

Ulrich Kaiser, Christoph Kecher

C/C++

Auf einen Blick

1	Einige Grundbegriffe	21
2	Einführung in die Programmierung	35
3	Ausgewählte Sprachelemente von C	43
4	Arithmetik	73
5	Aussagenlogik	97
6	Elementare Datentypen und ihre Darstellung	129
7	Modularisierung	181
8	Zeiger und Adressen	253
9	C-Referenz (Teil 1)	271
10	Kombinatorik	315
11	Leistungsanalyse und -messung von Algorithmen	349
12	Sortierverfahren	395
13	Datenstrukturen	441
14	C-Referenz (Teil 2)	497
15	Ausgewählte Datenstrukturen	519
16	Elemente der Graphentheorie	613
17	Projekt 2D-Grafikprogrammierung	701
18	C++-Referenz (Teil 1)	703
19	Objektorientierte Programmierung	743
20	Klassen in C++	759
21	Vererbung in C++	823
22	C++-Referenz (Teil 2)	893
23	Die C++-Standard-Library	983
24	Projekt: 3D-Grafikprogrammierung	1077
	Anhang	1079
	Lösungen	1083

Inhalt

Vorwort	17
Danksagung	18
Vorwort zur zweiten Auflage	19
Vorwort zur vierten Auflage	20

1 Einige Grundbegriffe 21

1.1 Algorithmus	23
1.2 Datenstruktur	27
1.3 Programm	29
1.4 Aufgaben	32

2 Einführung in die Programmierung 35

2.1 Die Programmierumgebung	40
2.1.1 Der Editor	40
2.1.2 Der Compiler	41
2.1.3 Der Linker	42
2.1.4 Der Debugger	42

3 Ausgewählte Sprachelemente von C 43

3.1 Programmrahmen	43
3.2 Zahlen	44
3.3 Variablen	44
3.4 Operationen	46
3.4.1 Zuweisungsoperationen	46
3.4.2 Rechenoperationen	47
3.4.3 Vergleichsoperationen	49
3.5 Kommentare	49
3.6 Elementare Ein-/Ausgabe	50
3.6.1 Bildschirmausgabe	50
3.6.2 Tastatureingabe	52
3.7 Kontrollfluss	52
3.7.1 Bedingte Befehlsausführung	53
3.7.2 Wiederholte Befehlsausführung	54
3.7.3 Verschachtelung von Kontrollstrukturen	60
3.8 Beispiele	61
3.8.1 Das erste C-Programm	61

3.8.2	Das zweite C-Programm	62
3.8.3	Das dritte C-Programm	66
3.9	Aufgaben	69

4 Arithmetik 73

4.1	Folgen	73
4.2	Summen	80
4.3	Produkte	87
4.4	Aufgaben	89

5 Aussagenlogik 97

5.1	Aussagen	97
5.2	Logische Operatoren	98
5.3	Darstellung boolescher Funktionen	106
5.4	Vereinfachung boolescher Ausdrücke	111
5.5	Logische Operatoren in C	119
5.6	Programmierbeispiele	120
5.6.1	Kugelspiel	120
5.6.2	Schaltung	121
5.7	Aufgaben	124

6 Elementare Datentypen und ihre Darstellung 129

6.1	Zahlendarstellungen	129
6.1.1	Dezimaldarstellung	131
6.1.2	Dualdarstellung	131
6.1.3	Oktalдарstellung	133
6.1.4	Hexadezimaldarstellung	134
6.2	Bits und Bytes	136
6.3	Skalare Datentypen in C	140
6.3.1	Ganze Zahlen	141
6.3.2	Aufzählungstypen	142
6.3.3	Gleitkommazahlen	142
6.3.4	Buchstaben	143
6.4	Bit-Operationen	147
6.5	Programmierbeispiele mit Zeichen, Zahlen und Bit-Operationen	150
6.5.1	Zeichensatz	150
6.5.2	Bit-Editor	152

6.6	Arrays und Zeichenketten	156
6.6.1	Arrays	156
6.6.2	Zeichenketten	161
6.7	Programmierbeispiele mit Arrays und Strings	167
6.7.1	Buchstaben zählen	167
6.7.2	Matrixdruck	170
6.8	Aufgaben	173

7 Modularisierung 181

7.1	Funktionen und Unterprogramme	181
7.2	Rekursion	188
7.3	Der Stack	196
7.4	Globale, lokale und statische Variablen	199
7.5	Die C-Runtime-Library	201
7.5.1	Mathematische Funktionen	201
7.5.2	Konvertierungs- und Klassifizierungsroutinen	204
7.5.3	Stringbearbeitung	206
7.5.4	Terminal I/O	208
7.5.5	Dateioperationen	213
7.5.6	Suchen und Sortieren	216
7.5.7	Variable Anzahl von Argumenten	216
7.5.8	Ausnahme- und Fehlerbehandlung	219
7.5.9	Assertions und Programmabbruch	223
7.5.10	Freispeicherverwaltung und Speicherfunktionen	225
7.5.11	Zeit- und Datum-Funktionen	225
7.5.12	Prozess-Steuerung	226
7.6	Beispiele	226
7.6.1	Das Damenproblem	226
7.6.2	Labyrinth	234
7.7	Aufgaben	240

8 Zeiger und Adressen 253

8.1	Zeigerarithmetik	257
8.2	Arrays und Zeiger	259
8.3	Funktionszeiger	262
8.4	Komplexe Variablendeklarationen	266
8.5	Aufgaben	268

9	C-Referenz (Teil 1)	271
9.1	Reservierte Wörter	271
9.2	Identifizier	271
9.3	Numerische Werte	272
9.4	Werte für Zeichen und Zeichenketten	273
9.5	Skalare Datentypen	274
9.6	Variablen	274
9.7	Arrays	279
9.8	Operatoren	279
9.8.1	Arithmetische Operatoren	284
9.8.2	Vergleichsoperatoren	285
9.8.3	Logische Operatoren	285
9.8.4	Bit-Operatoren	286
9.8.5	Zugriffsoperatoren	286
9.8.6	Auswertungsoperatoren	287
9.8.7	Datentyp-Operatoren	288
9.8.8	Ausdrücke und Zuweisungsoperatoren	289
9.9	Funktionen	293
9.10	Kontrollstrukturen	295
9.10.1	Alternativen	295
9.10.2	Sprungleisten	296
9.10.3	Schleifen	298
9.10.4	Sprunganweisungen	300
9.11	Der Preprozessor	302
9.11.1	Includes	302
9.11.2	Symbolische Konstanten	303
9.11.3	Makros	305
9.11.4	Bedingte Compilierung	306
9.12	Der Aufbau von Programmdateien	308
9.12.1	Header-Dateien	309
9.12.2	Quellcode-Dateien	310
9.13	Einige Coding-Standards	311
10	Kombinatorik	315
10.1	Kombinatorische Grundaufgaben	316
10.1.1	Permutationen mit Wiederholungen	316
10.1.2	Permutationen ohne Wiederholungen	317
10.1.3	Kombinationen ohne Wiederholungen	318

10.1.4	Kombinationen mit Wiederholungen	320
10.1.5	Zusammenfassung	322
10.2	Kombinatorische Algorithmen	324
10.2.1	Permutationen mit Wiederholungen	325
10.2.2	Kombinationen mit Wiederholungen	328
10.2.3	Kombinationen ohne Wiederholungen	330
10.2.4	Permutationen ohne Wiederholungen	332
10.3	Beispiele	335
10.3.1	Juwelenraub	335
10.3.2	Geldautomat	340
10.4	Aufgaben	345

