Università degli Studi
di Cassino

Corso di Fondamenti di Informatica

Il sistema dei tipi in C++

Anno Accademico 2008/2009

Francesco Tortorella

Struttura di un programma C++

```
// Programma semplice in C++
#include <iostream>

int main()
{
   cout << “Salve, mondo !\n”;
   return (0);
}
```

I tipi di dato in C++

In C++ sono disponibili vari tipi di dato.

- Tipi numerici:
  - **`int`**
  - **`float`**
  - **`double`**
- Tipi non numerici:
  - **`char`**
  - **`bool`**
  - **`void`**

Il tipo `int`

- È costituito da un sottoinsieme limitato dei numeri interi
- Caratteristiche:
  - Dimensione: 4 bytes
  - Valore minimo: -2147483648
  - Valore massimo: +2147483647
- Operazioni ammesse:
  - Assegnazione =
  - Somma +
  - Sottrazione -
  - Moltiplicazione *
  - Divisione /
  - Confronto > < >= <= == !=

Il tipo `float`

- È costituito da un sottoinsieme limitato e discreto dei numeri reali
- Caratteristiche:
  - Dimensione: 4 bytes
  - Valore minimo (abs): 3.4E- 38
  - Valore massimo (abs): 3.4E+38
- Operazioni ammesse:
  - Assegnazione =
  - Somma +
  - Sottrazione -
  - Moltiplicazione *
  - Divisione /
  - Confronto > < >= <= == !=

Il tipo `double`

- È costituito da un sottoinsieme limitato e discreto dei numeri reali, ma con range e precisione maggiore rispetto a `float` (doppia precisione)
- Caratteristiche:
  - Dimensione: 8 bytes
  - Valore minimo (abs): 1.7E- 308
  - Valore massimo (abs): 1.7E+308
- Operazioni ammesse:
  - Assegnazione =
  - Somma +
  - Sottrazione -
  - Moltiplicazione *
  - Divisione /
  - Confronto > < >= <= == !=

Il tipo char

- Consiste in un insieme di caratteri, alcuni stampabili (caratteri alfabetici, cifre, caratteri di punteggiatura, ecc.) ed altri non stampabili tramite i quali si gestisce il formato dell'input/output (caratteri di controllo).
- I sottoinsiemi delle lettere e delle cifre sono ordinati e coerenti.
- Per la rappresentazione interna, viene tipicamente usato il codice ASCII, che mette in corrispondenza ogni carattere con un numero intero compreso tra 0 e 255.

Il codice ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Il tipo char

- Di fatto anche `char` è un tipo numerico
- Caratteristiche:
  - Dimensione: 1 byte
  - Valore minimo: -128
  - Valore massimo: +127
- Sono permesse le operazioni aritmetiche tipiche degli interi

Il tipo `bool`

- È un tipo costituito da due soli valori **`false`** e **`true`**, corrispondenti a *falso* e *vero* e rappresentati da 0 e 1. Il tipo rappresenta le informazioni di tipo logico (es. il risultato di un confronto, il verificarsi di una situazione).
- Caratteristiche:
  - Dimensione: 1 byte
  - Valore minimo: `false`
  - Valore massimo: `true`
- Operazioni ammesse
  - assegnazione =
  - disgiunzione ||
  - congiunzione &&
  - negazione !

Operazioni sul tipo `bool`

		AND	OR
`x`	`y`	`x && y`	`x || y`
`false`	`false`	`false`	`false`
`false`	`true`	`false`	`true`
`true`	`false`	`false`	`true`
`true`	`true`	`true`	`true`

Operazioni sul tipo `bool`

NOT

`x`	`!x`
`false`	`true`
`true`	`false`

Modificatori di tipo

- Sono usati per creare nuovi tipi modificando i tipi base.
- **`signed`** e **`unsigned`**: senza ulteriori specificazioni, i tipi sono signed. Con il modificatore unsigned, il tipo è in grado di contenere soltanto valori non negativi
- **`unsigned char`**: 0…255
- **`unsigned int`**: 0…4294967295

Modificatori di tipo

- **`short`** e **`long`**: modificano l'estensione del tipo
  - **`short int`**: 2 byte
  - **`long int`**: 4 byte
  - **`long double`**: 12 byte
- I modificatori possono combinarsi:
  - **`unsigned short int`**
  - **`unsigned long int`**

