centro universitário FEI

CC8210 – NCA210 Programação Avançada I

Prof. Reinaldo A. C. Bianchi
Prof. Isaac Jesus da Silva
Prof. Danilo H. Perico

Manipulação de Strings

Caracteres

- A representação de um **caractere** é dada por um **número** inteiro, hexadecimal ou binário
- Esse número segue um padrão conhecido entre diversos sistemas computacionais:
  - ASCII - American Standard Code for Information Interchange (lê-se ASC2 OU asci)
  - ISO 8859 - é extensão ao código ASCII, inclui caracteres acentuados.
  - UTF - Unicode Transformation Format

Tabela ASCII

- **Tabela ASCII:**
  - 7 bits (números de 0 a 127)
- **Tabela ASCII Estendida**
  - 8 bits
    - Igual a tabela ASCII, porém contém mais caracteres:
      - Além do 0 ao 127,
      - Contém do 128 até o 255 (inclui os caracteres com acentos)

Tabela ASCII - 7 bits

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040	&#032;	Space	64	40	100	&#064;	@	96	60	140	&#096;	`
1	1	001	Start of Header	33	21	041	&#033;	!	65	41	101	&#065;	A	97	61	141	&#097;	a
2	2	002	Start of Text	34	22	042	&#034;	"	66	42	102	&#066;	B	98	62	142	&#098;	b
3	3	003	End of Text	35	23	043	&#035;	#	67	43	103	&#067;	C	99	63	143	&#099;	c
4	4	004	End of Transmission	36	24	044	&#036;	$	68	44	104	&#068;	D	100	64	144	&#100;	d
5	5	005	Enquiry	37	25	045	&#037;	%	69	45	105	&#069;	E	101	65	145	&#101;	e
6	6	006	Acknowledgment	38	26	046	&#038;	&	70	46	106	&#070;	F	102	66	146	&#102;	f
7	7	007	Bell	39	27	047	&#039;	'	71	47	107	&#071;	G	103	67	147	&#103;	g
8	8	010	Backspace	40	28	050	&#040;	(	72	48	110	&#072;	H	104	68	150	&#104;	h
9	9	011	Horizontal Tab	41	29	051	&#041;	)	73	49	111	&#073;	I	105	69	151	&#105;	i
10	A	012	Line feed	42	2A	052	&#042;	*	74	4A	112	&#074;	J	106	6A	152	&#106;	j
11	B	013	Vertical Tab	43	2B	053	&#043;	+	75	4B	113	&#075;	K	107	6B	153	&#107;	k
12	C	014	Form feed	44	2C	054	&#044;	,	76	4C	114	&#076;	L	108	6C	154	&#108;	l
13	D	015	Carriage return	45	2D	055	&#045;	-	77	4D	115	&#077;	M	109	6D	155	&#109;	m
14	E	016	Shift Out	46	2E	056	&#046;	.	78	4E	116	&#078;	N	110	6E	156	&#110;	n
15	F	017	Shift In	47	2F	057	&#047;	/	79	4F	117	&#079;	O	111	6F	157	&#111;	o
16	10	020	Data Link Escape	48	30	060	&#048;	0	80	50	120	&#080;	P	112	70	160	&#112;	p
17	11	021	Device Control 1	49	31	061	&#049;	1	81	51	121	&#081;	Q	113	71	161	&#113;	q
18	12	022	Device Control 2	50	32	062	&#050;	2	82	52	122	&#082;	R	114	72	162	&#114;	r
19	13	023	Device Control 3	51	33	063	&#051;	3	83	53	123	&#083;	S	115	73	163	&#115;	s
20	14	024	Device Control 4	52	34	064	&#052;	4	84	54	124	&#084;	T	116	74	164	&#116;	t
21	15	025	Negative Ack.	53	35	065	&#053;	5	85	55	125	&#085;	U	117	75	165	&#117;	u
22	16	026	Synchronous idle	54	36	066	&#054;	6	86	56	126	&#086;	V	118	76	166	&#118;	v
23	17	027	End of Trans. Block	55	37	067	&#055;	7	87	57	127	&#087;	W	119	77	167	&#119;	w
24	18	030	Cancel	56	38	070	&#056;	8	88	58	130	&#088;	X	120	78	170	&#120;	x
25	19	031	End of Medium	57	39	071	&#057;	9	89	59	131	&#089;	Y	121	79	171	&#121;	y
26	1A	032	Substitute	58	3A	072	&#058;	:	90	5A	132	&#090;	Z	122	7A	172	&#122;	z
27	1B	033	Escape	59	3B	073	&#059;	;	91	5B	133	&#091;	[	123	7B	173	&#123;	{
28	1C	034	File Separator	60	3C	074	&#060;	<	92	5C	134	&#092;	\	124	7C	174	&#124;	|
29	1D	035	Group Separator	61	3D	075	&#061;	=	93	5D	135	&#093;	]	125	7D	175	&#125;	}
30	1E	036	Record Separator	62	3E	076	&#062;	>	94	5E	136	&#094;	^	126	7E	176	&#126;	~
31	1F	037	Unit Separator	63	3F	077	&#063;	?	95	5F	137	&#095;	_	127	7F	177	&#127;	Del

asciicharstable.com

Extended ASCII - 8 bits

128	Ç	144	É	160	á	176	▒	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	▉	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	n
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

