



it
informatik

Bjarne Stroustrup

Einführung in die Programmierung mit C++

Einführung in die Programmierung mit C++

Einführung in die Programmierung mit C++

Inhaltsverzeichnis

Einführung in die Programmierung mit C++

Inhaltsübersicht

Teil I Die Grundlagen

Teil II Ein- und Ausgabe

Teil III Daten und Algorithmen

Teil IV Erweiterung des Blickwinkels

Teil V Anhang

Inhaltsverzeichnis

Vorwort zur amerikanischen Ausgabe

Vorwort zur deutschen Ausgabe

Kapitel 0 Hinweise für den Leser

Kapitel 1 Computer, Menschen und Programmierung

Teil I Die Grundlagen

Kapitel 2 Hello, World!

Kapitel 3 Objekte, Typen und Werte

Kapitel 4 Berechnungen und Anweisungen

Kapitel 5 Fehler

Kapitel 6 Ein Programm schreiben

Kapitel 7 Ein Programm fertigstellen

Kapitel 8 Technische Details: Funktionen und mehr

Kapitel 9 Technische Details: Klassen und mehr

Teil II Ein- und Ausgabe

Kapitel 10 Ein- und Ausgabestreams

Kapitel 11 Die Ein- und Ausgabe anpassen

Inhaltsverzeichnis

Kapitel 12 Ein Anzeigemodell

Kapitel 13 Grafikklassen

Kapitel 14 Grafikklassen-Design

Kapitel 15 Grafische Darstellung von Funktionen und Daten

Kapitel 16 Grafische Benutzerschnittstellen

Teil III Daten und Algorithmen

Kapitel 17 Vektoren und Freispeicher

Kapitel 18 Vektoren und Arrays

Kapitel 19 Vektor, Templates und Ausnahmen

Kapitel 20 Container und Iteratoren

Kapitel 21 Algorithmen und Maps

Teil IV Erweiterung des Blickwinkels

Kapitel 22 Ideale und Geschichte

Kapitel 23 Textmanipulation

Kapitel 24 Numerik

Kapitel 25 Programmierung eingebetteter Systeme

Kapitel 26 Testen

Kapitel 27 Die Programmiersprache C

Teil V Anhang

Anhang A Sprachübersicht

Anhang B Zusammenfassung der Standardbibliothek

Anhang C Erste Schritte mit Visual Studio

Anhang D FLTK-Installation

Anhang E GUI-Implementierung

Glossar

Literaturverzeichnis

Bildnachweis

Register

Farbteil

Vorwort zur amerikanischen Ausgabe

Inhaltsverzeichnis

Hinweis an die Studenten

Hinweis an die Lehrkräfte

Unterstützung

Danksagungen

Vorwort zur deutschen Ausgabe

Zur Handhabung des Buches

Für Dozenten und Studenten

Companion Website

0. Hinweise für den Leser

0.1 Die Struktur dieses Buches

0.1.1 Allgemeiner Ansatz

0.1.2 Aufgaben, Übungen usw.

0.1.3 Was kommt nach diesem Buch?

0.2 Eine Philosophie des Lehrens und Lernens

0.2.1 Die Reihenfolge der Themen

0.2.2 Programmierung und Programmiersprache

0.2.3 Portabilität

0.3 Programmierung und Informatik

0.4 Kreativität und Problemlösung

0.5 Feedback

0.6 Literaturhinweise

0.7 Biografien

Bjarne Stroustrup

Lawrence Pete Petersen

1 Computer, Menschen und Programmierung

1.1 Einführung

1.2 Software

1.3 Menschen

Inhaltsverzeichnis

