C-Kurs - zum Informatikumfeld

1 Die Programmiersprachen 1
1.1 Fünf Sprachgenerationen 1
1.2 Ein Stammbaum der Programmiersprachen 1
1.3 Die C-Entwicklungsstufen und die C-Normen 2
1.4 Zur Bedeutung von C 3
1.5 Grafische Hilfsmittel zur Programmentwicklung 4
1.6 Zeichensätze und Kodierung 4
1.6.1 ASCII 7 Bit-Code 4
1.6.2 8-Bit-Codes 5
1.6.3 Escape-Sequenzen 8
1.6.4 Unicode 8
1.7 C-Compiler 8
1.8 Literatur 8

1 Die Programmiersprachen

1.1 Fünf Sprachgenerationen

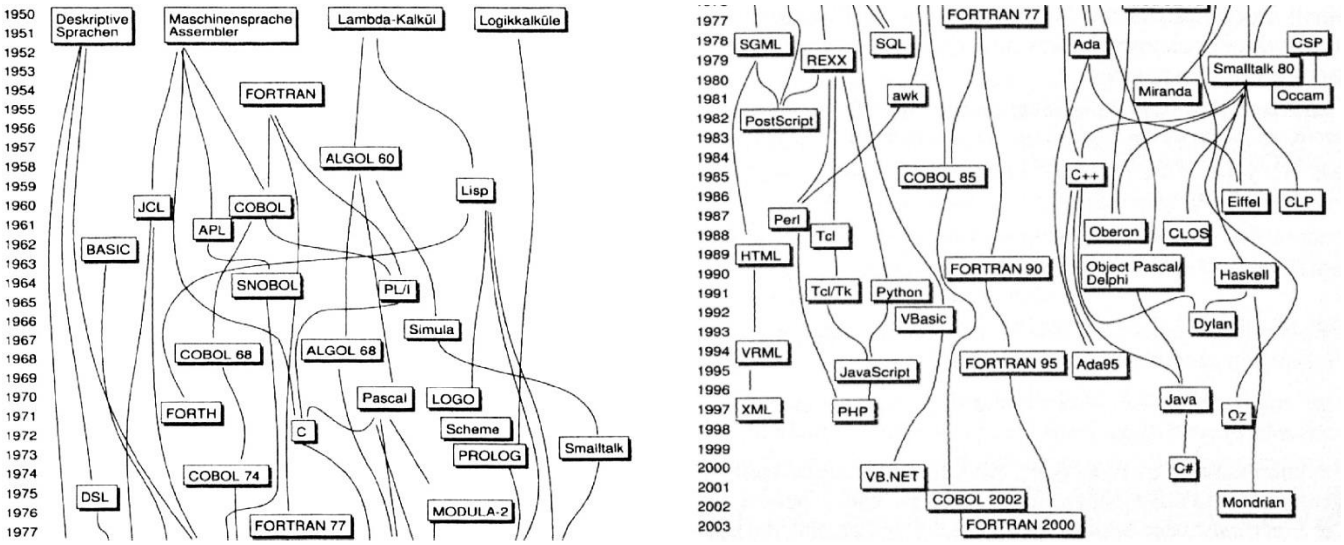
Heute werden fünf Generationen von Programmiersprachen unterschieden.

Gene-ration	Sprachtyp	prozessor-spezifisch?	Anmerkung
1	Maschinen-sprache - kaum noch verwendet	ja	Alle Anweisungen, auch die in ihnen ggf. vorkommenden Werte, werden ausschließlich mit den Ziffern 0 und 1, also binär, formuliert. Befehl: Addiere 2 und 5 Binärer Befehl: 00011010 0010 0101
2	Assembler-sprache - seit C wenig verwendet	ja	Die Anweisungen werden für Menschen verständlicher mit Schlüssel-worten und Dezimalzahlen geschrieben. Assemblerbefehl: ADD 2,5 Jeder Assembler-Befehl wird von einem Assemblierer in genau eine Anweisung in Maschinensprache übersetzt.
3	Höhere Program-mier-sprache	nein	Wie 2, aber jede Anweisung wird von einem Compiler oder Interpreter ggf. auch in mehrere Anweisungen in Maschinensprache übersetzt. Beispiele: FORTRAN, C, C++, C#, Visual C++. Manche Programmiersysteme gestatten zusätzlich die Integration von Maschinen- oder Assemblercode.
4	Makro-Sprache	nein	Beliebig lange Befehlsfolgen können zu einem Befehl zusammen-gefasst und mit diesem aufgerufen werden. Beispiele DMAP von der NASA und VBA von Microsoft. VBA ist in alle MS Office-Programme integriert und ermöglicht die Ausführung komplexer und auch ganz neuer Arbeitsabläufe sehr schnell und fehlerfrei.
5	KI-Sprache	nein	Die geistigen Fähigkeiten des Menschen sollen erreicht werden. Beispiele: LISP, PROLOG.

C gehört zu den höheren Programmiersprachen, hat aber zudem Assemblereigenschaften. Es ermöglicht deshalb auch die Erstellung maschinennaher, sehr effizienter Programme.

1.2 Ein Stammbaum der Programmiersprachen

Ein besonders umfangreicher unter den vielen veröffentlichten Stammbäumen der Programmiersprachen (aus Henning u. a., Taschenbuch Programmiersprachen):



Die erste höhere Programmiersprache FORTRAN (FORmula TRANslator) wurde bei IBM vor allem für mathematische Anwendungen ab 1954 entwickelt und im Jahr 1957 erstmals ausgeliefert. Das Konkurrenzprodukt ALGOL (ALGOrithmic Language) mit derselben Zielsetzung aber aus dem europäischen Hochschulbereich folgte als erster Diskussionsentwurf im Jahr 1958. Anfänglich auf einen anderen Schwerpunkt ausgerichtet, nämlich auf die Programmierung des Betriebssystems Unix, war das 1969 bis 1973 bei den Bell Laboratories entworfene C. Inzwischen werden mit C auch Anwendungen programmiert und es gilt als die bedeutendste und am häufigsten verwendete Programmiersprache.

