



# Linguaggio C++

## Istruzioni decisionali



# Struttura di controllo di selezione

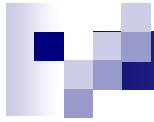
```
SE (cond)
  ALLORA
    INIZIO
      .....
    FINE
  [ ALTRIMENTI
    INIZIO
      .....
    FINE ]
FINE_SE
```

```
if (cond) {
  .....
} [ else {
  .....
}]
```



# Concetto di blocco

```
    {  
    .....  
    .....  
    .....  
}
```



# Variabili booleane

- In C non esiste il tipo boolean.
- In C:
  - falso: è lo zero
  - vero: è ciascun valore diverso da zero
- In C++ esiste il tipo **bool** (1 byte) i cui valori sono:
  - **false**
  - **true**



# Struttura di controllo di selezione

```
int num,esito;
```

```
...
```

```
    //scanf("%d",&num);
```

```
    cin>>num
```

```
    esito = num < 100;
```

```
    if(esito==1)
```

```
        //printf("\n minore di 100");
```

```
        cout<<"\n minore di 100";
```

```
int num,esito;
```

```
...
```

```
    //scanf("%d",&num);
```

```
    cin>>num;
```

```
    esito = num < 100;
```

```
    if(esito)
```

```
        //printf("\n minore di 100");
```

```
        cout<<"\n minore di 100";
```



# Struttura di controllo di selezione

```
int num;
```

```
...
```

```
    //scanf("%d",&num);
```

```
    cin>>num;
```

```
    if(num < 100)
```

```
        //printf("\n minore di 100");
```

```
        cout<<"\n minore di 100";
```




# Struttura di controllo di selezione

## Attenzione !!!

```
if (k = 5) {  
    .....  
}
```

```
if (k == 5) {  
    .....  
}
```

Warning: non esegui  
un confronto, ma  
un'assegnazione





# Struttura di controllo di selezione

## Attenzione !!!

```
if (5) {    //vero  
.....    //sempre eseguito  
}
```

```
if (temperatura < 10);  
    istruz1;
```

Attenzione al punto e virgola: istruzione nulla (non dà errore).





# Struttura di controllo di selezione

```
if (x > 0) {  
    .....  
}  
if (x == 0) {  
    .....  
}  
if (x < 0) {  
    .....  
}
```

Brutto!!!

```
if (x > 0) {  
    .....  
} else if (x == 0) {  
    .....  
} else {  
    .....  
}
```

La condizione (x < 0) è inutile.



# Struttura di controllo di selezione

```
if (a > 0)
    if (b > 0) {
        .....
    } else {
        .....
    }
```

- Istruzioni condizionali annidate (nidificate).
- Regola di “abbinamento della clausola else”
- “Abbina ogni else con l’ultima if non abbinata”.
- E se l’else lo voglio riferire al 1° if?



# Struttura di controllo di selezione

- Soluzione:

```
if (a > 0) {  
    if (b > 0) {  
        .....  
    }  
} else {  
    .....  
}
```

- Nello pseudocodice, la parola chiave FINE\_SE eliminava l'ambiguità.



# Espressioni aritmetiche

- Si definisce espressione aritmetica un insieme di variabili, costanti e richiami di funzioni (es. `abs()`, ...) connessi da operatori aritmetici.
- Il risultato di un'espressione aritmetica è sempre un valore numerico.
- Operatori aritmetici:
  - ☐ + addizione
  - ☐ - sottrazione
  - ☐ \* moltiplicazione
  - ☐ / divisione (fra interi e reali)
  - ☐ % modulo



# Operatore di assegnazione

- Operatore di assegnazione:  $=$
- Caratteristica del C: anche un'espressione ha un valore. Questo è il valore assegnato.
- Assegnazione multipla:  $altezza = base = 18;$
- Gerarchia degli operatori aritmetici e di assegnamento.



# Espressioni logiche: operatori logici

- Un'espressione logica genera come risultato un valore vero (per default 1, ma anche qualunque valore diverso da zero) o falso (0).
  - ! not
  - && and
  - || or
- Un semplice esempio di espressione logica è una variabile il cui contenuto può essere interpretato in due modi: vero, se diverso da zero, falso se uguale a zero.
- Le espressioni logiche possono contenere gli operatori relazionali.



