

Informatica per le discipline umanistiche

Lezione 9 – File di testo

`cristiano.longo@unict.it`



File

Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

File

Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

L'*interpretazione* di questi Byte è definita da un **media type** associato al file.

Esempi di media type: text/plain, audio/wav, image/jpeg, video/mp4

Il media type indica il *formato* dei dati nel file. È definito da chi genera il file e deve essere noto a chi vuole utilizzare il file

L'elenco completo dei media type è disponibile al seguente indirizzo

<https://www.iana.org/assignments/media-types/media-types.xhtml>

File di testo

Il media type **text/plain** indica che il contenuto del file è una sequenza di caratteri.

Il media type text/plain ha un parametro aggiuntivo, che indica come associare i Byte ai corrispondenti caratteri: **charset** . (codifica)

L'elenco dei charset è disponibile all'indirizzo

<https://www.iana.org/assignments/character-sets/character-sets.xhtml>

Vedremo due differenti charset: US-ASCII e UTF-8.

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
P	A	L	L	A

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
P	A	L	L	A

b ₇ b ₆ b ₅ → Bits ↓					0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1 Column → Row ↓							
b ₄	b ₃	b ₂	b ₁	Row	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(8	H	X	h	x
1	0	0	1	9	HT	EM)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[k	}
1	1	0	0	12	FF	FS	,	<	L	\	l	
1	1	0	1	13	CR	GS	-	=	M]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
P	A	L	L	A

Bits					Column	0	1	2	3	4	5	6	7
b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	0	0	0	1	1	0	1
				Row	0	1	2	3	4	5	6	7	
0	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	8	BS	CAN	(8	H	X	h	x
1	0	0	1	9	9	HT	EM)	9	I	Y	i	y
1	0	1	0	10	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	11	VT	ESC	+	;	K	[k	}
1	1	0	0	12	12	FF	FS	,	<	L	\	l	~
1	1	0	1	13	13	CR	GS	-	=	M]	m	}
1	1	1	0	14	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	15	SI	US	/	?	O	_	o	DEL

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

0101 0000	01000001	01001100	01001100	01000001
P	A	L	L	A

Bits					Column		Row								
b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	0	0	1	0	0	SOH	DC1	!	1	A	Q	~	q
0	0	0	1	0	0	0	0	STX	DC2	"	2	B	R	b	r
0	0	0	1	0	1	0	0	ETX	DC3	#	3	C	S	c	s
0	0	1	0	0	0	0	0	EOT	DC4	\$	4	D	T	d	t
0	0	1	0	0	1	0	0	ENQ	NAK	%	5	E	U	e	u
0	0	1	0	1	0	0	0	ACK	SYN	&	6	F	V	f	v
0	0	1	1	0	0	0	0	BEL	ETB	'	7	G	W	g	w
0	1	0	0	0	0	0	0	BS	CAN	(8	H	X	h	x
0	1	0	0	0	1	0	0	HT	EM)	9	I	Y	i	y
0	1	0	1	0	0	0	0	LF	SUB	*	:	J	Z	j	z
0	1	0	1	0	1	0	0	VT	ESC	+	;	K	[k	}
0	1	1	0	0	0	0	0	FF	FS	,	<	L	\	l	~
0	1	1	0	0	1	0	0	CR	GS	-	=	M]	m	}
0	1	1	1	0	0	0	0	SO	RS	.	>	N	^	n	~
0	1	1	1	1	0	0	0	SI	US	/	?	O	_	o	DEL

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
P	A	L	L	A

The diagram shows a byte structure with bits b7 to b0. Below it is a grid of ASCII characters. A blue box highlights the character 'P' at the intersection of bit pattern 01010000 and character 'P'. A black box highlights the bit pattern 01010000.