11 Leistungsanalyse und -messung von Algorithmen 349

11.1	Leistungsanalyse	352
11.2	Leistungsmessung	364
11.2.1	Überdeckungsanalyse	366
11.2.2	Performance-Analyse	367
11.3	Mathematische Grundfunktionen	369
11.3.1	Floor und Ceiling	369
11.3.2	Potenzfunktionen	371
11.3.3	Exponentialfunktionen	372
11.3.4	Logarithmen	373
11.4	Laufzeitklassen	375
11.5	Beispiele	382

12 Sortierverfahren 395

12.1	Bubblesort	396
12.2	Selectionsort	398
12.3	Insertionsort	401
12.4	Shellsort	403
12.5	Quicksort	407
12.6	Heapsort	412
12.7	Leistungsanalyse	418
12.7.1	Bubblesort	420
12.7.2	Selectionsort	421
12.7.3	Insertionsort	423
12.7.4	Shellsort	424
12.7.5	Quicksort	425
12.7.6	Heapsort	427

12.8	Vergleich und Bewertung	428
12.9	Grenzen der Optimierung von Sortierverfahren	434
12.10	Aufgaben	439

13 Datenstrukturen 441

13.1	Datensequenz (struct)	444
13.2	Datenalternative (union)	451
13.3	Optimierung von Datenstrukturen	457
13.4	Zeiger und Datenstrukturen	462
13.5	Dynamische Datenstrukturen	465
13.6	Verkettete Datenstrukturen (Listen)	470
13.7	Die Freispeicherverwaltung	474
13.8	Abstrakte Datentypen	478
13.8.1	Der abstrakte Datentyp »Stack«	480
13.8.2	Der abstrakte Datentyp »Queue«	485
13.9	Aufgaben	490

14 C-Referenz (Teil 2) 497

14.1	Einfache Strukturen	497
14.2	Zusammengesetzte Strukturen	500
14.3	Zugriff auf Strukturen	502
14.3.1	Direkter Zugriff	502
14.3.2	Indirekter Zugriff	503
14.4	Unions	505
14.5	Datenstrukturen und Funktionen	506
14.6	Dynamische Datenstrukturen	509
14.7	Zeiger in Datenstrukturen	511
14.8	Typvereinbarungen	515
14.9	Bitfelder	516

15 Ausgewählte Datenstrukturen 519

15.1	Aufgabenstellung	519
15.2	Schnittstellenvereinbarung	521
15.3	Anwendungsprogramm	524
15.4	Listen	530
15.4.1	Grundbegriffe	530
15.4.2	Arrays oder Listen	532
15.4.3	Speicherstruktur	533

15.4.4	Implementierung	537
15.4.5	Test	542
15.5	Bäume	543
15.5.1	Grundbegriffe	543
15.5.2	Traversierung von Bäumen	547
15.5.3	Speicherstruktur	559
15.5.4	Implementierung	562
15.5.5	Test	569
15.6	Ausgeglichene Bäume	570
15.6.1	Grundbegriffe	571
15.6.2	Speicherstruktur	575
15.6.3	Implementierung	578
15.6.4	Test	592
15.7	Hashtabellen	593
15.7.1	Grundbegriffe	594
15.7.2	Speicherstruktur	596
15.7.3	Implementierung	599
15.7.4	Test	602
15.8	Vergleich und Bewertung	606
15.8.1	Speicherkomplexität	606
15.8.2	Laufzeitmessungen	607
15.9	Aufgaben	612

16 Elemente der Graphentheorie 613

16.1	Grundbegriffe	615
16.2	Darstellung von Graphen durch Datenstrukturen	621
16.3	Ausgewählte graphentheoretische Probleme	627
16.3.1	Existenz von Wegen	629
16.3.2	Kürzeste Wege	649
16.3.3	Minimal spannende Bäume	682
16.3.4	Hamiltonsche Wege	688

17 Projekt 2D-Grafikprogrammierung 701

18 C++-Referenz (Teil 1) 703

18.1	Schlüsselwörter	703
18.2	Operatoren	704
18.3	Kommentare	707
18.4	Datentypen, Datenstrukturen und Variablen	708

18.4.1	Automatische Typisierung von Aufzählungstypen	708
18.4.2	Automatische Typisierung von Strukturen	708
18.4.3	Vorwärtsverweise auf Strukturen	709
18.4.4	Der Datentyp bool	710
18.4.5	Wide Character	711
18.4.6	Const-Deklarationen	712
18.4.7	Typumwandlungen	713
18.4.8	Definition von Variablen	716
18.4.9	Referenzen	716
18.5	Funktionen und Operatoren	722
18.5.1	Funktionsdeklarationen und Prototypen	722
18.5.2	Default-Werte	722
18.5.3	Inline-Funktionen	724
18.5.4	Der Scope-Resolution-Operator	725
18.5.5	Überladen von Funktionen	726
18.5.6	Überladen von Operatoren	728
18.5.7	Einbindung von C-Funktionen in C++-Programme	730
18.6	Namensräume	732
18.6.1	Erstellung von Namensräumen	733
18.6.2	Verwendung von Namensräumen	736
18.6.3	Der Namensraum std	740

19 Objektorientierte Programmierung 743

20 Klassen in C++ 759

20.1	Aufbau von Klassen	759
20.1.1	Daten-Member	761
20.1.2	Funktions-Member	764
20.1.3	Konstruktoren und Destruktoren	772
20.2	Instantiierung von Klassen	777
20.2.1	Automatische Instantiierung	779
20.2.2	Statische Instantiierung	781
20.2.3	Dynamische Instantiierung	782
20.2.4	Instantiierung von Arrays	784
20.3	Friends	785
20.4	Operatoren auf Klassen	788
20.5	Ein- und Ausgabe in C++	792
20.5.1	Bildschirmausgabe	793

20.5.2	Tastatureingabe	796
20.5.3	Dateioperationen	797
20.6	Der this-Pointer	799
20.7	Beispiele	800
20.7.1	Menge	800
20.7.2	Bingo	811
20.8	Aufgaben	818

21 Vererbung in C++ 823

21.1	Geschützte Member	829
21.2	Einfache Vererbung	832
21.3	Mehrfache Vererbung	833
21.4	Instantiierung abgeleiteter Klassen	833
21.5	Erweiterung abgeleiteter Klassen	835
21.6	Überladen von Funktionen der Basisklasse	836
21.7	Virtuelle Member-Funktionen	841
21.8	Rein virtuelle Member-Funktionen	844
21.9	Statische Member	845
21.10	Beispiele	850
21.10.1	Würfelspiel	850
21.10.2	Partnervermittlung	871

22 C++-Referenz (Teil 2) 893

22.1	Klassen und Instanzen	893
22.2	Member	894
22.2.1	Daten-Member	895
22.2.2	Funktions-Member	896
22.2.3	Konstante Member	899
22.2.4	Statische Member	900
22.2.5	Operatoren	902
22.3	Zugriff auf Member	903
22.3.1	Zugriff von außen	904
22.3.2	Zugriff von innen	907
22.3.3	Der this-Pointer	910
22.3.4	Zugriff durch Friends	911
22.4	Vererbung	913
22.4.1	Einfachvererbung	913
22.4.2	Mehrfachvererbung	917
22.4.3	Virtuelle Funktionen	922