1.3 Die C-Entwicklungsstufen und die C-Normen

Die Programmiersprache C wurde von Brian W. Kernighan und Dennis M. Ritchie in den Bell Labs entwickelt und von ihnen in der ersten Auflage des Buches „The C Programming Language“ von 1977 beschrieben (dt. Übersetzung 1983). Im Jahr 1988 wurde die verbesserte Variante C90 in der zweiten Auflage des Buches veröffentlicht (dt. Übersetzung 1990) und 1990 in der ISO/IEC 9899:1990 firmenunabhängig und international genormt. C90 wurde die Basis für viele Weiterentwicklungen, unter anderem für das gleichfalls firmenunabhängig und international genormte C++, das über Möglichkeiten zur objektorientierten und generischen Programmierung verfügt. C90 liegt auch diesem Kurs zugrunde. Verweise auf Änderungen durch die Versionen C95, C99 und C11 sind eingestreut. Weiterentwicklungen einzelner Firmen gibt es viele, z. B. Visual C++ und C# von Microsoft.

C-Standard	**Norm oder Werke ähnlicher Bedeutung**
C78 oder K&R C 1	Ritchie D. M. u. a., **The C Programming Language**, Computing Science Technical Report # 31, Bell Laboratories, 1975. TUB vorh. Kernighan Brian W. u. a., **The C Programming Language**, Prentice Hall, 1. A. 1978, 228 S. Das Buch (bekannt als **K&R1**) von den Erfindern von C ist die erste käufliche Darstellung von C. TUB vorh. Kernighan Brian W. u. a., **Programmieren in C**, Hanser, 1. A. 1983, 262 S. Übersetzung des davor stehenden Buchs ins Deutsche. TUB vorh.
C89 oder ANSI C oder K&R C 2	Norm **ANSI X3.159-1989**, Programming Language C, 1989. Normung von C zur Vereinheitlichung der ausufernden C-Dialekte und der Sicherung der Portabilität von C-Programmen. Der Begriff „Standard C“ entstand. Kernighan Brian W. u. a., **The C Programming Language**, Prentice Hall, 2. A. 1988, 272 S. Die Neuauflage (bekannt als **K&R2**) des Buchs beschreibt ANSI C. TUB vorh. Kernighan Brian W. u. a., **Programmieren in C**, Hanser, 2. A. 1990, 262 S. Übersetzung des davor stehenden Buchs ins Deutsche. TUB vorh.
C90	Norm **ISO/IEC 9899:1990**, Programming languages – C, First edition 1990-12-15. Fast identisch mit C89. Wurde nach ihrem Erscheinen inhaltlich von ANSI nur mit abweichender Nummerierung als Standard X3.159-1989 rückwärts übernommen, so dass der ANSI C-Standard in ursprünglicher Form nicht mehr existiert. Seitdem meinen C89 und C90 das gleiche. Normkorrektur **ISO/IEC 9899:1990/COR 1:1994**, Programming languages – C, Technical corrigendum 1, 1994-09-15
C95	Normänderung **ISO/IEC 9899:1990/AMD 1:1995**, Programming languages – C, Amendmend 1: C Integrity, 1995-04-01. Fehlerbehebungen und kleine Änderungen im Sprachumfang.

C-Standard	Norm oder Werke ähnlicher Bedeutung
	Normkorrektur **ISO/IEC 9899:1990/COR 2:1996**, Programming languages – C, Technical corrigendum 2, 1996-01-01
C99	Norm **ISO/IEC 9899:1999**, Programming languages – C, Second edition 1999-12-01. Ersetzt ISO/IEC 9899:1990. Erhebliche Erweiterungen. U. a. Aufnahme von Sprachkonzepten aus C++. Normkorrektur **ISO/IEC 9899:1999/COR 1:2001**, Programming languages – C, Technical corrigendum 1, 2001 Normkorrektur **ISO/IEC 9899:1999/COR 2:2004**, Programming languages – C, Technical corrigendum 2, 2004 Normkorrektur **ISO/IEC 9899:1999/COR 3:2007**, Programming languages – C, Technical corrigendum 3, 2007 Der Arbeitstitel des Normenausschusses war C9X. C99 zusammen mit den drei Korrekturen wird inoffiziell als C0X bezeichnet.
C11	Norm **ISO/IEC 9899:2011** Programming languages – C, 2011. Ersetzt ISO/IEC 9899:1999. Neue bindende und optionale Bestandteile. Bindende Bestandteile aus C99 wurden optional. Der Arbeitstitel war C1X und basierte auf C0X. Normkorrektur **ISO/IEC 9899:2011/COR 1:2012** Programming languages – C, Technical corrigendum 1, 2012

Mit dem Begriff **Standard C** meint man eigentlich irgendeine eine der genormten Versionen von C. Manche Autoren verwenden ihn aber nur für C90.

Zu diesem Thema interessante Stellen aus dem Internet

1) Die versionsweise eingeführten Änderungen

https://de.wikipedia.org/wiki/Varianten_der_Programmiersprache_C

http://www.schellong.de/c.htm

2) Überblick über die Entwicklungsstufen der C-Standardbibliotheken

https://de.wikipedia.org/wiki/C-Standard-Bibliothek

3) Ausführliche Beschreibungen der Standardbibliothek von C90

http://www2.hs-fulda.de/~klingebiel/c-stdlib/index.htm

Plauger P.J.: The Standard C Library. Prentice Hall, 1992, ISBN 0-13-131509-9

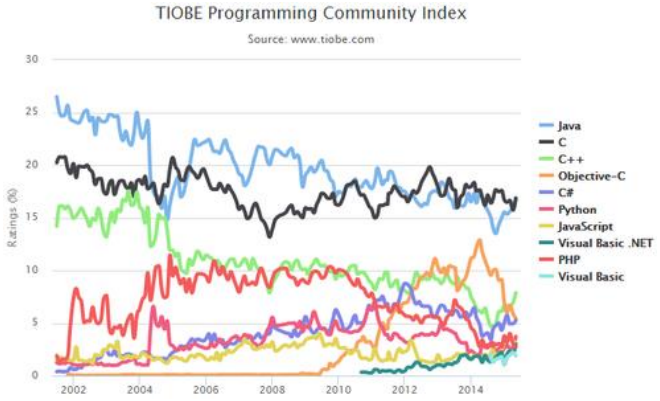
4) Die C-Sprachreferenz von Microsoft

https://msdn.microsoft.com/de-de/library/fw5abdx6.aspx

1.4 Zur Bedeutung von C

Nach dem monatlich aktualisierten „TIOBE Programming Community Index“ sind C und Java die populärsten Programmiersprachen.