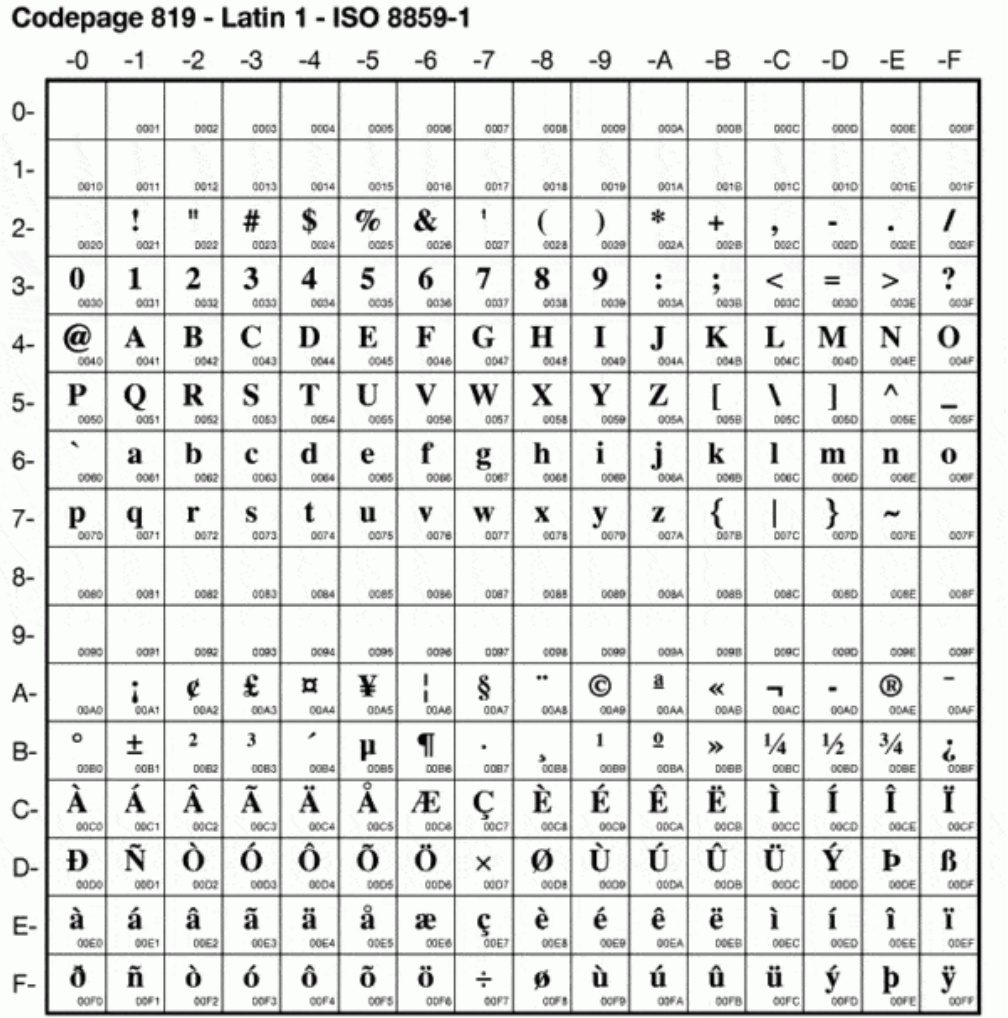
Source: www.LookupTables.com

ISO 8859

- ISO 8859 is a full series of 10 (and soon MORE) standardized multilingual single-byte coded (8bit) graphic character sets for writing in alphabetic languages:
  - ISO 8859-1: West European
  - ISO 8859-2: East European
  - ISO 8859-3: South European
  - ISO 8859-4: North European
  - ISO 8859-5: Cyrillic
  - ...

ISO 8859-1

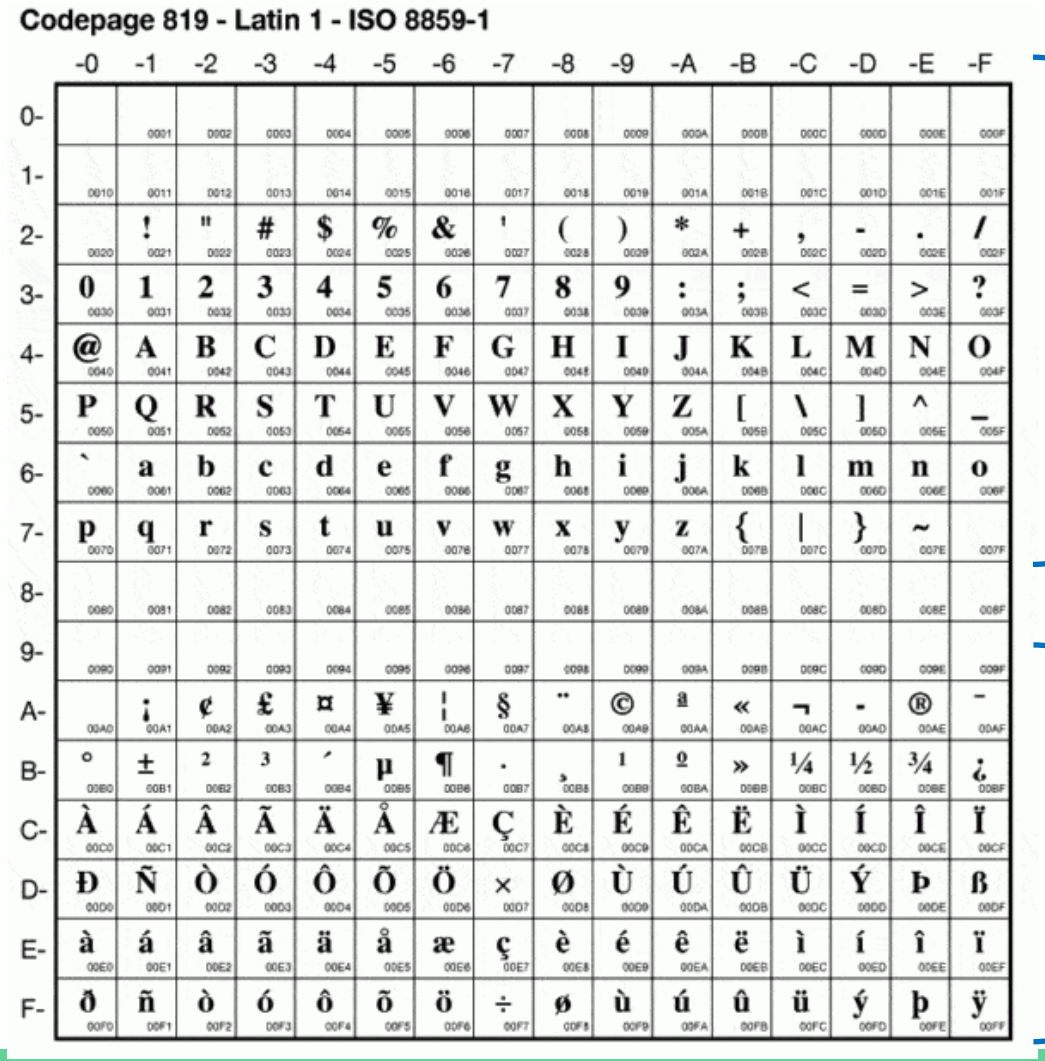
- Latin 1



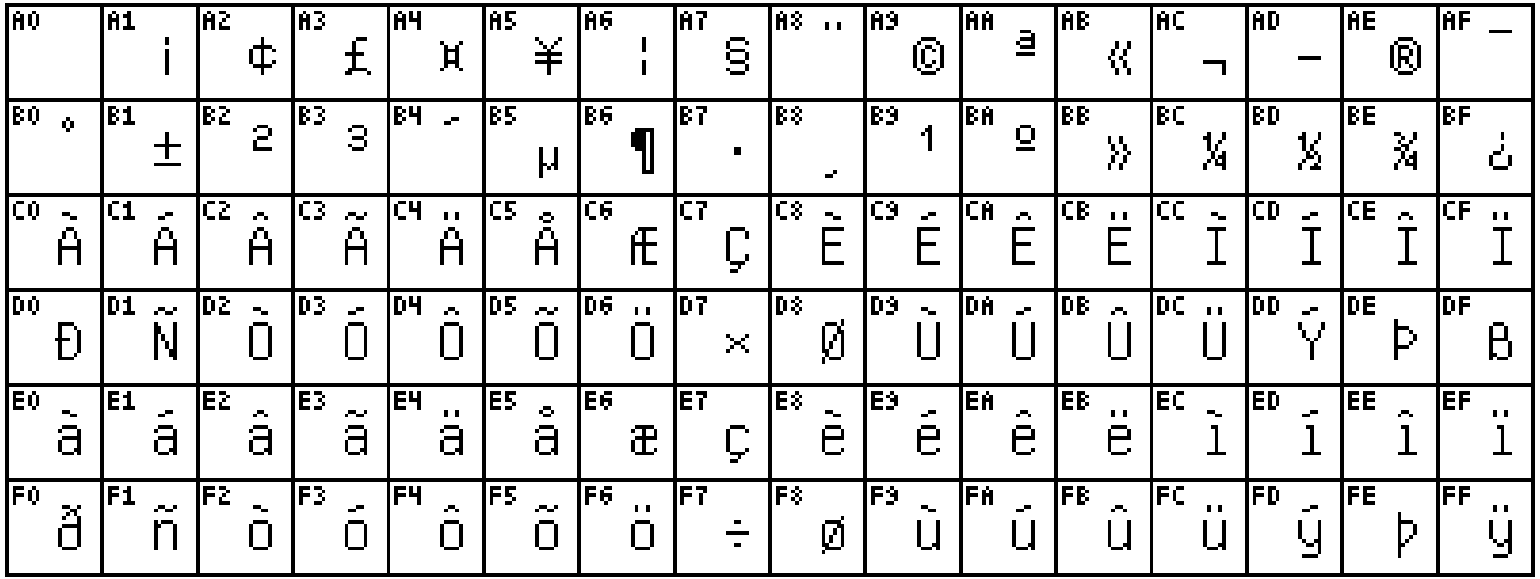
	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-		0001	0002	0003	0004	0005	0006	0007	0008	0009	000A	000B	000C	000D	000E	000F
1-	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	001A	001B	001C	001D	001E	001F
2-	0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3-	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4-	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5-	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
6-	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7-	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	007F
8-	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	008A	008B	008C	008D	008E	008F
