldatei	84
4.6	Zusammenfassung	86
5	PROGRAMMERZEUGUNG UND -AUSFÜHRUNG	91
5.1	Compiler	93
5.1.1	Lexikalische Analyse	94
5.1.2	Syntaxanalyse	95
5.1.3	Semantische Analyse	95
5.1.4	Optimierungen	96
5.1.5	Codeerzeugung	96
5.1.6	Laufzeitsystem des Compilers	97
5.2	Linker	97
5.3	Lader	99
5.4	Debugger	100
5.5	Integrierte Entwicklungsumgebungen	101
5.6	Zusammenfassung	101
6	LEXIKALISCHE KONVENTIONEN	105
6.1	Zeichenvorrat von C	106
6.1.1	Quell- und Ausführungszeichensatz nach C11	106
6.1.2	Zeichensatz nach C90	107
6.1.3	Der Unicode in C	107
6.1.4	Multibyte-Zeichen	108
6.1.5	Der Datentyp wchar_t	109
6.2	Lexikalische Einheiten	109
6.2.1	Reservierte Wörter	111
6.2.2	Bezeichner	113
6.2.3	Konstanten	114
6.2.4	Operatoren und Interpunktionszeichen	125
6.3	Zusammenfassung	126
6.4	Übungsaufgaben	131


7 DATENTYPEN UND VARIABLEN IN C	133
7.1 Übersicht über die Datentypen in C	135
7.1.1 Der Datentyp void	135
7.1.2 Klassifikation der Datentypen im Falle von C90	135
7.1.3 Ergänzung des Klassifikationsbaums der Datentypen bei C11	136
7.2 Einfache Datentypen in C	137
7.2.1 Übersicht über die Integer-Typen in C	137
7.2.2 Die einzelnen Standard-Integer-Typen von C90	139
7.2.3 Der zusätzliche Datentyp long long in C11	143
7.2.4 Aufzählungstypen	143
7.2.5 Die Standard-Gleitpunkt-Typen float und double	143
7.3 Variablen in C	146
7.3.1 Definitionen, Deklarationen und Vereinbarungen	147
7.3.2 Definition einfacher Variablen	148
7.3.3 Externe und interne Variablen	148
7.3.4 Initialisierung von Variablen	151
7.4 Qualifikatoren	152
7.4.1 Qualifikatoren in C90	152
7.4.2 Qualifikatoren in C11	153
7.5 Generelle Arten von Typen in C	154
7.6 Zusammenfassung	154
7.7 Übungsaufgaben	159
8 EINFÜHRUNG IN POINTER UND ARRAYS	161
8.1 Pointertypen und Pointervariablen	162
8.1.1 Definition von Pointervariablen	163
8.1.2 NULL-Pointer	164
8.1.3 Wertebereich von Pointern	165
8.1.4 Wertzuweisung an einen Pointer	165
8.1.5 Adressoperator	166
8.1.6 Zugriff auf ein Objekt über einen Pointer	167
8.1.7 Beispiele für das Referenzieren und Dereferenzieren	169
8.2 Pointer auf void	170
8.3 Eindimensionale Arrays	171
8.3.1 Beispiel für ein Array	172
8.3.2 Beispiel zum Speichern von Daten in einem Array	173
8.3.3 Weitere Erläuterungen zur Definition eines Arrays	174
8.3.4 Zeichenketten und Arrays	175
8.3.5 Beispiel für Zeichenketten und Arrays	175
8.4 Der Qualifikator restrict	176

8.5	Zusammenfassung	177
8.6	Übungsaufgaben	181
9	AUSDRÜCKE, ANWEISUNGEN UND OPERATOREN.....	183
9.1	Operatoren und Operanden.....	185
9.1.1	Stelligkeit der Operatoren.....	185
9.1.2	Postfix- und Präfixoperatoren.....	186
9.2	Ausdrücke und Anweisungen	186
9.3	Nebeneffekte.....	188
9.4	Auswertungsreihenfolge komplexer Ausdrücke	189
9.4.1	Einstellige und mehrstellige Operatoren.....	189
9.4.2	Operatoren gleicher Priorität	190
9.5	L-Werte und R-Werte.....	191
9.6	Zusammenstellung der Operatoren	194
9.6.1	Einstellige arithmetische Operatoren	194
9.6.2	Zweistellige arithmetische Operatoren	196
9.6.3	Zuweisungsoperatoren.....	198
9.6.4	Relationale Operatoren	200
9.6.5	Logische Operatoren.....	203
9.6.6	Bit-Operatoren	206
9.6.7	Sonstige Operatoren	210
9.6.8	Prioritätentabelle der Operatoren	217
9.7	Implizite Typumwandlung	219
9.7.1	Gewöhnliche arithmetische Konversionen	221
9.7.2	Zuweisungen, Rückgabewerte und Argumente von Funktionen	224
9.7.3	Konvertiervorschriften	225
9.7.4	Zwei Beispiele für Typkonvertierungen	225
9.8	Sequenzpunkte bei Nebeneffekten	227
9.9	Zusammenfassung	228
9.10	Übungsaufgaben	235
10	KONTROLLSTRUKTUREN	239
10.1	Blöcke – Kontrollstrukturen für die Sequenz.....	240
10.2	Kontrollstrukturen für die Selektion.....	241
10.2.1	Einfache Alternative – if und else	241
10.2.2	Mehrfache Alternative – else if	243
10.2.3	Mehrfache Alternative – switch	244
10.3	Kontrollstrukturen für die Iteration.....	249
10.3.1	Abweisende Schleife mit while	249

10.3.2	Abweisende Schleife mit for	250
10.3.3	Annehmende Schleife mit do while	258
10.3.4	Endlosschleife sowie leere Anweisung.....	259
10.4	Sprunganweisungen	261
10.4.1	break	261
10.4.2	continue.....	262
10.4.3	goto und Marken	264
10.5	Zusammenfassung	265
10.6	Übungsaufgaben	268
11	BLÖCKE UND FUNKTIONEN	271
11.1	Struktur eines Blockes in C90 und C11	272
11.2	Schachtelung von Blöcken	274
11.3	Gültigkeit, Sichtbarkeit und Lebensdauer	275
11.4	Definition und Aufruf von Funktionen.....	278
11.4.1	Definition von Funktionen.....	278
11.4.2	Formale und aktuelle Parameter	281
11.4.3	Syntax eines Funktionsaufrufs	282
11.4.4	Rücksprung mit oder ohne Rückgabewert – die return-Anweisung.....	282
11.4.5	Call by reference-Schnittstelle.....	284
11.4.6	Pointer über die Parameterliste	284
11.5	Vorwärtsdeklaration von Funktionen	286
11.5.1	Behandlung von Library-Funktionen	288
11.5.2	Eigene Header-Dateien.....	289
11.6	Gültigkeitsbereiche von Namen	289
11.7	Die Ellipse ... – ein Mittel für variable Parameteranzahlen	290
11.8	Rekursive Funktionen	292
11.8.1	Iteration und Rekursion	293
11.8.2	Iterative und rekursive Berechnung der Fakultätsfunktion	293
11.8.3	Beispiel für iterative und rekursive Berechnung der Binärdarstellung	301
11.9	Inline-Funktionen	304
11.10	Funktionen ohne Wiederkehr in C11	305
11.11	Zusammenfassung	305
11.12	Übungsaufgaben	311
12	FORTGESCHRITTENE PROGRAMMIERUNG MIT POINTERN .	319
12.1	Pointer und Elemente von Arrays	320
12.1.1	Äquivalenz von Array- und Pointernotation	321
12.1.2	Vergleich von Arrays	321