https://de.wikipedia.org/wiki/TIOBE-Index



1.5 Grafische Hilfsmittel zur Programmentwicklung

Datenflussplan, DIN 66001	Stellt den Datenfluss zwischen Datenträgern mit Symbolen und Verbindungslinien grafisch dar. https://de.wikipedia.org/wiki/Datenflussdiagramm
Programmablaufplan, DIN 66001	Der logische Ablauf eines Programms wird mit Grafiksymbolen und Verbindungslinien dargestellt. https://de.wikipedia.org/wiki/Programmablaufplan
Struktogramm oder Nassi-Shneiderman-Diagramm, DIN 66261	Gleiche Funktion wie der Programmablaufplan. Verwendet Grafiksymbole aber keine Verbindungslinien. Hat deshalb kein Symbol für Sprunganweisungen und soll so helfen, den gefürchteten Spaghetticode zu vermeiden. https://de.wikipedia.org/wiki/Nassi-Shneiderman-Diagramm

1.6 Zeichensätze und Kodierung

1.6.1 ASCII 7 Bit-Code

Als C-Programmierer bekommt man es unweigerlich mit den auf Computern verwendeten Zeichensätzen und ihren Codierungen zu tun. Unter Codierung versteht man Zuordnung von Zeichen zu Nummern (Ordnungszahlen), die in binärer Darstellung, mit den Ziffern 0 und 1, in den Computerspeichern stellvertretend für die Zeichen abgelegt werden. Zeichensatz und Codierung zusammen nennt man einen (Computer-) Code. Bei den PCs wird fast ausschließlich der ASCII 7 Bit-Code verwendet oder eine seiner verschiedenen Erweiterungen auf 8 und mehr Bits.

ASCII ist die Abkürzung von „American Standard Code for Information Interchange“. Der ASCII 7 Bit-Code für 128 Zeichen mit den dezimalen Ordnungszahlen 0 bis 127 wurde am 17. Juni 1963 von American Standards Association (ASA, inzwischen umbenannt in ANSI) als Standard ASA X3.4-1963 erstmals eingeführt und später mehrfach aktualisiert. Er definierte lange Zeit den wichtigsten Computercode.

Die folgenden beiden Tabellen zeigen den ASCII 7 Bit-Code mit den Ordnungszahlen in binärer, oktaler, hexadezimaler und dezimaler Darstellung.

Zeichengruppen im ASCII 7 Bit-Code

0-31, 127	Steuerzeichen	48-57	Ziffern
32	Leerzeichen	65-90	Großbuchstaben
32-126	druckbare Zeichen	97-122	Kleinbuchstaben

Eine ausführlichere Beschreibung der Steuerzeichen findet man in
https://de.wikipedia.org/wiki/Steuerzeichen

ASCII 7 Bit-Code, Zeichen 0 bis 31, Steuerzeichen

Binär	Okt	Hex	Dez	Zeichen	Bedeutung
00000000	0	0	0	**NUL**	null char.
00000001	1	1	1	**SOH**	start of header
00000010	2	2	2	**STX**	start of text
00000011	3	3	3	**ETX**	end of text
00000100	4	4	4	**EOT**	end of transmission
00000101	5	5	5	**ENQ**	enquiry
00000110	6	6	6	**ACK**	acknowledgment
00000111	7	7	7	**BEL**	bell
00001000	10	8	8	**BS**	backspace
00001001	11	9	9	**HAT**	horizontal tab
00001010	12	A	10	**LF**	line feed
00001011	13	B	11	**VT**	vertical tab
00001100	14	C	12	**FF**	form feed
00001101	15	D	13	**CR**	carriage return
00001110	16	E	14	**SO**	shift out
00001111	17	F	15	**SI**	shift in

Binär	Okt	Hex	Dez	Zeichen	Bedeutung
00010000	20	10	16	**DLE**	data link escape
00010001	21	11	17	**DC1**	xon(device control 1
00010010	22	12	18	**DC2**	device control 2
00010011	23	13	19	**DC3**	xoff(device control 3
00010100	24	14	20	**DC4**	device control 4
00010101	25	15	21	**NAK**	negative acknowledgement
00010110	26	16	22	**SYN**	synchronous idle
00010111	27	17	23	**ETB**	end of trans. block
00011000	30	18	24	**CAN**	cancel
00011001	31	19	25	**EM**	end of medium
00011010	32	1A	26	**SUB**	substitute
00011011	33	1B	27	**ESC**	escape
00011100	34	1C	28	**FS**	file separator
00011101	35	1D	29	**GS**	group separator
00011110	36	1E	30	**RS**	request to send
00011111	37	1F	31	**US**	unit separator

ASCII 7 Bit-Code, Zeichen 32 bis 126, druckbare Zeichen, Zeichen 127 Steuerzeichen

Binär	Okt	Hex	Dez	Zeichen
00100000	40	20	32	SP
00100001	41	21	33	!
00100010	42	22	34	"
00100011	43	23	35	#
00100100	44	24	36	$
00100101	45	25	37	%
00100110	46	26	38	&
00100111	47	27	39	'
00101000	50	28	40	(
00101001	51	29	41	)
00101010	52	2A	42	*
00101011	53	2B	43	+
00101100	54	2C	44	,
00101101	55	2D	45	-
00101110	56	2E	46	.
00101111	57	2F	47	/
00110000	60	30	48	0
00110001	61	31	49	1
00110010	62	32	50	2
00110011	63	33	51	3
00110100	64	34	52	4
00110101	65	35	53	5
00110110	66	36	54	6
00110111	67	37	55	7