# Operatori relazionali

- > maggiore
  - < minore
  - >= maggiore uguale
  - <= minore uguale
  - == uguale
  - != diverso
- 
- Esempio: `//printf("%d", a < b);` //stampa 1 se la condizione è vera, 0 altrimenti  
`cout<<(a < b);` //stampa 1 se la condizione è vera, 0 altrimenti



# Gerarchia di tutti gli operatori esaminati

In ordine di priorità dalla più alta alla più bassa.

- **! - (unario)**
- **\* / %**
- **+ -**
- **> >= < <=**
- **== !=**
- **&&**
- **||**
- **?:**
- **=**





# Condizioni o predicati

- In matematica:  $4 \leq \text{num} \leq 9$

- In informatica:

if ( (num >= 4) && (num <= 9) )

- Inoltre non si può scrivere: if (a == b == c)

ma: if( (a == b) && (b == c) )



# Espressioni condizionali: operatore ternario

- Operatore ternario    ? :

- Sintassi:

<var> = <cond> ? <valoreCondVera> : <valoreCondFalsa>

- Semantica:

SE (cond)

ALLORA

var ← valoreCondVera

ALTRIMENTI

var ← valoreCondFalsa

FINE\_SE



# Esempi: operatore ternario

- `bollo = (peso < 100) ? 0.50 : 0.80;`

- In modo equivalente:

```
if( peso < 100)
```

```
    bollo = 0.50;
```

```
else
```

```
    bollo = 0.80;
```



# Variabili carattere

```
char car = 'A';           //inizializzazione
char car2= 'a';           //inizializzazione
char car3 = ':';          //inizializzazione
char car4;                //definizione di una variabile
    car4 = '5';            //assegnazione
```

```
if (car == 'a')
```

```
    .....
```



# Variabili carattere

- Costanti carattere
  - `#define CAR 'Z'`
  - `const char CAR = 'Z';`
- Per stampare `%` → `%%`
- Per stampare `\` → `\\`
- Per stampare `"` → `\"`
- `'\n'` → carattere di new line
- N.B. `1` ≠ `'1'` ≠ `"1"` (numero intero 1, carattere 1, stringa formata dal carattere 1)



# Variabili carattere

- Caratteri → Codifica ASCII
- Ciascun carattere viene memorizzato con suo codice ASCII (American Standard Code for Information Interchange)
  - 0..127 caratteri standard
    - (0..31 → caratteri non stampabili, caratteri di controllo es. 26 → Ctrl-Z (fine file) nel “mondo” MS-DOS o sequenze di escape es. 10 → ‘\n’)
  - 128 ..255 caratteri non standard
- Relazione tra caratteri e numeri interi:

□ char	→ 1 byte	→	-128..+127
□ unsigned char	→ 1 byte	→	0..255

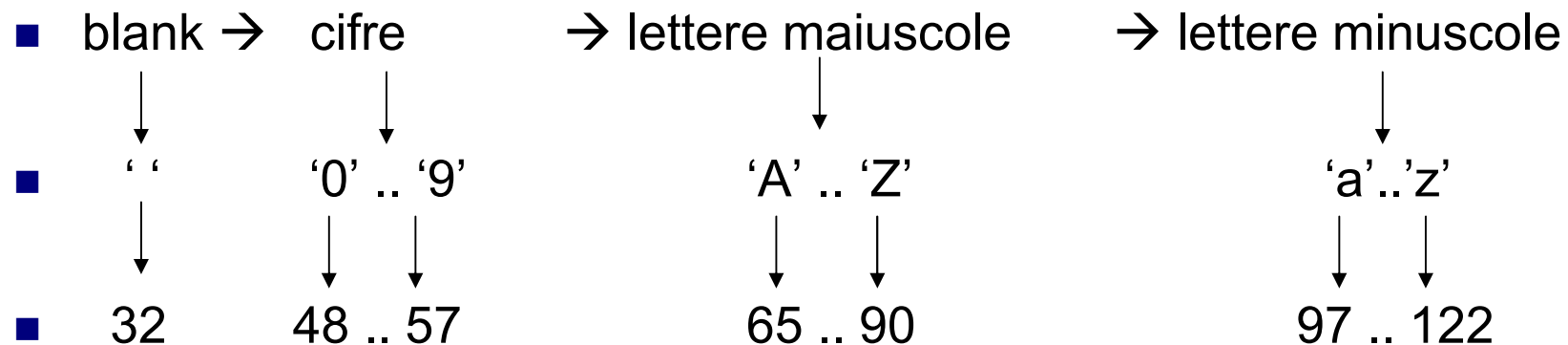
# ASCII - Acronimo di American Standard Code for Information Interchange

