COESIONE ITALIA 21-27 PIEMONTE
Cofinanziato dall'Unione europea
REGIONE PIEMONTE

Numeri binari, bit e byte

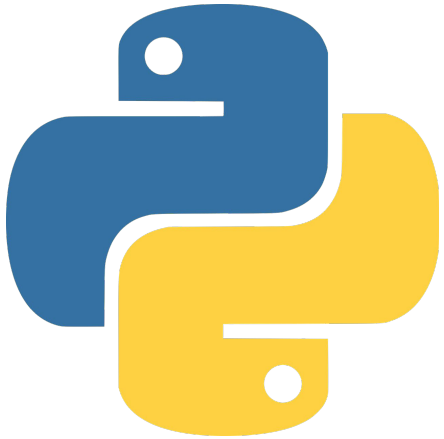
Introduzione ai numeri binari, ottali ed esadecimali con applicazioni in Python

Docente:
Enrico Zimuel

Intervento realizzato da ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Programma

- Numeri binari
- Python e i numeri binari
- Sistema ottale
- Sistema esadecimale
- Bit e Byte
- ASCII
- Byte in Python
- Operazioni bit a bit
- Base64
- Base64 in Python



Numeri binari

- Un **bit** è l'unità di misura fondamentale in informatica è può assumere due valori **0** e **1** (numero in base 2)
- Un numero **binario** è una sequenza di bit ossia di 0 e 1
  - 00, 01, 10, 11 equivale a 0, 1, 2, 3 in decimale
- Convertire un numero binario in decimale (es. 10111)

1	0	1	1	1
2^4	2^3	2^2	2^1	2^0
16	0	4	2	1

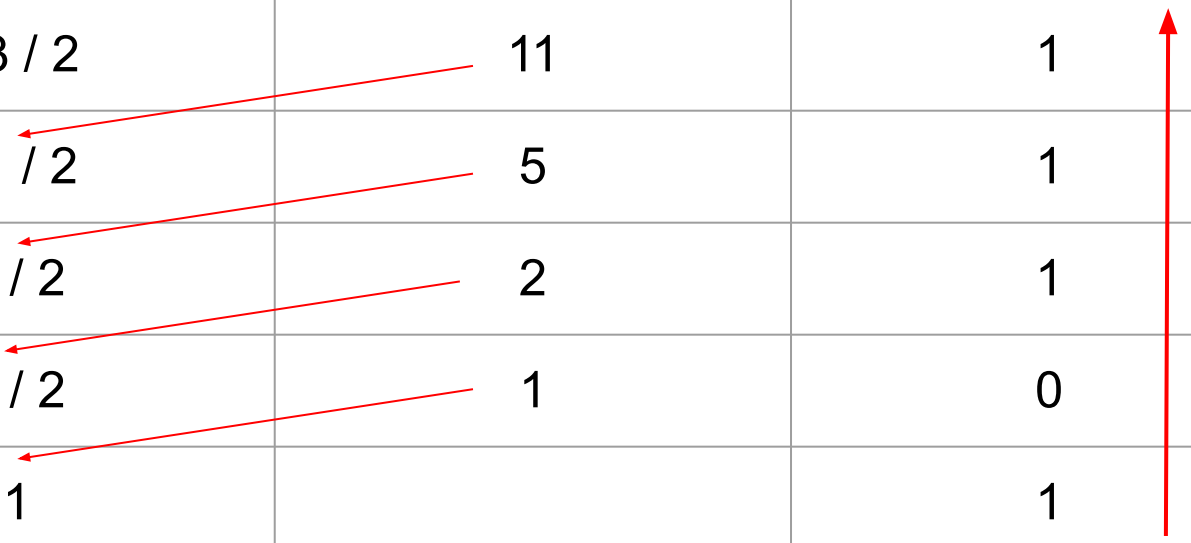
← peso di ogni bit

= 16 + 4 + 2 + 1 = 23

Numeri binari (2)

- Convertire un numero decimale in binario (es. 23)

Divisione per 2	Risultato	Resto
23 / 2	11	1
11 / 2	5	1
5 / 2	2	1
2 / 2	1	0
1		1



= 10111

Python e numeri binari

- In Python è possibile inserire un numero binario con la notazione **0b**

```
num = 0b10111
print(num) # 23
```

- Da numero intero a numero binario con la funzione **bin()**

```
num = 23
print(bin(num)) # 0b10111
```

- Per sapere il numero di bit necessari per rappresentare un numero si utilizza il metodo **bit_length()**

```
num = 23
print(num.bit_length()) # 5
```

Ottale

- Ci sono altri formati numerici utilizzati in informatica: **ottali** ed **esadecimali**
- Un numero ottale utilizza 8 simboli (numero in base 8): 0, 1, 2, 3, 4, 5, 6, 7
- Il numero in ottale 27 corrisponde a 23 in decimale

2	7
8^1	8^0
16	7

= 16 + 7 = 23

/ 8	Risultato	Resto
23 / 8	2	7
2		2

= 27

Ottali in Python

- Posso inserire un numero ottale con il prefisso **0o** (zero e o minuscolo)

```
num = 0o27
print(num) # 23
```

- Da numero intero a numero ottale con la funzione **oct()**

```
num = 23
print(oct(num)) # 0o27
```

Esadecimale

- Un numero esadecimale utilizza 16 simboli (numero in base 16):
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f
- Il numero in esadecimale 17 corrisponde a 23 in decimale