Binär	Okt	Hex	Dez	Zeichen
00111000	70	38	56	8
00111001	71	39	57	9
00111010	72	3A	58	:
00111011	73	3B	59	;
00111100	74	3C	60	<
00111101	75	3D	61	=
00111110	76	3E	62	>
00111111	77	3F	63	?
1000000	100	40	64	@
1000001	101	41	65	A
1000010	102	42	66	B
1000011	103	43	67	C
1000100	104	44	68	D
1000101	105	45	69	E
1000110	106	46	70	F
1000111	107	47	71	G
1001000	110	48	72	H
1001001	111	49	73	I
1001010	112	4A	74	J
1001011	113	4B	75	K
1001100	114	4C	76	L
1001101	115	4D	77	M
1001110	116	4E	78	N
1001111	117	4F	79	O

Binär	Okt	Hex	Dez	Zeichen
1010000	120	50	80	P
1010001	121	51	81	Q
1010010	122	52	82	R
1010011	123	53	83	S
1010100	124	54	84	T
1010101	125	55	85	U
1010110	126	56	86	V
1010111	127	57	87	W
1011000	130	58	88	X
1011001	131	59	89	Y
1011010	132	5A	90	Z
1011011	133	5B	91	[
1011100	134	5C	92	\
1011101	135	5D	93	]
1011110	136	5E	94	^
1011111	137	5F	95	_
1100000	140	60	96	`
1100001	141	61	97	a
1100010	142	62	98	b
1100011	143	63	99	c
1100100	144	64	100	d
1100101	145	65	101	e
1100110	146	66	102	f
1100111	147	67	103	g

Binär	Okt	Hex	Dez	Zeichen
1101000	150	68	104	h
1101001	151	69	105	i
1101010	152	6A	106	j
1101011	153	6B	107	k
1101100	154	6C	108	l
1101101	155	6D	109	m
1101110	156	6E	110	n
1101111	157	6F	111	o
1110000	160	70	112	p
1110001	161	71	113	q
1110010	162	72	114	r
1110011	163	73	115	s
1110100	164	74	116	t
1110101	165	75	117	u
1110110	166	76	118	v
1110111	167	77	119	w
1111000	170	78	120	x
1111001	171	79	121	y
1111010	172	7A	122	z
1111011	173	7B	123	{
1111100	174	7C	124	|
1111101	175	7D	125	}
1111110	176	7E	126	~
1111111	177	7F	127	DEL

Die druckbaren Zeichen des ASCII 7 Bit-Codes bilden - mit Ausnahme der Zeichen $, @ und ´ - genau den Quellzeichensatz von C, also den Zeichensatz mit dem C-Quellprogramme geschrieben werden. Die drei Ausnahmezeichen und alle anderen in einem Quelleditor mit seinem 8-Bit-Zeichensatz möglichen Zeichen können nur an zwei Stellen verwendet werden:

- In Kommentaren. Völlig problemlos, da der Compiler Sie im ersten Arbeitsschritt durch jeweils ein Leerzeichen ersetzt.
- In Zeichenliteralen und Zeichenkettenliteralen: Führt nicht auf eine Fehlermeldung aber in der Windows-Konsole werden diejenigen Zeichen abweichend ausgegeben, die in CP 1252 und CP 850 unterschiedlich kodiert sind.

Am einfachsten beschränkt man sich sowohl beim Schreiben eines C-Quellprogramms als auch beim Ablauf der exe in der Ein- und Ausgabe auf den in den C-Normen definierten C-Zeichensatz und hat dann keinerlei Probleme. Das bedeutet insbesondere den Verzicht auf die deutschen Umlaute ä, ö, ü und das ß.

1.6.2 8-Bit-Codes

Da 128 Zeichen für länderspezifische Besonderheiten nicht ausreichen, wurde das achte, ursprüngliche Prüfbit für verschiedene Erweiterungen auf 8 Bit und 256 Zeichen benutzt. Alle stimmen in den Zeichen 0 bis 127 mit dem ASCII 7 Bit-Code überein. Zwei wichtige Erweiterungen für die PCs sind die Codes mit den Bezeichnungen Codepage 437 (IBM PC ab 1981) und Codepage 850 (MS DOS 3.3 ab 1987), die sich in 47 Zeichen unterscheiden.

Codepage 437 https://de.wikipedia.org/wiki/Codepage_437

Die von PC DOS und MS-DOS verwendete Codepage 437, kurz CP437, auch bekannt als DOS-US oder OEM-US, ist der Original-Zeichensatz des IBM-PC ab 1981 und enthält folgende Zeichen: (Die Zahl unterhalb des Symbols ist der Unicode-Wert in hexadezimaler Schreibweise)