- 22.4.4 Virtuelle Destruktoren 925
- 22.4.5 Rein virtuelle Funktionen 926
- 22.4.6 Dynamische Typüberprüfungen 928
- 22.4.7 Dynamische Typumwandlung 929
- 22.5 Zugriffsschutz und Vererbung 935
 - 22.5.1 Geschützte Member 936
 - 22.5.2 Zugriff auf die Basisklasse 937
 - 22.5.3 Modifikation von Zugriffsrechten 941
- 22.6 Der Lebenszyklus von Objekten 942
 - 22.6.1 Konstruktion von Objekten 945
 - 22.6.2 Destruktion von Objekten 948
 - 22.6.3 Kopieren von Objekten 949
 - 22.6.4 Instantiierung von Objekten 953
 - 22.6.5 Implizite und explizite Verwendung von
Konstruktoren 956
 - 22.6.6 Initialisierung eingelagerter Objekte 957
 - 22.6.7 Initialisierung von Basisklassen 960
 - 22.6.8 Initialisierung virtueller Basisklassen 962
 - 22.6.9 Instantiierungsregeln 964
- 22.7 Pointer to Member 966
- 22.8 Generische Klassen (Templates) 970
- 22.9 Ausnahmefallbehandlung 976

23 Die C++-Standard-Library 983

- 23.1 Iteratoren 984
- 23.2 Strings (string) 987
 - 23.2.1 Konstruktion 988
 - 23.2.2 Ein-/Ausgabe 989
 - 23.2.3 Zugriff 989
 - 23.2.4 Manipulation 992
 - 23.2.5 Vergleich 997
 - 23.2.6 Suchen 998
 - 23.2.7 Speichermanagement 999
- 23.3 Bitsets (bitset) 1002
 - 23.3.1 Konstruktion 1002
 - 23.3.2 Zugriff 1003
 - 23.3.3 Manipulation 1004
- 23.4 Dynamische Arrays (vector) 1005
 - 23.4.1 Konstruktion 1006
 - 23.4.2 Zugriff 1007

23.4.3	Iteratoren	1008
23.4.4	Manipulation	1009
23.4.5	Speichermanagement	1012
23.5	Beidseitige Warteschlangen (deque)	1012
23.6	Listen (list)	1014
23.6.1	Konstruktion	1014
23.6.2	Zugriff	1015
23.6.3	Iteratoren	1015
23.6.4	Manipulation	1017
23.6.5	Speichermanagement	1027
23.7	Stacks (stack)	1027
23.8	Warteschlangen (queue)	1030
23.9	Prioritätswarteschlangen (priority_queue)	1032
23.10	Geordnete Paare (pair)	1038
23.11	Mengen (set und multiset)	1040
23.11.1	Konstruktion	1041
23.11.2	Zugriff	1042
23.11.3	Manipulation	1044
23.12	Relationen (map und multimap)	1045
23.12.1	Konstruktion	1046
23.12.2	Zugriff	1049
23.12.3	Manipulation	1049
23.13	Algorithmen der Standard-Library	1049
23.13.1	Iterieren	1052
23.13.2	Suchen und Finden	1053
23.13.3	Vergleichen	1055
23.13.4	Zählen	1056
23.13.5	Kopieren	1057
23.13.6	Tauschen	1058
23.13.7	Ersetzen	1059
23.13.8	Wertzuweisung	1061
23.13.9	Entfernen von Elementen	1062
23.13.10	Reorganisation	1063
23.13.11	Sortieren	1064
23.13.12	Binäre Suche	1065
23.13.13	Mischen	1066
23.13.14	Mengenoperationen	1067
23.13.15	Heap-Algorithmen	1069
23.13.16	Minima und Maxima	1070
23.13.17	Lexikografische Ordnung und Permutationen	1071
23.14	Vererbung und virtuelle Funktionen in Containern	1073

24 Projekt: 3D-Grafikprogrammierung	1077
--	-------------

Anhang	1079
---------------------	-------------

Lösungen	1083
Kapitel 1	1085
Kapitel 3	1103
Kapitel 4	1123
Kapitel 5	1155
Kapitel 6	1169
Kapitel 7	1195
Kapitel 8	1245
Kapitel 10.....	1253
Kapitel 12.....	1273
Kapitel 13.....	1277
Kapitel 15.....	1305
Kapitel 20.....	1315
 Index	 1333

1 Einige Grundbegriffe

Einem Kochbuch entnehmen wir das folgende Rezept zur Herstellung eines sogenannten Pharisäers:

Zutaten:

1/2 l heißer Kaffee
1/4 l Sahne
2 Essl. Zucker
4 Schnapsgläser 54%iger Rum (8 cl)

Zubereitung:

Den Kaffee aufbrühen und warmhalten. 4 Tassen mit heißem Wasser vorwärmen. Inzwischen die Sahne steif schlagen. Das Wasser aus den Tassen gießen, die Tassen abtrocknen und in jede Tasse 1–2 Teelöffel Zucker geben. Je 1 Schnapsglas Rum darüber gießen und mit dem Kaffee auffüllen. Die Schlag-
sahne als Haube auf jede Tasse Pharisäer setzen.

Das Rezept gliedert sich in zwei Teile. Im ersten Teil werden die erforderlichen Zutaten genannt, und im zweiten Teil wird ein Verfahren beschrieben, nach dem man aus den Zutaten das gewünschte Getränk herstellen kann. Die beiden Teile sind wesentlich verschieden und gehören doch untrennbar zusammen. Ohne Zutaten ist die Zubereitung nicht durchführbar, und ohne Zubereitung bleiben die Zutaten ungenießbar. Zu beachten ist auch, dass sich der Autor bei der Erstellung des Rezepts einer bestimmten Fachsprache (Essl., cl, Sahne steif schlagen, aufbrühen) bedient. Ohne diese Fachsprache wäre die Anleitung wahrscheinlich weit-schweifiger, umständlicher und vielleicht sogar missverständlich. Die Verwendung einer Fachsprache setzt allerdings voraus, dass sich Autor und Leser des Rezepts zuvor (ausgesprochen oder unausgesprochen) auf eine gemeinsame Terminologie verständigt haben.

Wir übertragen dieses Beispiel in unsere Welt – die Welt der Datenverarbeitung:

- ▶ Die Zutaten für das Rezept sind die Daten bzw. **Datenstrukturen**, die wir verarbeiten wollen.
- ▶ Die Zubereitungsvorschrift ist ein **Algorithmus**¹, der festlegt, wie die Daten zu verarbeiten sind.

1. Dieser Begriff geht zurück auf Abu Jafar Muhammad Ibn Musa Al-Khwarizmi, der als Bibliothekar des Kalifen von Bagdad um 825 ein Rechenbuch verfasste und dessen Name in der lateinischen Übersetzung von 1200 als »Algorithmus« angegeben wurde.

- ▶ Das Rezept ist ein **Programm**, das alle Datenstrukturen (Zutaten) und Algorithmen (Zubereitungsvorschriften) zum Lösen der gestellten Aufgabe (Erstellen des Gerichts) enthält.
- ▶ Die gemeinsame Terminologie, in der sich Autor und Leser des Rezepts verständigen, ist eine **Programmiersprache**, in der das Programm geschrieben ist. Die Programmiersprache muss dabei in der Lage sein, alle bezüglich der Zutaten und der Zubereitung bedeutsamen Informationen zweifelsfrei zu übermitteln.
- ▶ Die Küche ist die technische Infrastruktur zur Umsetzung von Rezepten in schmackhafte Gerichte und ist vergleichbar mit einem **Computer**, seinem **Betriebssystem** und den benötigten **Entwicklungswerkzeugen**.
- ▶ Der Koch übersetzt das Rezept in einzelne Arbeitsschritte in der Küche. Üblicherweise geht ein Koch in zwei Schritten vor. Im ersten Schritt bereitet er die Zutaten einzeln und unabhängig voneinander vor (z. B. Kartoffeln kochen), um die Einzelteile dann in einem zweiten Schritt zusammenzufügen und abzuschmecken. In der Datenverarbeitung sprechen wir in diesem Zusammenhang von **Compiler** und **Linker**.
- ▶ Das fertige Gericht ist das **lauffähige Programm**, das vom Benutzer (Esser) angewandt (verzehrt) werden kann.