1	7
16^1	16^0
16	7

= 16 + 7 = 23

/ 16	Risultato	Resto
23 / 16	1	7
1		1

= 17

Esadecimali in Python

- Posso inserire un numero ottale con il prefisso **0x**

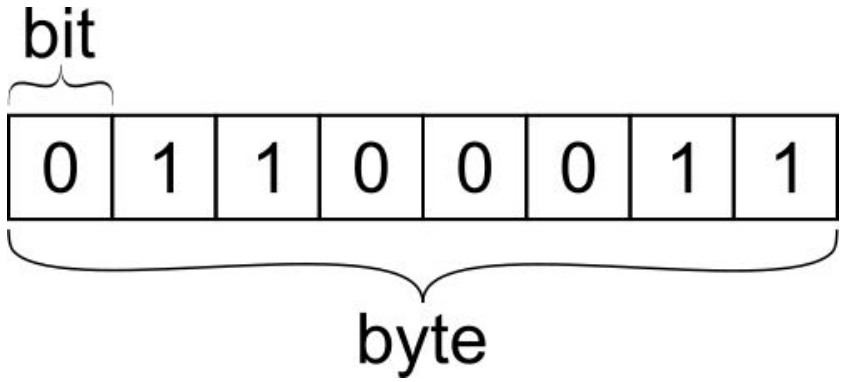
```
num = 0x17
print(num) # 23
```

- Da numero intero a numero esadecimale con la funzione **hex()**

```
num = 23
print(hex(num)) # 0x17
```

Byte

- Un **byte** è un insieme di 8 bit
- Il byte è l'unità di misura dello spazio occupato dai file
- Un byte corrisponde ad un carattere della tabella ASCII (256 simboli diversi)
- Alcuni di questi sono dei caratteri di controllo, non stampabili



ASCII

numeri e lettere	ASCII
0…9	48…57
A...Z	65…90
a…z	97…122

ASCII in Python

- La tabella ASCII in Python può essere gestita con le funzioni **chr()** e **ord()**
- E' possibile ricavare un simbolo ASCII partendo dalla sua posizione con la funzione **chr()**

```
num = 48
print(chr(num)) # 0
```

- Per sapere il valore ASCII di un carattere si utilizza **ord()**

```
num = "0"
print(ord(num)) # 48
```

Byte in Python

- E’ possibile specificare una stringa in byte in Python utilizzando il prefisso **b**

```
num = b'abc'
```

- Convertire un numero in bytes con il metodo **to_bytes(),** specificando il numero di bytes

```
num = 123
print(num.to_bytes(1)) # b'{'
print(num.to_bytes(3)) # b'\x00\x00{'
```

Byte e Bit

- E’ possibile convertire una stringa in byte con il metodo **encode()**

```
text = "abc"
print(text.encode("utf-8")) # b'abc'
print(text.encode()) # b'abc', default utf-8
```

- Da byte a stringa con il metodo **decode()**

```
textb = b'abc'
print(textb.decode("utf-8")) # abc
print(textb.decode()) # abc, default utf-8
```

Operazioni bit a bit

0b1011 **|** 0b0111 == 0b1111	OR
0b1011 **^** 0b0111 == 0b1100	XOR
0b1011 **&** 0b0111 == 0b0011	AND
0b1011 **<<** 2 == 0b101100	spostamento (shift) a sinistra di 2 bit
0b1011 **>>** 2 == 0b10	spostamento (shift) a destra di 2 bit
~ 0b1011 == 0b0100	NOT

Base64

- Un sistema di codifica molto utilizzato per l'interscambio di dati su Internet è il formato **Base64**
- Questo formato utilizza 64 caratteri: a…z, A…Z, 0…9, +, /
- 64 caratteri possono essere rappresentati con 6 bit
- Per codificare un testo in Base64 devo raggruppare i bit in sestetti
- Ogni 3 caratteri del testo (8 x 3 = 24 bit) equivalgono a 4 caratteri in Base64 (6 x 4 = 24 bit)
- Se la lunghezza del testo non è multipla di 3 si utilizza il carattere di **Padding** (=), uno o al massimo due

Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	0
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/

Base64 esempio

- La sequenza **Man** viene codificata in Base64 con **TWFu**

Source	Text (ASCII)	M							a								n							
	Octets	77 (0x4d)							97 (0x61)								110 (0x6e)							
Bits		0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
Base64 encoded	Sextets	19							22								5							46	
	Character	T							W								F							u	
	Octets	84 (0x54)							87 (0x57)								70 (0x46)							117 (0x75)	

Base64 esempio (2)

- La sequenza **Ma** viene codificata in Base64 con **TWE=**

Source	Text (ASCII)	M								a																	
	Octets	77 (0x4d)								97 (0x61)																	
Bits		0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0								
Base64 encoded	Sextets	19						22						4				Padding									
	Character	T						W						E				=									
	Octets	84 (0x54)						87 (0x57)						69 (0x45)				61 (0x3D)									

Base64 esempio (3)

- La lettera **M** viene codificata in Base64 con **TQ==**

Source	Text (ASCII)	M			
	Octets	77 (0x4d)			
Bits		0 1 0 0 1 1 0 1	0 0 0 0		
Base64 encoded	Sextets	19	16	Padding	Padding
	Character	T	Q	=	=
	Octets	84 (0x54)	81 (0x51)	61 (0x3D)	61 (0x3D)

Base64 in Python

- E’ possibile codificare una sequenza di byte in Base64 con il modulo base64

```
import base64
a = b'Man'
print(base64.b64encode(a)) # b'TWFu'
```

- Inverso, decodificare da Base64 in una sequenza di byte

```
import base64
a = b'TWFu'
print(base64.b64decode(a)) # b'Man'
```

Grazie!

Per informazioni: enrico.zimuel@its-ictpiemonte.it

Intervento realizzato da ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE