Introdução ao Processamento de Dados

Francisco Sant'Anna

francisco@ime.uerj.br

http://github.com/fsantanna/IPD

Strings

- São sequências (cadeias) de caracteres
- São parecidas com listas
  - possuem índices, comprimento, etc

```
nome = “Joaquim”
nome

nome[0]

len(nome)
```

Índices

- Positivos: do início para o fim
- Negativos: do fim para o início
- Intervalos: subpartes da string
- `len`: comprimento

```
print(nome[0], nome[-7])

print(nome[6], nome[-1])

nome[1:6]

len(nome)
```

Concatenação

- Operador +
  - une (concatena) strings

```
v = 'ola' + ' ' + 'mundo'

v
```

Representação Interna

- Na memória, cada caractere é guardado como um código numérico pré-determinado.

TABLE 3
ASCII CHARACTER CODES (DECIMAL)

0	Ctrl-@	32	Space	64	@	96	`
1	Ctrl-A	33	!	65	A	97	a
2	Ctrl-B	34	"	66	B	98	b
3	Ctrl-C	35	#	67	C	99	c
4	Ctrl-D	36	$	68	D	100	d
5	Ctrl-E	37	%	69	E	101	e
6	Ctrl-F	38	&	70	F	102	f
7	Ctrl-G	39	'	71	G	103	g
8	Backspace	40	(	72	H	104	h
9	Tab	41	)	73	I	105	i
10	Ctrl-J	42	*	74	J	106	j
11	Ctrl-K	43	+	75	K	107	k
12	Ctrl-L	44	,	76	L	108	l
13	Return	45	-	77	M	109	m
14	Ctrl-N	46	.	78	N	110	n
15	Ctrl-O	47	/	79	O	111	o
16	Ctrl-P	48	0	80	P	112	p
17	Ctrl-Q	49	1	81	Q	113	q
18	Ctrl-R	50	2	82	R	114	r
19	Ctrl-S	51	3	83	S	115	s
20	Ctrl-T	52	4	84	T	116	t
21	Ctrl-U	53	5	85	U	117	u
22	Ctrl-V	54	6	86	V	118	v
23	Ctrl-W	55	7	87	W	119	w
24	Ctrl-X	56	8	88	X	120	x
25	Ctrl-Y	57	9	89	Y	121	y
26	Ctrl-Z	58	:	90	Z	122	z
27	Escape	59	;	91	[	123	{
28	Ctrl-\	60	<	92	\	124	|
29	Ctrl-]	61	=	93	]	125	}
30	Ctrl-^	62	>	94	^	126	~
31	Ctrl-_	63	?	95	_	127	Delete

Representação Interna

- `ord`: converte **um** caractere para um número
- `chr`: converte número para **um** caractere
- `str`: converte um valor qualquer para uma string

```
ord('a')
chr(97)

str(True)
```

Exemplo

- Ler uma string e exibir a soma dos códigos numéricos de todos os caracteres

```
s = raw_input('string: ')
soma = 0
for i in range(0,len(s)):
    soma = soma + ord(s[i])
print 'SOMA: ', soma
```

Trabalhando com strings

- `s1.count(s2)`
  - conta quantas vezes `s2` aparece em `s1`
- `s1.lower() / s1.upper()`
  - converte `s1` para caixa baixa/alta

```
'banana'.count('na')
'banana'.count('aa')

'brasil'.upper()
'.JPG'.lower()
```

Trabalhando com strings

- `s1.isdigit()` / `s1.isalpha()` / `s1.isalnum()`
  - diz se `s1` é formado apenas por números / letras / números e letras
- `s1.replace(s2,s3)`
  - converte todas as ocorrências de `s2` em `s1` por `s3`

```
'123'.isdigit()
'maria'.isalpha()
'joao1991'.isalnum()

'brazil'.replace('z','s')
```

Trabalhando com strings

- `s1.split(s2)`
  - quebra `s1` em vários pedaços separados por `s2`
  - guarda os pedaços em uma nova lista

```
'1,2,3,4'.split(',')
```