Il codice binario

Mattia Cozzi
cozzimattia@gmail.com

a.s. 2023/2024

Contenuti

Definizioni

Computer

Il computer è una macchina elettronica capace di ricevere, trasmettere, memorizzare e soprattutto elaborare informazioni sotto forma di *dati*.

Hardware

L'hardware è l'insieme delle parti elettroniche e meccaniche che compongono fisicamente il computer

Software

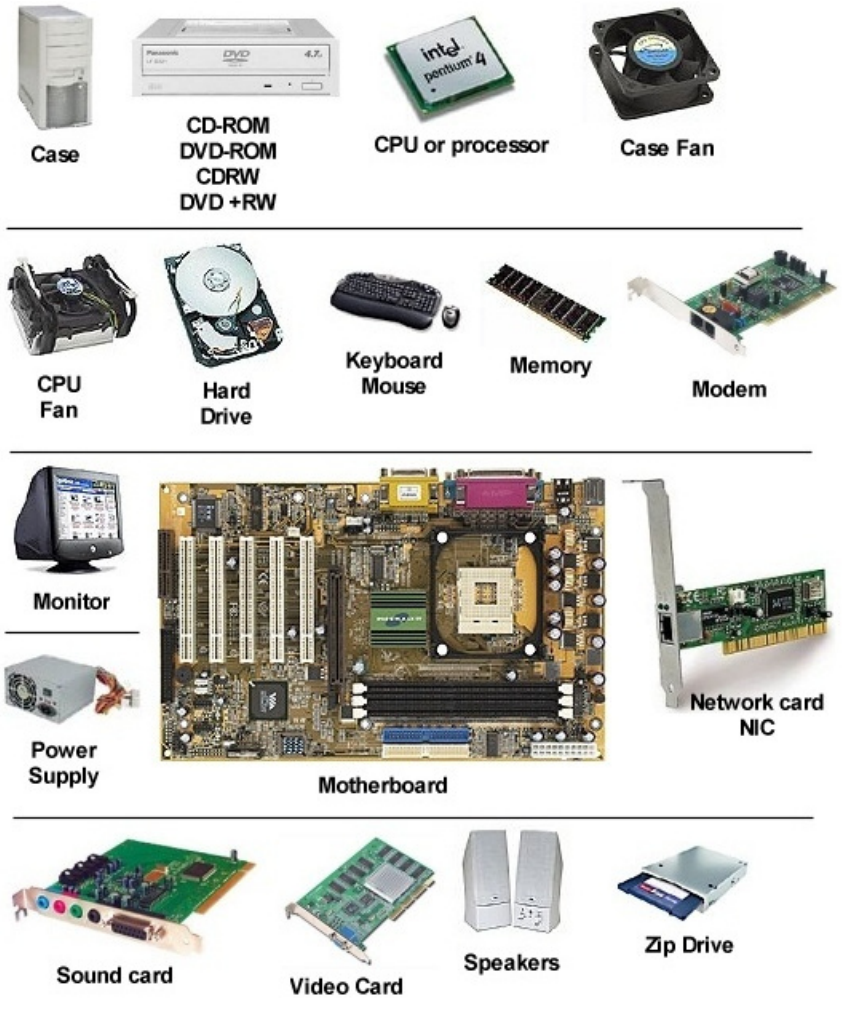
Il software è l'insieme delle parti immateriali a livello logico di un calcolatore (ad esempio un programma).

Hardware (1)

I componenti dell'hardware sono generalmente racchiusi dentro ad un *case* e sono, ad esempio:

- scheda madre;
- CPU;
- alimentatore elettrico;
- memoria primaria (RAM);
- memoria di massa;
- scheda di rete;
- scheda video;
- scheda audio.

Hardware (2)


Case
CD-ROM
DVD-ROM
CDRW
DVD +RW
CPU or processor
Case Fan
CPU
Fan
Hard
Drive
Keyboard
Mouse
Memory
Modem
Monitor
Power
Supply
Motherboard
Network card
NIC
Sound card
Video Card
Speakers
Zip Drive

Dati

Un calcolatore riceve una serie di dati (sequenze di numeri e lettere) in ingresso, esegue delle operazioni su di essi e restituisce altri dati in uscita.

I dati in ingresso sono chiamati in generale input.

I dati in uscita sono chiamati invece output.

Bit

I dati che un calcolatore elettronico può trattare sono scritti nelle sue memorie (primaria e di massa) sotto forma di bit.

La parola *bit* nasce dall'unione di ***bi****nary* e *digi****t***, cioè cifra binaria.

