Inhalt

Freitag

Teil 1 – Freitagabend 2

Lektion 1 – Was ist Programmierung? 3
1.1 Ein menschliches Programm 4
1.1.1 Der Algorithmus 4
1.2 Der Prozessor 4
1.3 Das Programm 5
1.4 Computerprozessoren 7
Zusammenfassung 7
Selbsttest 7

Lektion 2 – Ihr erstes Programm in Visual C++ 8
2.1 Installation von Visual C++ 8
2.2 Ihr erstes Programm 9
2.3 Erzeugen Ihres Programms 10
2.4 Ausführen Ihres Programms 14
2.5 Abschluss 15
2.5.1 Programmausgabe 16
2.5.2 Visual C++-Hilfe 16
Zusammenfassung 17
Selbsttest 17

Lektion 3 – Ihr erstes C++-Programm mit GNU C++ 17
3.1 Installation von CNU C++ 18
3.2 Ihr erstes Programm 20
3.2.1 Eingabe des C++-Codes 20
3.3 Erzeugen Ihres Programms 23
3.4 Ausführen Ihres Programms 25

3.5 Abschluss 26
3.5.1 Programmausgabe 26
3.5.2 GNU C++-Hilfe 27
Zusammenfassung 28
Selbsttest 28

Lektion 4 – C++-Instruktionen 29
4.1 Das Programm 29
4.2 Das C++-Programm erklärt 30
4.2.1 Der grundlegende Programmaufbau 30
4.2.2 Kommentare 31
4.2.3 Noch mal der Rahmen 32
4.2.4 Anweisungen 32
4.2.5 Deklarationen 32
4.2.6 Eingabe/Ausgabe 33
4.2.7 Ausdrücke 33
4.2.8 Zuweisung 34
4.2.9 Ausdrücke (Fortsetzung) 34
Zusammenfassung 34
Selbsttest 35
Freitagabend – Zusammenfassung 36

Samstag

Teil 2 – Samstagmorgen 40

Lektion 5 – Variablentypen 41
5.1 Dezimalzahlen 42
5.1.1 Begrenzungen von int in C++ 42
5.1.2 Lösen des Abschneideproblems 45
5.1.3 Grenzen von Gleitkommazahlen 46
5.2 Andere Variablentypen 47
5.2.1 Typen von Konstanten 48
5.2.2 Sonderzeichen 48
5.3 Gemischte Ausdrücke 50
Zusammenfassung 51
Selbsttest 52

Lektion 6 – Mathematische Operationen 53
6.1 Arithmetische Operatoren 54
6.2 Ausdrücke 54
6.3 Vorrang von Operatoren 55
6.4 Unäre Operatoren 56
6.5 Zuweisungsoperatoren 57
Zusammenfassung 58
Selbsttest 58

Lektion 7 – Logische Operationen 59
7.1 Einfache logische Operatoren 59
7.1.1 Kurze Schaltkreise und C++ 61
7.1.2 Logische Variablentypen 62
7.2 Binäre Zahlen 62
7.3 Bitweise logische Operationen 63
7.3.1 Die Einzelbit-Operatoren 63
7.3.2 Die bitweisen Operatoren 64
7.3.3 Ein einfacher Test 65
7.3.4 Warum? 66
Zusammenfassung 67
Selbsttest 68

Lektion 8 – Kommandos zur Flusskontrolle 69
8.1 Das Verzweigungskommando 69
8.2 Schleifenkommandos 71
8.2.1 Die while-Schleife 71
8.2.2 Die for-Schleife 74
8.2.3 Spezielle Schleifenkontrolle 75
8.3 Geschachtelte Kontrollkommandos 78
8.4 Können wir switchen? 79
Zusammenfassung 80
Selbsttest 80

Lektion 9 – Funktionen 81
9.1 Code einer Sammelfunktion 81
9.1.1 Sammelcode 82
9.2 Funktion. 84
9.2.1 Warum Funkionen? 84
9.2.2 Einfache Funktionen 85
9.2.3 Funktionen mit Argumenten 85
9.2.4 Mehrere Funktionen mit gleichem Namen 88
9.3 Funktionsprototypen 89
9.4 Verschiedene Speichertypen 90
Zusammenfassung 91
Selbsttest 91

