

HANSER



Leseprobe

zu

Programmieren trainieren

Luigi Lo Iacono
Stephan Wiefling
Michael Schneider

ISBN (Buch): 978-3-446-45486-6

ISBN (E-Book): 978-3-446-45503-0

Weitere Informationen und Bestellungen unter

<http://www.hanser-fachbuch.de/>

sowie im Buchhandel

© Carl Hanser Verlag, München

Inhalt

Vorwort	XVII
Danksagung	XIX
1 Einleitung	1
1.1 Wozu sollte ich programmieren lernen?	1
1.2 Wie kann mir dieses Buch dabei helfen?	2
1.3 Was muss ich mitbringen?	2
1.4 Wie geht das vonstatten?	3
1.5 Was muss ich sonst noch wissen?	5
2 Einführung in die Programmierung	7
2.1 Warmup	7
2.2 Workout	11
W.2.1 Three-Two-One – Mein erstes Programm	11
W.2.2 Weihnachtsbaum	13
W.2.3 Perlenkette	15
W.2.4 Die erste Zeichnung	16
W.2.5 Raupe Allzeitappetit	18
W.2.6 Klötzchen-Kunst	19
W.2.7 Nachteule	20
W.2.8 GhettoBlaster	21
W.2.9 Hallo Bello!	23
3 Variablen, Datentypen, Operatoren und Ausdrücke	25
3.1 Warmup	25
3.2 Workout	28
W.3.1 Einfache Rechenaufgaben	28
W.3.2 Perlenkette 2.0	29

W.3.3	Blutalkoholkonzentration	30
W.3.4	Stoffwechselrate	32
W.3.5	Baumstammvolumen	34
W.3.6	Körperoberfläche	35
W.3.7	RGB nach CMYK	36
W.3.8	Tic-Tac-Toe-Spielfeld	38
4	Kontrollstrukturen	39
4.1	Warmup	39
4.2	Workout	43
W.4.1	Maximum bestimmen	43
W.4.2	Summe berechnen	44
W.4.3	Tippspiel	45
W.4.4	PIN-Code-Generator	46
W.4.5	Dominosteine	47
W.4.6	Radialer Farbverlauf	48
W.4.7	Ladevorgang-Rädchen	49
W.4.8	Windrad	51
W.4.9	Rotierte Dreiecke	52
W.4.10	Moderne Kunst	53
W.4.11	Schachbrett	55
W.4.12	Sinuskurve	56
W.4.13	Zahlen-Palindrom	57
W.4.14	Interaktiver Button	58
5	Funktionen	61
5.1	Warmup	61
5.2	Workout	63
W.5.1	Endliches Produkt	63
W.5.2	Fakultät	64
W.5.3	Konfektionsgröße	65
W.5.4	Schaltjahr Prüfung	66
W.5.5	Literzahlen umwandeln	67
W.5.6	LKW-Maut	68
W.5.7	Analoger Uhrzeiger	69
W.5.8	Körperoberfläche 2.0	70
W.5.9	Sportwetten	71
W.5.10	GPS-Luftlinie	73
W.5.11	IBAN-Generator	75

W.5.12	Sanduhr.....	77
W.5.13	Der faire Würfel.....	78
W.5.14	Quadrat mit Kreisausschnitten.....	79
W.5.15	Wurfparabel.....	81
W.5.16	Bogenschießen-Spiel.....	83
6	Arrays.....	85
6.1	Warmup.....	85
6.2	Workout.....	88
W.6.1	Wochentag.....	88
W.6.2	Tankfüllung.....	90
W.6.3	Rückwärtsausgabe.....	91
W.6.4	Bildvergrößerung.....	92
W.6.5	Partnervermittlung.....	93
W.6.6	Sitzplatzreservierung.....	94
W.6.7	Platztausch.....	95
W.6.8	Bestimmung minimale Distanz.....	96
W.6.9	Morsecode.....	97
W.6.10	Endlose Animation.....	98
W.6.11	Spiegeln.....	99
W.6.12	Reflexion.....	100
W.6.13	Greenscreen.....	102
W.6.14	Bild umdrehen und invertieren.....	103
W.6.15	Bild mit Schatten.....	104
W.6.16	Bild rotieren.....	105
W.6.17	Bildverkleinerung.....	106
W.6.18	Minimale Punktdistanz.....	107
W.6.19	Glatte Kurven.....	109
W.6.20	Bildausschnitt.....	111
W.6.21	Bild mit Rahmen.....	112
W.6.22	Memory-Spielfeldgenerator.....	113
W.6.23	Sudoku-Check.....	115
W.6.24	Medianfilter.....	116
7	Strings und Stringverarbeitung.....	119
7.1	Warmup.....	119
7.2	Workout.....	121
W.7.1	String-Kompression.....	121
W.7.2	Split-Funktion.....	122

W.7.3	Geldschein-Blütencheck	123
W.7.4	Starkes Passwort	125
W.7.5	E-Mail-Check	126
W.7.6	Prüfen auf eine korrekte Klammerung	127
W.7.7	Sternchenmuster	128
W.7.8	URL-Encoding	129
W.7.9	Telefonbuch bearbeiten	130
W.7.10	Webserver-Antwort verarbeiten	132
W.7.11	IMDB-Einträge verarbeiten	134
W.7.12	Geheimsprache	135
W.7.13	Ähnlich klingende Wörter	136
W.7.14	Textrahmen	138
W.7.15	JSON-Array	139
W.7.16	Barcode-Generator	140
8	Objektorientierung	143
8.1	Warmup	143
8.2	Workout	146
W.8.1	Schrittzähler	146
W.8.2	Body-Mass-Index	148
W.8.3	Songtextsuche	150
W.8.4	Passwortklasse	151
W.8.5	Kopffitness	153
W.8.6	Fernbedienung	155
W.8.7	Stoppuhr	156
W.8.8	Druckerwarteschlange	157
W.8.9	Tic Tac Toe	159
W.8.10	Zwischenablage	161
W.8.11	Temperaturgraph	163
W.8.12	Ambient Light	165
W.8.13	Verschlüsselung	169
W.8.14	Mastermind	171
9	Referenzdatentypen	173
9.1	Warmup	173
9.2	Workout	175
W.9.1	Kreis-Klasse	175
W.9.2	Mathematischer Bruch	176
W.9.3	Highscore-Liste	177

W.9.4	Adressbuch	179
W.9.5	Digitaler Bilderrahmen	181
W.9.6	Musikalbenanwendung	183
W.9.7	Koch-Website	185
W.9.8	Hotelzimmerverwaltung	187
W.9.9	Flughafen-Check-in	189
W.9.10	Polygonzug	191
W.9.11	Twitterwall	193
W.9.12	Fototagebuch	195
W.9.13	Partygäste	197
10	Vererbung	199
10.1	Warmup	199
10.2	Workout	201
W.10.1	Online-Shop	201
W.10.2	Gewässer	203
W.10.3	To-do-Liste	204
W.10.4	Lampen	206
W.10.5	Meeting-Protokoll	208
W.10.6	E-Book	210
W.10.7	Zoo	212
W.10.8	Audioeffekt-Player	213
W.10.9	Fahrtenbuch	214
A	Lösungen in Java	215
A.1	Download und Verwendung der elektronischen Lösungen	215
A.1.1	Download von GitHub	215
A.1.2	Öffnen der Programme	216
A.1.3	Tipp: Debugger	217
A.2	Einführung in die Programmierung	219
A.2.1	Three-Two-One – Mein erstes Programm	219
A.2.2	Weihnachtsbaum	219
A.2.3	Perlenkette	219
A.2.4	Die erste Zeichnung	220
A.2.5	Raupe Allzeitappetit	220
A.2.6	Klötzchen-Kunst	221
A.2.7	Nachteule	222
A.2.8	Ghettoblaster	223
A.2.9	Hallo Bello!	224