Nur, welche Rolle spielen **wir** in diesem Szenario? Sollte für uns kein Platz vorgesehen sein? Wir suchen uns die interessantesten Aufgaben aus:

- ▶ Wir sind Autoren, die sich neue, schmackhafte Gerichte für unterschiedliche Anlässe ausdenken und Rezepte bzw. Kochbücher mit den besten Kreationen veröffentlichen.
- ▶ Gelegentlich probieren wir auch einen Pharisäer, um uns an unseren eigenen Schöpfungen zu berauschen und um festzustellen, ob die Speise gelungen ist.

Was müssen wir lernen, um unsere Rolle ausfüllen zu können?

- ▶ Wir müssen die Sprache beherrschen, in der Rezepte formuliert werden.
- ▶ Wir müssen einen Überblick über die üblicherweise verwendeten Zutaten, deren Eigenschaften und Zubereitungsmöglichkeiten haben.
- ▶ Wir müssen einen Vorrat an Standard-Zubereitungsverfahren bzw. Rezepten abrufbereit im Kopf haben.
- ▶ Wir müssen wissen, welche Zutaten oder Verfahren miteinander harmonieren und welche nicht.
- ▶ Wir müssen wissen, was in einer Küche üblicherweise an Hilfsmitteln vorhanden ist und wie bzw. wozu diese Hilfsmittel verwendet werden.

- ▶ Bei anspruchsvolleren Gerichten müssen wir wissen, in welcher Reihenfolge und mit welchem Timing die Einzelteile zuzubereiten sind und wie die einzelnen Aufgaben verteilt werden müssen, damit alles zeitgleich serviert werden kann.
- ▶ Wir müssen auch wissen, worauf ein potentieller, späterer Esser Wert legt und worauf nicht. Dies ist besonders wichtig, wenn wir Rezepte für einen ganz besonderen Anlass erstellen.

Letztlich wollen wir komplette Festmenüs und deren Speisefolge komponieren und benötigen dazu eine Mischung aus Phantasie, Kreativität, logischer Strenge, Ausdauer und Fleiß, wie sie auch ein guter Komponist oder Architekt benötigt.

Zurück zu den Grundbegriffen der Informatik. Wir haben informell die Begriffe

- ▶ Datenstruktur,
- ▶ Algorithmus und
- ▶ Programm

eingeführt und dabei bereits erkannt, dass diese Begriffe untrennbar zusammengehören und eigentlich nur unterschiedliche Facetten ein und desselben Themenkomplexes sind.

- ▶ Algorithmen arbeiten auf Datenstrukturen. Algorithmen ohne Datenstrukturen sind leere Formalismen.
- ▶ Datenstrukturen benötigen Algorithmen, die auf ihnen operieren und sie damit zum »Leben« erwecken.
- ▶ Programme realisieren Datenstrukturen und Algorithmen. Algorithmen und Datenstrukturen sind zwar ohne Programme denkbar, aber viele Datenstrukturen und Algorithmen wären ohne Programmierung allenfalls von akademischem Interesse.

In einem ersten Wurf versuchen wir, die Begriffe »Algorithmus«, »Datenstruktur« und »Programm« einigermaßen exakt zu erfassen.

1.1 Algorithmus

Um unsere noch sehr vage Vorstellung von einem Algorithmus zu präzisieren, starten wir mit einer Definition:

Ein **Algorithmus** ist eine endliche Menge von genau beschriebenen Anweisungen, die unter Benutzung von vorgegebenen Anfangsdaten in einer genau festgelegten Reihenfolge auszuführen sind, um die Lösung eines Problems in endlich vielen Schritten zu ermitteln.

Bei dem Begriff »Algorithmus« denkt man heute sofort an »Programmierung«. Das war nicht immer so. In der Tat gab es Algorithmen schon lange, bevor man auch nur entfernt an Programmierung dachte. Bereits im antiken Griechenland wurden Algorithmen zur Lösung mathematischer Probleme formuliert, so zum Beispiel der Euklidische Algorithmus zur Bestimmung des größten gemeinsamen Teilers zweier Zahlen oder das sogenannte Sieb des Eratosthenes zur Bestimmung aller Primzahlen unterhalb einer vorgegebenen Schranke.²

Ihnen ist der Algorithmus zur schrittweisen Berechnung des Quotienten zweier Zahlen schon lange vertraut. Um beispielsweise 84 durch 16 zu dividieren, geht man wie folgt vor:

$$84 : 16 = 5,25$$

$$\begin{array}{r} 80 \\ \hline 40 \\ 32 \\ \hline 80 \\ 80 \\ \hline 0 \end{array}$$

Wenn wir versuchen, dieses Verfahren im Detail zu beschreiben, finden wir alle in der obigen Definition genannten Elemente wieder:

Problem:

Berechne den Quotienten zweier natürlicher Zahlen!

Anfangsdaten:

z = Zähler ($z \geq 0$),

n = Nenner ($n > 0$) und

a = Anzahl der zu berechnenden Nachkommastellen.³

Anweisungen:

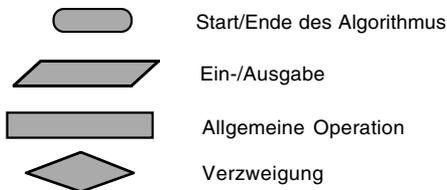
1. Bestimme die größte ganze Zahl x mit $nx \leq z$! Dies ist der Vorkomma-Anteil der gesuchten Zahl.
2. Zur Bestimmung der Nachkommastellen fahre wie folgt fort:
 - 2.1 Sind noch Nachkommastellen zu berechnen (d.h. $a > 0$)? Wenn nein, dann beende das Verfahren!
 - 2.2 Setze $z = 10(z - nx)$!
 - 2.3 Ist $z = 0$, so beende das Verfahren!

-
2. Eukleides von Alexandria (um 300 vor Chr.) und Eratosthenes von Kyrene (um 200 vor Chr.).
 3. Anfänglich ist a die Anzahl der zu berechnenden Nachkommastellen. Im Verfahren verwenden wir a als die Anzahl der **nach** zu berechnenden Nachkommastellen. Wir werden den Wert von a in jedem Verfahrensschritt herunterzählen, bis $a = 0$ ist und keine Nachkommastellen mehr zu berechnen sind.

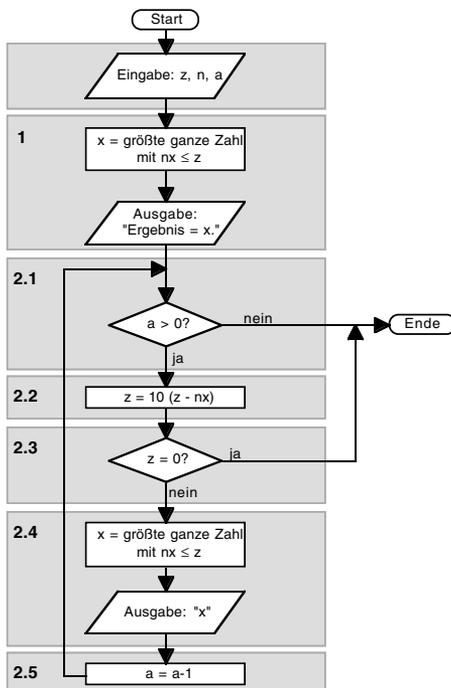
2.4 Bestimme die größte ganze Zahl x mit $nx \leq z$! Dies ist die nächste Ziffer.