Bits	b7	b6	b5	b4	b3	b2	b1	b0	Column	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0		NUL	DLE	SP	0	@	P	^	p
0	0	0	0	1	1	1	1	1		SOH	DC1	!	1	A	Q	u	q
0	0	1	0	0	0	0	0	0		STX	DC2	"	2	B	R	b	r
0	0	1	1	1	1	1	1	1		ETX	DC3	#	3	C	S	c	s
0	1	0	0	0	0	0	0	0		EOT	DC4	\$	4	D	T	d	t
0	1	0	1	0	0	0	0	0		ENQ	NAK	%	5	E	U	e	u
0	1	1	0	0	0	0	0	0		ACK	SYN	&	6	F	V	f	v
0	1	1	1	1	1	1	1	1		BEL	ETB	'	7	G	W	g	w
1	0	0	0	0	0	0	0	0		BS	CAN	(8	H	X	h	x
1	0	0	1	0	0	0	0	0		HT	EM)	9	I	Y	i	y
1	0	1	0	0	0	0	0	0		LF	SUB	*	:	J	Z	j	z
1	0	1	1	1	1	1	1	1		VT	ESC	+	;	K	[k	}
1	1	0	0	0	0	0	0	0		FF	FS	,	<	L	\	l	~
1	1	0	1	0	0	0	0	0		CR	GS	-	=	M]	m	}
1	1	1	0	0	0	0	0	0		SO	RS	.	>	N	^	n	~
1	1	1	1	1	1	1	1	1		SI	US	/	?	O	_	o	DEL

Charset US-ASCII

US-ASCII (RFC20) associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
P	A	L	L	A

Bits					Column									
b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	0	0	0	0	1	1	1	1
					0	0	1	1	0	0	1	1	0	0
					0	1	2	3	4	5	6	7		
0	0	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	0	0	0	STX	DC2	"	2	B	R	b	r
0	0	1	1	1	1	1	ETX	DC3	#	3	C	S	c	s
0	1	0	0	0	0	0	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	1	1	1	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	0	0	0	ACK	SYN	&	6	F	V	f	v
0	1	1	1	1	1	1	BEL	ETB	'	7	G	W	g	w
1	0	0	0	0	0	0	BS	CAN	(8	H	X	h	x
1	0	0	1	1	1	1	HT	EM)	9	I	Y	i	y
1	0	1	0	0	0	0	LF	SUB	*	:	J	Z	j	z
1	0	1	1	1	1	1	VT	ESC	+	;	K	[k	}
1	1	0	0	0	0	0	FF	FS	,	<	L	\	l	~
1	1	0	1	1	1	1	CR	GS	-	=	M]	m	}
1	1	1	0	0	0	0	SO	RS	.	>	N	^	n	~
1	1	1	1	1	1	1	SI	US	/	?	O	_	o	DEL

Caratteri di controllo (non vengono visualizzati).

Charset US-ASCII

US-ASCII associa ad ogni Byte un carattere. *(esadecimale)*

01010000	01000001	01001100	01001100	01000001
50	41	4C	4C	41
P	A	L	L	A

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Caratteri di controllo

- ...
- 0A Line Feed
- ...
- 0D Carriage Return
- ...

Charset US-ASCII

US-ASCII associa ad ogni Byte un carattere.

01010000	01000001	01001100	01001100	01000001
50	41	4C	4C	41
P	A	L	L	A

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Utilizza solo 7 bit (quello più a sinistra è sempre 0).

Ha 128 caratteri.

Charset US-ASCII – esempio 1

A	C	Q	U	A

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Charset US-ASCII – esempio 1

41	43	51	55	41
A	C	Q	U	A

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Charset US-ASCII – esempio 2

53	50	41	44	41

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Charset US-ASCII – esempio 2

53	50	41	44	41
S	P	A	D	A

Tabella di codifica

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	[Redacted]															
1:	[Redacted]															
2:	[Redacted]	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	[Redacted]

Unicode

Nel 1991 nasce Unicode (vedi <https://unicode.org>) con l'intento di fornire una codifica sufficiente per tutte le lingue del mondo e una grande varietà di simboli.

Esempi di caratteri trattati da Unicode (raggruppati per tipo di scrittura)

Latin: æ, à, £, ů, ĺ

Cyrillic: Ѡ, Ѳ, ж

CJK: 丁, 丕, 弗, 𠂇, 乖


Arrows: ←, ⇔, ⇌, ↷, ↶

Emoticons: 😐, 😱, 🐵, 😎

Unicode

Nel 1991 nasce Unicode (vedi <https://unicode.org>) con l'intento di fornire una codifica sufficiente per tutte le lingue del mondo e una grande varietà di simboli.