A.3	Variablen, Datentypen, Operatoren und Ausdrücke	226
A.3.1	Einfache Rechenaufgaben	226
A.3.2	Perlenkette 2.0	227
A.3.3	Blutalkoholkonzentration	227
A.3.4	Stoffwechselrate	228
A.3.5	Baumstammvolumen	228
A.3.6	Körperoberfläche	228
A.3.7	RGB nach CMYK	229
A.3.8	Tic-Tac-Toe-Spielfeld	229
A.4	Kontrollstrukturen	230
A.4.1	Maximum bestimmen	230
A.4.2	Summe berechnen	230
A.4.3	Tippspiel	231
A.4.4	PIN-Code-Generator	231
A.4.5	Dominosteine	232
A.4.6	Radialer Farbverlauf	232
A.4.7	Ladevorgang-Rädchen	233
A.4.8	Windrad	233
A.4.9	Rotierte Dreiecke	233
A.4.10	Moderne Kunst	234
A.4.11	Schachbrett	235
A.4.12	Sinuskurve	235
A.4.13	Zahlen-Palindrom	236
A.4.14	Interaktiver Button	237
A.5	Funktionen	239
A.5.1	Endliches Produkt	239
A.5.2	Fakultät	239
A.5.3	Konfektionsgröße	240
A.5.4	Schaltjahr Prüfung	241
A.5.5	Literzahlen umwandeln	242
A.5.6	LKW-Maut	242
A.5.7	Analoger Uhrzeiger	244
A.5.8	Körperoberfläche 2.0	245
A.5.9	Sportwetten	245
A.5.10	GPS-Luftlinie	246
A.5.11	IBAN-Generator	247
A.5.12	Sanduhr	249
A.5.13	Der faire Würfel	250

A.5.14	Quadrat mit Kreisausschnitten	251
A.5.15	Wurfparabel	252
A.5.16	Bogenschießen-Spiel	253
A.6	Arrays	258
A.6.1	Wochentag	258
A.6.2	Tankfüllung	259
A.6.3	Rückwärtsausgabe	260
A.6.4	Bildvergrößerung	261
A.6.5	Partnervermittlung	262
A.6.6	Sitzplatzreservierung	263
A.6.7	Platztausch	265
A.6.8	Bestimmung minimale Distanz	265
A.6.9	Morsecode	266
A.6.10	Endlose Animation	268
A.6.11	Spiegeln	269
A.6.12	Reflexion	270
A.6.13	Greenscreen	272
A.6.14	Bild umdrehen und invertieren	273
A.6.15	Bild mit Schatten	274
A.6.16	Bild rotieren	276
A.6.17	Bildverkleinerung	277
A.6.18	Minimale Punktdistanz	279
A.6.19	Glatte Kurven	280
A.6.20	Bildausschnitt	281
A.6.21	Bild mit Rahmen	283
A.6.22	Memory-Spielfeldgenerator	284
A.6.23	Sudoku-Check	286
A.6.24	Medianfilter	289
A.7	Strings und Stringverarbeitung	291
A.7.1	String-Kompression	291
A.7.2	Split-Funktion	292
A.7.3	Geldschein-Blütencheck	293
A.7.4	Starkes Passwort	296
A.7.5	E-Mail-Check	297
A.7.6	Prüfen auf eine korrekten Klammerung	298
A.7.7	Sternchenmuster	299
A.7.8	URL-Encoding	301
A.7.9	Telefonbuch bearbeiten	302

A.7.10	Webserver-Antwort verarbeiten.....	304
A.7.11	IMDB-Einträge verarbeiten.....	306
A.7.12	Geheimsprache.....	307
A.7.13	Ähnlich klingende Wörter.....	308
A.7.14	Textrahmen.....	309
A.7.15	JSON-Array.....	310
A.7.16	Barcode-Generator.....	311
A.8	Objektorientierung.....	316
A.8.1	Schrittzähler.....	316
A.8.2	Body-Mass-Index.....	317
A.8.3	Songtextsuche.....	318
A.8.4	Passwortklasse.....	318
A.8.5	Kopffitness.....	321
A.8.6	Fernbedienung.....	322
A.8.7	Stoppuhr.....	323
A.8.8	Druckerwarteschlange.....	325
A.8.9	Tic Tac Toe.....	326
A.8.10	Zwischenablage.....	329
A.8.11	Temperaturgraph.....	330
A.8.12	Ambient Light.....	333
A.8.13	Verschlüsselung.....	335
A.8.14	Mastermind.....	338
A.9	Referenzdatentypen.....	341
A.9.1	Kreis-Klasse.....	341
A.9.2	Mathematischer Bruch.....	342
A.9.3	Highscore-Liste.....	343
A.9.4	Adressbuch.....	345
A.9.5	Digitaler Bilderrahmen.....	349
A.9.6	Musikalbenanwendung.....	351
A.9.7	Koch-Website.....	353
A.9.8	Hotelzimmerverwaltung.....	356
A.9.9	Flughafen-Check-in.....	358
A.9.10	Polygonzug.....	361
A.9.11	Twitterwall.....	363
A.9.12	Fototagebuch.....	364
A.9.13	Partygäste.....	367
A.10	Vererbung.....	370
A.10.1	Online-Shop.....	370

A.10.2	Gewässer	373
A.10.3	To-do-Liste	376
A.10.4	Lampen	380
A.10.5	Meeting-Protokoll	381
A.10.6	E-Book	384
A.10.7	Zoo	389
A.10.8	Audioeffekt-Player	391
A.10.9	Fahrtenbuch	394
B	Lösungen in Python	397
B.1	Download und Verwendung der elektronischen Lösungen	397
B.1.1	Download von GitHub	397
B.1.2	Öffnen der Programme	397
B.2	Einführung in die Programmierung	399
B.2.1	Three-Two-One – Mein erstes Programm	399
B.2.2	Weihnachtsbaum	399
B.2.3	Perlenkette	399
B.2.4	Die erste Zeichnung	400
B.2.5	Raupe Allzeitappetit	400
B.2.6	Klötzchenkunst	401
B.2.7	Nachteule	402
B.2.8	Ghettoblaster	403
B.2.9	Hallo Bello!	404
B.3	Variablen, Datentypen, Operatoren und Ausdrücke	406
B.3.1	Einfache Rechenaufgaben	406
B.3.2	Perlenkette 2.0	407
B.3.3	Blutalkoholkonzentration	407
B.3.4	Stoffwechselrate	408
B.3.5	Baumstammvolumen	408
B.3.6	Körperoberfläche	408
B.3.7	RGB nach CMYK	409
B.3.8	Tic-Tac-Toe-Spielfeld	409
B.4	Kontrollstrukturen	410
B.4.1	Maximum bestimmen	410
B.4.2	Summe berechnen	410
B.4.3	Tippspiel	410
B.4.4	PIN-Code-Generator	411
B.4.5	Dominosteine	411