1.4 Informatik

1.5 Computer sind allgegenwärtig

1.5.1 Mit und ohne Bildschirm

1.5.2 Schifffahrt

1.5.3 Telekommunikation

1.5.4 Medizin

1.5.5 EDV

1.5.6 Eine vertikale Betrachtung

1.5.7 Na und?

1.6 Ideale für Programmierer

Fragen

Übungen

Schlüsselbegriffe

Teil I Die Grundlagen

2 Hello, World!

2.1 Programme

2.2 Das klassische erste Programm

2.3 Kompilierung

2.4 Linken

2.5 Programmierumgebungen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

3 Objekte, Typen und Werte

3.1 Eingabe

3.2 Variablen

3.3 Eingabe und Typ

3.4 Operationen und Operatoren

3.5 Zuweisung und Initialisierung

Inhaltsverzeichnis

3.5.1 Ein Beispiel: Wortwiederholungen löschen

3.6 Zusammengesetzte Zuweisungsoperatoren

3.6.1 Ein Beispiel: Wortwiederholungen nummerieren

3.7 Namen

3.8 Typen und Objekte

3.9 Typsicherheit

3.9.1 Sichere Typumwandlungen

3.9.2 Unsichere Typumwandlungen

Testen Sie Ihr Können

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

4 Berechnungen und Anweisungen

4.1 Berechnungen

4.2 Ziele und Werkzeuge

4.3 Ausdrücke

4.3.1 Konstante Ausdrücke

4.3.2 Operatoren

4.3.3 Typumwandlungen

4.4 Anweisungen

4.4.1 Auswahlanweisungen

4.4.2 Schleifen

4.5 Funktionen

4.5.1 Wozu brauchen wir Funktionen?

4.5.2 Funktionsdeklarationen

4.6 Vektor

4.6.1 Einen Vektor vergrößern

4.6.2 Ein Zahlenbeispiel

4.6.3 Ein Textbeispiel

4.7 Sprachkonstrukte

Aufgaben

Fragen

Inhaltsverzeichnis

Übungen

Schlüsselbegriffe

5 Fehler

5.1 Einführung

5.2 Fehlerquellen

5.3 Kompilierfehler

5.3.1 Syntaxfehler

5.3.2 Typfehler

5.3.3 Fehler, die keine sind (non-errors)

5.4 Linkerfehler

5.5 Laufzeitfehler

5.5.1 Der Aufrufer behandelt die Fehler

5.5.2 Die aufgerufene Funktion behandelt die Fehler

5.5.3 Fehler melden

5.6 Ausnahmen

5.6.1 Ungültige Argumente

5.6.2 Bereichsfehler

5.6.3 Unzulässige Eingaben

5.6.4 Fehler durch Einengung

5.7 Logische Fehler

5.8 Abschätzen

5.9 Debuggen

5.9.1 Praktische Debug-Hinweise

5.10 Vor- und Nachbedingungen

5.10.1 Nachbedingungen

5.11 Testen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

6 Ein Programm schreiben

6.1 Das Problem

Inhaltsverzeichnis

6.2 Über das Problem nachdenken

6.2.1 Entwicklungsphasen

6.2.2 Strategie

6.3 Zurück zum Taschenrechner!

6.3.1 Erster Versuch

6.3.2 Token

6.3.3 Token implementieren

6.3.4 Token verwenden

6.3.5 Zurück ans Reißbrett

6.4 Grammatiken

6.4.1 Ein Exkurs: deutsche Grammatik

6.4.2 Eine Grammatik schreiben

6.5 Eine Grammatik in Code umwandeln

6.5.1 Grammatikregeln implementieren

6.5.2 Ausdrücke

6.5.3 Terme

6.5.4 Faktoren

6.6 Die erste Version ausprobieren

6.7 Die zweite Version ausprobieren

6.8 Token-Streams