Unicode assegna ad ogni carattere un **nome** e un **code point**, che si rappresenta con "U+" seguito da un numero in notazione esadecimale.

	Nome	Code Point
Σ	LATIN CAPITAL LETTER ESH	U+01A9
A	LATIN CAPITAL LETTER A	U+0041
Æ	LATIN CAPITAL LETTER AE	U+00C6
	GRINNING FACE	U+1F600

L'elenco dei caratteri Unicode è disponibile a
<https://www.unicode.org/charts/>

Codifica UTF-8

UTF-8 è una codifica per convertire i code point di Unicode in Byte e viceversa.

È una codifica a lunghezza variabile: i caratteri codificati possono occupare uno, due, tre o quattro Byte. **Esaminiamo solo i casi con uno o due Byte.**

Se un Byte inizia per 0 allora il code point si ottiene dai successivi 7 bit. In questo caso la codifica coincide con US-ASCII.

01000001

Codifica UTF-8

UTF-8 è una codifica per convertire i code point di Unicode in Byte e viceversa.

È una codifica a lunghezza variabile: i caratteri codificati possono occupare uno, due, tre o quattro Byte. **Esaminiamo solo i casi con uno o due Byte.**

Se un Byte inizia per 0 allora il code point si ottiene dai successivi 7 bit. In questo caso la codifica coincide con US-ASCII.

0100 0001 → U+0041 “(“

Codifica UTF-8

UTF-8 è una codifica per convertire i code point di Unicode in Byte e viceversa.

È una codifica a lunghezza variabile: i caratteri codificati possono occupare uno, due, tre o quattro Byte. **Esaminiamo solo i casi con uno o due Byte.**

Se un Byte inizia per 0 allora il code point si ottiene dai successivi 7 bit. In questo caso la codifica coincide con US-ASCII.

01000001 → U+0041 “(“

Se un Byte inizia per **110** allora si deve prendere in considerazione il successivo, che deve iniziare per **10**.

11000110 10101001

Codifica UTF-8

UTF-8 è una codifica per convertire i code point di Unicode in Byte e viceversa.

È una codifica a lunghezza variabile: i caratteri codificati possono occupare uno, due, tre o quattro Byte. **Esaminiamo solo i casi con uno o due Byte.**

Se un Byte inizia per 0 allora il code point si ottiene dai successivi 7 bit. In questo caso la codifica coincide con US-ASCII.

01000001 → U+0041 “(“

Se un Byte inizia per 110 allora si deve prendere in considerazione il successivo, che deve iniziare per 10. In questo caso il code point si ottiene concatenando gli ultimi 5 bit del primo Byte

1100**110** 10101001 → 00110

Codifica UTF-8

UTF-8 è una codifica per convertire i code point di Unicode in Byte e viceversa.

È una codifica a lunghezza variabile: i caratteri codificati possono occupare uno, due, tre o quattro Byte. **Esaminiamo solo i casi con uno o due Byte.**

Se un Byte inizia per 0 allora il code point si ottiene dai successivi 7 bit. In questo caso la codifica coincide con US-ASCII.

01000001 → U+0041 “(“

Se un Byte inizia per 110 allora si deve prendere in considerazione il successivo, che deve iniziare per 10. In questo caso il code point si ottiene concatenando gli ultimi 5 bit del primo Byte e gli ultimi 6 del secondo.

11000110 10**101001** → 001 10**101001** → U+01A9 “Σ”

Comma-Separated Values (CSV)

CSV (RFC4180) è un formato testuale identificato col media type **text/csv**. Il media type ha due parametri aggiuntivi

- **charset** che indica il set di caratteri utilizzato, e
- **header** che indica la presenza o meno di un *header* (vedremo dopo).

Si utilizza per dati tabellari.

CSV - definizione

Un file CSV è un file di testo con le seguenti caratteristiche

- 1) È costituito da un insieme di **record** (righe), separate dalla sequenza CRLF (accapo)
- 2) Ogni record, a sua volta, è costituito da un insieme di **campi** separati da “,”.