Una cifra binaria può valere 0 oppure 1, mentre una cifra decimale può valere 0, 1, 2, 3, 4, 5, 6, 7, 8 oppure 9.

A livello fisico, un bit può corrispondere a diversi stati: un interruttore alzato o abbassato, una corrente che può passare o meno, un campo magnetico orientato in un verso o nell'altro, etc.

▸ Bit con tavolette di legno

Il sistema di numerazione decimale

Il sistema di numerazione che utilizziamo abitualmente è un sistema decimale: ogni numero può essere scritto combinando le cifre da 0 a 9.

La posizione di ogni cifra ne determina il valore.

Sappiamo bene che 27 è ben diverso da 72: utilizzando le stesse cifre in ordine diverso otteniamo numeri differenti.

Valore di una cifra e posizione

Quando scriviamo il numero 5812 intendiamo dire:

5 migliaia, più 8 centinaia, più 1 decina, più 2 unità

ovvero, in notazione polinomiale:

$$(5 \cdot 10^3) + (8 \cdot 10^2) + (1 \cdot 10^1) + (2 \cdot 10^0)$$

Vediamo come il valore di una cifra sia determinato da una potenza di 10 (perché 10 sono le cifre che usiamo).

Attenzione: si parte da $10^0 = 1$.

Il sistema di numerazione binario (1)

Il sistema binario utilizza lo stesso procedimento, usando però due sole cifre (0 e 1) e di conseguenza le potenze di 2.

$$2^0 = 1 \qquad 2^1 = 2 \qquad 2^2 = 4 \qquad 2^3 = 8$$

$$2^4 = 16 \qquad 2^5 = 32 \qquad 2^6 = 64 \qquad 2^7 = 128$$

$$2^8 = 256 \qquad 2^9 = 512 \qquad 2^{10} = 1024$$

▸ Un gioco interessante

Il sistema di numerazione binario (2)

Esempi di numeri binari sono:

101 11101 10001000

Per convertirli in numerazione decimale basta lavorare come abbiamo fatto in precedenza per la numerazione decimale, moltiplicando ogni cifra per la potenza di 2 corrispondente alla posizione occupata dalla cifra.

Conversione da binario a decimale (1)

110

Questo numero sarà convertito in numerazione decimale utilizzando la scrittura polinomiale:

$$(1 \cdot 2^2) + (1 \cdot 2^1) + (0 \cdot 2^0) = 4 + 2 + 0 = 6$$

Basterà sommare le potenze di 2 associate alla presenza della cifra 1 (perché $0 \cdot 2^n$ fa sempre 0).

Conversione da binario a decimale (2)

Convertiamo:

$$111001$$

Cominciamo da destra per iniziare comodamente con 2^0:

$$(1 \cdot 2^0) + (1 \cdot 2^3) + (1 \cdot 2^4) + (1 \cdot 2^5) = 1 + 8 + 16 + 32 = 57$$

Prova a convertire:

$$11111 \qquad 1000 \qquad 10001$$

...

Al mondo ci sono 10 tipi di persone – quelle che capiscono la numerazione binaria e quelle che non la capiscono.

Al mondo ci sono 10 tipi di persone – quelle che capiscono la numerazione ternaria, quelle che non la capiscono e quelle che la confondono con la numerazione binaria.

Conversione da decimale a binario (1)

Per convertire un numero da decimale a binario si procede come segue:

- si divide il numero per 2 (con resto 0 se è pari o 1 se è dispari);
- si divide il quoziente ottenuto ancora per 2 (con resto 0 oppure 1);
- si prosegue fino a ottenere quoziente zero;
- si scrivono i resti, iniziando da sinistra con l'ultimo resto ottenuto.

Conversione da decimale a binario (2)

Ad esempio, per convertire il numero 159:

$159 : 2 = 79$, resto 1

$79 : 2 = 39$, resto 1

$39 : 2 = 19$, resto 1

$19 : 2 = 9$, resto 1

$9 : 2 = 4$, resto 1

$4 : 2 = 2$, resto 0

$2 : 2 = 1$, resto 0

$1 : 2 = 0$, resto 1

Il numero 159, convertito in numerazione binaria, sarà quindi:

10011111

Byte

Un *byte* è una sequenza di 8 bit, cioè un numero binario costituito da 8 cifre.

I valori minimo e massimo che un byte può assumere sono:

$$(00000000)_2 = (0)_{10} \qquad (11111111)_2 = (255)_{10}$$

Un byte può pertanto assumere $2^8 = 256$ valori possibili.