6.8.1 Token_stream implementieren

6.8.2 Token lesen

6.8.3 Zahlen lesen

6.9 Programmstruktur

Testen Sie Ihr Können

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

7 Ein Programm fertigstellen

7.1 Einführung

7.2 Eingabe und Ausgabe

7.3 Fehlerbehandlung

Inhaltsverzeichnis

7.4 Negative Zahlen

7.5 Rest: %

7.6 Aufräumarbeiten

7.6.1 Symbolische Konstanten

7.6.2 Einsatz von Funktionen

7.6.3 Code-Layout

7.6.4 Kommentare

7.7 Wiederaufnahme der Programmausführung nach Auftreten eines Fehlers

7.8 Variablen

7.8.1 Variablen und Definitionen

7.8.2 Namen einführen

7.8.3 Vordefinierte Namen

7.8.4 Sind wir fertig?

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

8 Technische Details: Funktionen und mehr

8.1 Technische Details

8.2 Deklarationen und Definitionen

8.2.1 Arten von Deklarationen

8.2.2 Variablen- und Konstantendeklarationen

8.2.3 Standardinitialisierung

8.3 Headerdateien

8.4 Gültigkeitsbereich

8.5 Funktionsaufrufe und -rückgabewerte

8.5.1 Argumente und Rückgabetyt deklarieren

8.5.2 Rückgabewerte

8.5.3 Pass-by-value

8.5.4 Pass-by-const-reference

8.5.5 Pass-by-reference

8.5.6 Pass-by-value kontra pass-by-reference

8.5.7 Argumentüberprüfung und -umwandlung

Inhaltsverzeichnis

8.5.8 Implementierung von Funktionsaufrufen

8.6 Auswertungsreihenfolge

8.6.1 Auswertung von Ausdrücken

8.6.2 Globale Initialisierung

8.7 Namensbereiche

8.7.1 using-Deklarationen und using-Direktiven

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

9 Technische Details: Klassen und mehr

9.1 Benutzerdefinierte Typen

9.2 Klassen und Klassenmember

9.3 Schnittstelle und Implementierung

9.4 Eine Klasse entwickeln

9.4.1 Strukturen und Funktionen

9.4.2 Memberfunktionen und Konstruktoren

9.4.3 Halten Sie Details privat

9.4.4 Memberfunktionen definieren

9.4.5 Objektbezug

9.4.6 Fehlerbehandlung

9.5 Aufzählungen

9.6 Operatorenüberladung

9.7 Klassenschnittstellen

9.7.1 Argumenttypen

9.7.2 Kopieren

9.7.3 Standardkonstruktoren

9.7.4 Konstante Memberfunktionen

9.7.5 Member und Hilfsfunktionen

9.8 Die Klasse

Aufgaben

Fragen

Übungen

Inhaltsverzeichnis

Schlüsselbegriffe

Teil II Ein- und Ausgabe

10 Ein- und Ausgabestreams

10.1 Ein- und Ausgabe

10.2 Das E/A-Stream-Modell

10.3 Dateien

10.4 Dateien öffnen

10.5 Dateien lesen und schreiben

10.6 E/A-Fehlerbehandlung

10.7 Einzelne Werte lesen

10.7.1 Das Problem in handliche Teilprobleme zerlegen

10.7.2 Trennung von Kommunikation und Funktion

10.8 Benutzerdefinierte Ausgabeoperatoren

10.9 Benutzerdefinierte Eingabeoperatoren

10.10 Standardlösung für eine Einleseschleife

10.11 Eine strukturierte Datei lesen

10.11.1 Repräsentation im Speicher