2.5 Jetzt ist eine Ziffer weniger zu bestimmen. Vermindere also den Wert von a um 1 und fahre anschließend bei 2.1 fort!

Die einzelnen Anweisungen und ihre Abfolge können wir uns durch ein sogenanntes **Flussdiagramm** veranschaulichen. In einem solchen Diagramm werden alle beim Ablauf des Algorithmus möglicherweise vorkommenden Wege unter Verwendung bestimmter Symbole grafisch beschrieben. Die dabei zulässigen Symbole sind in einer Norm (DIN 66001) festgelegt. Von den zahlreichen in der Norm festgelegten Symbolen wollen wir hier nur einige einführen:



Mit diesen Symbolen können wir den zuvor nur sprachlich beschriebenen Algorithmus auch grafisch darstellen, wenn wir zusätzlich die Abfolge der einzelnen Operationen durch Richtungspfeile kennzeichnen.



Index

2D-Grafikprogrammierung 701

A

Abstrakte Klasse 746, 844, 927
Abstrakter Datentyp 479
 Queue 485
 Stack 480
adjacent_find 1054
Adjazenzliste 623
Adjazenzmatrix 623
Adressbus 137
Adresse 137, 254
Aggregation 753
Aktualparameter 182
Algorithmus 23
 Algorithmus von Dijkstra 661
 Algorithmus von Floyd 652
 Algorithmus von Ford 672
 Algorithmus von Kruskal 682
 Algorithmus von Warshall 637
Alignment 459
and 703
and_eq 703
Anweisung
 break 59, 297, 300
 catch-Anweisung 978
 continue 59, 300
 do...while 300
 extern 184
 for 56, 298
 goto 300
 if 53
 if...else 295
 return 184, 293
 struct 444, 497
 switch 297
 throw-Anweisung 978
 try-Anweisung 978
 union 451, 505
 while 299
Argument 182
Array 156, 279
 eindimensionaler Array 156
 mehrdimensionaler Array 160, 279

Array (Forts.)
 zweidimensionaler Array 158
Arrays und Zeiger 259
ASCII-Zeichensatz 143
Assertions 223
Assoziativgesetz 104
Attribut 744, 759
Aufzählungstyp 142
 automatische Typisierung 708
Ausdruck
 arithmetischer Ausdruck 290
 logischer Ausdruck 290
 L-Wert 291
 relationaler Ausdruck 290
 R-Wert 291
 Zuweisungs-Ausdruck 290
Ausgabe
 Ausgabe in C++ 793
 Ausgabe in eine Datei 213
 Ausgabe von ganzen Zahlen 51
 Ausgabe von Gleitkommazahlen 51
 Ausgabe von Text 50
 Bildschirmausgabe 208
Ausgeglichene Bäume 570
Ausnahmefallbehandlung 219, 976
Aussage 97
Aussagenlogik 97
Automatische Objekte 943

B

Basisklasse
 Zugriff auf die Basisklasse 937
 Zugriffsspezifikation private 940
 Zugriffsspezifikation protected 939
 Zugriffsspezifikation public 937
Baum 543, 619
 absteigend sortierter Baum 546
 aufsteigend sortierter Baum 546
 ausgeglichener Baum 573
 AVL-Baum 573, 578
 Balance 572
 Baum-Modul 559
 Binärbaum 545
 Blätter 545

- Baum (Forts.)
 - Inorder-Traversierung* 550
 - Knoten* 545
 - Level* 545
 - Level-Order-Traversierung* 557
 - linker Teilbaum* 546
 - Nachfolger* 544
 - Postorder-Traversierung* 552
 - Preorder-Traversierung* 549
 - rechter Teilbaum* 546
 - Teilbaum* 545
 - Tiefe* 545
 - Traversierung* 547
 - vollständiger Baum* 571
 - Wurzel* 544
- Bedingte Compilierung 306
- Befehlsausführung
 - bedingte Befehlsausführung* 53
 - wiederholte Befehlsausführung* 54
- Beidseitige Warteschlangen 1012
- Beispiel
 - Algorithmus von Dijkstra* 666
 - Algorithmus von Floyd* 656
 - Algorithmus von Ford* 674, 679
 - Algorithmus von Kruskal* 684
 - Algorithmus von Warshall* 637
 - Ausnahmebehandlung* 220
 - AVL-Baum-Modul* 575
 - Baum-Modul* 559
 - Bingo* 811
 - Bit-Editor* 152
 - Bubblesort* 397
 - Buchstaben zählen* 167
 - Damenproblem* 226
 - Darlehen* 74
 - Dateioperationen* 214
 - Division ganzer Zahlen* 61
 - Fakultäten* 88, 188
 - Geldautomat* 340
 - Hamiltonsche Wege* 690
 - Hashtabellen-Modul* 596
 - Heapsort* 416
 - Hofstadter-Funktion* 190
 - Inorder-Traversierung* 550
 - Insertionsort* 402
 - Juwelenraub* 335
 - Kombinationen mit Wiederholungen* 328
 - Kombinationen ohne Wiederholungen* 330
- Beispiel (Forts.)
 - Kugelspiel* 62, 120
 - Labyrinth* 234
 - Legotreppe* 81
 - Level-Order-Traversierung* 558
 - Listen-Modul* 533
 - Matrixdruck* 170
 - Menge* 800
 - Näherungslösung Travelling Salesman* 697
 - Partnervermittlung* 871
 - Permutationen* 193
 - Permutationen mit Wiederholungen* 325
 - Permutationen ohne Wiederholungen* 332
 - Postorder-Traversierung* 552
 - Preorder-Traversierung* 549
 - Queue* 486
 - Quicksort* 411
 - Schaltung* 121
 - Selectionsort* 400
 - Shellsort* 405
 - Stack* 481
 - Summation von Zahlen* 66
 - Travelling Salesman Problem* 693
 - Traversierung von Graphen* 631
 - Variable Anzahl von Argumenten* 216
 - Vereinigungssuche* 645
 - Würfelspiel* 850
 - Zeichensatz* 150
 - Zusammenhangskomponenten* 645
- Bildschirmausgabe 793
- Binärbaum 545
- binary_search 1066
- Binomialkoeffizient 319
- Bit 136
 - invertieren* 149
 - löschen* 149
 - prüfen* 149
 - setzen* 149
- bitand 703
- Bitfelder 516
- Bit-Operationen 147
- bitor 703
- bitset 1002
- Bitsets 1002
- Block 274
 - Schachtelung von Blöcken* 275
 - Variablen in Blöcken* 275

bool 703, 710
 Boolesche Funktion 100
 break-Anweisung 59, 300
 Bubblesort 396
 Byte 136

C

Callback 262
 calloc 466, 510
 case-Label 297
 catch 703, 978
 class 703, 759, 893
 Coding-Standards 311
 Compiler 41
 Compileschalter 306
 compl 703
 const_cast 703
 const-Deklaration 278
 continue-Anweisung 59, 300
 copy 1058
 copy_backward 1058
 count 1056
 count_if 1056
 Crosscast 932
 C-Runtime-Library 201