Il byte è l'unità di misura fondamentale della quantità di dati in informatica e i suoi multipli sono il kB, il MB, GB e TB (ognuno costituito da 1000 unità precedenti).

Numerazione esadecimale (1)

In informatica usiamo anche la numerazione esadecimale, che utilizza sedici simboli:

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$$

Le sei lettere che compaiono stanno ad indicare i numeri dopo il 9, cioè:

$$A = 10, B = 11, C = 12, D = 13, E = 14, F = 15$$

Numerazione esadecimale (2)

In numerazione esadecimale, con due simboli possiamo esprimere ben $16^2 = 256$ numeri, con tre simboli ne possiamo esprimere $16^3 = 4096$!

Convertiamo ad esempio $2DF$:

$$(2 \cdot 16^2) + (13 \cdot 16^1) + (15 \cdot 16^0) =$$

$$= 512 + 208 + 15 = 735$$

Calcolo binario

Con in numeri scritti in numerazione binaria è possibile eseguire ovviamente le solite operazioni:

- addizione;
- sottrazione;
- moltiplicazione;
- divisione.

Addizione e sottrazione

La somma segue lo stesso principio della somma decimale: si sommano le cifre corrispondenti dei due addendi, ricordando di "riportare" un 1 quando una somma tra cifre fa 2 o più.

Calcoliamo $1010 + 1110$:

riporto	1	1	1		
I addendo		1	0	1	0 +
II addendo		1	1	1	0 =
risultato	1	1	0	0	0

In modo analogo possiamo eseguire la differenza, ricordando che invece del "riporto" dovremo usare dei "prestiti".

Moltiplicazione

Anche la moltiplicazione segue le solite regole, ricordando ancora che 2 unità di un certo ordine equivalgono ad 1 unità dell'ordine superiore (così come 10 centinaia fanno 1 migliaio).

Eseguiamo 110×101:

I fattore			1	1	0	$\times$
II fattore			1	0	1	$=$
I prodotto parziale			1	1	0	
II prodotto parziale		0	0	0	-	
III prodotto parziale	1	1	0	-	-	
risultato	1	1	1	1	0	

Le operazioni logiche

Con i numeri binari possiamo anche eseguire operazioni particolari dette operazioni logiche o booleane (dal nome del logico del XIX secolo George Boole).

Le operazioni logiche più comuni sono:

- AND (congiunzione), indicata anche con & oppure $\wedge$;
- OR (disgiunzione), indicata anche con $\vee$;
- NOT (negazione).

Se consideriamo la cifra 0 come valore di verità "falso" e la cifra 1 come "vero", possiamo comprendere meglio il senso delle operazioni logiche.

NOT

Corrisponde alla negazione e tramuta uno 0 in un 1 e viceversa.

X	NOT(X)
0	1
1	0

È un operatore unario perché riceve in ingresso un solo numero (bit).

Esempio

X	1	0	1	1
	↓	↓	↓	↓
NOT(X)	0	1	0	0

AND

Corrisponde alla congiunzione e restituisce un 1 solo se entrambi i bit in ingresso valgono 1.

X	Y	X AND Y
1	1	1
1	0	0
0	1	0
0	0	0

È un operatore binario perché riceve in ingresso due bit.

Esempio

X	1	0	0	1
Y	1	1	0	1
	↓	↓	↓	↓
X AND Y	1	0	0	1

OR

Corrisponde alla disgiunzione e restituisce un 1 se almeno uno dei due bit in ingresso vale 1.

X	Y	X OR Y
1	1	1
1	0	1
0	1	1
0	0	0

È un operatore binario perché riceve in ingresso due bit.

Esempio

X	1	0	0	1
Y	1	1	0	1
	↓	↓	↓	↓
X OR Y	1	1	0	1

Codificare un'informazione (1)

Codificare un'informazione significa tradurre un certo dato in uno specifico codice, nel nostro caso codice binario.

Per esempio, se vogliamo codificare $16 = 2^4$ simboli (15 lettere e uno spazio vuoto) ci serviranno 4 bit:

A	0000
B	0001
C	0010
D	0011

E	0100
F	0101
G	0110
H	0111

I	1000
L	1001
M	1010
N	1011

O	1100
P	1101
Q	1110
␣	1111

Codificare un'informazione (2)

Prova, con la codifica precedente, a decodificare questo messaggio:

000010011000111100111000111111011100100110011100

Ci rendiamo subito conto che è possibile decodificare un messaggio solo se ogni simbolo è rappresentato dallo stesso numero di bit (non sapremmo infatti dire altrimenti dove finisce e dove inizia un nuovo simbolo).