12.1.3	Arrayname als nicht modifizierbarer L-Wert	321
12.1.4	Pointerarithmetik	322
12.1.5	Spielerische Erkundung von Fischer und Friederich	326
12.1.6	Initialisierung von Arrays	328
12.1.7	Mehrdimensionale Arrays.....	331
12.1.8	Konstante Zeichenketten – Arrays aus Zeichen.....	333
12.1.9	char-Arrays.....	334
12.2	Übergabe von Arrays und Zeichenketten an Funktionen	335
12.2.1	Übergabe von Arrays	335
12.2.2	Übergabe von Zeichenketten	336
12.2.3	Ausgabe von Zeichenketten und von char-Arrays.....	336
12.3	Vergleich von char-Arrays und Pointern auf Zeichenketten	337
12.4	Das Schlüsselwort const bei Pointern und Arrays.....	339
12.5	Beispiele für das Kopieren von Zeichenketten von Hand	341
12.6	Standardfunktionen zur Stringverarbeitung und Speicherbearbeitung	343
12.6.1	Einige Stringverarbeitungsfunktionen.....	343
12.6.2	Funktionen zur Speicherbearbeitung	350
12.7	Beispiele für eindimensionale Arrays von Pointern und Pointer auf Pointer	355
12.7.1	Beispiele für eindimensionale Arrays von Pointern	355
12.7.2	Beispiel für Pointer auf Pointer.....	357
12.7.3	Beispiel für die Initialisierung von Arrays von Pointern.....	358
12.7.4	Vergleich zweidimensionales Array gegen eindimensionales Array von Pointern.....	359
12.8	Pointer auf Funktionen	360
12.8.1	Vereinbarung eines Pointers auf eine Funktion.....	361
12.8.2	Aufruf einer Funktion.....	361
12.8.3	Beispiel für Pointer auf eine Funktion.....	362
12.8.4	Beispiel Nullstellen mit dem Newtonschen Iterationsverfahren	363
12.9	Zusammenfassung	365
12.10	Übungsaufgaben	369
13	STRUKTUREN, UNIONEN UND BITFELDER.....	375
13.1	Records auf der Platte und Strukturen	376
13.1.1	Vereinbarung von Strukturtypen und Strukturvariablen.....	377
13.1.2	Verschiedene Möglichkeiten zur Definition von Strukturvariablen.....	379
13.1.3	Zulässige Operationen	380
13.1.4	Selektion der Komponenten	381
13.1.5	Übergabe von Strukturvariablen an Funktionen und Rückgabe.....	383
13.1.6	Initialisierung einer Struktur mit einer Initialisierungsliste	384

13.1.7	Initialisierung einzelner Komponenten einer Struktur	385
13.1.8	Stringvariablen in Strukturen	386
13.1.9	Anwendungsmöglichkeiten von Strukturen	387
13.2	Unionen	390
13.3	Anonyme Typen und Objekte	393
13.3.1	Anonyme Strukturen und Unionen	394
13.3.2	Compound Literals	394
13.4	Bitfelder – Komponenten von Strukturen und Unionen	396
13.5	Zusammenfassung	401
13.6	Übungsaufgaben	404
14	KOMPLEXERE VEREINBARUNGEN, EIGENE TYPNAMEN UND NAMENSRÄUME	411
14.1	Komplexere Vereinbarungen	412
14.1.1	Komplexere Vereinbarungen von Variablen und Funktionen	412
14.1.2	Komplexere Datentypen	414
14.2	Vereinbarung eigener Typnamen als Aliasnamen	414
14.2.1	Beispiel für einen zusammengesetzten Datentyp	415
14.2.2	Verwendung von typedef bei einfachen Datentypen	415
14.2.3	Syntax einer typedef-Anweisung	415
14.2.4	Neuer Typname plus zusätzlicher Aliasname	416
14.2.5	Vorwärtsdeklaration und Information Hiding mit typedef	416
14.2.6	Bedeutung von typedef für portable Datentypen	417
14.3	Namensräume	417
14.4	Zusammenfassung	419
15	SPEICHERKLASSEN	421
15.1	Segmente des Adressraums eines Programms	423
15.2	Der Linker	424
15.2.1	Virtuelle Adressen	425
15.2.2	Bindungen	426
15.3	Programme aus mehreren Dateien – die Speicherklasse extern	427
15.3.1	Beispielprogramm extern-Deklaration ohne Header-Datei	429
15.3.2	Beispielprogramm extern-Deklaration mit Header-Datei	429
15.3.3	Die extern-Deklaration bei Arrays	431
15.3.4	Beispiel für die extern-Deklaration bei Arrays	431
15.4	Programme aus mehreren Dateien – die Speicherklasse static für externe Definitionen	432
15.5	Speicherklassen bei lokalen Variablen	433