9-	0090	0091	0092	0093	0094	0095	0096	0097	0098	0099	009A	009B	009C	009D	009E	009F
A-	00A0	¡ 00A1	¢ 00A2	£ 00A3	¤ 00A4	¥ 00A5	¦ 00A6	§ 00A7	¨ 00A8	© 00A9	ª 00AA	« 00AB	¬ 00AC	- 00AD	® 00AE	¯ 00AF
B-	° 00B0	± 00B1	2 00B2	3 00B3	´ 00B4	µ 00B5	¶ 00B6	· 00B7	¸ 00B8	1 00B9	º 00BA	» 00BB	¼ 00BC	½ 00BD	¾ 00BE	¿ 00BF
C-	À 00C0	Á 00C1	Â 00C2	Ã 00C3	Ä 00C4	Å 00C5	Æ 00C6	Ç 00C7	È 00C8	É 00C9	Ê 00CA	Ë 00CB	Ì 00CC	Í 00CD	Î 00CE	Ï 00CF
D-	Ð 00D0	Ñ 00D1	Ò 00D2	Ó 00D3	Ô 00D4	Õ 00D5	Ö 00D6	× 00D7	Ø 00D8	Ù 00D9	Ú 00DA	Û 00DB	Ü 00DC	Ý 00DD	Þ 00DE	ß 00DF
E-	à 00E0	á 00E1	â 00E2	ã 00E3	ä 00E4	å 00E5	æ 00E6	ç 00E7	è 00E8	é 00E9	ê 00EA	ë 00EB	ì 00EC	í 00ED	î 00EE	ï 00EF
F-	ð 00F0	ñ 00F1	ò 00F2	ó 00F3	ô 00F4	õ 00F5	ö 00F6	÷ 00F7	ø 00F8	ù 00F9	ú 00FA	û 00FB	ü 00FC	ý 00FD	þ 00FE	ÿ 00FF

ISO 8859-1

- Latin 1

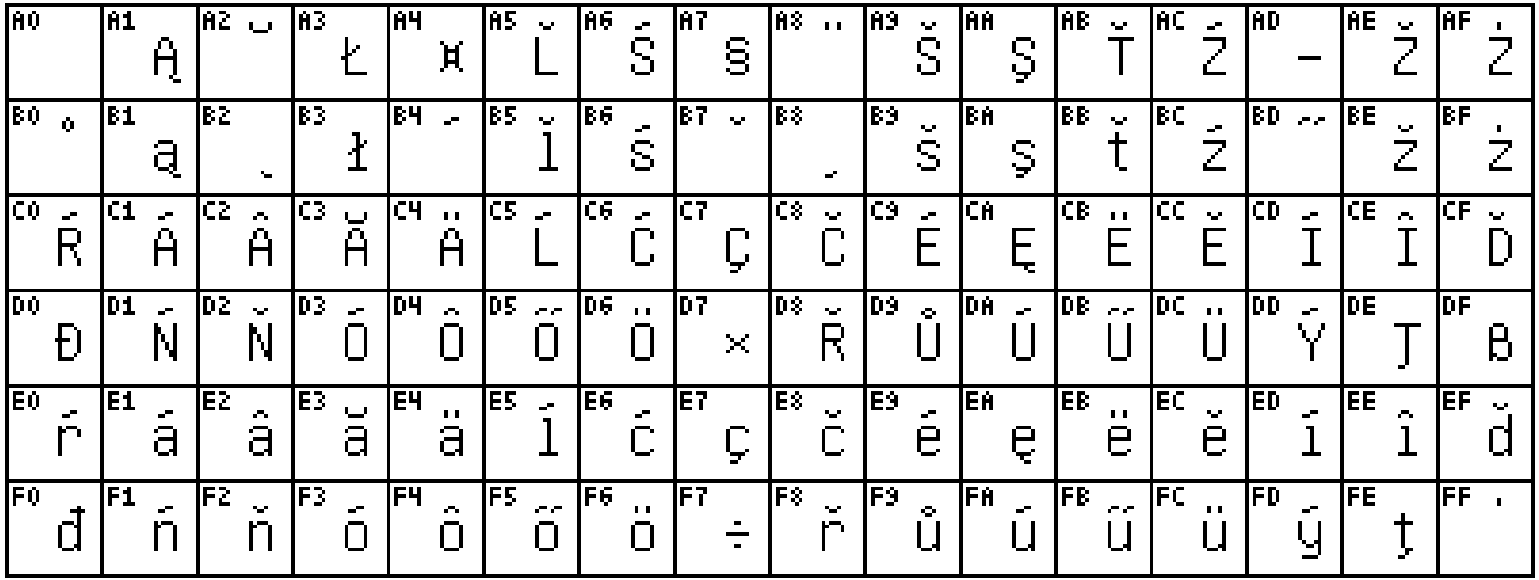


ISO 8859-1: West European



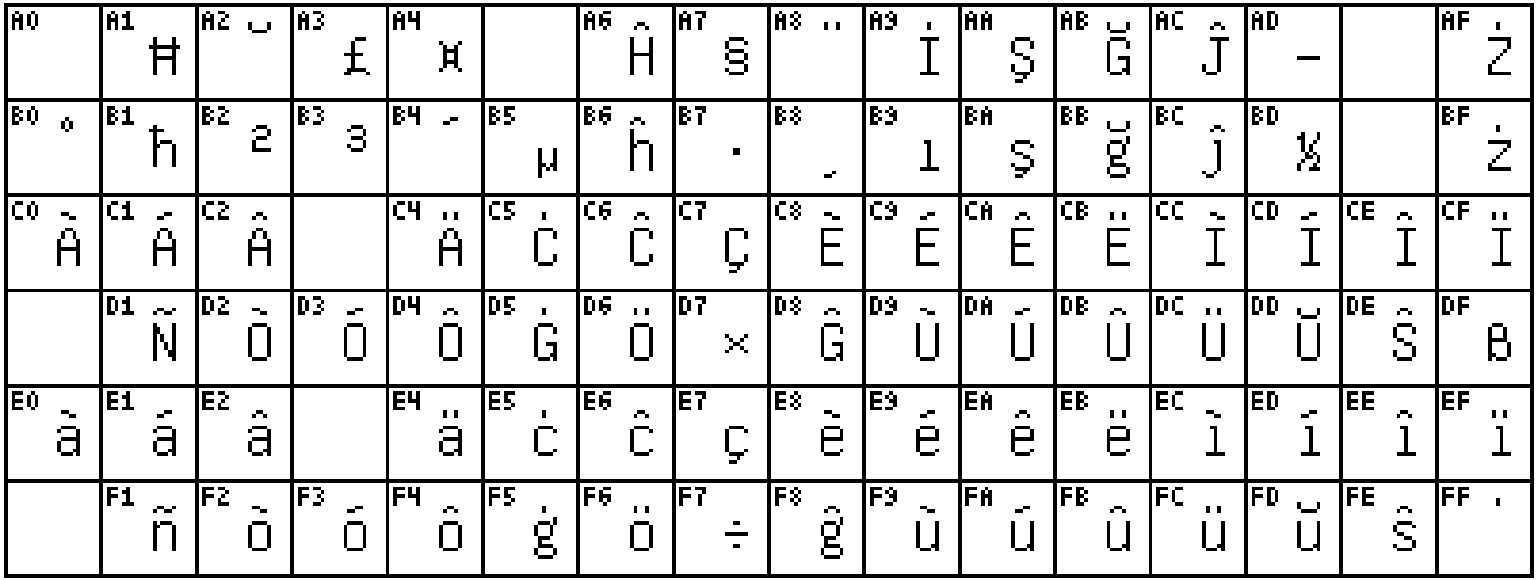
French (fr), Spanish (es), Catalan (ca), Basque (eu), Portuguese (pt), Italian (it), Albanian (sq), Rhaeto-Romanic (rm), Dutch (nl), German (de), Danish (da), Swedish (sv), Norwegian (no), Finnish (fi), Faroese (fo), Icelandic (is), Irish (ga), Scottish (gd), and English (en), incidentally also Afrikaans (af) and Swahili (sw), thus in effect also the entire American continent, Australia and much of Africa.

ISO 8859-2: East European



Czech (cs), Hungarian (hu), Polish (pl), Romanian (ro), Croatian (hr), Slovak (sk), Slovenian (sl), Sorbian.