Esempio di file CSV

Cristiano,Longo,Catania
Mario,Rossi,Roma

Valori rappresentati nel file

Cristiano	Longo	Catania
Mario	Rossi	Roma

CSV - definizione

Un file CSV è un file di testo con le seguenti caratteristiche

- 1) È costituito da un insieme di **record** (righe), separate dalla sequenza CRLF (accapo)
- 2) Ogni record, a sua volta, è costituito da un insieme di **campi** separati da “,”.
- 3) Il primo record può essere l'**header** (informativo).

Esempio di file CSV

```
Nome,Cognome,Nascita  
Cristiano,Longo,Catania  
Mario,Rossi,Roma
```

Valori rappresentati nel file

Nome	Cognome	Nascita
Cristiano	Longo	Catania
Mario	Rossi	Roma

CSV - definizione

Un file CSV è un file di testo con le seguenti caratteristiche

- 1) È costituito da un insieme di **record** (righe), separate dalla sequenza CRLF (accapo)
- 2) Ogni record, a sua volta, è costituito da un insieme di **campi** separati da “,”.
- 3) Il primo record può essere l'**header** (informativo).
- 4) Ogni campo può essere racchiuso tra doppi apici (nel caso in cui contenga , o CRLF).

Esempio di file CSV

```
Nome,Cognome,Nascita  
Cristiano,Longo,"Catania, Italia"  
Mario,Rossi,"Roma, Italia"
```

Valori rappresentati nel file

Nome	Cognome	Nascita
Cristiano	Longo	Catania, Italia
Mario	Rossi	Roma, Italia

CSV - definizione

Un file CSV è un file di testo con le seguenti caratteristiche

- 1) È costituito da un insieme di **record** (righe), separate dalla sequenza CRLF (accapo)
- 2) Ogni record, a sua volta, è costituito da un insieme di **campi** separati da “,”.
- 3) Il primo record può essere l'**header** (informativo).
- 4) Ogni campo può essere racchiuso tra doppi apici (nel caso in cui contenga , o CRLF).
- 5) Per indicare un doppio apice nel contenuto di un campo si usa una coppia di doppi apici “”.

Esempio di file CSV

```
Nome,Cognome,Nascita  
Cristiano,Longo,"Catania, Italia"  
Mario,Rossi,"Roma, Italia"  
Antonio ""Nino"", Sardo, "Como,Italia"
```

Valori rappresentati nel file

Nome	Cognome	Nascita
Cristiano	Longo	Catania, Italia
Mario	Rossi	Roma, Italia
Antonio "Nino"	Sardo	Como, Italia

CSV e dove trovarli

Dati di tipo aperto, i dati che presentano le seguenti caratteristiche: 1) sono disponibili secondo i termini di una licenza che ne permetta l'utilizzo da parte di chiunque, anche per finalità commerciali, in formato disaggregato; 2) sono accessibili attraverso le tecnologie dell'informazione e della comunicazione, ivi comprese le reti telematiche pubbliche e private, in formati aperti ai sensi della lettera 3), sono adatti all'utilizzo automatico da parte di programmi per elaboratori e sono provvisti dei relativi metadati;

Da Art. 68, comma 3, del CAD, come modificato dalle norme contenute nell'art. 9 del DL n. 179/2012, convertito con la Legge n. 221/2012, recante "Ulteriori misure urgenti per la crescita del Paese"

Alcuni *cataloghi* di dati aperti:

- data.europa.eu
- dati.gov.it
- dati.istruzione.it
- www.comune.siracusa.it/area_tematica/portale-open-data-del-comune-di-siracusa

Alcuni strumenti per l'utilizzo di CSV

- Fogli di calcolo;
- datawrapper.de strumento di visualizzazione dei dati;
- Tabula.technology permette di estrarre le tabelle da un file PDF.

Linguaggi di Programmazione

Per realizzare Unix, fu necessario prima inventare un linguaggio di *alto livello*: il **linguaggio C**.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```

Linguaggi di Programmazione

Per realizzare Unix, fu necessario prima inventare un linguaggio di *alto livello*: il **linguaggio C**. Sviziati altri linguaggi di programmazione seguirono negli anni: BASIC, Fortran, Pascal, Python, Java, ...