Lo "spazio" occupato dipende ovviamente dalla lunghezza del messaggio e dal tipo di codifica.

Esempio

Prova, con la codifica precedente, a decodificare questo messaggio:

La tabella ASCII (1)

Il codice ASCII fu inventato molti anni fa per le comunicazioni fra telescriventi, per poi diventare uno standard mondiale.

ASCII sta per "American Standard Code for Information Interchange", ovvero "Standard americano per lo scambio di informazioni".

Mediante questa codifica era possibile far riferimento ad un carattere di testo mediante un numero binario a 7 bit, fino ad un massimo di $2^7 = 128$ caratteri.

Ad esempio il carattere "@" è rappresentato dal codice ASCII "64", "Y" dall'"89", ecc.

La tabella ASCII (2)

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29	)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,
010 1101	055	45	2D	-
010 1110	056	46	2E	.
010 1111	057	47	2F	/
011 0000	060	48	30	0
011 0001	061	49	31	1
011 0010	062	50	32	2
011 0011	063	51	33	3
011 0100	064	52	34	4
011 0101	065	53	35	5
011 0110	066	54	36	6
011 0111	067	55	37	7
011 1000	070	56	38	8
011 1001	071	57	39	9
011 1010	072	58	3A	:
011 1011	073	59	3B	;
011 1100	074	60	3C	<
011 1101	075	61	3D	=
011 1110	076	62	3E	>
011 1111	077	63	3F	?

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L
100 1101	115	77	4D	M
100 1110	116	78	4E	N
100 1111	117	79	4F	O
101 0000	120	80	50	P
101 0001	121	81	51	Q
101 0010	122	82	52	R
101 0011	123	83	53	S
101 0100	124	84	54	T
101 0101	125	85	55	U
101 0110	126	86	56	V
101 0111	127	87	57	W
101 1000	130	88	58	X
101 1001	131	89	59	Y
101 1010	132	90	5A	Z
101 1011	133	91	5B	[
101 1100	134	92	5C	\
101 1101	135	93	5D	]
101 1110	136	94	5E	^
101 1111	137	95	5F	_

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l
110 1101	155	109	6D	m
110 1110	156	110	6E	n
110 1111	157	111	6F	o
111 0000	160	112	70	p
111 0001	161	113	71	q
111 0010	162	114	72	r
111 0011	163	115	73	s
111 0100	164	116	74	t
111 0101	165	117	75	u
111 0110	166	118	76	v
111 0111	167	119	77	w
111 1000	170	120	78	x
111 1001	171	121	79	y
111 1010	172	122	7A	z
111 1011	173	123	7B	{
111 1100	174	124	7C	|
111 1101	175	125	7D	}
111 1110	176	126	7E	~

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29	)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,
010 1101	055	45	2D	-
010 1110	056	46	2E	.
010 1111	057	47	2F	/
011 0000	060	48	30	0
011 0001	061	49	31	1
011 0010	062	50	32	2
011 0011	063	51	33	3
011 0100	064	52	34	4
011 0101	065	53	35	5
011 0110	066	54	36	6
011 0111	067	55	37	7
011 1000	070	56	38	8
011 1001	071	57	39	9
011 1010	072	58	3A	:
011 1011	073	59	3B	;
011 1100	074	60	3C	<
011 1101	075	61	3D	=
011 1110	076	62	3E	>
011 1111	077	63	3F	?

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L
100 1101	115	77	4D	M
100 1110	116	78	4E	N
100 1111	117	79	4F	O
101 0000	120	80	50	P
101 0001	121	81	51	Q
101 0010	122	82	52	R
101 0011	123	83	53	S
101 0100	124	84	54	T
101 0101	125	85	55	U
101 0110	126	86	56	V
101 0111	127	87	57	W
101 1000	130	88	58	X
101 1001	131	89	59	Y
101 1010	132	90	5A	Z
101 1011	133	91	5B	[
101 1100	134	92	5C	\
101 1101	135	93	5D	]
101 1110	136	94	5E	^
101 1111	137	95	5F	_

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l
110 1101	155	109	6D	m
110 1110	156	110	6E	n
110 1111	157	111	6F	o
111 0000	160	112	70	p
111 0001	161	113	71	q
111 0010	162	114	72	r
111 0011	163	115	73	s
111 0100	164	116	74	t
111 0101	165	117	75	u
111 0110	166	118	76	v
111 0111	167	119	77	w
111 1000	170	120	78	x
111 1001	171	121	79	y
111 1010	172	122	7A	z
111 1011	173	123	7B	{
111 1100	174	124	7C	|
111 1101	175	125	7D	}
111 1110	176	126	7E	~