B.4.6	Radialer Farbverlauf	412
B.4.7	Ladevorgang-Rädchen	412
B.4.8	Windrad	413
B.4.9	Rotierte Dreiecke	413
B.4.10	Moderne Kunst	414
B.4.11	Schachbrett	414
B.4.12	Sinuskurve	415
B.4.13	Zahlen-Palindrom	416
B.4.14	Interaktiver Button	417
B.5	Funktionen	418
B.5.1	Endliches Produkt	418
B.5.2	Fakultät	418
B.5.3	Konfektionsgröße	419
B.5.4	Schaltjahr Prüfung	420
B.5.5	Literzahlen umwandeln	420
B.5.6	LKW-Maut	421
B.5.7	Analoger Uhrzeiger	422
B.5.8	Körperoberfläche 2.0	423
B.5.9	Sportwetten	423
B.5.10	GPS-Luftlinie	424
B.5.11	IBAN-Generator	425
B.5.12	Sanduhr	426
B.5.13	Der faire Würfel	427
B.5.14	Quadrat mit Kreisabschnitten	428
B.5.15	Wurfparabel	429
B.5.16	Bogenschießen-Spiel	430
B.6	Arrays	434
B.6.1	Wochentag	434
B.6.2	Tankfüllung	435
B.6.3	Rückwärtsausgabe	436
B.6.4	Bildvergrößerung	436
B.6.5	Partnervermittlung	438
B.6.6	Sitzplatzreservierung	439
B.6.7	Platztausch	440
B.6.8	Bestimmung minimale Distanz	441
B.6.9	Morsecode	442
B.6.10	Endlose Animation	443
B.6.11	Spiegeln	444

B.6.12	Reflexion.....	445
B.6.13	Greenscreen.....	446
B.6.14	Bild umdrehen und invertieren	447
B.6.15	Bild mit Schatten	448
B.6.16	Bild rotieren.....	450
B.6.17	Bildverkleinerung	451
B.6.18	Minimale Punktdistanz	453
B.6.19	Glatte Kurven	454
B.6.20	Bildausschnitt	455
B.6.21	Bild mit Rahmen	457
B.6.22	Memory-Spielfeldgenerator	458
B.6.23	Sudoku-Check	460
B.6.24	Medianfilter	462
B.7	Strings und Stringverarbeitung	464
B.7.1	String Kompression	464
B.7.2	Split-Funktion	465
B.7.3	Geldschein-Blütencheck	465
B.7.4	Starkes Passwort.....	467
B.7.5	E-Mail-Check	468
B.7.6	Prüfen auf eine korrekten Klammerung	469
B.7.7	Sternchenmuster	470
B.7.8	URL-Encoding.....	471
B.7.9	Telefonbuch bearbeiten	471
B.7.10	Webserver-Antwort verarbeiten.....	473
B.7.11	IMDB-Einträge verarbeiten.....	474
B.7.12	Geheimsprache.....	475
B.7.13	Ähnlich klingende Wörter.....	476
B.7.14	Textrahmen	477
B.7.15	JSON-Array.....	478
B.7.16	Barcode-Generator	479
B.8	Objektorientierung	482
B.8.1	Schrittzähler	482
B.8.2	Body-Mass-Index.....	482
B.8.3	Songtextsuche	483
B.8.4	Passwortklasse.....	484
B.8.5	Kopffitness	486
B.8.6	Fernbedienung	487
B.8.7	Stoppuhr	488

B.8.8	Druckerwarteschlange	489
B.8.9	Tic Tac Toe.....	490
B.8.10	Zwischenablage	492
B.8.11	Temperaturgraph.....	494
B.8.12	Ambient Light	496
B.8.13	Verschlüsselung	499
B.8.14	Mastermind	502
B.9	Referenzdatentypen	504
B.9.1	Kreis-Klasse	504
B.9.2	Mathematischer Bruch	505
B.9.3	Highscore-Liste.....	506
B.9.4	Adressbuch.....	507
B.9.5	Digitaler Bilderrahmen	510
B.9.6	Musikalbenanwendung	511
B.9.7	Koch-Website	513
B.9.8	Hotelzimmerverwaltung	515
B.9.9	Flughafen-Check-in	517
B.9.10	Polygonzug.....	519
B.9.11	Twitterwall	520
B.9.12	Fototagebuch	522
B.9.13	Partygäste	524
B.10	Vererbung	527
B.10.1	Online-Shop	527
B.10.2	Gewässer	529
B.10.3	To-do-Liste	532
B.10.4	Lampen	534
B.10.5	Meeting-Protokoll.....	536
B.10.6	E-Book	538
B.10.7	Zoo	541
B.10.8	Audioeffekt-Player	543
B.10.9	Fahrtenbuch	545
C	Installation Processing	549
C.1	Einleitung	549
C.2	Windows	549
C.3	Mac OS X.....	550
C.4	Linux.....	551
C.5	Aktivierung des Python Mode.....	552

Vorwort

Nerds sind in. Diese liebenswerten Zeitgenossen mit dem vielen Spezialwissen und den kindlichen Vorlieben für Superhelden werden lange nicht mehr nur komisch beäugt. Im Gegenteil. Sie selbst sind nunmehr Stars in vielen Fernsehserien, und ihr modischer Stil ist allgemein akzeptiert. Diese Entwicklung kommt auch der Programmierung zugute. Lange Zeit galt diese Fertigkeit als ein Gebiet, das den Nerds vorbehalten ist. Dem ist nicht so! Es muss nur der Mut aufgebracht werden, sich damit auseinanderzusetzen. Dann wird schnell klar, was mit der Programmierung alles umgesetzt werden kann. Die Bandbreite ist groß und wird durch aktuelle Trends stetig befeuert. Insbesondere durch die Digitalisierung und Vernetzung vieler Alltagsgegenstände finden sich Softwareprogramme vermehrt jenseits gängiger Anwendungsfälle im betrieblichen Kontext von Unternehmen wieder. Also, keine Scheu und ran ans Programmieren!

Mir selbst bereitet das Programmieren viel Freude. Zudem ist es mir eine Herzensangelegenheit, mein Programmier-Knowhow und meine Erfahrung an andere weiterzugeben. Ich weiß aus vielen Schulungen sehr genau, was es für Hürden und Stolpersteine beim Programmieren lernen gibt und wie diesen zu begegnen ist. **Gutem Trainingsmaterial kommt dabei eine zentrale Rolle zu.**

Die Autoren Lo Iacono, Wiefling und Schneider schließen hier eine wichtige Lücke. Sie versorgen dich mit vielen Trainingsaufgaben, die dir helfen werden, die wesentlichen Programmierkonzepte wirklich zu verstehen. Und mehr noch. Du kannst und solltest so lange mit den vielen Aufgaben trainieren, bis der Groschen tatsächlich gefallen ist. Das ist wichtig. Denn erst dann wirst Du in der Lage sein, mit dem erlernten Handwerkszeug auch selbstständig Entwicklungsaufgaben bewältigen und lösen zu können. Genau da sollst du hin. Viele Lehrformate gehen hier nicht weit genug. Die falsche Annahme ist dabei häufig, dass ein Beispiel und eine Übungsaufgabe zum Verständnis ausreichen. Weit gefehlt. Es fängt schon damit an, dass nicht jeder mit dem gegebenen Beispiel oder der gestellten Übungsaufgabe etwas anfangen kann. Hier schafft das vorliegende Buch Abhilfe, und es gehört damit in die „Einstieg in die Programmierung“-Ecke deines Bücherregals.

Dirk Louis, im Januar 2018

1

Einleitung

■ 1.1 Wozu sollte ich programmieren lernen?

Weil Du es kannst und weil die Programmierung **das Werkzeug des 21. Jahrhunderts** ist. Die Bundeskanzlerin Frau Angela Merkel hat erst unlängst in einem Interview mit YouTubern das Programmieren auf eine Stufe mit den Grundfertigkeiten Lesen, Schreiben und Rechnen gestellt (<https://youtu.be/Uq2zIzscPgY?t=12m18s>). Programmieren ist lange nicht mehr nur etwas für Experten, die das studiert haben. Durch den Einzug des Digitalen in alle Branchen und den Alltag können viel mehr als nur Informatiker von der Programmierung profitieren und damit ihre Ideen erproben und verwirklichen. Beispiele kannst du unzählige finden. Lass' uns hier nur einige zur Verdeutlichung kurz anreißen. Dir fallen dann bestimmt selbst viele weitere Beispiele ein.

Angenommen, du bist **Künstler** und hast bisher mit den klassischen Materialien und Techniken deiner Disziplin gearbeitet. Für deine neueste Projektidee möchtest du mit regelmäßigen Formen und Farben experimentieren, wie es z.B. Sol LeWitt in seinem künstlerischen Schaffen getan hat (https://de.wikipedia.org/wiki/Sol_LeWitt). Das erfordert viel Fleiß, Geduld und Präzision. Da du deine Zeit lieber damit verbringen möchtest, an spannenden neuen Konstruktionen und deren Wirkung zu experimentieren, anstatt diese in langwierigen und teils monotonen Arbeitsschritten erst erstellen zu müssen, wünschst du dir einen Automatismus dafür, der das für dich erledigt. Dies kann ein eigens geschriebenes Computerprogramm leisten. Ist ein solches geschrieben, liegen die Vorteile auf der Hand. Veränderungen an den Farben, der Größe sowie Anordnungen der Formen usw. sind umgehend gemacht. Auch das Ausgabeformat kann leicht angepasst werden, um das Kunstwerk in vielfältiger Art und Weise zu drucken oder aus einem Rohling zu fräsen. Pioniere der computergenerierten Kunst sind z.B. Manfred Mohr, Joseph Nechvatal, Olga Kisseleva und John Lansdown.

Als **Veranstaltungstechniker** sieht man sich heute immer stärker mit Anforderungen von Kunden herausgefordert, die nach noch nicht dagewesenen Hinguckern verlangen. Hierfür gibt es naturgemäß keine fertigen Lösungen, die man aus dem Regal ziehen kann. Somit siehst du dich auf der einen Seite immer mit neuen spannenden Entwicklungsaufgaben konfrontiert, musst dafür aber auf der anderen Seite adäquate Lösungen entwickeln. Diese bedingen eigentlich immer auch Software, die es zu programmieren gilt.

Im letzten fiktiven Szenario wollen wir ins **Internet der Dinge** abtauchen. Mit diesem Schlagwort wird der allgemeine Trend bezeichnet, mit dem die Digitalisierung und die Vernetzung

im Gewand des Internets stetig in Gegenstände des alltäglichen Gebrauchs diffundieren. Der smart gewordene Fernseher ist ein Paradebeispiel hierfür. Einige neue Anwendungen findest du toll, willst aber noch nicht in neue Produkte investieren. Die alten tun es ja noch. So findest du es z.B. praktisch, im Supermarkt einen Blick in deinen Kühlschrank werfen zu können, um zu sehen, ob es genügend Frühstückseier fürs Wochenende gibt. Der Kühlschrank ist schnell für diesen Anwendungsfall erweitert. Mit einer batteriebetriebenen Kamera, einem LED-Licht und etwas Programmierung kannst du bald via Smartphone-App in deinen Kühlschrank gucken.

Das soll zeigen, was dir alles an Möglichkeiten offen steht, wenn du die Programmierung als ein Werkzeug verstehst und dich dessen bemächtigst.

■ 1.2 Wie kann mir dieses Buch dabei helfen?

Vor den Erfolg haben die Götter allerdings den Schweiß gesetzt. Diese Tatsache hat der griechische Dichter und Geschichtsschreiber *Hesiod* bereits vor langer Zeit festgestellt und dann so zutreffend formuliert. Dieser Ausspruch trifft unseres Erachtens kaum besser auf etwas zu als die Programmierung. Es gehört eine ordentliche Portion Arbeit dazu, bis der Groschen fällt und man die wesentlichen Programmierkonzepte verstanden hat. Erst dann wird man in der Lage sein, Aufgabenstellungen jeglicher Couleur selbstständig angehen und erfolgreich bewältigen zu können. Wir wollen dir diese notwendigen Mühen nicht verschweigen. Unserer Erfahrung nach scheitert so mancher Einstieg genau an dieser Hürde.

Unser Ansatz ist deshalb, durch **viele spannende Programmieraufgaben** das nötige Material zum Trainieren bereitzustellen. Du wirst in diesem Buch im Wesentlichen Aufgabenstellungen von uns bekommen, an denen du Aufgabe für Aufgabe alle relevanten Programmierkonzepte üben kannst. Damit dir dabei nicht die Laune vergeht, haben wir uns viel Mühe beim Zusammenstellen der Aufgaben gegeben. Es wird deutlich über die meisten "Lehrbuchaufgaben" hinausgehen und sich, soweit möglich, erheblich näher an praktischen Anwendungsszenarien orientieren. Der klassische Lehrbuchstil handelt sich meist an Aufgabenstellungen aus der Mathematik entlang. Das ist wichtig, und daher haben auch wir das ab und an mit dir vor. Wir können aber auch verstehen, wenn derartige Aufgaben nicht jedem liegen, um etwas Neues zu lernen. Daher programmierst du eher Anwendungen, die etwas Nützliches tun oder etwas hübsch Anzuschauendes generieren. Als Appetitanreger haben wir im nachfolgenden Bild 1.1 schon mal drei Beispiele aus dem Buch für dich.

Diese drei Bilder zeigen exemplarisch, was du mit unserem Trainingsprogramm programmieren sollst und auch können wirst, wenn du fleißig am Ball bleibst. Es lohnt sich!

■ 1.3 Was muss ich mitbringen?

Nicht viel! Mit diesem Buch können wir es nicht leisten, dir die Basics beizubringen. Das musst du halt selbst tun oder du bekommst es in irgendeiner Form gezeigt. Wir erklären zu

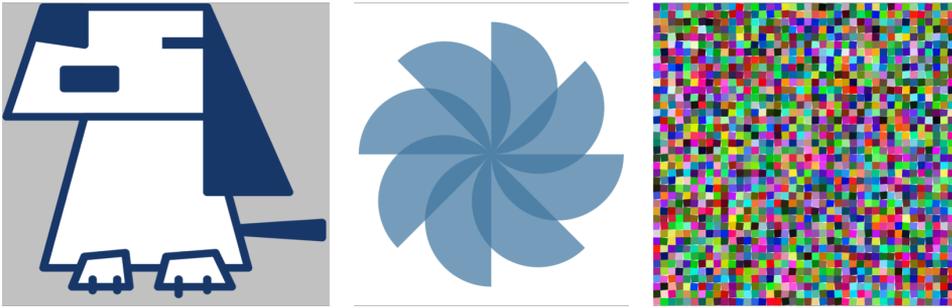


Bild 1.1 Drei Beispielbilder, die du programmieren wirst

Beginn eines jeden Kapitels nochmal kurz die im Fokus stehenden Übungsschwerpunkte. Das ist mehr eine Gedächtnisstütze und soll als Warm-up dienen, falls du es überhaupt brauchst. Wir gehen dabei nochmals kurz auf die wesentlichen Konzepte ein und erläutern Besonderheiten in den Programmierumgebungen, für die wir Beispiellösungen bereitstellen.