ISO 8859-3: South European



ISO 8859-5: Cyrillic

A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
	Ё	Ђ	Ѓ	Є	Ѕ	І	Ї	Ј	Љ	Њ	Ћ	Ќ		Ў	Џ
B0	**B1**	**B2**	**B3**	**B4**	**B5**	**B6**	**B7**	**B8**	**B9**	**BA**	**BB**	**BC**	**BD**	**BE**	**BF**
А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
C0	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**CA**	**CB**	**CC**	**CD**	**CE**	**CF**
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
D0	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**	**D9**	**DA**	**DB**	**DC**	**DD**	**DE**	**DF**
а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**	**EA**	**EB**	**EC**	**ED**	**EE**	**EF**
р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
F0	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**FA**	**FB**	**FC**	**FD**	**FE**	**FF**
№	ё	ђ	ѓ	є	ѕ	і	ї	ј	љ	њ	ћ	ќ	§	ў	џ

Bulgarian (bg), Byelorussian (be), Macedonian (mk), Russian (ru), Serbian

Unicode

- Unicode é um padrão que permite aos computadores representar e manipular, de forma consistente qualquer sistema de escrita existente.
- O padrão consiste de quase 138 mil caracteres:
  - Define uma metodologia para codificação e um conjunto de codificações padrões de caracteres, uma enumeração de propriedades de caracteres como caixa alta e caixa baixa, além de regras para normalização, decomposição, ordenação alfabética e renderização.
  - Os 256 primeiros códigos Unicode são idênticos aos do padrão ISO 8859-1

UTF-8

- Um dos padrões mais utilizados da atualidade é o UTF-8 (8-bit Unicode Transformation Format)
- É um tipo de codificação binária (Unicode) de comprimento variável.
- UTF-8 pode representar qualquer caractere universal padrão do Unicode, sendo também compatível com o ASCII
    - Por esta razão, está lentamente a ser adaptado como tipo de codificação padrão para e-mail, páginas web, e outros locais onde os caracteres são armazenados.

Unicode

A:A B:B C:C D:D E:E

U+1F600 : 😀	U+263A : ☺	U+1F61F : 😟	U+1F61E : 😞
U+1F603 : 😃	U+1F61A : 😚	U+1F641 : 🙁	U+1F648 : 🙈
U+1F604 : 😄	U+1F619 : 😙	U+2639 : ☹	U+1F649 : 🙉
U+1F601 : 😁	U+1F60B : 😋	U+1F62E : 😮	U+1F64A : 🙊
U+1F606 : 😆	U+1F61B : 😛	U+1F62F : 😯	U+1F44B : 👋
U+1F605 : 😅	U+1F61C : 😜	U+1F632 : 😲	U+1F91A : 🤚
U+1F923 : 🤣	U+1F92A : 🤪	U+1F633 : 😳	U+1F590 : 🖐
U+1F602 : 😂	U+1F61D : 😝	U+1F626 : 😦	U+270B : ✋
U+1F642 : 🙂	U+1F911 : 🤑	U+1F627 : 😧	U+1F596 : 🖖
U+1F643 : 🙃	U+1F917 : 🤗	U+1F628 : 😨	U+1F44C : 👌
U+1F609 : 😉	U+1F92D : 🤭	U+1F630 : 😰	U+270C : ✌
U+1F60A : 😊	U+1F92B : 🤫	U+1F625 : 😥	U+1F91E : 🤞
U+1F607 : 😇	U+1F914 : 🤔	U+1F622 : 😢	U+1F91F : 🤟
U+1F60D : 😍	U+1F60E : 😎	U+1F62D : 😭	U+1F918 : 🤘
U+1F929 : 🤩	U+1F913 : 🤓	U+1F631 : 😱	U+1F919 : 🤙
U+1F618 : 😘	U+1F9D0 : 🧐	U+1F616 : 😖	U+1F44D : 👍
U+1F617 : 😗	U+1F615 : 😕	U+1F623 : 😣	U+1F44E : 👎

№	Code	Brow.	Chart	Apple	Goog.	One	Twtr.	Wind.	GMail	DCM	KDDI	SB	Name
1	U+1F600									*missing*	*missing*	*missing*	GRINNING FACE
2	U+1F601												GRINNING FACE WITH SMILING EYES
3	U+1F602									*missing*			FACE WITH TEARS OF JOY
4	U+1F603												SMILING FACE WITH OPEN MOUTH
5	U+1F604									*missing*	*missing*		SMILING FACE WITH OPEN MOUTH AND SMILING EYES
6	U+1F605										*missing*	*missing*	SMILING FACE WITH OPEN MOUTH AND COLD SWEAT
7	U+1F606										*missing*	*missing*	SMILING FACE WITH OPEN MOUTH AND TIGHTLY-CLOSED EYES
8	U+1F609												WINKING FACE
9	U+1F60A									*missing*			SMILING FACE WITH SMILING EYES
10	U+1F60B										*missing*	*missing*	FACE SAVOURING DELICIOUS FOOD