Sistema di codifica dei caratteri proposto nel 1961 dall'ingegnere dell'IBM Bob Bemer(1920, 2004)

ASCII control characters			ASCII printable characters			Extended ASCII characters										
00	NULL	(Null character)	32	space	64	@	96	`	128	Ç	160	á	192	Ł	224	Ó
01	SOH	(Start of Header)	33	!	65	A	97	a	129	ü	161	í	193	ł	225	ô
02	STX	(Start of Text)	34	"	66	B	98	b	130	é	162	ó	194	Ł	226	Ô
03	ETX	(End of Text)	35	#	67	C	99	c	131	â	163	ú	195	ł	227	Õ
04	EOT	(End of Trans.)	36	\$	68	D	100	d	132	ä	164	ñ	196	—	228	ö
05	ENQ	(Enquiry)	37	%	69	E	101	e	133	à	165	Ñ	197	†	229	Ö
06	ACK	(Acknowledgement)	38	&	70	F	102	f	134	â	166	ª	198	ä	230	µ
07	BEL	(Bell)	39	'	71	G	103	g	135	ç	167	º	199	Ä	231	þ
08	BS	(Backspace)	40	(	72	H	104	h	136	ê	168	¿	200	Ł	232	ƒ
09	HT	(Horizontal Tab)	41	)	73	I	105	i	137	ë	169	®	201	ł	233	ù
10	LF	(Line feed)	42	*	74	J	106	j	138	è	170	™	202	Ł	234	Û
11	VT	(Vertical Tab)	43	+	75	K	107	k	139	ï	171	½	203	ł	235	Ü
12	FF	(Form feed)	44	,	76	L	108	l	140	î	172	¼	204	Ł	236	Ý
13	CR	(Carriage return)	45	-	77	M	109	m	141	ì	173	î	205	=	237	Ÿ
14	SO	(Shift Out)	46	.	78	N	110	n	142	Ā	174	«	206	≠	238	—
15	SI	(Shift In)	47	/	79	O	111	o	143	Ă	175	»	207	≠	239	'
16	DLE	(Data link escape)	48	0	80	P	112	p	144	É	176	⋮	208	ð	240	≡
17	DC1	(Device control 1)	49	1	81	Q	113	q	145	æ	177	⋮	209	Ð	241	±
18	DC2	(Device control 2)	50	2	82	R	114	r	146	Æ	178	⋮	210	Ê	242	≡
19	DC3	(Device control 3)	51	3	83	S	115	s	147	ô	179	⋮	211	Ë	243	¾
20	DC4	(Device control 4)	52	4	84	T	116	t	148	ö	180	⋮	212	È	244	¶
21	NAK	(Negative acknowl.)	53	5	85	U	117	u	149	ò	181	À	213	Ì	245	§
22	SYN	(Synchronous idle)	54	6	86	V	118	v	150	û	182	Ā	214	Í	246	÷
23	ETB	(End of trans. block)	55	7	87	W	119	w	151	ù	183	Ă	215	Î	247	°
24	CAN	(Cancel)	56	8	88	X	120	x	152	ÿ	184	©	216	Ï	248	°
25	EM	(End of medium)	57	9	89	Y	121	y	153	Ö	185	⋮	217	Ĵ	249	°
26	SUB	(Substitute)	58	:	90	Z	122	z	154	Ü	186	⋮	218	Ŕ	250	°
27	ESC	(Escape)	59	;	91	[	123	{	155	ø	187	⋮	219	Ŗ	251	°
28	FS	(File separator)	60	<	92	\	124		156	£	188	⋮	220	Ÿ	252	°
29	GS	(Group separator)	61	=	93	]	125	}	157	Ø	189	¢	221	Ź	253	°
30	RS	(Record separator)	62	>	94	^	126	~	158	×	190	¥	222	ı	254	■
31	US	(Unit separator)	63	?	95	_			159	f	191	ſ	223	■	255	nbs
127	DEL	(Delete)														