Lektion 10 – Debuggen 92
10.1 Fehlertypen 92
10.2 Die Technik der Ausgabeanweisungen. 93
10.3 Abfangen von Bug Nr. 1 94
10.3.1 Visual C++ 95
10.3.2 GNU C++ 96
10.4 Abfangen von Bug Nr. 2 97
Zusammenfassung 100
Selbsttest 100
Samstagmorgen – Zusammenfassung 101

Teil 3 – Samstagnachmittag 104

Lektion 11 – Das Array 105
11.1 Was ist ein Array? 105
11.1.2 Ein Array in der Praxis 108
11.1.3 Initialisierung eines Array 110
11.1.4 Warum Arrays benutzen? 110
11.1.5 Arrays von Arrays 111
11.2 Arrays von Zeichen 111
11.3 Manipulation von Zeichenketten 113
11.3.1 Unsere eigene Verbindungsfunktion 114
11.3.2 Funktionen für C++-Zeichenketten 116
11.3.3 Wide Character 117
11.4 Obsolete Ausgabefunktionen 117
Zusammenfassung 118
Selbsttest 118

Lektion 12 – Einführung in Klassen 119
12.1 Gruppieren von Daten 119
12.1.1 Ein Beispiel 120
12.1.2 Das Problem 122
12.2 Die Klasse 122
12.2.2 Beispielprogramm 124
12.2.3 Vorteile 126
Zusammenfassung 126
Selbsttest 126

Lektion 13 – Einstieg C++-Zeiger 127
13.1 Was ist deine Adresse? 128
13.2 Einführung in Zeigervariablen 129
13.3 Typen von Zeigern 132
13.4 Übergabe von Zeigern an Funktionen 133
13.4.1 Wertübergabe 133
13.4.2 Übergabe von Zeigerwerten 134
13.4.3 Referenzübergabe 135
13.5 Heap-Speicher 135
13.5.1 Geltungsbereich 135
13.5.2 Das Geltungsbereichsproblem 137
13.5.3 Die Heap-Lösung 137
Zusammenfassung 138
Selbsttest 139

Lektion 14 – Mehr über Zeiger 140
14.1 Zeiger und Arrays 140
14.1.1 Operationen auf Zeigern 141
14.1.2 Zeichenarrays 145
14.1.3 Operationen auf unterschiedlichen Zeigertypen 148
14.1.4 Unterschiede zwischen Zeigern und Arrays 149

14.2 Argumente eines Programms 150
14.2.1 Arrays von Zeigern 150
14.2.2 Arrays von Zeichenketten 151
14.2.3 Die Argumente von main( ) 152
Zusammenfassung 154
Selbsttest 155

Lektion 15 – Zeiger auf Objekte 156
15.1 Zeiger auf Objekte 156
15.1.1 Übergabe von Objekten 157
15.1.2 Referenzen 159
15.1.3 Rückgabe an den Heap 159
15.2 Die Datenstruktur Array 160
15.3 Verkettete Listen 160
15.3.1 Anfügen am Kopf der verketteten Liste 161
15.3.2 Andere Operationen auf verketteten Listen 162
15.3.3 Eigenschaften verketteter Listen 164
15.4 Ein Programm mit verkettetem NameData 165
15.5 Andere Container 168
Zusammenfassung 168
Selbsttest 168

Lektion 16 – Debuggen II 169
16.1 Welcher Debugger? 170
16.2 Das Testprogramm 170
16.3 Einzelschritte durch ein Programm 172
16.4 Einzelschritte in eine Funktion hinein 173
16.5 Verwendung von Haltepunkten 175
16.6 Ansehen und Modifizieren von Variablen 176
16.7 Verwendung des Visual C++-Debuggers 179
Zusammenfassung 181
Selbsttest 182
Samstagnachmittag – Zusammenfassung 183