10.11.2 Strukturierte Werte einlesen

10.11.3 Austauschbare Darstellungen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

11 Die Ein- und Ausgabe anpassen

11.1 Regelmäßigkeit und Individualität

11.2 Formatierung der Ausgabe

11.2.1 Ausgabe ganzer Zahlen

11.2.2 Eingabe ganzer Zahlen

11.2.3 Ausgabe von Gleitkommazahlen

11.2.4 Genauigkeit

11.2.5 Felder

11.3 Dateien öffnen

Inhaltsverzeichnis

11.3.1 Öffnungsmodi für Dateien

11.3.2 Binärdateien

11.3.3 Festlegen der Schreib- und Leseposition in Dateien

11.4 Stringstreams

11.5 Zeilenorientierte Eingabe

11.6 Zeichenklassifizierung

11.7 Verwendung eigener Trennzeichen

11.8 Und es gibt noch so viel mehr

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

12 Ein Anzeigemodell

12.1 Wozu Grafik?

12.2 Ein Anzeigemodell

12.3 Ein erstes Beispiel

12.4 Programmieren mit GUI-Bibliotheken

12.5 Koordinaten

12.6 Formen

12.7 Programmieren mit Grafikprimitiven

12.7.1 Die Grafik-Header und

12.7.2 Ein nahezu leeres Fenster

12.7.3 Achsen

12.7.4 Grafische Darstellung von Funktionen

12.7.5 Polygone

12.7.6 Rechtecke

12.7.7 Füllen

12.7.8 Text

12.7.9 Bilder

12.7.10 Und vieles mehr

12.8 Ausführung des Grafikbeispiels

12.8.1 Die Quelldateien

Aufgaben

Inhaltsverzeichnis

Fragen

Übungen

Schlüsselbegriffe

13 Grafikklassen

13.1 Überblick über die Grafikklassen

13.2 Point und Line

13.3 Lines

13.4 Color

13.5 Line-Style

13.6 Open_polyline

13.7 Closed poly_line

13.8 Polygon

13.9 Rectangle

13.10 Arbeiten mit unbenannten Objekten

13.11 Text

13.12 Circle

13.13 Ellipse

13.14 Marked_polyline

13.15 Marks

13.16 Mark

13.17 Image

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

14 Grafikklassen-Design

14.1 Designprinzipien

14.1.1 Typen

14.1.2 Operationen

14.1.3 Namensgebung

14.1.4 Zugriff und Veränderung

Inhaltsverzeichnis

14.2 Shape

- 14.2.1 Eine abstrakte Klasse
- 14.2.2 Zugriffskontrolle
- 14.2.3 Shape-Objekte zeichnen
- 14.2.4 Kopieren und Zugriffskontrolle

14.3 Basisklassen und abgeleitete Klassen

- 14.3.1 Objekt-Layout
- 14.3.2 Klassen ableiten und virtuelle Funktionen definieren
- 14.3.3 Überschreibung
- 14.3.4 Zugriff
- 14.3.5 Rein virtuelle Funktionen

14.4 Vorteile der objektorientierten Programmierung

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

15 Grafische Darstellung von Funktionen und Daten

15.1 Einführung

15.2 Grafische Darstellung einfacher Funktionen

15.3 Function

- 15.3.1 Vorgabeargumente
- 15.3.2 Weitere Beispiele

15.4 Achsen

15.5 Approximation

15.6 Darstellung von Daten

- 15.6.1 Einlesen aus Dateien
- 15.6.2 Allgemeines Layout
- 15.6.3 Skalierung
- 15.6.4 Aufbau des Graphen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