D

Darstellung boolescher Funktionen 106
 Darstellungssatz für boolesche Funktionen 109
 Dateioperationen 213
 Dateioperationen in C++ 797
 Datenabstraktion 442
 Datenbus 137
 Daten-Member 759, 761, 895
 Datenstruktur 27, 441
 anlegen 445, 498
 Arrays in Datenstrukturen 501
 automatische Typisierung 708
 Bitfelder 516
 Datenstrukturen und Funktionen 506
 Datenstrukturen und Zeiger 511
 direkter Zugriff 503
 dynamische Datenstrukturen 465, 509
 indirekter Zugriff 503
 Initialisierung 445, 498

Datenstruktur (Forts.)
 Initialisierung zusammengesetzter Datenstrukturen 500
 Liste 470, 513
 lokale Datenstruktur 499
 Optimierung 457
 struct 497
 Typvereinbarungen 515
 unbenannte Datenstruktur 499
 union 505
 Vorwärtsverweise 709
 Zeiger auf Datenstrukturen 503
 Zeiger und Datenstrukturen 462
 zusammengesetzte Datenstruktur 500
 Zuweisung von Datenstrukturen 502
 Datentyp
 char 141
 double 142
 enum 142
 float 142
 int 141
 long 141
 long double 142
 short 141
 signed 141
 skalarer Datentyp 140, 274
 unsigned 141
 void 188
 Datum-Funktionen 225
 De Morgan'sches Gesetz 104
 Debugger 42
 Default-Werte 722
 Default-Konstruktor 780
 default-Label 297
 delete 703, 782
 delete-Operator 782
 deque 1012
 Dereferenzierung 254
 Destruktion 780
 Destruktor 479, 772, 948
 Dezimaldarstellung 131
 Disjunktive Normalform 109
 Diskriminante 452, 505
 Distributivgesetz 104
 Division mit Rest 130
 do...while-Anweisung 300
 Downcasts 932
 Dualdarstellung 131
 Dynamic Cast 716

dynamic_cast 703, 933
 Dynamische Arrays 1005
 Dynamische Datenstrukturen 465, 509
 Dynamische Objekte 944
 Dynamische Typüberprüfungen 928

E

Editor 40
 Einerkomplement 138
 Einfachvererbung 913
 Eingabe
 Eingabe aus einer Datei 213
 Eingabe in C++ 796
 Eingabe von ganzen Zahlen 52
 Eingabe von Zeichenketten 162
 Tastatureingabe 208
 Eingabeparameter 182
 equal 1055
 equal_range 1066
 Erweiterung abgeleiteter Klassen 835
 Escape-Sequenz 145, 273
 Eulerscher Weg 614
 Exception-Handling 976
 explicit 703
 Explizite Instanziierung 956
 export 703
 extern 730
 extern-Anweisung 184

F

Fakultät (n!) 88, 188
 false 703
 Fehlerbehandlung 219
 fill 1061
 fill_n 1061
 find 1054
 find_end 1054
 find_first_of 1054
 find_if 1054
 Flussdiagramm 25
 Folgen 73
 for_each 1053
 for-Anweisung 56, 298
 Formalparameter 182
 Format-Anweisungen
 Ausgabe 209
 Eingabe 211

free 466, 510
 Freispeicherverwaltung 225, 467, 474
 friend 703
 Friends 785, 911
 Function Name Encoding 727
 Funktion 182, 293
 Default-Werte 722
 Function Name Encoding 727
 Funktionskörper 184
 Funktionsprototyp 183, 184, 294, 722
 globale Funktion 293
 Implementierung 183
 Inline-Funktion 724
 Parameter 186, 293
 Parametersignatur 727
 Schnittstelle 184, 293
 Schnittstellenvariablen 186
 statische Funktion 294
 Überladen von Funktionen 726, 836
 Funktions-Member 759, 764, 896
 Funktionsprototyp 184, 722
 Funktionszeiger 262
 fwprintf 712
 fwscanf 712

G

Generalisierung 749
 generate 1061
 generate_n 1061
 Generische Klasse 970
 Geordnete Paare 1038
 Geschützte Member 936
 getch 712
 Gigabyte 140
 Gleitkommazahlen 44
 Globale Variablen 199
 goto-Anweisung 301
 Graph 616
 Adjazenzliste 623
 Adjazenzmatrix 623
 Algorithmus von Dijkstra 661
 Algorithmus von Floyd 652
 Algorithmus von Ford 672
 Anfangsknoten einer Kante 616
 Anfangsknoten eines Weges 617
 Baum 619
 bewerteter Graph 620
 Bewertungsfunktion 620

Graph (Forts.)
Endknoten einer Kante 616
Endknoten eines Weges 617
Existenz von Wegen 629
gerichteter Graph 616
geschlossener Weg 617
Hamiltonscher Weg 689
Inzidenzmatrix 625
Kante 616
Kantentabelle 626
Kantenzug 617
Knoten 616
Kostenfunktion 620
Kosten-Wege-Matrix 623
Kreis 617
kürzeste Wege 649
Länge eines Weges 617
minimal spannender Baum 682
Pfad in einem Graphen 617
Schleife 617
schleifenfreier Weg 617
schwach zusammenhängender Graph 619
Spannbaum 682
stark zusammenhängender Graph 619
symmetrischer Graph 616
Travelling Salesman Problem 692
Traversierung von Graphen 630
ungerichteter Graph 616
unzerlegbarer Graph 618
Verbindbarkeit von Knoten 618
Verfahren von Warshall 637
Weg in einem Graphen 617
Wegematrix 634
Wurzelbaum 620
zusammenhängender Graph 619
Zusammenhangskomponenten 641
 Graphentheorie 613

H

Hamiltonscher Weg 689
 Hashing 593
 Hashtabelle 594
 Hashfunktion 595
 Kollision 595
 Synonymkette 595
 Hauptprogramm 43, 182
 Header-Datei 41, 186, 200, 294, 309
 Heap 413, 467

Heap-Bedingung 413
 Heapsort 412
 Hexadezimaldarstellung 134

I

Idempotenzgesetz 104
 Identifier 271
 if-Anweisung 53
 Implementierung 35, 37
 Implizite Instantiierung 956
 Include-Direktive 302
 includes 1068
 Indirektzugriff 254
 Initialisierung
 eingelagerter Objekte 957
 virtueller Basisklassen 962
 von Basisklassen 960
 inline 703
 Inline-Funktion 724
 Inorder-Traversierung 550
 inplace_merge 1066
 Insertionsort 401
 Instantiierbare Klassen 746
 Instantiierung 747, 953
 automatische Instantiierung 778, 779
 dynamische Instantiierung 778, 782
 Instantiierung abgeleiteter Klassen 833
 Instantiierung von Arrays 784
 Instantiierungsregeln 964
 statische Instantiierung 778, 781
 von Klassen 777
 Instanz 747, 893
 Inzidenzmatrix 625
 iter_swap 1058
 Iteratoren 984

K

Kantentabelle 626
 Karnaugh-Veitch-Diagramm 113
 Keyword 271
 Kilobyte 140
 Klasse 746, 759, 893
 abgeleitete Klasse 913
 abstrakte Klasse 844
 Aufbau 759
 Basisklasse 913
 Instantiierung 777

Klasse (Forts.)
Vererbung 823
virtuelle Basisklasse 920
Zugriff von außen 904
Zugriff von innen 907
Klassenhierarchie 750
Klassifizierungsfunktionen 204
Kombinationen mit Wiederholungen 320
Kombinationen ohne Wiederholungen 318
Kombinatorik 315
Kombinatorische Algorithmen 324
Kommentar 49, 707
Kommutativgesetz 104
Komplementgesetz 104
Konfigurationsmanagement 36
Königsberger Brückenproblem 613
Konstante 278, 712
Konstante Member 899
Konstruktor 479, 772, 945
Default-Konstruktor 780
Kontrollabstraktion 442
Kontrollstrukturen 295
Konvertierungsfunktionen 204
Kosten-Wege-Matrix 623
Kürzeste Wege 649

L

Label 301
Laufzeitanalyse
Block 361
Fallunterscheidung 357
Schleife 355
Laufzeitfunktion 369
Laufzeitklassen 375
Laufzeitkomplexität 377
exponentielle 381
logarithmische 379
polynomiale 379
Laufzeitsystem 197
Leistungsanalyse 352
Bubblesort 420
Heapsort 427
Insertionsort 423
Quicksort 425
Selectionsort 421
Shellsort 424
Leistungsmessung 364

Level-Order-Traversierung 557
lexicographical_compare 1072
LIFO-Prinzip 197
Linker 42
list 1014
Liste 471, 513, 530, 1014
doppelt verkettete Liste 531
einfach verkettete Liste 531
Listenanfang 530
Listenende 530
logische Sicht 531
Nachfolger 530
physikalische Sicht 531
Vergleich mit Array 532
Vorgänger 530
Listenanker 471
Lokale Variablen 199
longjmp 977
lower_bound 1066
L-Wert 291

M

main 43, 293
make_heap 1070
Makros 305
malloc 466, 509
map 1045
Mathematische Funktionen 201
Mathematische Grundfunktionen 369
ceiling 369
Exponentialfunktionen 372
floor 369
Logarithmen 373
Potenzfunktionen 371
Matrizenprodukt 635
max 1070
max_element 1070
Megabyte 140
Mehrdeutige Vererbung 752, 917
Mehrfachvererbung 751, 917
Member 759, 894
Daten-Member 761, 895
Funktions-Member 764, 896
geschützte Member 760, 829, 936
öffentliche Member 760
private Member 760
rein virtuelle Member-Funktion 844, 926
statische Member 845, 900

Member (Forts.)
virtuelle Member-Funktion 841, 922
Zugriff auf Member 903
Mengen 1040
merge 1066
Message-Passing 755
Methode 744, 759
min 1070
min_element 1070
mismatch 1055
Modifikation von Zugriffsrechten 941
Modularisierung 181
multimap 1045
multiset 1040
mutable 703, 900

N

Nachricht 755
Namensraum std 740
Namensräume 732
namespace 703, 733
new 703, 782
new-Operator 782
next_permutation 1073
Nil-Zeiger 471
not 703
not_eq 703
nth_element 1065
Null-Zeiger 471

O

Objectfile 42
Objekt 744
Objektorientierte Programmierung 743
abstrakte Klasse 746
Aggregation 753
Attribut 744
dynamisches Binden 756
Generalisierung 749
instanzierbare Klasse 746
Instantiierung 747
Instanz 747
Klasse 746
Klassenhierarchie 750
mehrdeutige Vererbung 752
Mehrfachvererbung 751
Message-Passing 755

Objektorientierte Programmierung
(Forts.)
Methode 744
Nachricht 755
persistente Attribute 745
Polymorphismus 756
Relation 754
Spezialisierung 749
transiente Attribute 745
Vererbung 748
wiederholte Vererbung 752
Oktaldarstellung 133
Operator 279, 704, 902
Adressoperator (&) 254
Äquivalenz 103
arithmetische Operatoren 47, 284
Assoziativität 280
Auswertungsoperatoren 287
Bit-Operatoren 147, 286
bitshift links (<<) 149
bitshift rechts (>>) 149
bitweises entweder oder (^) 149
bitweises Komplement (~) 148
bitweises oder (|) 148
bitweises und (&) 148
Cast-Operator 466
Class-Member-Zugriff 704
Datentyp-Operatoren 288
Dekrement-Operator 293
delete 704
delete-Operator 782
Dereferenzierungsoperator ()* 254, 503
Globalzugriff 704
Implikation 105
Indirektzugriff (->) 463, 504, 904
Infix-Notation 280
Inkrement-Operator 293
logische Operatoren 98, 285
new 704
new-Operator 782
nicht-Operator 100
nicht-Operator (!) 119
oder-Operator 101
oder-Operator (||) 119
Operatoren auf Klassen 788
Operatoren und Ausdrücke 289
Pointer to Member 966
Pointer to Member ()* 704
Pointer to Member (->)* 704

Operator (Forts.)

Postfix-Notation 280
Präfix-Notation 280
Priorität 102, 280
Scope Resolution 725
sizeof-Operator 509
Stelligkeit 280
Strukturzugriff (.) 445, 503
Überladen von Operatoren 728, 788
und-Operator 101
und-Operator (&&) 119
Vergleichsoperatoren 49, 285
Zugriff (.) 762
Zugriffsoperator (.) 904
Zugriffsoperatoren 286
Zuweisungsoperator (=) 46
Zuweisungsoperatoren 289
operator 703
or 703
or_eq 703

P

pair 1038
Parameter 182
Parametersignatur 726
partial 1065
partial_sort 1065
partial_sort_copy 1065
partition 1064
Performance-Analyse 367
Permutationen mit Wiederholungen 316
Permutationen ohne Wiederholungen 317
Pointer 254
Pointer to Member 966
Polymorphe Klasse 928
Polymorphismus 756
pop_heap 1069
Postorder-Traversierung 552
Preorder-Traversierung 549
Preprozessor 302
Preprozessor-Direktiven 302
prev_permutation 1073
Prioritätswarteschlangen 1032
priority_queue 1032
private 703, 894, 904, 936, 940
Produkte 87
Programm 29

Programmabbruch 223
Programmcode 43
Programmdateien 308
Programmiersprache 29
 maschinenorientierte 30
 spezielle 30
 universelle 30
Programmrahmen 43
Projektplan 36
Projektplanung 36
protected 703, 894, 904, 936, 939
Prozessor 137
Prozesssteuerung 226
public 703, 894, 936, 937
push_heap 1069
putwchar 712

Q

Qualitätssicherung 37
Quellcode-Datei 41, 310
Queue 485, 558
 get 486
 put 486
queue 1030
Quicksort 407

R

random_shuffle 1064
Realisierung 35
realloc 466, 511
Referenz 717
 Initialisierung von Referenzen 720
 konstante Referenz 720
 Referenz auf Variablen 720
 Referenzen als Rückgabewert 718
 Referenzen in Funktionsschnittstellen 717
Rein virtuelle Funktion 927
Reinterpret Cast 715
reinterpret_cast 703
Rekursion 188, 196
Relationen 754, 1045
remove 1062
remove_copy 1062
remove_copy_if 1062
remove_if 1062
replace 1059
replace_copy 1060