Teil 4 – Samstagabend 186

Lektion 17 – Objektprogrammierung 187
17.1 Abstraktion und Mikrowellen 187
17.1.1 Funktionale Nachos 188
17.1.2 Objektorientierte Nachos 188
17.2 Klassifizierung und Mikrowellen 189
17.2.1 Warum solche Objekte bilden? 189
17.2.2 Selbstenthaltende Klassen 190
Zusammenfassung 191
Selbsttest 191

Lektion 18 – Aktive Klassen 192
18.1 Klassenrückblick 192
18.2 Grenzen von struct 193
18.2.1 Eine funktionale Lösung 194
18.3 Definition einer aktiven Klasse 195
18.3.1 Namengebung für Elementfunktionen 196
18.4 Definition einer Elementfunktion einer Klasse 197
18.5 Schreiben von Elementfunktionen außerhalb einer Klasse 198
18.5.1 Include-Dateien 199
18.6 Aufruf einer Elementfunktion 200
18.6.1 Aufruf einer Elementfunktion über einen Zeiger 201
18.6.2 Zugriff auf andere Elemente von einer Elementfunktion aus 202
18.7 Überladen von Elementfunktionen 204
Zusammenfassung 205
Selbsttest 205

Lektion 19 – Erhalten der Klassenintegrität 206
19.1 Erzeugen und Vernichten von Objekten 206
19.1.1 Der Konstruktor 207
19.1.2 Der Destruktor 212
19.2 Zugriffskontrolle 214
19.2.1 Das Schlüsselwort protected 214
19.2.2 Statische Datenelemente 217
Zusammenfassung 218
Selbsttest 219

Lektion 20 – Klassenkonstuktoren II 220
20.1 Konstruktoren mit Argumenten 220
20.2 Konstruktion von Klassenelementen 223
20.3 Reihenfolge der Konstruktion 228
20.3.1 Lokale Objekte werden in der Reihenfolge konstruiert 229
20.3.2 Statische Objekte werden nur einmal angelegt 229
20.3.3 Alle globalen Variablen werden vor main( ) erzeugt 229
20.3.4 Keine bestimmte Reihenfolge für globale Objekte 229
20.3.5 Elemente werden in der Reihenfolge ihrer Deklaration konstruiert 231
20.3.6 Destruktoren in umgekehrter Reihenfolge wie Konstruktoren 231
20.4 Der Kopierkonstruktor 231
20.4.1 Flache Kopie gegen tiefe Kopie 233
20.4.2 Ein Fallback-Kopierkonstruktor 234
Zusammenfassung 235
Selbsttest 236
Samstagabend – Zusammenfassung 237

Sonntag

Teil 5 – Sonntagmorgen 240

Lektion 21 – Vererbung 241
21.1 Vorteile der Vererbung 241
21.2 Faktorieren von Klassen 242
21.3 Implementierung von Vererbung in C++ 243
21.4 Unterklassen konstruieren 246
21.5 Die Beziehung HAS_A 249
Zusammenfassung 251
Selbsttest 251

Lektion 22 – Polymorphie 252
22.1 Elementfunktionen überschreiben 252
22.2 Einstieg in Polymorphie 254
22.3 Polymorphie und objektorientierte Programmierung 254
22.4 Wie funktioniert Polymorphie? 257
22.5 Was ist eine virtuelle Funktion nicht? 259
22.6 Überlegungen zu virtual 261
Zusammenfassung 262
Selbsttest 262

Lektion 23 – Abstrakte Klassen und Faktorieren 263
23.1 Faktorieren 263
23.2 Abstrakte Klassen 267
23.2.1 Deklaration einer abstrakten Klasse 267
23.2.2 Erzeugung einer konkreten Klasse aus einer abstrakten Klasse 269
23.2.3 Warum ist eine Unterklasse abstrakt? 271
23.2.4 Ein abstraktes Objekt an eine Funktion übergeben 272
23.2.5 Warum werden rein virtuelle Funktionen benötigt? 273
Zusammenfassung 274
Selbsttest 275