Inhaltsverzeichnis

16 Grafische Benutzerschnittstellen

16.1 Verschiedene Benutzerschnittstellen

16.2 Die Schaltfläche Weiter

16.3 Ein einfaches Fenster

16.3.1 Eine Callback-Funktion

16.3.2 Eine Warteschleife

16.4 Schaltflächen und andere Widgets

16.4.1 Widgets

16.4.2 Schaltflächen

16.4.3 in_box und Out_box

16.4.4 Menüs

16.5 Ein Beispiel

16.6 Umkehrung der Steuerung

16.7 Ein Menü hinzufügen

16.8 GUI-Code debuggen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

Teil III Daten und Algorithmen

18 Vektoren und Arrays

18.1 Einführung

18.2 Kopieren

18.2.1 Kopierkonstruktoren

18.2.2 Zuweisungsoperatoren

18.2.3 Terminologie

18.3 Essenzielle Operationen

18.3.1 Explizite Konstruktoren

18.3.2 Konstruktoren und Destruktoren debuggen

18.4 Zugriff auf Vektor-Elemente

18.4.1 Überladung für const-Objekte

18.5 Arrays

Inhaltsverzeichnis

18.5.1 Zeiger auf Array-Elemente

18.5.2 Zeiger und Arrays

18.5.3 Array-Initialisierung

18.5.4 Probleme mit Zeigern

18.6 Beispiele: Palindrom

18.6.1 Palindrome mit string

18.6.2 Palindrome mit Arrays

18.6.3 Palindrome mit Zeigern

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

19 Vektor, Templates und Ausnahmen

19.1 Problematik

19.2 Die Größe ändern

19.2.1 Darstellung

19.2.2 reserve und capacity

19.2.3 resize

19.2.4 push_back

19.2.5 Zuweisung

19.2.6 Aktueller Stand unseres Vektors

19.3 Templates

19.3.1 Typen als Template-Parameter

19.3.2 Generische Programmierung

19.3.3 Container und Vererbung

19.3.4 Integer als Template-Parameter

19.3.5 Deduktion von Template-Argumenten

19.3.6 Verallgemeinerung von vector

19.4 Bereichsüberprüfung und Ausnahmen

19.4.1 Eine Nebenbemerkung: Überlegungen zum Design

19.4.2 Eine Beichte: Makros

19.5 Ressourcen und Ausnahmen

19.5.1 Potenzielle Probleme mit der Ressourcenverwaltung

19.5.2 Ressourcenbelegung ist Initialisierung (RAII)

Inhaltsverzeichnis

19.5.3 Garantien

19.5.4 auto_ptr

19.5.5 RAII für vector

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

20 Container und Iteratoren

20.1 Daten speichern und verarbeiten

20.1.1 Mit Daten arbeiten

20.1.2 Code allgemein halten

20.2 STL-Ideale

20.3 Sequenzen und Iteratoren

20.3.1 Zurück zum Beispiel

20.4 Verkettete Listen

20.4.1 Listenoperationen

20.4.2 Iteration

20.5 Weitere Verallgemeinerung des vector-Typs

20.6 Ein Beispiel: ein einfacher Texteditor

20.6.1 Die Zeilen

20.6.2 Iteration

20.7 vector, list und string

20.7.1 Einfügen und Löschen

20.8 Unseren Vektor an die STL anpassen

20.9 Annäherung der integrierten Arrays an die STL

20.10 Überblick über die Container

20.10.1 Iterator-Kategorien

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

21 Algorithmen und Maps

Inhaltsverzeichnis

21.1 Algorithmen der Standardbibliothek

21.2 Der einfachste Algorithmus: find()

21.2.1 Einige generische Anwendungsbereiche

21.3 Die allgemeine Suche:

21.4 Funktionsobjekte

21.4.1 Allgemeine Darstellung des Konzepts der Funktionsobjekte

21.4.2 Prädikate für Klassenmember

21.5 Numerische Algorithmen

21.5.1 Akkumulator

21.5.2 accumulate() - allgemeine Version

21.5.3 Das innere Produkt

21.5.4 inner_product() allgemeine Version

21.6 Assoziative Container

21.6.1 Maps

21.6.2 Maps ein Überblick

21.6.3 Ein weiteres map-Beispiel

21.6.4 unordered_map

21.6.5 Sets

21.7 Kopieren

21.7.1 Kopieren

21.7.2 Stream-Iteratoren

21.7.3 Mit set Ordnung halten

21.7.4 copy_if

21.8 Sortieren und suchen

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

17 Vektoren und Freispeicher

17.1 Einführung

17.2 Vektor-Grundlagen

17.3 Speicher, Adressen und Zeiger

17.3.1 Der sizeof-Operator

Inhaltsverzeichnis

17.4 Freispeicher und Zeiger

- 17.4.1 Freispeicher reservieren (Allokation)
- 17.4.2 Zugriff über Zeiger
- 17.4.3 Bereiche
- 17.4.4 Initialisierung
- 17.4.5 Der Nullzeiger
- 17.4.6 Freispeicher freigeben (Deallokation)

17.5 Destruktoren

- 17.5.1 Automatisch generierte Destruktoren
- 17.5.2 Destruktoren und Freispeicher

17.6 Zugriff auf Elemente

17.7 Zeiger auf Klassenobjekte

17.8 Eingriff ins Typensystem: void* und Casts

17.9 Zeiger und Referenzen

- 17.9.1 Zeiger- und Referenzparameter
- 17.9.2 Zeiger, Referenzen und Vererbung
- 17.9.3 Ein Beispiel: Listen
- 17.9.4 Operationen für Listen
- 17.9.5 Verwendung von Listen

17.10 Der this-Zeiger

- 17.10.1 Weitere Anwendungsbeispiele

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

Teil IV Erweiterung des Blickwinkels

22 Ideale und Geschichte

22.1 Geschichte, Ideale und Professionalität

- 22.1.1 Programmiersprachen Ziele und Philosophien
- 22.1.2 Programmierideale
- 22.1.3 Stile/Paradigmen