- replace_copy_if 1060
- replace_if 1059
- return-Anweisung 184, 293
- Returnwert 182
- reverse 1064
- reverse_copy 1064
- Review 36
- rotate 1064
- rotate_copy 1064
- Rückgabeparameter 182
- Rückgabewert 182
- Rücksprungadresse 198
- Runtime-Library
 - abort* 224
 - abs* 202
 - acos* 202
 - asctime* 225
 - asin* 202
 - atan* 202
 - atan2* 202
 - atexit* 224
 - atof* 204
 - atoi* 204
 - atol* 204
 - bsearch* 216
 - calloc* 225, 466, 510
 - ceil* 202
 - clearerr* 223
 - clock* 226
 - cos* 202
 - cosh* 202
 - ctime* 225
 - difftime* 225
 - div* 202
 - exit* 224
 - exp* 202
 - fabs* 202
 - fclose* 213
 - feof* 213
 - ferror* 213
 - fflush* 213
 - fgetc* 213
 - fgetpos* 213
 - fgets* 213
 - floor* 202
 - fmod* 202
 - fopen* 213
 - fprintf* 213
 - fputc* 213
- Runtime-Library (Forts.)
 - fputs* 213
 - fread* 213
 - free* 225, 466, 510
 - frexp* 203
 - fscanf* 213
 - fseek* 213
 - fsetpos* 213
 - ftell* 213
 - fwrite* 213
 - getc* 213
 - getchar* 208
 - getenv* 226
 - gets* 208
 - gmtime* 225
 - isalnum* 204
 - isalpha* 204
 - iscntrl* 204
 - isdigit* 205
 - isgraph* 205
 - islower* 205
 - isprint* 205
 - ispunct* 205
 - isspace* 205
 - isupper* 205
 - isxdigit* 205
 - labs* 202
 - ldexp* 203
 - ldiv* 202
 - localtime* 225
 - log* 203
 - log10* 203
 - longjmp* 220
 - malloc* 225, 466, 509
 - memchr* 225
 - memcmp* 225
 - memcpy* 225
 - memmove* 225
 - memset* 225
 - mktime* 225
 - modf* 203
 - perror* 223
 - pow* 203
 - printf* 208
 - putc* 213
 - putchar* 208
 - puts* 208
 - qsort* 216
 - raise* 226

Runtime-Library (Forts.)

rand 203
realloc 225, 466, 511
remove 213
rename 213
rewind 213
scanf 206, 208
setbuf 213
setjmp 220
setvbuf 213
signal 226
sin 203
sinh 203
sprintf 206
sqrt 203
srand 203
strcat 206
strchr 206
strcmp 206
strcpy 206
strcspn 207
strerror 223
strlen 207
strncat 207
strncmp 207
strncpy 207
strpbrk 207
strrchr 207
strspn 207
strstr 207
strtod 205
strtok 207
strtol 205
strtoul 205
system 226
tan 203
tanh 203
time 225
tmofile 214
tmpnam 214
toascii 205
tolower 205
toupper 205
ungetc 214
va_arg 216
va_end 216
va_start 216
vfprintf 219
vprintf 219

Runtime-Library (Forts.)

vsprintf 219
 R-Wert 291

S

Schleife 298

Initialisierung 54
Inkrement 55
Kopf 55
Körper 55
Test 54

Schlüsselwörter 271, 703

and 703
and_eq 703
bitand 703
bitor 703
bool 703
catch 703
class 703
compl 703
const_cast 703
delete 703
dynamic_cast 703
explicit 703
export 703
false 703
friend 703
inline 703
mutable 703
namespace 703
new 703
not 703
not_eq 703
operator 703
or 703
or_eq 703
private 703
protected 703
public 703
reinterpret_cast 703
static_cast 703
template 703
this 703
throw 703
true 703
try 703
typeid 703
typename 703

Schlüsselwörter (Forts.)
using 703
virtual 703
wchar_t 703
xor 703
xor_eq 703
Schnittstelle 182, 293
search 1054
search_n 1054
Seiteneffekt 182, 187
Selectionsort 398
set 1040
set_difference 1068
set_intersection 1068
set_symmetric_difference 1068
set_union 1068
setjmp 977
Shellsort 403
sizeof-Operator 509
sort 1065
sort_heap 1070
Sortierverfahren 395, 434
 Bubblesort 396
 Distributionsort 435
 Heapsort 412
 Insertionsort 401
 Leistungsanalyse 418
 Quicksort 407
 Selectionsort 398
 Shellsort 403
 Vergleich und Bewertung 428
Speicherfunktionen 225
Spezialisierung 749
Sprunganweisung 300
Sprungleisten 296
stable_partition 1064
stable_sort 1065
Stack 197, 480, 553
 pop 197, 480
 push 197, 480
stack 1027
Stackpointer 197
Stacks 1027
static 845
Static Cast 714
static_cast 703
Statische Objekte 943

Statische Variablen 199
string 987
Stringbearbeitung 206
Strings 161, 987
struct-Anweisung 444, 497
Suchen und Sortieren 216
Suffix 272
Summen 80
swap 1058
swap_ranges 1058
Symbolische Konstanten 303
Systemanalyse 35, 37
Systementwurf 35, 37

T

Tastatureingabe 796
Tautologie 104
Template 970
template 703
Terabyte 140
this 703
this-Pointer 799, 910
throw 703, 978
transform 1053
Travelling Salesman Problem 692
Traversierung 547
true 703
try 703, 978
type_info 928
typeid 703, 928
typename 703
Typumwandlungen 713
Typvereinbarungen 515

U

Überdeckungsanalyse 366
Überladen von Funktionen 726
Überladen von Operatoren 728
union-Anweisung 451, 505
unique 1062
unique_copy 1062
Unterprogramm 182
Upcast 931
upper_bound 1066
using 703, 737

V

Variable Anzahl von Argumenten 216
 Variablen 45, 274
 automatische Variablen 277
 Definition von Variablen 716
 Externverweis 200
 globale statische Variablen 277
 globale Variablen 187, 199, 277
 Initialisierung von Variablen 276
 lokale statische Variablen 277
 lokale Variablen 199, 276
 Register-Variablen 277
 statische Variablen 199
 Variablendefinitionen 43
 Variablendeklarationen 266
 vector 1005
 Vereinigungssuche 644
 Vererbung 748, 823, 913
 Einfachvererbung 832, 913
 mehrdeutige Vererbung 917
 Mehrfachvererbung 833, 917
 Modifikation von Zugriffsrechten 941
 wiederholte Vererbung 918
 Zugriffsschutz 935
 Zugriffsspezifikation 936
 Verfahren von Warshall 637
 Verschmelzungsgesetz 104
 Virtual 703, 842
 Virtuelle Basisklasse 920
 Virtuelle Destruktoren 925
 Virtuelle Funktion 923
 Virtuelle Member-Funktionen 841
 vollständiger Baum 571
 Vorgehensmodell 35

W

Wahrheitstafel 100
 Warteschlangen 1030
 wchar_t 703, 711
 wscat 712
 wscmp 712
 wcsncpy 712

wcslen 712
 Wegematrix 634
 while-Anweisung 299
 Wide Character 711
 Wiederholte Vererbung 752
 wprintf 712
 wscanf 712
 Wurzel 620
 Wurzelbaum 620

X

xor 703
 xor_eq 703

Z

Zahlen
 Dezimalzahlen 131, 272
 Dualzahlen 131
 ganze Zahlen 44, 129
 Gleitkommazahlen 44, 142
 Hexadezimalzahlen 134, 272
 Oktalzahlen 133, 272
 Zahlendarstellungen 129
 Zahlenfolge 73
 explizite Definition 73
 induktive Definition 74
 Zeichen 273
 Bildschirm-Steuerzeichen 144
 druckbares Zeichen 143
 nicht druckbares Zeichen 143
 Zeichenkette 161, 273, 279
 Länge berechnen 166
 Terminator 161
 Zeichenkette kopieren 165
 Zeichenketten vergleichen 165
 Zeichensatz 143
 Zeiger 254, 463
 Zeigerarithmetik 257
 Zeigervariable 254
 Zeit-Funktionen 225
 Zusammenhangskomponente 641
 Zweierkomplement 138