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F	
0_	NULL 0	☺ 263A	☻ 263B	♥ 2665	♦ 2666	♣ 2663	♠ 2660	• 2022	◘ 25D8	○ 25CB	◙ 25D9	♂ 2642	♀ 2640	♪ 266A	♫ 266B	☼ 263C	0_
1_	► 25BA	◄ 25C4	↕ 2195	‼ 203C	¶ B6	§ A7	▬ 25AC	↨ 21A8	↑ 2191	↓ 2193	→ 2192	← 2190	∟ 221F	↔ 2194	▲ 25B2	▼ 25BC	1_
2_	20	! 21	" 22	# 23	$ 24	% 25	& 26	' 27	(28	) 29	* 2A	+ 2B	, 2C	- 2D	. 2E	/ 2F	2_
3_	0 30	1 31	2 32	3 33	4 34	5 35	6 36	7 37	8 38	9 39	: 3A	; 3B	< 3C	= 3D	> 3E	? 3F	3_
4_	@ 40	A 41	B 42	C 43	D 44	E 45	F 46	G 47	H 48	I 49	J 4A	K 4B	L 4C	M 4D	N 4E	O 4F	4_
5_	P 50	Q 51	R 52	S 53	T 54	U 55	V 56	W 57	X 58	Y 59	Z 5A	[5B	\ 5C	] 5D	^ 5E	_ 5F	5_
6_	` 60	a 61	b 62	c 63	d 64	e 65	f 66	g 67	h 68	i 69	j 6A	k 6B	l 6C	m 6D	n 6E	o 6F	6_
7_	p 70	q 71	r 72	s 73	t 74	u 75	v 76	w 77	x 78	y 79	z 7A	{ 7B	| 7C	} 7D	~ 7E	⌂ 2302	7_
8_	Ç C7	ü FC	é E9	â E2	ä E4	à E0	å E5	ç E7	ê EA	ë EB	è E8	ï EF	î EE	ì EC	Ä C4	Å C5	8_
9_	É C9	æ E6	Æ C6	ô F4	ö F6	ò F2	û FB	ù F9	ÿ FF	Ö D6	Ü DC	¢ A2	£ A3	¥ A5	₧ 20A7	ƒ 192	9_
A_	á E1	í ED	ó F3	ú FA	ñ F1	Ñ D1	ª AA	º BA	¿ BF	⌐ 2310	¬ AC	½ BD	¼ BC	¡ A1	« AB	» BB	A_
B_	░ 2591	▒ 2592	▓ 2593	│ 2502	┤ 2524	╡ 2561	╢ 2562	╖ 2556	╕ 2555	╣ 2563	║ 2551	╗ 2557	╝ 255D	╜ 255C	╛ 255B	┐ 2510	B_
C_	└ 2514	┴ 2534	┬ 252C	├ 251C	─ 2500	┼ 253C	╞ 255E	╟ 255F	╚ 255A	╔ 2554	╩ 2569	╦ 2566	╠ 2560	═ 2550	╬ 256C	╧ 2567	C_
D_	╨ 2568	╤ 2564	╥ 2565	╙ 2559	╘ 2558	╒ 2552	╓ 2553	╫ 256B	╪ 256A	┘ 2518	┌ 250C	█ 2588	▄ 2584	▌ 258C	▐ 2590	▀ 2580	D_
E_	α 3B1	ß DF	Γ 393	π 3C0	Σ 3A3	σ 3C3	µ B5	τ 3C4	Φ 3A6	Θ 398	Ω 3A9	δ 3B4	∞ 221E	φ 3C6	ε 3B5	∩ 2229	E_
F_	≡ 2261	± B1	≥ 2265	≤ 2264	⌠ 2320	⌡ 2321	÷ F7	≈ 2248	° B0	∙ 2219	· B7	√ 221A	ⁿ 207F	² B2	■ 25A0	NBSP A0	F_
	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F	

Unterschiede zu Codepage 850

Codepage 850 https://de.wikipedia.org/wiki/Codepage_850

Codepage 850 ist eine von MS-DOS und PC DOS verwendete Codepage (daher auch bekannt als DOS-Latin-1). Sie wurde 1987 mit DOS 3.3 eingeführt.
Sie modifiziert die Codepage 437 dahingehend, dass die griechischen Buchstaben (mit Ausnahme des β, welches auch für das deutsche ß benutzt wurde, und des µ, das als Vorsilbe für 10-6 dient) und die gemischten Rahmenzeichen (die aus einfachen und doppelten Rahmenabschnitten bestehen) entfernt wurden und durch die in CP 437 fehlenden Zeichen aus dem ISO-8859-1-Zeichensatz ersetzt wurden.
Die Codepage 850 enthält folgende Zeichen: (Die Zahl unterhalb des Symbols ist der Unicode-Wert in hexadezimaler Schreibweise)

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F	
0_	NULL 0	☺ 263A	☻ 263B	♥ 2665	♦ 2666	♣ 2663	♠ 2660	• 2022	◘ 25D8	○ 25CB	◙ 25D9	♂ 2642	♀ 2640	♪ 266A	♫ 266B	☼ 263C	0_
1_	► 25BA	◄ 25C4	↕ 2195	‼ 203C	¶ B6	§ A7	▬ 25AC	↨ 21A8	↑ 2191	↓ 2193	→ 2192	← 2190	∟ 221F	↔ 2194	▲ 25B2	▼ 25BC	1_
2_	20	! 21	" 22	# 23	$ 24	% 25	& 26	' 27	(28	) 29	* 2A	+ 2B	, 2C	- 2D	. 2E	/ 2F	2_
3_	0 30	1 31	2 32	3 33	4 34	5 35	6 36	7 37	8 38	9 39	: 3A	; 3B	< 3C	= 3D	> 3E	? 3F	3_
4_	@ 40	A 41	B 42	C 43	D 44	E 45	F 46	G 47	H 48	I 49	J 4A	K 4B	L 4C	M 4D	N 4E	O 4F	4_
5_	P 50	Q 51	R 52	S 53	T 54	U 55	V 56	W 57	X 58	Y 59	Z 5A	[5B	\ 5C	] 5D	^ 5E	_ 5F	5_
6_	` 60	a 61	b 62	c 63	d 64	e 65	f 66	g 67	h 68	i 69	j 6A	k 6B	l 6C	m 6D	n 6E	o 6F	6_
7_	p 70	q 71	r 72	s 73	t 74	u 75	v 76	w 77	x 78	y 79	z 7A	{ 7B	| 7C	} 7D	~ 7E	⌂ 2302	7_
8_	Ç C7	ü FC	é E9	â E2	ä E4	à E0	å E5	ç E7	ê EA	ë EB	è E8	ï EF	î EE	ì EC	Ä C4	Å C5	8_
9_	É C9	æ E6	Æ C6	ô F4	ö F6	ò F2	û FB	ù F9	ÿ FF	Ö D6	Ü DC	ø F8	£ A3	Ø D8	× D7	ƒ 192	9_
A_	á E1	í ED	ó F3	ú FA	ñ F1	Ñ D1	ª AA	º BA	¿ BF	® AE	¬ AC	½ BD	¼ BC	¡ A1	« AB	» BB	A_
B_	░ 2591	▒ 2592	▓ 2593	│ 2502	┤ 2524	Á C1	Â C2	À C0	© A9	╣ 2563	║ 2551	╗ 2557	╝ 255D	¢ A2	¥ A5	┐ 2510	B_
C_	└ 2514	┴ 2534	┬ 252C	├ 251C	─ 2500	┼ 253C	ã E3	Ã C3	╚ 255A	╔ 2554	╩ 2569	╦ 2566	╠ 2560	═ 2550	╬ 256C	¤ A4	C_
D_	ð F0	Ð D0	Ê CA	Ë CB	È C8	ı 131	Í CD	Î CE	Ï CF	┘ 2518	┌ 250C	█ 2588	▄ 2584	¦ A6	Ì CC	▀ 2580	D_
E_	Ó D3	ß DF	Ô D4	Ò D2	õ F5	Õ D5	µ B5	þ FE	Þ DE	Ú DA	Û DB	Ù D9	ý FD	Ý DD	¯ AF	´ B4	E_
F_	SHY AD	± B1	‗ 2017	¾ BE	¶ B6	§ A7	÷ F7	¸ B8	° B0	¨ A8	· B7	¹ B9	³ B3	² B2	■ 25A0	NBSP A0	F_
	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F	

Unterschiede zu Codepage 437

Die mit 128 bzw. 256 Zeichen im Vergleich zum neueren Unicode kleinen Zeichensätze sind auch heute noch der Standard für viele Anwendungsbereiche, insbesondere für viele C-Compiler.