■ 1.4 Wie geht das vonstatten?

Wie schon gesagt, ist das hier ein Trainingsbuch fürs Programmieren. Wir stellen dir **120 Übungsaufgaben mit Lösungsvorschlägen** zum Training bereit. Die grundlegende Struktur gleicht dabei derer gängiger Ressourcen zur Einführung in die Programmierung. Es geht mit dem Aufbau erster einfacher Programme los und wird durch das Hinzukommen von Programmierkonzepten wie Variablen, Datentypen, Operatoren, Ausdrücke, bedingte Anweisungen, Wiederholungsanweisungen, Funktionen, Arrays, Strings bis hin zur Objektorientierung stetig erweitert. Damit wir uns - ohne unnötiges Störfeuer und Ablenkung - auf das Kernthema des jeweiligen Kapitels konzentrieren können, bestehen die ersten Programme der Kapitel 2, 3 und 4 zunächst aus Anweisungen. Erst in den nachfolgenden Kapiteln 5 bis 9 kommen dann Strukturelemente für den Programmcode in Form von Funktionen sowie Klassen und Objekten hinzu. Wir werden dir Kapitel für Kapitel Trainingsaufgaben stellen, für die du dir ein passendes Programm überlegen und dieses dann in einer Programmiersprache vollständig angeben sollst.

Jede Trainingsaufgabe ist nach einem **festen Schema** aufgebaut (siehe Bild 1.2). Um jede Aufgabe eindeutig identifizieren zu können, haben wir diese mit einem eindeutigen Namen, einer eindeutigen Nummer und einem Piktogramm versehen. Das wird dir insbesondere dabei helfen, dich mit Freunden, Mitschülern, Kommilitonen, Kollegen oder der Community über die Aufgaben auszutauschen. Auch unsere Lösungsvorschläge im Anhang und online wirst du auf diese Weise spielend der jeweiligen Aufgabe zuordnen können. Die Identifizierungsnummer ist dem jeweiligen Buchkapitel zugeordnet. Das einfache Zurückspringen zur Aufgabenstellung im Buch ist damit auch gewährleistet.

Die Aufgaben haben wir unserem Dafürhalten nach in den Kategorien **Schwierigkeit, Kreativität** und **Zeitaufwand** bewertet und sortiert. Einfachere Aufgaben, die im Vergleich eher

Wenn du völlig auf dem Schlauch stehst und gar keinen Zugang findest, sind am Ende einer jeden Aufgabe **algorithmische Tipps** aufgelistet, die dich einem möglichen Lösungsansatz näher bringen sollen. Bitte nutze diese Tipps und versuche unbedingt, selbst eine Lösung herzustellen, bevor du dir unsere Lösungsvorschläge im Anhang des Buchs anschaust. Nur auf diese Weise kommst du genügend ins Schwitzen, um nachher wirklich behaupten zu können, das Programmieren auch selbstständig zu beherrschen.

■ 1.5 Was muss ich sonst noch wissen?

Damit du möglichst effektiv und fokussiert trainieren kannst, haben wir die Trainingsumgebung für dich von unnötigem Ballast entschlackt. Du sollst nicht schon bei der Installation, Konfiguration und Verwendung der Programmierumgebung die Lust am Programmieren bzw. die Sicht auf das Wesentliche verlieren. Wir stützen uns daher auf ein einziges Werkzeug, mit dem du in **Java und Python** das Programmieren trainieren kannst. Es handelt sich dabei um das frei und kostenlos verfügbare Tool mit dem Namen **Processing**, das du unter der Web-Adresse <https://processing.org/> abrufen kannst. Hier findest du auch viele weitere Informationen und Dokumentationen rund um Processing. Im Anhang C haben wir Installationsanleitungen für die gängigen Betriebssysteme Windows, macOS und Linux beigelegt.

Dass wir auf Processing abstellen, soll aber nicht heißen, dass du mit dem Erlernten in der Praxis nicht viel anfangen kannst. Ganz im Gegenteil! Die Programmiersprachen Java und Python gehören zu den am weitest verbreiteten Sprachen, und schließlich kommt es im Wesentlichen auf die Programmierkonzepte an. Wenn du diese intensiv trainiert und dadurch verinnerlicht hast, dann bist du bereit, alle möglichen Aufgabenstellungen programmatisch zu lösen. Dann haben wir unser gemeinsames Ziel erreicht. Die Verwendung professionellerer Programmierumgebungen wie z.B. Eclipse, IntelliJ oder PyCharm ist dann ein Klacks. Darüber müssen wir dann nicht mehr viele Worte verlieren.

Die Quelltexte zum Buch – unsere Lösungsvorschläge – haben wir auf dem Onlinedienst **GitHub** für dich bereitgelegt. Du findest sie unter der Adresse <https://github.com/protrain>. Wie du damit umgehst, erklären wir für alle gängigen Desktop-Betriebssysteme im Anhang A.1 (für Java) bzw. Anhang B.1 (für Python).

Mit dem Kauf des Buches soll aber noch nicht Schluss sein. Wir würden uns sehr freuen, von dir zu hören. Über GitHub kannst du uns z.B. auf Fehler im Buch oder in den Lösungen aufmerksam machen. Wir tragen das dann in die Erata-Liste ein bzw. korrigieren die Programme. Außerdem kannst du uns auch deine Lösung(en) bereitstellen. Wenn diese einen eigenen Lösungsweg zeigen, nehmen wir sie mit in das Repositorie auf. Selbiges gilt für Lösungen in anderen Programmiersprachen. Achte aber bitte hierbei darauf, dass es für die Sprache eine ähnliche einfache und umfangreiche Programmierumgebung gibt, wie es Processing für Java und Python ist. Wenn du sonstige Anregungen zur Verbesserung hast oder Ideen für weitere Aufgaben beisteuern möchtest, freuen wir uns von dir zu hören.

Hoffentlich konnten wir dein Interesse wecken und dir unseren Ansatz zum Programmieren lernen schmackhaft machen. Jedenfalls würden wir uns sehr freuen, gemeinsam mit dir das Programmieren zu trainieren.



2

Einführung in die Programmierung

■ 2.1 Warmup

Dein Training beginnt in diesem Kapitel mit ersten einfachen Programmen. Dazu musst du wissen, wie der grundlegende Aufbau eines Programms sowie der Aufbau der Anweisungen in einer bestimmten Programmiersprache sind. Letzteres gehört zur sogenannten **Syntax** einer Programmiersprache. So wie z.B. die Syntax einer natürlichen Sprache Prinzipien und Regeln des Wort- und Satzbaus festlegt, so legt die Syntax einer Programmiersprache das Vokabular und den Aufbau von Anweisungen fest.

Für die allerersten Programme, die du entwickeln sollst, genügt zunächst die allereinfachste Struktur überhaupt. Hierbei werden Programme als eine lineare Abfolge von Anweisungen angegeben. Anweisungen verfügen immer über einen Namen und eine Liste von Parametern, die die Anweisung verarbeiten soll. Um den Anweisungsnamen von der Parameterliste unterscheiden zu können, werden die Parameter häufig eingeklammert und dem Anweisungsnamen nachgestellt.

```
nameAnweisung(parameter);
```

Verfügt die Parameterliste über mehrere Einträge, so werden diese mit Komma (,) voneinander getrennt.

```
nameAnweisung(parameter1, parameter2);
```

Parameterlose Anweisungen sind durch ein leeres Klammersymbol gekennzeichnet.

```
nameAnweisung();
```

Um mehrere Anweisungen voneinander unterscheiden zu können, wird dafür ein Trennzeichen in der Syntax einer Programmiersprache festgelegt. In Java ist das das Semikolon (;). Das folgende Beispiel zeigt ein abstraktes Programm, das sich aus sieben Anweisungen zusammensetzt, die in der angegebenen Reihenfolge ausgeführt werden. Die lineare Programmabfolge führt die programmierten Anweisungen zeilenweise von links nach rechts beginnend mit der obersten Zeile aus.

```
Anweisung1(); Anweisung2(); Anweisung3(); Anweisung4(); Anweisung5();  
Anweisung6(); Anweisung7();
```

Durch diese Syntaxregel können die einzelnen Anweisungen separiert werden, unabhängig davon, wie du diese in die Quelltextdatei schreibst. Zur besseren Lesbarkeit empfehlen wir dir aber, dich auf eine Anweisung pro Zeile zu beschränken und die Anweisungen untereinander zu schreiben.

```
Anweisung1();  
Anweisung2();  
Anweisung3();  
Anweisung4();  
Anweisung5();  
Anweisung6();  
Anweisung7();
```

Die Programmiersprache Python legt in seiner Syntax als Trennzeichen von Anweisungen den Zeilenumbruch fest. Ein Zeilenumbruch kann je nach Betriebssystem aus einem oder zwei Zeichen bestehen ('\n', '\r' oder '\r\n').

In der Programmierliteratur hat sich das „Hello World!“-Programm als einführendes Beispiel zur Darstellung der grundlegenden Syntax eines einfachen Programms in einer bestimmten Programmiersprache etabliert. Das Hello-World-Programm gibt in der Konsole einen einfachen Text aus, nämlich Hello World!. Wir wollen es zur Konkretisierung der einführenden Erläuterungen verwenden.

Java:

```
print("Hallo_Welt!");
```

Python:

```
print("Hallo_Welt!")
```

Die `print()`-Anweisung bekommt einen Parameter übergeben. Dieser enthält den Text, den die Anweisung in der Konsole ausgeben soll. Um den Text eingrenzen zu können, wird dieser von doppelten Anführungszeichen (") eingerahmt.

Die Aufgaben dieses Kapitels drehen sich um derartige Programme. Deine Aufgabe wird es sein, die zur Lösung der Aufgabenstellung benötigten Anweisungen zu identifizieren und diese dann in einer geeigneten Abfolge zu platzieren. Welche Anweisungen eine Programmiersprache im Standardumfang bereitstellt, sind in der Referenzdokumentation aufgeführt. Die Referenz der von Processing bereitgestellten Anweisungen kann im Internet eingesehen werden:

- <https://processing.org/reference/> (Java)
- <http://py.processing.org/reference/> (Python)

Referenzen sind sehr umfangreich. Dies gilt auch für die von Processing. Es kann daher etwas dauern, bist du dich darin zurechtfindest. Für die in diesem Kapitel bereitgestellten Trainingsaufgaben sind insbesondere Funktionen zur Ausgabe von Texten in der Konsole und Funktionen zur Ausgabe elementarer geometrischer Formen im grafischen Ausgabefenster wichtig. Um dir das Auffinden dieser Anweisungen zu erleichtern, führen wir dir in der nachfolgenden Auflistung die relevanten auf.

- *Konsolenausgabe*
 - https://processing.org/reference/print_.html (Java)
 - <http://py.processing.org/reference/print.html> (Python)
- *Linie*
 - https://processing.org/reference/line_.html (Java)
 - <http://py.processing.org/reference/line.html> (Python)

- *Dreieck*
 - https://processing.org/reference/triangle_.html (Java)
 - <http://py.processing.org/reference/triangle.html> (Python)
- *Rechteck*
 - https://processing.org/reference/rect_.html (Java)
 - <http://py.processing.org/reference/rect.html> (Python)
- *Viereck*
 - https://processing.org/reference/quad_.html (Java)
 - <http://py.processing.org/reference/quad.html> (Python)
- *Ellipse*
 - https://processing.org/reference/ellipse_.html (Java)
 - <http://py.processing.org/reference/ellipse.html> (Python)
- *Kreisausschnitt*
 - https://processing.org/reference/arc_.html (Java)
 - <http://py.processing.org/reference/arc.html> (Python)

Um sich mit der Funktionsweise der Anweisungen vertraut zu machen, empfehlen wir dir, die Beschreibung in der Referenz aufmerksam zu lesen. Dies ist eine wichtige Grundfertigkeit, die zum Programmieren dazu gehört.

Verwendet werden wir in diesem Buch die Entwicklungsumgebung Processing. Hiermit können wir Programme sowohl in Java als auch in Python schreiben. Processing bietet nicht nur den Vorteil der einfachen Installation auf nahezu allen Betriebssystemen. Wir können damit auch sehr einfach (grafische) Programme auf Basis von Anweisungen schreiben. Aber auch höherwertige Konzepte, wie wir sie in den späteren Kapiteln umsetzen werden, sind in Processing möglich. Perfekte Voraussetzungen also zum Trainieren deiner Programmier Techniken mit diesem Buch.

Alle Installationsschritte von Processing findest du in Anhang C.1. Wie du an die digitalen Quelltexte unserer Lösungsvorschläge zu einzelnen Aufgaben kommst und wie du sie in Processing öffnest, steht im Anhang A.1.1 für Java und Anhang B.1.1 für Python.

Dateien mit Quelltext können wir in Processing mit Klick auf *Datei* → *Öffnen...* laden. In Bild 2.1 haben wir zum Beispiel eine solche Datei geöffnet. Dort können wir gut die grafische Benutzeroberfläche von Processing erkennen:

- Mit dem Start- und Stopp-Button (1) kannst du deinen Java- bzw. Python-Code ausführen.
- Um vom Java- auf den Python-Modus zu wechseln, kannst du den Modus-Auswahlreiter (2) verwenden. Wie du den Python-Modus in Processing installierst, steht in Anhang C.5. Links neben diesem Button ist der integrierte Debugger, den du zur Analyse von Java-Code verwenden kannst. Mehr dazu findest du in Anhang A.1.3.
- In der Mitte der Benutzeroberfläche (3) steht der eigentliche Quelltext. In diesen Bereich kannst du deinen Java- bzw. Python-Code hineinschreiben.
- Entsprechende Ausgaben in der Konsole findest du im darunter liegenden Bereich (4). Hier werden auch auftretende Fehler im Code angezeigt, sofern es welche gibt.

Nach der Einrichtung von Processing und dem Lesen der Einführung solltest du für dieses Kapitel ausgerüstet sein. In dem Sinne: Viel Spaß bei den ersten Aufgaben!

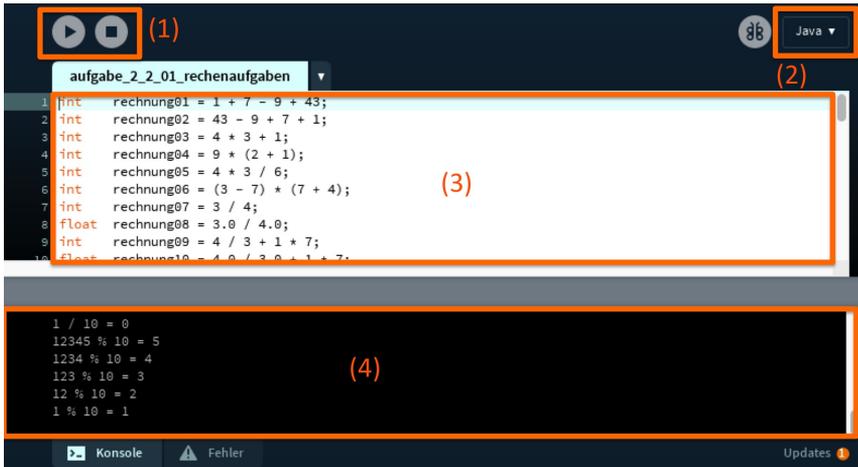


Bild 2.1 So sieht die grafische Benutzeroberfläche von Processing aus.