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```

Compilatori

Un **compilatore** è un programma che traduce un programma in un linguaggio di alto livello, contenuto in un file detto **sorgente**, in un file eseguibile in linguaggio macchina.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```



helloworld

00000010

00000110

01000011

01001001

01000001

01001111

...

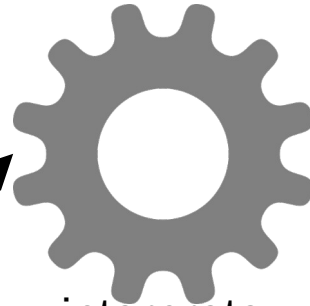
mov 0x024, AX
add AX, 3
mov AX, 0x024
...

Interpreti

Un **interprete** è un programma che **esegue** programmi scritti in un linguaggio di alto livello.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```



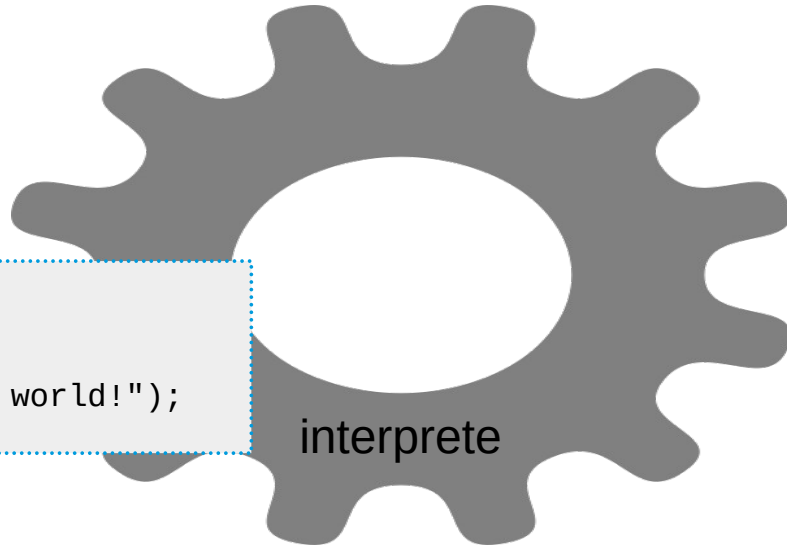
interprete

Interpreti

Un **interprete** è un programma che **esegue** programmi scritti in un linguaggio di alto livello.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```



interprete

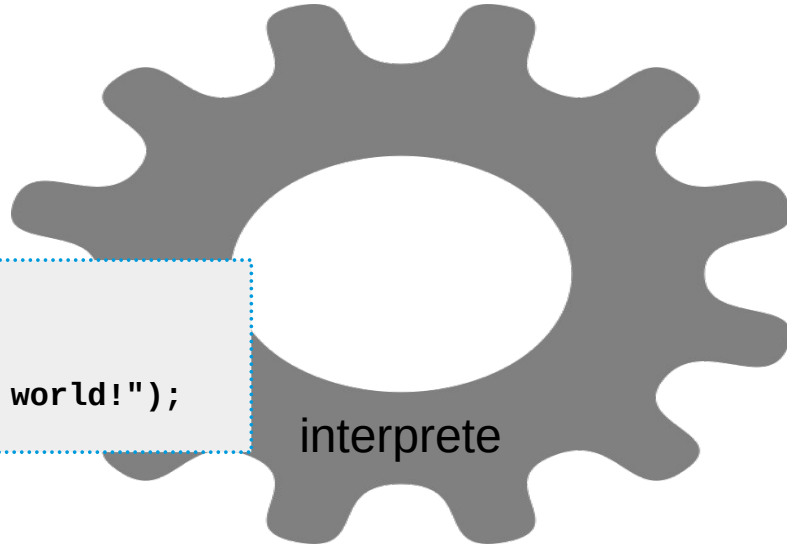
Output

Interpreti

Un **interprete** è un programma che **esegue** programmi scritti in un linguaggio di alto livello.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```