Windows, insbesondere auch die unter Windows verwendeten Editoren, mit denen C-Programme geschrieben werden, verwenden als Standard die Codepage 1252 (auch bekannt als CP 1252, Windows-1252, Windows Latin 1 und fälschlich als ANSI-Code). Codepage-Einstellungen in der Windows-Registry sind

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\CodePage
```

```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls\CodePage\ACP
```

Die Ausgabe eines ablaufenden C-Programms erfolgt unter Windows dagegen in einem Kommandozeilenfenster (Windows-Konsole, DOS-Fenster), das in deutschen Installationen standardmäßig die Codepage 850, also einen

anderen 8-Bit-Zeichensatz verwendet.

Die Konsole kann man auch mit dem Befehl cmd im Windows-Suchfeld aufrufen, dann in ihr mit dem Befehl chcp die aktuelle Codepage anzeigen lassen und auch ändern z. B. in Codepage 1252 mit dem Befehl chcp 1252. Mit charmap kann man in der Konsole die Windows-Zeichentabelle aufrufen und in deren erweiterten Ansicht den Namen des Zeichensatzes anzeigenlassen. Wer will, kann in den Eigenschaften des Kommandozeilenfensters (Kontextmenü des Fenstersymbols in der linken oberen Ecke und dann Eigenschaften) eine andere nichtproportionale (monospace), deshalb zum Programmieren gut geeignete, Schrift einstellen, z. B. mit Serifen aus der Courier-Familie oder ohne Serifen "Consolas", "Lucida Console" oder aus der OCR-Familie.

Beispiel 1