■ 2.2 Workout

W.2.1 Three-Two-One – Mein erstes Programm

Schwierigkeit



Zeitaufwand



Kreativität



Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau von Programmanweisungen
- Ausgabe in der Konsole

Beschreibung

Wir wollen ein erstes Programm schreiben. Der Klassiker hierfür ist die Ausgabe eines Texts – meist der Text `Hello World` – in der Konsole. Dazu braucht es in der Regel nur eine einzige Anweisung. An dieser kannst du aber bereits den Aufbau von Anweisungen und einfachen Programmen nachvollziehen und trainieren. Los geht's!

Aufgabenstellung

Schreibe ein Programm, das den Text `Three-Two-One - Takeoff!` in der Konsole ausgibt. Wenn dein Programm funktioniert, solltest du den angegebenen Text in der Konsole lesen können, so wie nachfolgend exemplarisch zu sehen ist:

```
Three-Two-One - Takeoff!
```

Wenn das geklappt hat, dann mach' doch einfach weiter und modifiziere dein erstes Programm nach deinen Wünschen. Ändere z.B. den Text oder füge weitere Anweisungen zur Textausgabe hinzu. Reflektiere dabei, wie dein Programm auf die Änderungen reagiert. Wenn du das Resultat hast kommen sehen und es ist nichts Unerwartetes bei der Ausführung deines Programms passiert, hast du es im Griff und verstanden, wie Anweisungen und einfache Programme aufgebaut sind.

Testfälle

Zum Testen deines Programms brauchst du in dieser Aufgabe noch keine Testdaten. Starte dein Programm und prüfe, ob die geforderte Ausgabe in der Konsole ausgegeben wird.

(Algorithmische) Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Gib nicht auf. Du solltest es solange probieren, bis es klappt. Das nennt man *Trial and Error* (Versuch und Irrtum). Versuch es weiter! Vermutlich bist du schon nah dran an der Lösung, denn der Fehler liegt sehr häufig im Detail.

- Wir benötigen eine passende Anweisung, die uns die Programmiersprache zur Ausgabe von Daten in der Konsole bereitstellt. Wie lautet diese?
- Anweisungen folgen einem festgelegten Aufbau. Hier schleichen sich schon mal Tippfehler ein. Was sagen denn die Fehlermeldungen, wenn du versuchst, dein Programm zu starten?

W.2.2 Weihnachtsbaum

Schwierigkeit



Zeitaufwand



Kreativität



Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe in der Konsole

Beschreibung

Wir wollen jetzt ein erstes Muster in die Konsole schreiben. Dafür werden wir bestimmte Zeichen so oft hinter- und untereinander schreiben, bis sich daraus eine Form ergibt. Diese Form des „Malens“ ist bei vielen Konsolenprogrammen üblich und wird auch heute noch verwendet.

Aufgabenstellung

Schreibe ein Programm, das das folgende Muster in der Konsole ausgibt:

```

*
***
*****
*****
*****
*****
*****
*****
*****
***

```

Testfälle

Wenn die Tanne wie angegeben in der Konsole ausgegeben wird, dann hast du alles richtig gemacht und diese Aufgaben erfolgreich bearbeitet. Gesetzt den Fall, dass du noch weitere Programme diesen Typs erstellen willst, geben wir dir hier noch weitere Anregungen (kannst dir aber auch gerne selbst was überlegen!):

```

Sanduhr:  *****
          ***
          *
          ***
          *****

Pizzastück:  *****
             *   *
             *   *
             *   *
             **

Diamant:    **
            * *
            * *
            * *
            **

```

Für diese zusätzlichen Trainingseinheiten bieten wir dir keine Lösungsvorschläge mehr an. Wir sind fest davon überzeugt, dass du das selbst hinbekommst und unsere Hilfe hierfür nicht mehr benötigst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Schau dir die Aufgabe 2.2.1 doch noch einmal an und überlege dir, wie die Ausgabe für jede Zeile von oben nach unten aussehen muss.
- In Processing für Java gibt es zwei Befehle, mit denen du Text in die Konsole schreiben kannst. Der eine fügt eine neue Zeile hinzu, der andere hingegen nicht.
- Das Sternchen- und das Leerzeichen führen zum Ziel!

W.2.3 Perlenkette

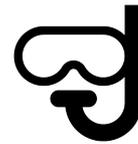
Schwierigkeit



Zeitaufwand



Kreativität



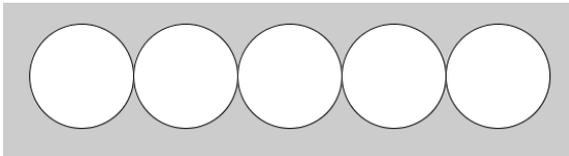
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir die unten dargestellte Perlenkette programmieren:



Die Kette besteht aus fünf Perlen, die als Kreise mit schwarzer Linie und weißer Füllung dargestellt sind.

Aufgabenstellung

Programmiere das angegebene Bild mithilfe der grafischen Grundelemente von Processing.

Testfälle

Wenn die geforderten Grundformen in Art, Größe, Farbe und Lage wie in der Aufgabenstellung gefordert gezeichnet werden, dann hast du eine Lösung gefunden und die Aufgabenstellung richtig gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Alle Processing-Befehle kannst du auf der offiziellen Homepage nachlesen (Java: <https://processing.org/reference/>, Python: <http://py.processing.org/reference/>). Hier kannst du nachschauen, um die entsprechenden Befehle für das Programm zu finden.
- In Processing gibt es keine Funktion zum Zeichnen von Kreisen. Aber es gibt eine Funktion zum Malen von Ellipsen. Ab wann wird eine Ellipse zum Kreis?
- Wenn du die Ausmaße des Bildschirmfensters weißt, wo wird wohl die Mitte des Bildschirmfensters liegen?

W.2.4 Die erste Zeichnung

Schwierigkeit



Zeitaufwand



Kreativität



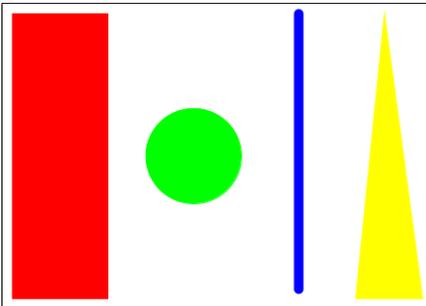
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir die Processing-Grundelemente besser kennenlernen. Dazu wollen wir folgende Grafik programmieren:



Die Grafik hat unter anderem folgende Eigenschaften:

- *Fenstergröße*: 450 Pixel breit und 320 Pixel hoch
- *Rechteck*:
 - x-Position: 10
 - y-Position: 10
 - Größe: 100 Pixel breit und 300 Pixel hoch
 - Farbe: rot
- *Kreis*:
 - x-Position: 200
 - y-Position: Mitte des Bildschirmsfensters
 - Radius: 100 Pixel
 - Farbe: grün
- *Linie*:
 - Breite: 10
 - Start: 310 (x), 10 (y)
 - Ziel: 310 (x), 300 (y)
 - Farbe: blau

- *Dreieck:*
 - Eckpunkte:
 - * 400 (x), 10 (y)
 - * 370 (x), 310 (y)
 - * 440 (x), 310 (y)
 - * Farbe: gelb

Aufgabenstellung

Programmiere das angegebene Bild mithilfe der grafischen Grundelemente von Processing.