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29	)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,
010 1101	055	45	2D	-
010 1110	056	46	2E	.
010 1111	057	47	2F	/
011 0000	060	48	30	0
011 0001	061	49	31	1
011 0010	062	50	32	2
011 0011	063	51	33	3
011 0100	064	52	34	4
011 0101	065	53	35	5
011 0110	066	54	36	6
011 0111	067	55	37	7
011 1000	070	56	38	8
011 1001	071	57	39	9
011 1010	072	58	3A	:
011 1011	073	59	3B	;
011 1100	074	60	3C	<
011 1101	075	61	3D	=
011 1110	076	62	3E	>
011 1111	077	63	3F	?

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L
100 1101	115	77	4D	M
100 1110	116	78	4E	N
100 1111	117	79	4F	O
101 0000	120	80	50	P
101 0001	121	81	51	Q
101 0010	122	82	52	R
101 0011	123	83	53	S
101 0100	124	84	54	T
101 0101	125	85	55	U
101 0110	126	86	56	V
101 0111	127	87	57	W
101 1000	130	88	58	X
101 1001	131	89	59	Y
101 1010	132	90	5A	Z
101 1011	133	91	5B	[
101 1100	134	92	5C	\
101 1101	135	93	5D	]
101 1110	136	94	5E	^
101 1111	137	95	5F	_

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l
110 1101	155	109	6D	m
110 1110	156	110	6E	n
110 1111	157	111	6F	o
111 0000	160	112	70	p
111 0001	161	113	71	q
111 0010	162	114	72	r
111 0011	163	115	73	s
111 0100	164	116	74	t
111 0101	165	117	75	u
111 0110	166	118	76	v
111 0111	167	119	77	w
111 1000	170	120	78	x
111 1001	171	121	79	y
111 1010	172	122	7A	z
111 1011	173	123	7B	{
111 1100	174	124	7C	|
111 1101	175	125	7D	}
111 1110	176	126	7E	~

Altre codifiche

La tabella ASCII è stata poi estesa in vari modi utilizzando l'ottavo bit disponibile di un byte.

Lo standard per la codifica del testo attualmente utilizzato è UNICODE, compatibile con ASCII.

Ad oggi UNICODE conta circa 150.000 caratteri codificati, comprendendo anche lingue antiche, simboli ed emoji.

La codifica delle immagini

Un'immagine su uno schermo è costituita da un insieme di pixel affiancati, ognuno con uno specifico colore.

Per digitalizzarla (trasformarla in una sequenza di 0 e 1) possiamo memorizzare il colore di ogni singolo pixel.

Questo metodo tuttavia non è molto efficiente: molti dati si ripetono uguali.

La *palette di colori*

HTML name	R G B Hex	Decimal
Colori rosa		
Pink	FF C0 CB	255 192 203
LightPink	FF B6 C1	255 182 193
HotPink	FF 69 B4	255 105 180
DeepPink	FF 14 93	255 20 147
PaleVioletRed	DB 70 93	219 112 147
MediumVioletRed	C7 15 85	199 21 133
Colori rossi		
LightSalmon	FF A0 7A	255 160 122
Salmon	FA 80 72	250 128 114
DarkSalmon	E9 96 7A	233 150 122
LightCoral	F0 80 80	240 128 128
IndianRed	CD 5C 5C	205 92 92
Crimson	DC 14 3C	220 20 60
FireBrick	B2 22 22	178 34 34
DarkRed	8B 00 00	139 0 0
Red	FF 00 00	255 0 0
Colori arancioni		
OrangeRed	FF 45 00	255 69 0
Tomato	FF 63 47	255 99 71
Coral	FF 7F 50	255 127 80
DarkOrange	FF 8C 00	255 140 0
Orange	FF A5 00	255 165 0

Palette di colori X11

Un'altra opzione è creare una tavolozza di colori (*palette*), per poi dare il colore di ogni singolo pixel in riferimento alla *palette*.

Questo metodo è utile per immagini di grandi dimensioni, in cui lo spazio utilizzato per memorizzare la *palette* è ampiamente ripagato dal risparmio di dati sull'intera immagine.

Codice alfanumerico

Un codice alfanumerico è una stringa costituita soltanto da numeri e lettere (maiuscole o minuscole).

Se per il codice ASCII servono 8 bit, cioè 1 byte, per un codice alfanumerico ne bastano 6.

Infatti, considerando le lettere maiuscole e minuscole $(26 + 26)$ e i numeri (10), si ottiene un numero totale di simboli minore di $2^6 = 64$.