Lektion 24 – Mehrfachvererbung 276
24.1 Wie funktioniert Mehrfachvererbung? 276
24.2 Uneindeutigkeiten bei der Vererbung 278
24.3 Virtuelle Vererbung 279
24.4 Konstruktion von Objekten bei Mehrfachnennung 285
24.5 Eine Meinung dagegen 285
Zusammenfassung 286
Selbsttest 287

Lektion 25 – Große Programme 288
25.1 Warum Programme aufteilen? 288
25.2 Trennung von Klassendefinition und Anwendungsprogramm 289
25.2.1 Aufteilen des Programms 289
25.2.2 Die #include-Direktive 290
25.2.3 Anwendungscode aufteilen 292
25.2.4 Projektdatei 293
25.2.5 Erneute Betrachtung des Standard-Programm-Templates 296
25.2.6 Handhabung von Outline-Elementfunktionen 296
Zusammenfassung 297
Selbsttest 298

Lektion 26 – C++-Präprozessor 299
26.1 Der C++-Präprozessor 299
26.2 Die #include-Direktive. 300
26.3 Die Direktive #define 301
26.3.1 Definition von Makros 301
26.3.2 Häufige Fehler bei der Verwendung von Makros 302
26.4 Compiler-Kontrolle 303
26.4.1 Die #if-Direktive 303
26.4.2 Die #ifdef-Direktive 304
26.5 Namensräume 306
Zusammenfassung 307
Selbsttest 308
Sonntagmorgen – Zusammenfassung 309

Teil 6 – Sonntagnachmittag 312

Lektion 27 – Überladen von Operatoren 313
27.1 Warum sollte ich Operatoren überladen? 314
27.2 Was ist die Beziehung zwischen Operatoren und Funktionen? 314
27.3 Wie funktioniert das Überladen von Operatoren? 315
27.3.1 Spezielle Überlegungen 318
27.4 Ein detaillierterer Blick 318
27.5 Operatoren als Elementfunktionen 320
27.6 Eine weitere Irritation durch Überladen 322
27.7 Wann sollte ein Operator ein Element sein? 323
27.8 Cast-Operator 323
Zusammenfassung 326
Selbsttest 326

Lektion 28 – Der Zuweisungsoperator 327
28.1 Warum ist das Überladen des Zuweisungsoperators kritisch? 327
28.1.1 Vergleich mit dem Kopierkonstruktor 328
28.2 Wie den Zuweisungsoperator überladen? 328
28.2.1 Zwei weitere Details des Zuweisungsoperators 331
28.3 Ein Schlupfloch 332
Zusammenfassung 333
Selbsttest 333

Lektion 29 – Stream-I/O 334
29.1 Wie funktioniert Stream-I/O? 334
29.2 Die Unterklassen fstream 335
29.3 Die Unterklassen strstream 339
29.3.1 Vergleich von Techniken der Zeichenkettenverarbeitung 340
29.4 Manipulatoren 343
29.5 Benutzerdefinierte Inserter 345
29.6 Schlaue Inserter 346
29.7 Aber warum die Shift-Operatoren? 348
Zusammenfassung 349
Selbsttest 349

Lektion 30 – Ausnahmen 350
30.1 Konventionelle Fehlerbehandlung 350
30.2 Warum benötigen wir einen neuen Fehlermechanismus? 341
30.3 Wie arbeiten Ausnahmen? 352
30.3.1 Warum ist der Ausnahmemechanismus eine Verbesserung? 354
30.4 Abfangen von Details, die für mich bestimmt sind 354
30.4.1 Was kann ich »werfen«? 356
30.5 Verketten von catch-Blöcken. 359
Zusammenfassung 360
Selbsttest 361
Sonntagnachmittag – Zusammenfassung 362

Anhang A: Antworten auf die Wiederholungsfragen 365
Anhang B: Ergänzende Probleme 383
Anhang C: Was ist auf der CD-Rom 395
Index 397