```
printf("A b § € ä ö ü ß");
```

Programmiert man mit dieser Anweisung in einem C-Quellcode die Ausgabe der Zeichenfolge A b § € ä ö ü ß, so erhält man bei der Programmausführung in der Konsole standardmäßig angezeigt A b º Ç õ ÷ ³ ■. Ursache sind die Kodierung mit CP 1252 und die Dekodierung mit CP 850.

Beispiel 2

Die Zeichenfolge A b § ? ä ö ü ß aus einer mit CP 850 und aus einer CP 1252 kodierten Textdatei lesen. Mit printf in der Konsole mit ihrer Kodierung CP 850 ausgeben ergibt: A b § ? ä ö ü ß und A b º ? õ ÷ ³ ■.

International wurden 15 verschiedene auf 8 Bits erweiterte Codes ISO 8859-1 bis ISO 8859-16 genormt. ISO 8859-12 wurde verworfen.

ISO8859

-1	Latin-1, Westeuropäisch	-9	Latin-5, Türkisch
-2	Latin-2, Mitteleuropäisch	-10	Latin-6, Nordisch
-3	Latin-3, Südeuropäisch	-11	Thai
-4	Latin-4, Nordeuropäisch	-12	(existiert nicht)
-5	Kyrillisch	-13	Latin-7, Baltisch
-6	Arabisch	-14	Latin-8, Keltisch
-7	Griechisch	-15	Latin-9, Westeuropäisch
-8	Hebräisch	-16	Latin-10, Südosteuropäisch

Windows-1252 deckt die Zeichensätze von ISO 8859-1 und ISO 8859-15 ab. Den Zeichensatz von ISO 8859-1 findet man auch in der DIN 66303:2000-06.

https://de.wikipedia.org/wiki/ISO_8859

Zu den Begriffen

- **Zeichenvorrat** (character repertoire): Eine Menge von Zeichen, die man verwenden will.
- **Zeichensatz** (Alphabet, character set, kurz charset) Eine Zeichenmenge mit durchnummerierten Zeichen und der dadurch bestimmten Reihenfolge.
- **Computercode** (Zeichencode, Code, character encoding): Ein Zeichensatz mit Nummerierung in binärer Form zwecks Abbildung der Zeichen auf Bytes. Zum Zeichensatz Unicode gibt es verschiedene Codierungen, z. B. die verbreiteten UTF-8 und UTF-16. Die Unterscheidung zwischen Zeichensatz und Zeichencode wird in der Praxis oft vernachlässigt.

  https://wiki.selfhtml.org/wiki/Zeichenkodierung
- **Single-Byte-Kodierung:** Zur Darstellung aller Zeichen eines Zeichensatzes wird jeweils ein Byte verwendet.
- **Multi-Byte-Kodierung:** Zur Darstellung aller oder einiger Zeichen eines Zeichensatzes werden jeweils mehr als ein Byte verwendet.
- **Codepoint** (Ordnungszahl)**:** Die Nummer eines Zeichens. Im Unicode normalerweise angegeben mit hexadezimalen Zahlen und vorangestelltem „U+“. Die Unicode-Codepoints reichen von U+0000 bis U+10FFFF. Die Zeichen werden mit einem bis vier Bytes codiert.
- **Schriftart:** Wird für die grafische Darstellung von Zeichen verwendet, z. B. Arial, Courier.
- **Glyphe:** Die grafische Darstellung eines Zeichens.
- **Font:** Eine Computerdatei, die eine Schriftart zur Verfügung stellt.
- Der gelegentlich für Windows-1252 verwendete Name **ANSI-Zeichensatz** ist eine Fehlbezeichnung, weil der Zeichensatz nicht durch eine ANSI-Norm festgelegt ist, sondern von Microsoft.

- Der Begriff **Codepage** (Codierungstabelle) ist historisch entstanden. Er kommt ursprünglich von den IBM Großcomputern. Die Nummer einer code page war eine Seitennummer im IBM-Handbuch der Standardzeichensätze.
- **Erweiterte ASCII 7 Bit-Codes** nennt man alle Codes mit acht oder mehr Bits, die in den Ordnungszahlen 0 bis 127 mit dem ASCII-Code übereinstimmen.

1.6.3 Escape-Sequenzen

Eine Escape-Sequenz ist eine Folge von wenigen Zeichen mit einer festgelegten Funktion. Wird sie beim Verarbeiten einer Zeichenfolge erreicht, so steigt das Programm aus der normalen Verarbeitung aus und löst die zugeordnete Funktion aus. Anschließend wird die normale Verarbeitung der Zeichenfolge fortgesetzt.

Mit einer Escape-Sequenz wird z. B. ein Gerät angesteuert oder sie dient als Ersatzdarstellung für ein anderes Zeichen. Im Fall von C kann man so in ein Quellprogramm auch Zeichen einfügen, die auf der Tastatur nicht vorkommen, z. B. die Steuerzeichen des ASCII 7 Bit-Codes.

Escape-Sequenzen sind von verschiedenen Institutionen genormt worden. Die in den C-Normen definierten C-Escape-Sequenzen werden von einem Backslash \ eingeleitet und die von der ANSI festgelegten ANSI-Escape-Sequenzen von dem ESC-Zeichen gefolgt von der eckigen Klammer auf [. Anschließend folgen in beiden Fällen ASCII 7 Bit-Zeichen, welche die zugeordnete Funktion angeben.

Beispiel

\x1b ist die C-Escape-Sequenz für das ASCII-Steuerzeichen ESC mit der hexadezimalen Ordnungszahl 1b.

\x1b[2J ruft in einem C-Quellcode eine ANSI-Escape-Sequenz mit der Funktion 2J auf, löschen des Bildschirms.

https://de.wikipedia.org/wiki/Escape-Sequenz

1.6.4 Unicode

Unicode ist ein internationaler Standard, in dem jedes weltweit bekannte Schriftelement mit seinen Eigenschaften und geeigneten Kodierungsmöglichkeiten vorkommen soll. Er wird ständig ergänzt. Unicode mit der Codierung UTF 8 stimmt in den ersten 128 Zeichen genau mit dem ASCII 7 Bit-Code überein. In C-Quelltexten kann Unicode seit C95 verwendet werden.

1.7 C-Compiler

Eine Liste von C-Compilern findet man jeweils in

https://en.wikipedia.org/wiki/Category:C_compilers

https://en.wikipedia.org/wiki/List_of_compilers#C_compilers

Die Beispiele in diesem Kurs wurden mit dem freien Entwicklungssystem **dev C++** (C und C++) getestet.

Eine ebenfalls frei verfügbare Alternative ist das Entwicklungssystem **Code::Blocks** (C, C++, FORTRAN).

http://wiki.codeblocks.org/index.php/Category:User_Documentation

http://blog.newtrics.com/?p=1338

http://cbfortran.sourceforge.net/

Zu den früher berühmten Compilern von Borland: Turbo C++ für Windows Benutzerhandbuch, Borland, 2. A., 1992. Produziert von tewi in Zusammenarbeit mit Borland.

1.8 Literatur

Das Informatikumfeld

Böttcher Axel, Rechneraufbau und Rechnerarchitektur, Springer, 1. A. 2006, 210 S. TUB vorh.

Hellmann Roland, Rechnerarchitektur: Einführung in den Aufbau moderner Computer, Oldenbourg, 1. A. 2013. 394 S. TUB vorh.

Herrmann Paul, Rechnerarchitektur: Aufbau, Organisation und Implementierung, Inklusive 64-Bit-Technologie und Parallelrechner, Vieweg, 4. A. 2011, 456 S.. TUB vorh.

Herold Helmut, Grundlagen der Informatik, Pearson, 2. A. 2012, 800 S. Mit einer Einführung in C und Java.

Kersken, S., IT-Handbuch für Fachinformatiker, Galileo Press, 8. A. 2017, 1304 S. Wenig Hardware, viel über Betriebssysteme, noch viel mehr zum Programmieren. Mit nicht sehr weitgehenden Einführungen in C, Java, Perl, Ruby, XLM, HTML, XHTML, CSS, PHP, Java Script, Ajax und andere Sprachen. 6. A. als open Book bei Rheinwerk Computing. Stadtbib. vorh., 6. A. TUB vorh.

Malz Helmut, Rechnerarchitektur, eine Einführung für Ingenieure und Informatiker, Vieweg, 2. A. 2004, 244 S. Hardwarekenntnisse auffüllen. TUB im Lesesaal.

Ortmann Jürgen, Einführung in die PC – Grundlagen, Addison-Wesley, 8. A. 2003, 528S. Kapitel 1.1 meines alten Skripts stammte aus Ortmann.

Rechenberg Peter u. a., Informatik Handbuch, Hanser, 4. A. 2006, 1256 S. Gute Begriffserklärungen. TUB Lehrbuchsammlung.

Tanenbaum Andrew S. u. a., Rechnerarchitektur: Von der digitalen Logik zum Parallelrechner, Pearson, 2014, 800 S. TUB vorh.

Vogt Carsten, C für Java-Programmierer, Hanser, 2007, 255 S.

Verschiedene Programmiersprachen

Henning Peter A. u. a. (Autorenkollektiv), Handbuch Programmiersprachen, Hanser, 2006, 786 S.

Henning Peter A. u. a. (Autorenkollektiv), Taschenbuch Programmiersprachen, Hanser, 2. A. 2007, 631 S. TUB vorh.

C

Dausmann M. u. a., C als erste Programmiersprache: Vom Einsteiger zum Profi, mit den Konzepten von C11, Springer, 8. überarb. u. erw. A. 2014, 727 S. Leicht verständliche, vergleichsweise vollständige und doch noch kompakte Darstellung. Beschreibung der Struktogramme (Nassi-Shneidermann-Diagramme). Kurze Geschichte und sehr guter Stammbaum der höheren Programmiersprachen bis Java und C#. Mit Syntaxen und Struktogrammen. Ausführliche Beschreibung aller Operatoren. Anhang: Kurzbeschreibung aller Bibliotheken und ihrer Funktionen, wenn auch ohne Anwendungsbeispiele. Anhang Dateizugriff. Anhang Umwandeln zwischen Zahlensystemen. Auf CD: MS Visual C++ 6.0 Compiler in einer Evaluationsversion. TUB vorh.

Gaicher H. u. a., AVR Mikrocontroller - Programmierung in C: Eigene Projekte selbst entwickeln und verstehen, tredition, 2016, 368 S.

Herglotz Walter, Das Einsteigerseminar C, bhv Verlag, 5. und letzte Auflage 2001, 217 S. Die am einfachsten lesbare Einführung aber nur in die wichtigsten Sprachteile, vermeidet schwierige Fachsprache.

Kernighan Brian W. u. a., The C Programming Language, Prentice Hall, 1. A. 1978, 228 S. Von den Erfindern von C., TUB vorh.

Kernighan Brian W. u. a., Programmieren in C, Hanser, 1. A. 1983, 262 S. Übersetzung ins Deutsche des davor stehenden Buchs, TUB vorh.

Kernighan Brian W. u. a., The C Programming Language, Prentice Hall, 2. A. 1988, 272 S., ISBN 0131103628. Von den Erfindern von C, TUB vorh. Indian Reprint, PHI Learning Private Límited, 2013.

Kernighan Brian W. u. a., Programmieren in C, Hanser, 2. A. 1990, 262 S. Übersetzung ins Deutsche des davor stehenden Buchs, TUB vorh.

Kernighan Brian W. u. a., The Practice of Programming, Addison Wesley, 1999, 267 S. Unabhängig von C. Setzt aber C-, C++- oder Java-Kenntnisse voraus, TUB vorh.

Kernighan Brian W. u. a., Programmierpraxis, Prinzipien zur effizienten Programmierung, praktische Beispiele in C, C++ und Java, Addison-Wesley, 2000, 302 S. TUB vorh.

Kirch Ulla u.a., C - Lernen und professionell anwenden: Mit Microsoft Visual Studio Express 2012 und Open Watcom-Compiler auf der DVD, mitp Professional, 3. A. 2013, 912 S. Über C hinaus: Windows-Programmierung (GUI Grafical User Interface, Windows API Application Programming Interface, windows.h), GDI Graphics Defice Interface, Menüprogrammierung, Dialogprogrammierung, Hardware nahe Programmierung. Viel Profikritik in Amazon. Wird eher von Anwendern gelobt.

Knuth Donald E., The Art of Computer Programming, Volume 1: Fundamental Algorithms, Volume 2: Seminumerical Algorithms, Volume 3: Sorting and Searching, Addison Wesley. Hochgelobtes Referenzwerk für Algorithmen und Datenstrukturen.

Prinz Peter u. a., C Einführung und professionelle Anwendung, mitp, 2. A. 2007, ca. 1000 S. Sehr gute Begriffserklärungen. Mit einer Autorenversion des Visual C++-Compilers. Autorenversion bedeutet voller Sprachumfang aber Verbot, die erzeugten exe-Dateien zu vertreiben. TUB vorh.

Prinz Peter, C kurz & gut, O'Reilly, 1. A. 2002, 120 S. Behandelt ANSI C 99. Kompaktes Nachschlagewerk für jemanden, der sich in der Programmierwelt auskennt.

Prinz Peter, C - Das Übungsbuch: Testfragen und Aufgaben mit Lösungen, mitp, 2010, 320 S. Amazon: nur 4 und

5 Sterne.

Prinz Peter, C - Lernen und professionell anwenden: Mit Microsoft Visual Studio Express 2012 und Open Watcom-Compiler auf der DVD, mitp, 3. Auflage 2013, 912 S. Viel Profikritik in Amazon. Wird eher von Anwendern gelobt.

Reese R. M., Understanding and Using C Pointers, O'Reilly, 2013, 223 S. TUB 00, TUB Fernleihe verfügbar.

Schellong Helmut, Moderne C-Programmierung, Springer, 3. A. 2014, 335 S. Beschreibt C90, enthält aber auch je ein Kapitel über C99 und C11. Mit einem Abschnitt „C im Vergleich“ und ein Ausblick auf C++. TUB vorh.

Schröder K., C in der Praxis, Oldenburg, 1993. Enthält wenig über Pointer und Strukturen. Deshalb nur sinnvoll zusammen mit dem folgenden Buch.

Schröder K., C - Das Pointer-Buch, Oldenburg, 1992. Das Thema umfassend behandelt, wenn auch mit den beinahe schon üblichen sprachlichen Unpräzission. Auch mehrdimensionale Arrays, Strukturen, Unions, Bitfelder, Pointerfelder sowie dynamische Speicherverwaltung.

Sedgewick Robert, Algorithms in C, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching (3rd Edition) (Pts. 1-4), Pearson Studium, 2005, 744 S. Auch für andere Programmiersprachen erhältlich.

Straker David, C-Style, Standards and Guidelines, Prentice Hall, 1992. Kein Buch über C, sondern über guten Programmierstil mit C für Kenner der Sprache. TUB vorh.

Summit S., C Programming FAQs: Frequently Asked Questions, Addison Wesley, 1996. Amazon ausschl. 5 Sterne. Auch kostenlos im Internet.

Tondo, Clovis L. u. a., Das C-Lösungsbuch zu "Kernighan & Ritchie, Programmieren in C, Hanser, 2. A. 1990, 146 S.. TUB im Regal.

Wiegelmann Jörg, Softwareentwicklung in C für Mikroprozessoren und Mikrocontroller: C-Programmierung für Embedded-Systeme, VDE, 6. A. 2011, 319 S. Amazon: Eher für Anfänger als für Profis.

Wolf Jürgen, C von A bis Z, Galileo Computing, 3. A. 2009, 1190 S., ISBN 978-3-8362-1411-7. Oft gelobt. Keine Frage bleibt offen. Über C hinaus: CGI (Common Gate Interfcae, Webprogrammierung), MySQL Datenbankprogrammierung, Netzwerkprogrammierung, Prallelprocessing, GUI Grafikprogrammierung. Hinweise auf Fehlerquellen und den Standard C99. . TUB vorh. Auch als „openbook“ im Internet verfügbar:

http://openbook.rheinwerk-verlag.de/c_von_a_bis_z/