15.5.1	Automatische Variablen	433
15.5.2	Speicherklasse static für lokale Variablen	436
15.6	Automatische und nicht automatische Initialisierung von Variablen	438
15.7	Überblick über die Speicherklassen sequentieller Programme	439
15.8	Zusammenfassung	439
15.9	Übungsaufgaben	443
16	EIN- UND AUSGABE.....	445
16.1	Speicherung von Daten in Dateisystemen.....	447
16.1.1	Dateien unter UNIX – das Streamkonzept	448
16.1.2	Schichtenmodell von UNIX für die Ein- und Ausgabe	449
16.2	Das Ein-/Ausgabe-Konzept von C	450
16.3	Standardeingabe und -ausgabe in C	451
16.3.1	Umlenkung der Ein- und Ausgabe	452
16.3.2	Pipelining.....	453
16.3.3	Funktionen für die Standardeingabe und -ausgabe	454
16.4	High- und Low-Level-Bibliotheksfunktionen zur Ein- und Ausgabe	454
16.5	Der File-Pointer bei High-Level-Funktionen.....	456
16.5.1	Öffnen und Schliessen von Dateien	458
16.5.2	Puffersteuerung.....	459
16.5.3	Navigieren innerhalb von Dateien	460
16.5.4	Fehlerbehandlung	461
16.5.5	Beispiel für den Umgang mit File-Pointern.....	462
16.5.6	Operationen auf Dateien	463
16.6	Schreiben in Dateien mit High-Level-Funktionen	463
16.6.1	Schreiben von Objekten	464
16.6.2	Schreiben von formatierten Strings	465
16.7	Lesen von Dateien mit High-Level-Funktionen.....	472
16.7.1	Lesen von Objekten	473
16.7.2	Lesen von formatierten Strings	474
16.8	Alternative Funktionen für mehr Sicherheit nach C11	477
16.8.1	Die Funktion gets_s().....	479
16.8.2	Die Funktion fopen_s().....	480
16.9	Zusammenfassung	480
16.10	Übungsaufgaben	486
17	PROGRAMMAUFRUF UND -BEENDIGUNG	493
17.1	Übergabe von Argumenten beim Programmaufruf.....	494
17.2	Beendigung von Programmen 	496

17.2.1	Einige Funktionen der Standardbibliothek.....	497
17.2.2	Beispiel für eine Abfrage des Rückgabewertes.....	501
17.3	Zusammenfassung	502
18	DYNAMISCHE SPEICHERZUWEISUNG	505
18.1	Reservierung von Speicher	507
18.1.1	Die Funktion malloc()	507
18.1.2	Die Funktion calloc().....	509
18.1.3	Die Funktion realloc()	510
18.2	Freigabe von Speicher.....	511
18.2.1	Beispiel.....	512
18.2.2	Speicherlecks.....	513
18.3	Dynamisch erzeugte Arrays.....	513
18.4	Zusammenfassung	515
18.5	Übungsaufgaben	516
19	DYNAMISCHE DATENSTRUKTUREN.....	519
19.1	Verkettete Listen	520
19.1.1	Datentyp eines Listenelements	520
19.1.2	Einfach verkettete Liste	521
19.1.3	Andere Listenarten	527
19.2	Baumstrukturen	528
19.2.1	Allgemeine Darstellung von Baumstrukturen	528
19.2.2	Formale Definition eines Baumes	529
19.2.3	Binäre Bäume	531
19.2.4	Durchlaufen von binären Bäumen.....	533
19.2.5	Beispielprogramm für binäre Bäume.....	536
19.3	Zusammenfassung	543
19.4	Übungsaufgaben	546
20	SORTIEREN UND SUCHEN	549
20.1	Interne Sortierverfahren.....	551
20.1.1	Konzept für das iterative Sortieren durch direktes Auswählen	552
20.1.2	Konzept für das rekursive Sortieren mit dem Quicksort-Verfahren	553
20.1.3	Beispielprogramme für interne Sortierverfahren.....	556
20.1.4	Sortierverfahren im Vergleich	559
20.1.5	Die Quicksort-Funktion der Standardbibliothek	560
20.2	Einfache Suchverfahren	562
20.2.1	Sequenzielles Suchen.....	563
20.2.2	Halbierungssuchen	563