# Variabili carattere

- Il codice ASCII determina l'ordinamento fra i caratteri.







# Variabili carattere

- Istruzione che associa ad ogni lettera maiuscola un numero intero a partire da 0:

`int numero = carattereLetto - 'A';`



`[0, 25] → 26 numeri interi`



# Variabili carattere

- Convertire un carattere numerico (cifra) in un numero decimale:

```
int numero = carattereLetto - '0';
```



$[0, 9] \rightarrow 10$  numeri interi



# Variabili carattere in C++

```
char car = 'A';
```

```
cout<<car;      Output:    A
```

```
char car = 65;
```

```
cout<<car;      Output:    A
```

$(65)_{10} \rightarrow \text{decimale}$   
↓  
 $(01000001)_2 \rightarrow \text{binario}$

$$(01000001)_2 = 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = (65)_{10}$$

```
char car = 'A':    cout<<(car+1);        //66
```

```
cout<<(char)(car+1); //B
```



# Esempio di conversione da base 10 a base 2

$$(64)_{10} \leftrightarrow (01000000)_2$$

$$64 / 2 = 32 \text{ con resto } 0 \quad (2^0)$$

$$32 / 2 = 16 \text{ con resto } 0 \quad (2^1)$$

$$16 / 2 = 8 \text{ con resto } 0 \quad (2^2)$$

$$8 / 2 = 4 \text{ con resto } 0 \quad (2^3)$$

$$4 / 2 = 2 \text{ con resto } 0 \quad (2^4)$$

$$2 / 2 = 1 \text{ con resto } 0 \quad (2^5)$$

$$1 / 2 = 0 \text{ con resto } 1 \quad (2^6)$$

(il ciclo si interrompe quando il quoziente diviene zero).



# Variabili carattere in C++

I/O caratteri:

```
cin>>car;  
cout<<car;
```

Posso scrivere anche: `cout<<"/";`

Ci sono altre forme per leggere/scrivere caratteri, ad es.

```
car = cin.get();  
cout.put(car);
```

(per alcune forme di lettura può essere necessario effettuare, subito dopo la lettura, una pulizia del buffer di tastiera).

# Variabili carattere in C

```
char car = 'A';
```

```
printf("%c", car);
```

```
printf("%d", car);
```

Output: A

$(65)_{10}$

→ decimale



$(01000001)_2$

→ binario

$$(01000001)_2 = 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = (65)_{10}$$



# Variabili carattere in C

I/O caratteri:

`scanf("%c", &car);`    → problema: buffer tastiera  
`printf("%c", car);`

`car = getchar();`        → problema: buffer tastiera  
`putchar(car);`



# Variabili carattere in C

- Soluzione al problema: buffer (di tastiera) “sporco”:

```
fflush(stdin);
```

oppure

```
while(getchar() != '\n');
```

Ti ricordo che, in C, il return (invio):

carattere: `'\n'` ha codice ASCII (decimale) 10

(vedi disegno alla lavagna)





# Variabili carattere in C

- I/O caratteri (funzioni non ANSI C):
- `car = getche();` → legge e visualizza un carattere, non attende il return (invio).
- `car = getch();` → legge un carattere senza visualizzarlo; non attende il return (invio).