Definizione di variabili

- Per usare una variabile, questa deve essere dapprima *definita*.
- La definizione rende disponibile la variabile che mantiene il tipo assegnato nella definizione fino al termine del programma.
- La sintassi è `<tipo> nome_variabile;`
- Esempi:
  - `int a;`
  - `int a,b,c;`
  - `float x,y,z;`

Definizione di variabili

- Il nome della variabile non può coincidere con una delle parole chiave riservate del C++:
```
asm, auto, bool, break, case, catch, char, class,
const, const_cast, continue, default, delete, do,
double, dynamic_cast, else, enum, explicit,
export, extern, false, float, for, friend, goto,
if, inline, int, long, mutable, namespace, new,
operator, private, protected, public, register,
reinterpret_cast, return, short, signed, sizeof,
static, static_cast, struct, switch, template,
this, throw, true, try, typedef, typeid, typename,
union, unsigned, using, virtual, void, volatile,
wchar_t, while
```
- I caratteri ammessi sono lettere, cifre e carattere di sottolineatura (underscore _), messi in qualunque ordine, purché il primo carattere del nome sia una lettera o l'underscore (sconsigliato).
- C'è differenza tra caratteri minuscoli e maiuscoli, per cui **a** e **A** sono due variabili diverse.
- Nello scegliere il nome per le variabili, è consigliabile orientarsi verso nomi significativi del ruolo della variabile nel programma.

Costanti

- Il C++ prevede tre modalità per definire delle costanti:
  - **Costanti letterali** (literals)
  - **Costanti definite** (`#define`)
  - **Costanti dichiarate** (`const`)

Costanti letterali

- **Il valore della costante è rappresentato direttamente.**
- **Costanti di tipo intero**

  Sono definite come sequenze di cifre decimali, eventualmente precedute da un segno (+ o -):
  0 -1 3256 +34 12L 33U 5321UL 0713 0X12FF 0XFUL
- **Costanti di tipo reale**

  Sono definite come sequenze di cifre decimali, eventualmente precedute da un segno (+ o -), strutturate in virgola fissa o in virgola mobile (floating point):
  0.1 -3.7 0.0001 1.0E-4 -7.6E12 4.
- **Costanti di tipo carattere**

  sono definite come caratteri racchiusi tra singoli apici (‘):
  ‘x’ ‘A’ ‘\n’ ‘\t’ ‘2’
- **Costanti di tipo stringa di caratteri**

  sono definite come sequenze di caratteri racchiusi tra doppi apici (“):
  “Pippo” “Valore di x: ” “x”
- **Costanti di tipo logico**

  sono solo due: `false` e `true`

Costanti definite

- Viene utilizzata la direttiva `#define` che permette di associare **testualmente** un valore ad un identificatore

```
#define MAX  12
#define PI 3.14
```

- All'atto della compilazione il *preprocessore* sostituisce ogni occorrenza dell'identificatore con il valore corrispondente

Costanti dichiarate

- Il valore viene associato ad un identificatore e ne viene specificato anche il tipo:

```
const int MIN=0;
const float PI=3.14;
```

- In questo caso l'identificatore è a tutti gli effetti una variabile non modificabile.

Operatori

- Un *operatore* specifica un'operazione da eseguire su uno o due *operandi* definendo un'*espressione*.
- L'ordine con cui sono valutati definisce la *precedenza* degli operatori. Può essere alterata con l'uso delle parentesi ().

Alcuni operatori	Assoc.
! **–** (*unario*) **+** (*unario*)	destra
***** (*moltiplicazione*) **/** **%**	sinistra
+ (*binario*) **–** (*binario*)	sinistra
< **<=** **>** **>=**	sinistra
== **!=**	sinistra
&&	sinistra
||	sinistra
=	destra

Espressioni

- Un’espressione consiste in un operando o in una combinazione di operandi e operatori.
- La valutazione di un’espressione porta al calcolo di un valore di un certo tipo.
- Esempi:

```
2+3          2.1*3.5+pigreco   a>=2
(7+2)/3      (a>2)&&(b==0)     !trovato
```

Espressioni

- Nel caso ci siano operandi di tipo diverso, l'espressione assume il tipo "più ampio" tra quelli presenti.
- Esempio:
  `7+5*2.4` → `19.0`
- **Achtung !** Qual è il valore dell'espressione?
  `7.5+1/2`
- Uso del casting: `7.5+(float)1/2`