20.2.3	Beispielprogramm mit bsearch()	566
20.3	Suchen nach dem Hashverfahren	567
20.3.1	Einführendes konzeptionelles Beispiel	567
20.3.2	Einfache Schlüsseltransformation und Konflikte	568
20.3.3	Konfliktlösungen für ausreichend große Tabellen	570
20.3.4	Direkte Verkettung bei Kollisionen	580
20.3.5	Weitere Schlüsseltransformationen	584
20.3.6	Vor- und Nachteile der Hashverfahren	585
20.4	Suchen mit Hilfe von Backtracking	586
20.5	Zusammenfassung	592
20.6	Übungsaufgaben	596
21	PRÄPROZESSOR	599
21.1	Aufgaben des Präprozessors	600
21.2	Einfügen von Header-Dateien des lokalen Verzeichnisses und der Standardbibliotheken	601
21.3	Symbolische Konstanten und Makros mit Parametern	603
21.3.1	Beispiel für Komplikationen bei der Textersetzung und ihre Beseitigung	604
21.3.2	Probleme bei Makros bestehend aus mehreren Anweisungen	606
21.4	Bedingte Kompilierung	607
21.5	Informationen über den Übersetzungskontext	609
21.5.1	Die Präprozessor-Direktive <code>line</code>	610
21.5.2	Die Präprozessor-Direktive <code>error</code>	610
21.6	Weitere Präprozessor-Direktiven	610
21.6.1	Parameter mit <code>#</code> in Strings umwandeln	610
21.6.2	Parameter mit <code>##</code> verknüpfen	611
21.6.3	Generische Auswahl in C11	612
21.6.4	Statische Zusicherungen in C11	613
21.7	Der Pragma-Operator in C11	614
21.8	Zusammenfassung	614
21.9	Übungsaufgaben	616
22	MODULAR DESIGN IN C	619
22.1	Konzept des Structured Design	620
22.2	Konzept des Modular Design	621
22.2.1	Kapselung und Information Hiding	621
22.2.2	Export-Schnittstellen	622
22.2.3	Import-Schnittstellen	623
22.2.4	Das Modulkonzept – eine Vorstufe der Objektorientierung	624

22.3	Umsetzung des Modular Design in C	625
22.3.1	Information Hiding mit static	625
22.3.2	Behandlung der Header	625
22.4	Realisierung eines Stacks mit Modular Design in C	629
22.4.1	Die Header-Datei stack.h	630
22.4.2	Arbeitsteilige Entwicklung	631
22.4.3	Implementierung des Moduls main.c	631
22.4.4	Implementierung des Moduls stack.c	632
22.5	Zusammenfassung	636
22.6	Übungsaufgaben	639
23	THREADS NACH C11	641
23.1	Betriebssystemprozesse und Threads	642
23.2	Threads im C11-Standard	645
23.2.1	Erzeugen von Threads	647
23.2.2	Beenden von Threads	648
23.3	Synchronisation von Threads	651
23.3.1	Wechselseitiger Ausschluss mit Hilfe von Mutexen	652
23.3.2	Synchronisation mit Zustandsvariablen	657
23.3.3	Deadlocks, Livelocks und Starvation	665
23.4	Fortgeschrittene Programmierung mit Threads	667
23.4.1	Lokale und statische Variablen	668
23.4.2	Thread-lokaler Speicher	670
23.4.3	Funktionen für einen Thread-spezifischen Speicher	672
23.4.4	Atomare Operationen	675
23.5	Zusammenfassung	676
	BEGRIFFSVERZEICHNIS	681
	ANHANG A STANDARD BIBLIOTHEKSFUNKTIONEN	683
	ANHANG B LOW-LEVEL-DATEIZUGRIFFSFUNKTIONEN	691
	ANHANG C WANDLUNGEN ZWISCHEN ZAHLENSYSTEMEN	701
	ANHANG D ASCII- UND UNICODE-CODIERUNGEN	707
	ANHANG E KOMPLEXE ZAHLEN	711
	ANHANG F BEISPIELPROGRAMM FÜR POINTER AUF VOID ALS FORMALEN PARAMETER	713
	LITERATURVERZEICHNIS	715
	INDEX	717