Unicode em Python

- ***print()*** com unicode:

```
print(u'\u00ae')
```

```
®
```

```
print(u'\u0061')
```

```
a
```

```
print(u'\u00c7')
```

```
Ç
```

- No Python, aspas simples ou duplas são intercambiáveis para representar strings

Unicode em Python

```
# grinning face
print("\U0001f600")
```

😀

```
# grinning squinting face
print("\U0001F606")
```

😆

```
# rolling on the floor laughing
print("\U0001F923")
```

🤣

ord()

- Imprimir o valor ASCII/Unicode de um caractere - função ord( ):

```
print( ord("a") )
```

```
97
```

```
print( ord("A") )
```

```
65
```

```
print( ord("á") )
```

```
225
```

Unicode – ord()

```
print(ord('a'))
```

```
97
```

```
print(ord('á'))
```

```
225
```

```
print(ord('ç'))
```

```
231
```

```
print(ord('😀'))
```

```
128512
```

```
print(hex(ord('😀')))
```

```
0x1f600
```

chr()

- Imprimir o **caractere** a partir de um valor **ASCII/Unicode** - função **chr( ):**

```
print( chr(97) )
```

```
a
```

```
print( chr(65) )
```

```
A
```

```
print( chr(225) )
```

```
á
```

Exemplo: entrada em hexadecimal

```
print( chr(0xe1) )
```

```
á
```

Strings

- Strings são cadeias de caracteres:
  - Sequência de caracteres
- String em Python são cercados por aspas simples ou aspas duplas:
  - 'hello' é igual a "hello".
- Exemplo:
  - `print("Hello")`
  - `print('Hello')`

Strings – aspas triplas

- O Python também têm a opção de aspas triplas: *"""texto """*
- Este comando pode ser chamado de **bloco de string**
- Aspas triplas são bastante úteis para textos com múltiplas linhas
- Exemplo:

```
print("""Este
texto
tem
muitas
linhas""")
```

```
Este
texto
tem
muitas
linhas
```

Atribuir string a uma variável

- Para atribuir uma string a uma variável se faz como se fosse qualquer outro literal:
  - Com o nome variável seguido por um sinal igual e a sequência:
- Exemplo:
  - `a = "Hello"`
  - `print(a)`

Caracteres de escape

- Para inserir caracteres que são ilegais em uma sequência, use um caractere de escape – *Escape Character*
- Um caractere de escape é uma barra invertida seguido pelo caractere que você deseja inserir.
- Um exemplo de um caractere ilegal é uma citação dupla dentro de uma String que é cercada por citações duplas:
  - txt = "We are the so-called "Vikings" from the north.”
  - Gera erro! - Deve ser usado \"

Caracteres de escape

- \’ Single Quote
- \\ Backslash
- \n New Line
- \r Carriage Return
- \t Tab
- \b Backspace
- \f Form Feed
- \xhh Hex value

```
txt = "We are the so-called \"Vikings\" from the north."
print(txt)
```

```
We are the so-called "Vikings" from the north.
```

Strings são Arrays !!!

- Strings are Arrays
- Como muitas outras linguagens de programação populares, strings em Python são arrays, tuplas de bytes representando caracteres Unicode.
- No entanto, Python não tem um tipo de dados de caractere:
  - um único caractere é simplesmente uma sequência com um comprimento de 1.
- Colchetes podem ser usados para acessar elementos da String.

Strings – Índices

- De uma maneira geral, *strings* são **tuplas** de caracteres
- Podemos acessar cada caractere utilizando o índice de sua posição dentro da *String*
- Exemplos:

```
s = "spam"

print(s[0])
```

```
s
```

```
s = "spam"

print(s[2])
```

```
a
```

```
s = "spam"

print(s[-2])
```

```
a
```

Strings – Fatiamento (*slicing*)

- Podemos também utilizar o fatiamento (*slicing*) nas strings
- O fatiamento serve para extrairmos uma seção específica da *string*
- Exemplos:

```
s = "spam"

print(s[1:3])
```

```
pa
```

```
s = "spam"

print(s[:3])
```

```
spa
```

```
s = "spam"

print(s[1:])
```

```
pam
```

Strings – Imutabilidade

- As strings seguem o conceito de imutabilidade
- Isso quer dizer que não é possível alterar um único caractere de uma string
- Por exemplo: Alterar o “s” por “z” na palavra “spam”

```
s = "spam"

s[0] = "z"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-13-6ae494b2e9c0> in <module>
      1 s = "spam"
      2
----> 3 s[0] = "z"

TypeError: 'str' object does not support item assignment
```

Strings – Imutabilidade

- Para alterarmos um ou mais caracteres de uma string, normalmente criamos outra variável
- Exemplo: alterar ‘s’ por ‘z’ na palavra “spam”

```
s = "spam"

s_novo = "z" + s[1:]

print(s_novo)
```

```
zpam
```