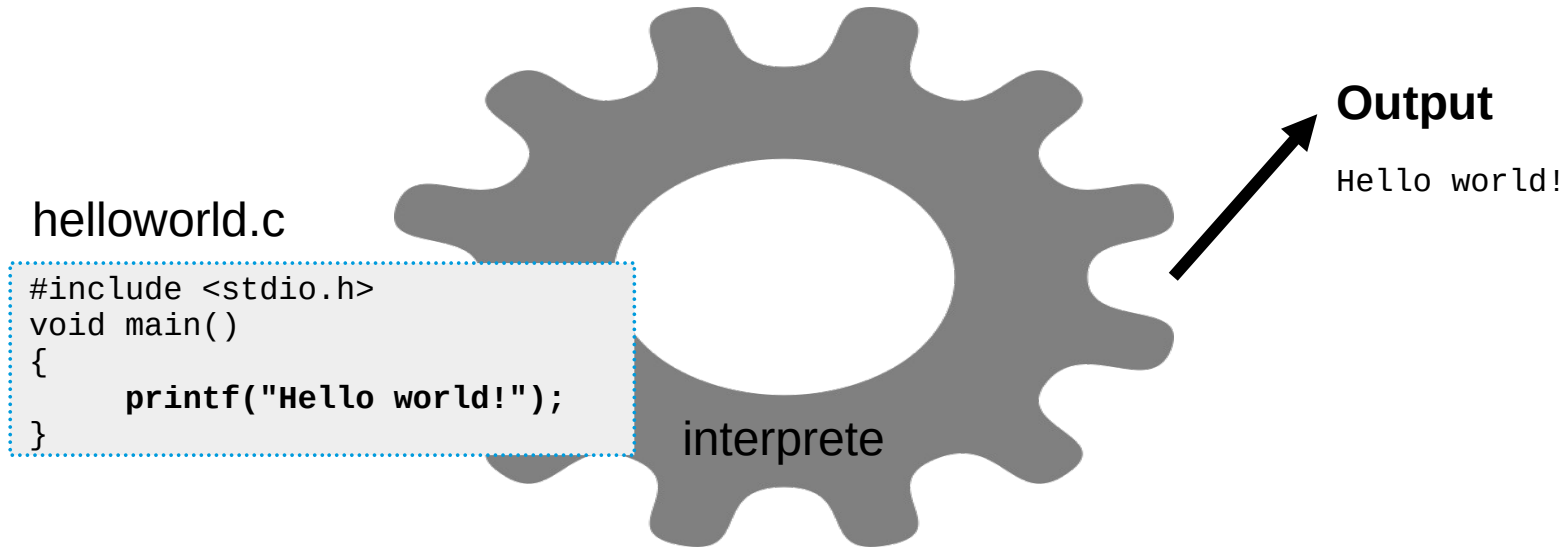


interprete

Output

Interpreti

Un **interprete** è un programma che **esegue** programmi scritti in un linguaggio di alto livello.



Riepilogo - file

Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

Riepilogo - file

Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

<pre>mov 0x024, AX add AX, 3 ...</pre>
--

I file che contengono istruzioni in linguaggio macchina (programmi) sono detti **eseguibili**.

Riepilogo - file

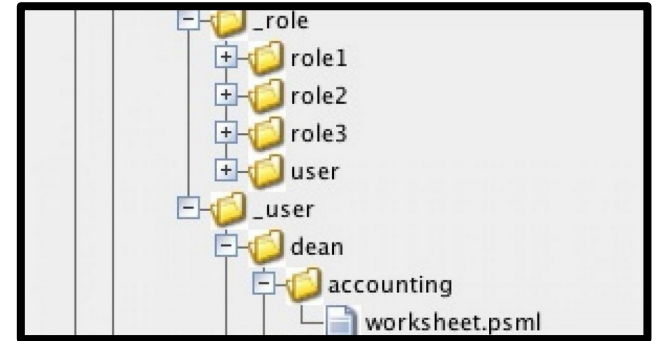
Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

```
mov 0x024, AX  
add AX, 3  
...
```

I file che contengono istruzioni in linguaggio macchina (programmi) sono detti **eseguibili** e rappresentano.