22.2 (Kurze) Geschichte der Programmiersprachen

- 22.2.1 Die frühesten Sprachen

Inhaltsverzeichnis

22.2.2 Die Wurzeln der modernen Sprachen

22.2.2.1 Fortran

22.2.2.2 COBOL

22.2.2.3 Lisp

22.2.3 Die Algol-Familie

22.2.3.1 Algol60

22.2.3.2 Pascal

22.2.3.3 Ada

22.2.4 Simula

22.2.5 C

22.2.6 C++

22.2.7 Heute

22.2.8 Informationsquellen

Fragen

Übungen

Schlüsselbegriffe

23 Textmanipulation

23.1 Text

23.2 Strings

23.3 E/A-Streams

23.4 Maps

23.4.1 Implementierungsdetails

23.5 Ein Problem

23.6 Die Idee der regulären Ausdrücke

23.7 Suchen mithilfe regulärer Ausdrücke

23.8 Syntax der regulären Ausdrücke

23.8.1 Zeichen und Sonderzeichen

23.8.2 Zeichenklassen

23.8.3 Quantifizierer

23.8.4 Gruppierung

23.8.5 Alternativen

23.8.6 Zeichensätze und -bereiche

23.8.7 Fehler bei regulären Ausdrücken

23.9 Abgleich mit regulären Ausdrücken

Inhaltsverzeichnis

23.10 Literaturhinweise

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

24 Numerik

24.1 Einführung

24.2 Größe, Genauigkeit und Überlauf

24.2.1 Numerische Grenzwerte

24.3 Arrays

24.4 Mehrdimensionale Arrays im C-Stil

24.5 Die Matrix-Bibliothek

24.5.1 Dimensionen und Zugriff

24.5.2 1D-Matrix

24.5.3 2D-Matrix

24.5.4 Matrix-E/A

24.5.5 3D-Matrix

24.6 Ein Beispiel: lineare Gleichungen

24.6.1 Klassische Gaußsche Elimination

24.6.2 Pivotisierung

24.6.3 Testen

24.7 Zufallszahlen

24.8 Die mathematischen Standardfunktionen

24.9 Komplexe Zahlen

24.10 Literaturhinweise

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

25 Programmierung eingebetteter Systeme

25.1 Eingebettete Systeme

25.2 Grundlegende Konzepte

Inhaltsverzeichnis

25.2.1 Vorhersagbarkeit

25.2.2 Ideale

25.2.3 Mit dem Scheitern leben

25.3 Speicherverwaltung

25.3.1 Probleme mit dem Freispeicher

25.3.2 Alternativen zum üblichen Freispeicher

25.3.3 Pool-Beispiel

25.3.4 Stack-Beispiel

25.4 Adressen, Zeiger und Arrays

25.4.1 Ungeprüfte Umwandlungen

25.4.2 Ein Problem: dysfunktionale Schnittstellen

25.4.3 Eine Lösung: eine Schnittstellenklasse

25.4.4 Vererbung und Container

25.5 Bits, Bytes und Words

25.5.1 Bits und Bitoperationen

25.5.2 bitset

25.5.3 signed und unsigned

25.5.4 Bitmanipulation

25.5.5 Bitfelder

25.5.6 Ein Beispiel: einfache Verschlüsselung

25.6 Codierstandards

25.6.1 Wie sollte ein Codierstandard aussehen?

25.6.2 Beispielregeln

25.6.3 Konkrete Codierstandards

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

26 Testen

26.1 Worum geht es uns?

26.1.1 Warnung

26.2 Beweise

26.3 Testen

Inhaltsverzeichnis

- 26.3.1 Regressionstests
- 26.3.2 Unit-Tests
- 26.3.3 Algorithmen und Nicht-Algorithmen
- 26.3.4 Systemtests
- 26.3.5 Klassen testen
- 26.3.6 Annahmen aufspüren, die nicht standhalten

26.4 Testfreundliches Design

26.5 Debuggen

26.6 Performance

- 26.6.1 Zeitmessungen

26.7 Literaturhinweise

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

27 Die Programmiersprache C

27.1 C und C++: Geschwister

- 27.1.1 C/C++-Kompatibilität
- 27.1.2 C++-Features, die in C fehlen
- 27.1.3 Die C-Standardbibliothek

27.2 Funktionen

- 27.2.1 Keine Überladung von Funktionsnamen
- 27.2.2 Typprüfung von Funktionsargumenten
- 27.2.3 Funktionsdefinitionen
- 27.2.4 C von C++ aus und C++ von C aus aufrufen
- 27.2.5 Zeiger auf Funktionen

27.3 Kleinere Sprachunterschiede

- 27.3.1 Namensbereich des struct-Tags
- 27.3.2 Schlüsselwörter
- 27.3.3 Definitionen
- 27.3.4 Typumwandlungen im C-Stil
- 27.3.5 Umwandlung von void*
- 27.3.6 enum