Strings – Métodos

- Existem muitos métodos próprios para serem utilizados com strings
- Alguns exemplos:
  - capitalize( )
  - count( )
  - find( )
  - lower( )
  - islower( )
  - isdigit( )
  - isalpha( )
  - isupper( )
  - split( )
  - strip( )
  - replace( )

Strings – Métodos

```
s = "spammy"
# conta as ocorrências de "m"
s.count("m")
```

```
2
```

```
# coloca a primeira letra em maiúscula
s.capitalize()
```

```
'Spammy'
```

```
# transforma tudo em maiúsculo
s.upper()
```

```
'SPAMMY'
```

```
# retorna o índice de "a"
s.find("a")
```

```
2
```

```
# verifica se os caracteres são minúsculos
s.islower()
```

```
True
```

```
# substitui "p" por "d"
s.replace("p", "d")
```

```
'sdammy'
```

```
# divide uma frase no argumento utilizado
s = "teste*de*spam"
s.split("*")
```

```
['teste', 'de', 'spam']
```

```
# remove o espaço vazio no começo e no fim da string
s = "teste de spam\n"
s.strip()
```

```
'teste de spam'
```

Strings – Métodos

```
s = "22"
s.isdigit()
```

```
True
```

```
s = "22"
s.isalpha()
```

```
False
```

```
s = "Aula"
s.isdigit()
```

```
False
```

```
s = "Aula"
s.isalpha()
```

```
True
```

```
s = "Aula de Python, Python 3"
# conta as ocorrências da palavra "Python"
s.count("Python")
```

```
2
```

```
s = "  Aula de Python, Python 3  "
# remove o espaço vazio no começo e no fim da String
s.strip()
```

```
'Aula de Python, Python 3'
```

```
s = "Aula de Python, Python 3"
# encontra a primeira ocorrência da palavra "Python"
s.find("Python")
```

```
8
```

```
s = "Aula de Python, Python 3"
# substitui "Python" por "C++"
s.replace("Python", "C++")
```

```
'Aula de C++, C++ 3'
```

Strings – Método len()

- Assim como nas listas ou tuplas, podemos utilizar também o método len( ) para obter o tamanho da string.
- Tamanho significa o número de caracteres
- Exemplo:

```
s = "olá mundo!"
print(len(s))
```

```
10
```

String – Método format()

- Python 3 introduziu uma nova maneira de fazer a formatação de strings que também foi posteriormente portado para Python 2.7.
- Esta formatação de string "novo estilo" se livra da sintaxe usando o “%” e torna a sintaxe para formatação de strings mais regular.
- A formatação agora é tratada chamando .format() em um objeto de sequência.

String – Método format()

```
name = "John"
'Hello, {}'.format(name)
```

```
'Hello, John'
```

```
age = 36
txt  = "My name is John, and I am {}"
print(txt.format(age))
```

```
My name is John, and I am 36
```

String – Método format()

```
name = "Bob"
error = 0xbadc0ffee
print('Hey {name}, there is a 0x{errno:x} error!'.format(name=name, errno=error))
```

```
Hey Bob, there is a 0xbadc0ffee error!
```

String – Método format()

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want {} pieces of item {} for {} dollars."
print(myorder.format(quantity, itemno, price))
```

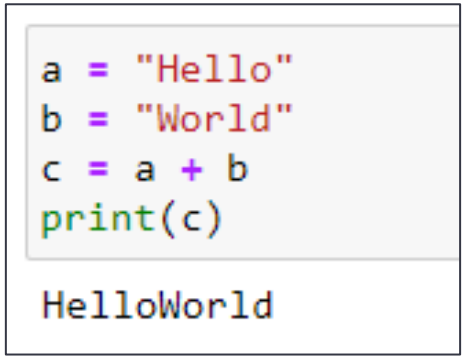
```
I want 3 pieces of item 567 for 49.95 dollars.
```

Concatenação de Strings

- Para concatenar, ou combinar, duas strings você pode usar o operador +.

- Exemplo:

```
a = "Hello"
b = "World"
c = a + b
print(c)
```



Conclusão

- Vimos na aula de hoje alguns comandos básicos para trabalhar com strings em Python...

- Já podemos começar a fazer exercícios!

Exercício em aula de teoria

- Peça ao usuário uma string e imprima se essa string é um palíndromo ou não.
- Palíndromo é uma palavra ou frase (normalmente, ignorando-se os espaços em branco) que se pode ler, indiferentemente, da esquerda para a direita ou vice-versa.
- Exemplos: “ovo”; “a grama é amarga”; “A sacada da casa”, “Luz azul”.

Exercício em aula de teoria

- Faça um algoritmo que conte a quantidade de incidências de todas as palavras em uma String, assim listando todas as palavras e suas quantidades, considere como palavras as que tenha uma quantidade igual ou maior que duas letras.

Fim


How am I so good at this? I've never written code before!
python
light roast comics