I file sono *organizzati* dentro le memorie dai **file system**.



Riepilogo - file

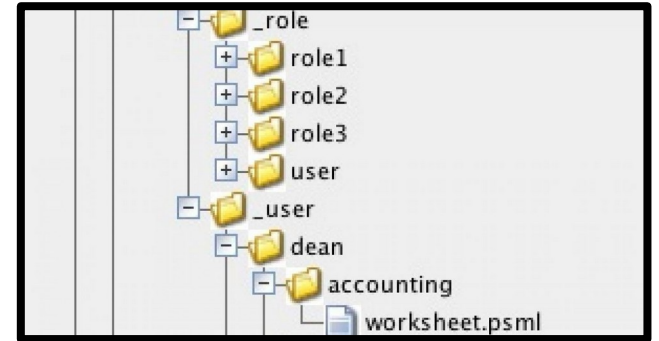
Un file è una sequenza finita di Byte.

01000011	01001001	01000001	01001111
----------	----------	----------	----------

```
mov 0x024, AX  
add AX, 3  
...
```

I file che contengono istruzioni in linguaggio macchina (programmi) sono detti **eseguibili** e rappresentano.

I file sono *organizzati* dentro le memorie dai **file system**.



Il **media type** associato ad un file indica il *formato* dei dati nel file.

Riepilogo – file di testo

Il media type **text/plain** indica che il contenuto del file è una sequenza di caratteri. La *codifica dei caratteri* viene specificata col parametro **charset**.

La codifica **US-ASCII** associa ad ogni Byte un carattere.

50	41	4C	4C	41
P	A	L	L	A

Tabella di codifica US-ASCII

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2:	█	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5:	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7:	p	q	r	s	t	u	v	w	x	y	z	{		}	~	█


Riepilogo – file di testo

Il media type **text/plain** indica che il contenuto del file è una sequenza di caratteri. La *codifica dei caratteri* viene specificata col parametro **charset**.

La codifica **US-ASCII** associa ad ogni Byte un carattere.

50	41	4C	4C	41
P	A	L	L	A

Unicode assegna ad ogni carattere un nome e un code point.

	Nome	Code Point
Σ	LATIN CAPITAL LETTER ESH	U+01A9
A	LATIN CAPITAL LETTER A	U+0041
Æ	LATIN CAPITAL LETTER AE	U+00C6
	GRINNING FACE	U+1F600

Riepilogo – file di testo

Il media type **text/plain** indica che il contenuto del file è una sequenza di caratteri. La *codifica dei caratteri* viene specificata col parametro **charset**.

La codifica **US-ASCII** associa ad ogni Byte un carattere.

50	41	4C	4C	41
P	A	L	L	A

Unicode assegna ad ogni carattere un nome e un code point.

UTF-8 è una codifica a *lunghezza variabile* per convertire i code point di Unicode in Byte e viceversa.

11000110 10101001 → U+01A9 “Σ”

Riepilogo – formati testuali

Il media type **text/csv** indica un formato testuale per rappresentare dati tabellari: I *campi* di ogni riga sono separati dal carattere “,”.

Esempio di file CSV

```
Nome,Cognome,Nascita  
Cristiano,Longo,"Catania, Italia"  
Mario,Rossi,"Roma, Italia"
```

Valori rappresentati nel file

Nome	Cognome	Nascita
Cristiano	Longo	Catania, Italia
Mario	Rossi	Roma, Italia

Riepilogo – formati testuali

Il media type **text/csv** indica un formato testuale per rappresentare dati tabellari: I *campi* di ogni riga sono separati dal carattere “,”.

Il **file sorgente** contengono programmi in un linguaggio di *alto livello*.

helloworld.c

```
#include <stdio.h>
void main()
{
    printf("Hello world!");
}
```

Riepilogo – formati testuali

Il media type **text/csv** indica un formato testuale per rappresentare dati tabellari: I *campi* di ogni riga sono separati dal carattere “,”.

Il **file sorgente** contengono programmi in un linguaggio di *alto livello*.

Il **compilatore** è un programma che traduce i file sorgente in istruzioni macchina.