Testfälle

Wenn die geforderten Grundformen in Art, Größe, Farbe und Lage, wie in der Aufgabenstellung gefordert, gezeichnet werden, dann hast du die Lösung gefunden und umgesetzt.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Alle Processing-Befehle kannst du auf der offiziellen Homepage nachlesen (Java: <https://processing.org/reference/>, Python: <http://py.processing.org/reference/>). Hier kannst du nachschauen, um die entsprechenden Befehle für das Programm zu finden.
- In Processing gibt es keine Funktion zum Zeichnen von Kreisen. Aber es gibt eine Funktion zum Malen von Ellipsen. Ab wann wird eine Ellipse zum Kreis?
- Wenn du die Ausmaße des Bildschirmfensters weißt, wo wird wohl die Mitte des Bildschirmfensters liegen?

W.2.5 Raupe Allzeitappetit

Schwierigkeit



Zeitaufwand



Kreativität



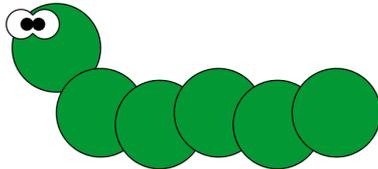
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir eine Raupe zeichnen:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Wenn deine Raupe grundsätzlich mit der abgebildeten Raupe übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Überlege dir zunächst, welche Grundelemente dieses Bild beinhaltet und wo diese platziert sind. Achte dabei auch auf eventuell „unsichtbare“ Grundelemente.
- Die Augen der Raupe bestehen entweder aus fünf (!) Kreisen oder zwei ganzen und zwei halben Kreisen. Beides ist möglich.
- Bei Ellipsen/Kreisen wird immer der Mittelpunkt angegeben und nicht die linke obere Ecke.

W.2.6 Klötzchen-Kunst

Schwierigkeit



Zeitaufwand



Kreativität



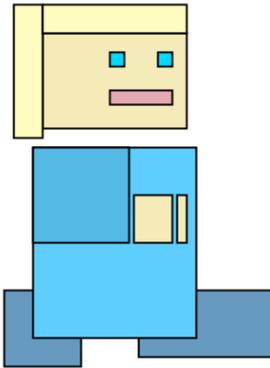
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir einen Menschen aus Rechtecken programmieren:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Wenn dein Klötzchen-Männchen grundsätzlich mit der abgebildeten Raupe übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Schau dir die vorherigen Zeichenaufgaben noch einmal an.
- Bei der Höhe und Breite des Rechtecks kannst du auch negative Werte angeben, um das Rechteck in die umgekehrte Richtung zu zeichnen.

W.2.7 Nachteule

Schwierigkeit



Zeitaufwand



Kreativität



Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

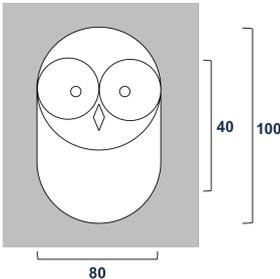
Beschreibung

In dieser Aufgabe wollen wir eine Eule nach dem folgenden Vorbild zeichnen:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente. Die folgenden Konstruktionsüberlegungen sollen dir dabei eine Hilfestellung bieten:



Testfälle

Wenn deine Eule grundsätzlich mit der abgebildeten Eule übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Bei Ellipsen und dem Spezialfall der Kreise wird immer der Mittelpunkt angegeben und nicht die linke obere Ecke.
- Achte auf die Reihenfolge!

W.2.8 Ghettoblaster

Schwierigkeit



Zeitaufwand



Kreativität



Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

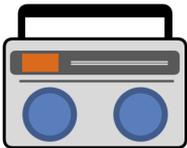
Beschreibung

Der sogenannte Ghettoblaster gilt quasi als der Vorgänger der Bluetooth-Box. Er bestand aus zwei Lautsprechern und meistens auch einem UKW-Radio, mit welchem man unterwegs Musik hören konnte.

Einen solchen Ghettoblaster wollen wir in dieser Aufgabe als Grafik realisieren.

Aufgabenstellung

Programmiere in Processing die Zeichnung eines Ghettoblasters. Er soll in dieser Form gestaltet werden:



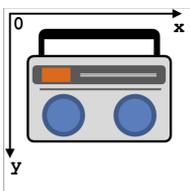
Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Bevor du dich an die Programmierung setzt, solltest du dir die Umsetzung überlegen. Am besten skizzierst du dir das Bild auf ein kariertes Blatt Papier. Danach zeichnest du das Koordinatensystem des Ausgabefensters ein. Wichtig hierbei ist, dass die y-Achse des Koordinatensystems von oben nach unten geht und der Nullpunkt in der linken oberen Ecke liegt:



Auf dem Blatt Papier kannst du anschließend bei allen Elementen die Höhe und Breite der einzelnen Elemente einzeichnen. Ebenso kannst du planen, wo die Koordinatenposition liegen wird.

- Nach der Planung kannst du mit der Programmierung beginnen. Hierbei wird es sehr helfen, wenn du die einzelnen Elemente deines Bildes mit entsprechenden Kommentaren versiehst. So behältst du immer den Überblick, an welcher Stelle welches Element gezeichnet wird. Das könnte in Java zum Beispiel so aussehen:

```
// Blaue Lautsprecherbox unten links  
... (hier steht dann der entsprechende Code)
```

- Sollten Elemente nicht an der vermuteten Stelle gezeichnet werden: Prüfe die entsprechende Stelle im Code und schau nach, ob sich nicht ein Gedanken- oder Tippfehler eingeschlichen hat. Probiere auch gerne verschiedene Werte in den Zeichenfunktionen aus. Das wird dir beim Verstehen der Funktionen sicher weiterhelfen.

W.2.9 Hallo Bello!

Schwierigkeit



Zeitaufwand



Kreativität



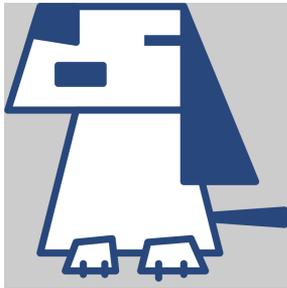
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

Auch einen Hund können wir mit einfachen geometrischen Objekten selber programmieren. Folgendes Bild haben wir als Vorgabe bekommen:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- In diesem Bild haben wir viele Elemente, die sich nur über die Eckpunkte beschreiben lassen (Trapez, Linien, Dreiecke). Deshalb solltest du hier besonders vor dem Programmieren die genauen Positionen der Elemente planen. Nimm dir dazu ein (kariertes) Blatt Papier und zeichne die Elemente und deren Position ein. Ist dann alles genau geplant, läuft die Programmierung wesentlich einfacher.
- Achte darauf, welches Element über welches andere Element gelegt werden soll. Dies kannst du über die Reihenfolge festlegen, mit der du die Elemente in das Ausgabefenster zeichnest.
- Es ist empfehlenswert, wenn du zunächst alle Formen einzeichnest. Stimmt die Zeichnung dann mit dem Ergebnis einigermaßen überein, kannst du die Eigenschaften der Elemente noch hinzufügen (Farbe, Liniendicke, Linienart etc.).