Inhaltsverzeichnis

27.3.7 Namensbereiche

27.4 Freispeicher

27.5 C-Strings

27.5.1 C-Strings und const

27.5.2 Byte-Operationen

27.5.3 Ein Beispiel: strcpy()

27.5.4 Eine Stilfrage

27.6 Eingabe/Ausgabe: stdio

27.6.1 Ausgabe

27.6.2 Eingabe

27.6.3 Dateien

27.7 Konstanten und Makros

27.8 Makros

27.8.1 Funktionsähnliche Makros

27.8.2 Syntax-Makros

27.8.3 Bedingte Kompilierung

27.9 Ein Beispiel: aufdringliche Container

Aufgaben

Fragen

Übungen

Schlüsselbegriffe

Teil V Anhang

A Sprachübersicht

A.1 Allgemein

A.1.1 Terminologie

A.1.2 Programmstart und -beendigung

A.1.3 Kommentare

A.2 Literale

A.2.1 Integer-Literale

A.2.1.1 Zahlensysteme

A.2.2 Gleitkommalliterale

A.2.3 Boolesche Literale

A.2.4 Zeichenliterale

Inhaltsverzeichnis

A.2.5 String-Literale

A.2.6 Das Zeigerliteral

A.3 Bezeichner

A.3.1 Schlüsselwörter

A.4 Gültigkeitsbereich, Speicherklasse und Lebensdauer

A.4.1 Gültigkeitsbereich

A.4.2 Speicherklassen

A.4.3 Lebensdauer

A.5 Ausdrücke

A.5.1 Benutzerdefinierte Operatoren

A.5.2 Automatische Typumwandlung

A.5.2.1 Promotionen

A.5.2.2 Umwandlungen

A.5.2.3 Benutzerdefinierte Umwandlungen

A.5.3 Konstante Ausdrücke

A.5.4 sizeof

A.5.5 Logische Ausdrücke

A.5.6 new and delete

A.5.7 Casts

A.6 Anweisungen

A.7 Deklarationen

A.7.1 Definitionen

A.8 Integrierte Typen

A.8.1 Zeiger

A.8.2 Arrays

A.8.3 Referenzen

A.9 Funktionen

A.9.1 Auflösung von Überladungen

A.9.2 Vorgabeargumente

A.9.3 Unspezifizierte Argumente

A.9.4 Bindspezifikationen

A.10 Benutzerdefinierte Typen

A.10.1 Überladen von Operatoren

A.11 Aufzählungen

Inhaltsverzeichnis

A.12 Klassen

A.12.1 Memberzugriff

A.12.1.1 Der this-Zeiger

A.12.1.2 Friends

A.12.2 Klassenmemberdefinitionen

A.12.3 Konstruktion, Destruktion und Kopieren

A.12.4 Abgeleitete Klassen

A.12.4.1 Virtuelle Funktionen

A.12.4.2 Abstrakte Klassen

A.12.4.3 Generierte Operationen

A.12.5 Bitfelder

A.12.6 Unions

A.13 Templates

A.13.1 Template-Argumente

A.13.2 Template-Instanzierung

A.13.3 Template-Membertypen

A.14 Ausnahmen

A.15 Namensbereiche

A.16 Aliase

A.17 Präprozessor-Direktiven

A.17.1 #include

A.17.2 #define

B Zusammenfassung der Standardbibliothek

B.1 Überblick

B.1.1 Headerdateien

B.1.2 Namensbereich

B.1.3 Notation

B.2 Fehlerbehandlung

B.2.1 Ausnahmen

B.3 Iteratoren

B.3.1 Iterator-Modell

B.3.2 Iterator-Kategorien

B.4 Container

B.4.1 Überblick

Inhaltsverzeichnis

- B.4.2 Membertypen
- B.4.3 Konstruktoren, Destruktoren und Zuweisungen
- B.4.4 Iteratoren
- B.4.5 Elementzugriff
- B.4.6 Stack- und Warteschlangenoperationen
- B.4.7 Listenoperationen
- B.4.8 Größe und Kapazität
- B.4.9 Weitere Operationen
- B.4.10 Operationen für assoziative Container

B.5 Algorithmen

- B.5.1 Nichtmodifizierende Sequenzalgorithmen
- B.5.2 Modifizierende Sequenzalgorithmen
- B.5.3 Utility-Algorithmen
- B.5.4 Sortieren und Suchen
- B.5.5 Mengen-Algorithmen
- B.5.6 Heap-Algorithmen
- B.5.7 Permutationen
- B.5.8 min und max

B.6 Hilfskomponenten

- B.6.1 Insert-Iteratoren
- B.6.2 Funktionsobjekte
- B.6.3 pair

B.7 E/A-Streams

- B.7.1 Hierarchie der E/A-Streams
- B.7.2 Fehlerbehandlung
- B.7.3 Eingabeoperationen
- B.7.4 Ausgabeoperationen
- B.7.5 Formatierung
- B.7.6 Standardmanipulatoren

B.8 Stringmanipulation

- B.8.1 Klassifizierung von Zeichen
- B.8.2 Strings
- B.8.3 Reguläre Ausdrücke

B.9 Numerik

Inhaltsverzeichnis

- B.9.1 Numerische Grenzwerte
- B.9.2 Mathematische Standardfunktionen
- B.9.3 Komplexe Zahlen
- B.9.4 Valarrays
- B.9.5 Generische Numerik-Algorithmen

B.10 C-Funktionen der Standardbibliothek

- B.10.1 Dateien
- B.10.2 Die printf()-Familie
- B.10.3 C-Strings
- B.10.4 Speicher
- B.10.5 Datum und Uhrzeit
- B.10.6 Weitere Funktionen

B.11 Andere Bibliotheken

C Erste Schritte mit Visual Studio

- C.1 Ein Programm zur Ausführung bringen
- C.2 Visual Studio installieren
- C.3 Ein Programm erzeugen und ausführen
 - C.3.1 Ein neues Projekt anlegen
 - C.3.2 Die Headerdatei `std_lib_facilities.h` verwenden
 - C.3.3 Dem Projekt eine C++-Quelldatei hinzufügen
 - C.3.4 Quellcode eingeben
 - C.3.5 Ein ausführbares Programm erstellen
 - C.3.6 Das Programm ausführen
 - C.3.7 Das Programm speichern

C.4 Später

D FLTK-Installation

- D.1 Einführung
- D.2 Das FLTK herunterladen
- D.3 Das FLTK installieren
- D.4 Das FLTK in Visual Studio verwenden
- D.5 Testen, ob alles funktioniert

E GUI-Implementierung

- E.1 Callback-Implementierung

Inhaltsverzeichnis

E.2 Widget-Implementierung

E.3 Window-Implementierung

E.4 Vector_ref

E.5 Ein Beispiel: Widgets manipulieren

Glossar

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z

Inhaltsverzeichnis

Literaturverzeichnis

Bildnachweis

Register

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Inhaltsverzeichnis

Z

Ins Internet: Weitere Infos zum Buch, Downloads, etc.

Copyright



Copyright

Daten, Texte, Design und Grafiken dieses eBooks, sowie die eventuell angebotenen eBook-Zusatzdaten sind urheberrechtlich geschützt. Dieses eBook stellen wir lediglich als persönliche Einzelplatz-Lizenz zur Verfügung!

Jede andere Verwendung dieses eBooks oder zugehöriger Materialien und Informationen, einschliesslich

- der Reproduktion,
- der Weitergabe,
- des Weitervertriebs,
- der Platzierung im Internet, in Intranets, in Extranets,
- der Veränderung,
- des Weiterverkaufs
- und der Veröffentlichung

bedarf der schriftlichen Genehmigung des Verlags.

Insbesondere ist die Entfernung oder Änderung des vom Verlag vergebenen Passwortschutzes ausdrücklich untersagt!

Bei Fragen zu diesem Thema wenden Sie sich bitte an: info@pearson.de

Zusatzdaten

Möglicherweise liegt dem gedruckten Buch eine CD-ROM mit Zusatzdaten bei. Die Zurverfügungstellung dieser Daten auf unseren Websites ist eine freiwillige Leistung des Verlags. Der Rechtsweg ist ausgeschlossen.

Hinweis

Dieses und viele weitere eBooks können Sie rund um die Uhr und legal auf unserer Website



herunterladen