# Variabili carattere

`#include <ctype.h>` solo in C (non si mette in C++)

`int tolower (int num);` → converte una lettera maiuscola in minuscola

`int toupper (int num);` → converte una lettera minuscola in maiuscola



# Selezione multipla

- Struttura di selezione (2 forme)
  - selezione binaria (a 2 vie): `if ... else ...` (forma principale)
  - selezione multipla (n-aria, a n vie): `switch() ... case`  
(si può simulare con la selezione binaria)

Non vediamo né il flow-chart (molto complesso), né lo pseudocodice (facoltativo: vedi libro di testo).



# Selezione multipla

```
switch( espressione ) {                                //tipo: int, char
    case costante1:
        sequenza istruz.;
        break;
    case costante2:
        sequenza istruz.;
        break;
    .....
    [ default :
        sequenza istruz; ]
}
```



# Selezione multipla

## Problema

Il prezzo di un abbonamento autobus urbano mensile è di 30,00€.

Su tale prezzo vengono fatte le seguenti riduzioni:

- ☐ il 50% per i bambini sotto i 10 anni
- ☐ il 40% per gli studenti
- ☐ l'80% per i militari
- ☐ il 30% per i pensionati.

Determinare le diverse tariffe di abbonamento ogni volta che l'utente lo richiede.



# Interfaccia utente

- Il computer come intermediario tra l'uomo e l'universo delle macchine.

**uomo ↔ computer ↔ universo delle macchine**

Progettare una interfaccia uomo-computer significa entrare nella sfera mentale dell'operatore allo scopo di presentare dati/informazioni in modo ottimale, minimizzare gli sforzi mnemonici, guidare l'utente nella esecuzione di compiti e nella risoluzione di problemi.

La prima domanda che il progettista si deve porre quando progetta un'interfaccia utente per una applicazione è:

**Quale utente?**



# Interfaccia utente

- Evoluzione dei paradigmi di interazione uomo-computer e correlazione con l'avvento di nuove tecnologie.

## **Tecnologia**

- ☐ Telescrivente
- ☐ Video
- ☐ Video grafico
- ☐ Disco ottico

## **Paradigma di interazione**

- stampare
- scegliere (menù)
- indicare (icone)
- collegare (ipermedia)

- ☐ Interazione multisensoriale: vista, voce, tatto, udito.



# Interfaccia utente

- Video grafico: indicare (icone)
  - La metafora della scrivania: lo schermo del computer diventa il piano di una scrivania (icone che rappresentano documenti, calcolatrice, ora, posta (in/out), archivi, cestino, ...). La scelta viene fatta mediante un dito elettronico: il mouse.
  - Questa filosofia di interazione tra utente e computer è ben descritta dallo slogan divenuto famoso: “What you see is what you get” (WYSIWYG): ciò che vedi è ciò che ottieni.
- Video grafico “touch screen”: la scelta si può fare usando un dito reale nel caso di schermi sensibili al tatto.
  - Oggi il piano (reale) della mia scrivania può essere utilizzato come un video touch screen. [zoom\\_image.jpg](#)





# Interfaccia utente

## ■ s/w di base o di sistema:

- La parte del sistema operativo dedicato al dialogo con l'utilizzatore. I sistemi operativi DOS e UNIX hanno un'interfaccia utente di tipo testuale, cioè i comandi vanno digitati sulla tastiera (interfaccia a riga di comando), invece i sistemi operativi Windows e MacOS hanno un'interfaccia grafica di tipo GUI (Interfaccia Grafica per l'Utente, si tratta di un'interfaccia basata su simboli grafici e non su testo).

## ■ s/w applicativo

- Interfaccia a menù
- Interfaccia grafica con finestre, menù, icone.



# Selezione multipla

- Esercizi\CASE.CPP



# Selezione multipla

```
int main() {
    char num;

    cout<<"Digita una cifra"<<endl;           //printf("Digita una cifra\n");
    cin>>num;                                  //scanf("%c",&num);

    switch(num) {
        case '0':
        case '2':
        case '4':
        case '6':
        case '8':
            cout<<"Pari"<<endl;                 //printf("Pari\n");
            break;

        case '1':
        case '3':
        case '5':
        case '7':
        case '9':
            cout<<"Dispari"<<endl;              //printf("Dispari\n");
            break;

    }
    cin.get();                                //getchar();
    return 0;
}
```