

Esercizi su

---

Operatori relazionali e logici

Istruzioni di scelta: selezione

# Esercizi precedenti

---

- Dubbi?
- Domande?

# Tipo booleano

---

- Scrivere un programma che
  - definisca una variabile di tipo *bool*,
  - le assegni *true* e la stampi
  - successivamente assegni *false* alla stessa variabile e la stampi di nuovo
  - *stampa\_bool.cc*

# Espr. logiche semplici 1/2

---

- Scrivere un programma che:
  - chieda all'utente di inserire due valori interi  $a$  e  $b$ ;
  - stampi 1 se  $a < b$ , 0 altrimenti
- Attenzione: l'operatore  $<<$  ha precedenza minore di  $<$ ,  $>$ ,  $>=$ , ...

# Espr. logiche semplici 2/2

---

- Esempio di output:

Inserisci i valori di a e di b: 3 4

Valore di  $3 < 4$ : 1

- *stampa\_logica\_semplice.cc*

# Espressioni logiche composte

---

- Scrivere un programma che:
  - chieda all'utente di inserire tre valori interi  $a$ ,  $b$  e  $c$ ;
  - stampi 1 se  $a < b$  oppure se  $a < c$ , 0 altrimenti
- Attenzione di nuovo alle precedenze
- *stampa\_logica\_composta.cc*

# Errore logico 1/2

---

- Quanto vale l'espressione logica:

$$1 < 3 < 2 ?$$

- Equivale a

$$(1 < 3) < 2$$

- Ossia

$$true < 2$$

# Errore logico 2/2

---

- `true` è convertito ad `1`, quindi
- $1 < 2$
- Quindi: *true* !!!!!!!!
- Problema: abbiamo confuso le regole di valutazione di una formula matematica con quelle di una espressione logica in C/C++



- Scrivere un programma che:
  - chieda all'utente di inserire tre valori interi  $a$ ,  $b$  ed  $x$ ;
  - stampi 1 se  $a \leq x \leq b$ , 0 altrimenti
- *stampa\_1\_se\_in\_intervallo.cc*

# Indentazione 1/3

---

- Se C1 e' la colonna rispetto alla quale sono allineate
- l'intestazione di una funzione,
- una istruzione condizionale o una istruzione iterativa,
- o l'inizio di una istruzione composta o di un blocco


# Indentazione 2/3



---

- Tutte le istruzioni appartenenti al loro corpo, devono essere allineate a partire da una colonna C2,
- spostata a destra di un numero prefissato di spazi rispetto a C1.

# Indentazione 3/3

---

```
int main()
{
    int i ;
    ...
     if (...)
        cout<<"messaggio" ;

      return 0 ;
}
```

- Scrivere/progettare i programmi prima carta e penna
- Rileggerli mettendosi nei panni
  - del compilatore prima
  - e del computer (esecutore) dopo
- Guardare le soluzioni solo quando si è sicuri di non essere in grado di risolvere l'esercizio da soli

# Esercizio 1/2

---

- Scrivere un programma che legge un numero intero da *standard input (cin)* e stampa

*Il numero inserito è positivo*

se il numero inserito è positivo.  
Altrimenti non stampa nulla ed esce.

# Esercizio 2/2

---

```
int main()
{
  int i ;
  cin >> i ;
  if (i > 0)
    cout << "Il numero inserito è positivo\n" ;

  return 0 ;
}
```

# Esercizio 1/2

---

- Scrivere un programma che legge un numero intero da *stdin* e stampa

*Il numero inserito è non negativo*

se il numero inserito è positivo o nullo. Altrimenti stampa

*Il numero inserito è negativo*



# Esercizio 2/2

---

```
int main()
{
  int i ;
  cin>>i ;

  if (i >= 0)
    cout<<"Il numero inserito è non negativo\n" ;
  else
    cout<<"Il numero inserito è negativo" ;

  return 0 ;
}
```

- Scrivere un programma che stampi il massimo tra due numeri interi letti dallo *stdin*
- Esempio:  
*Inserire i due numeri interi: 21 -3*  
*Il massimo tra 21 e -3 è 21*
- Nota: non è necessario andare a capo quando si immettono i due numeri
- Soluzione: *stampa\_max.cc*

# Messaggi di errore g++

---

- Ogni riga inizia con il nome del file sorgente
- Poi c'è
  - il nome della funzione
  - oppure il numero di riga e colonna in cui si è verificato l'errore
- Poi il termine *error* o *warning*
- Infine la descrizione del problema (può proseguire su più righe)

# Esercizio: divisione intera 1/4

---

- Riprendiamo l'esercizio di calcolo della divisione intera tra due numeri interi

*Inserisci i due numeri: 5 2*

$$5 / 2 = 2$$

# Esercizio: divisione intera 2/4

---

```
int main()
{
    int i, j ;
    cout<<Inserisci due numeri interi: " ;
    cin>>i>>j ;

    cout<<i<<" / "<<j<<"="<<i/j<<endl;

    return 0 ;
}
```

# Esercizio: divisione intera 3/4

---

- Proviamo ad inserire ad esempio 3 e 0

*Inserisci i due numeri: 3 0*

*????????*

- Cosa è successo?

# Alcune cause comuni ...

---

- ... di **fallimenti** a tempo di esecuzione:
  - Le variabili non sono inizializzate
  - I valori dei parametri attuali o dei valori letti non sono quelli attesi
  - C'è stato un *overflow*
    - Lo vedremo meglio nella prossima esercitazione

# Correggiamo ...

---

- ... l'esercizio sulla divisione intera



# Esercizio: divisione intera 4/4

---

```
int main()
{
    int i, j ;
    cout<<Inserisci due numeri interi: " ;
    cin>>i>>j ;

    if (j == 0)
        cout<<"Attenzione: il divisore è nullo\n" ;
    else
        cout<<i<<" / "<<j<<"="<<i/j<<endl;

    return 0 ;
}
```

# Gestione delle eccezioni

---

- Spesso è necessario controllare il valore dei parametri attuali o dei valori letti e prendere contromisure
- Segnalare errore ed uscire
- Modificare i valori per riportarli forzatamente in intervalli validi

# Esercizio 1/2

---

- Scrivere un programma che definisca ed inizializzi due costanti intere  $a$  e  $b$ , poi legga in ingresso un numero intero e scriva un messaggio se il numero non è compreso nell'intervallo  $[a, b]$

*Inserisci un numero intero: 101*

*101 non è in [1, 100]*

# Esercizio 2/2

---

```
int main()
{
    int i, a = 1, b = 100 ;

    cout<<"Inserisci un numero intero: " ;
    cin>>i ;

    if (i < a || i > b)
        cout<<i<<" non è in ["<<a<<" , "<<b<<" ]\n";

    return 0 ;
}
```

- La precedente soluzione contiene una imprecisione rispetto alla traccia
  - Provate ad individuarla prima di guardare la risposta nella slide successiva

- $a$  e  $b$  sono variabili e non costanti

# Esercizio 1/2

---

- Scrivere un programma che definisca ed inizializzi due costanti intere  $a$  e  $b$ , poi legga in ingresso un numero intero e scriva un messaggio se il numero è compreso nell'intervallo  $[a, b]$

*Inserisci un numero intero: 5*

*5 è in [1, 100]*

- Non utilizzare l'operatore `||`

# Esercizio 2/2

---

```
int main()
{
    const int i, a = 1, b = 100 ;

    cout<<"Inserisci un numero intero: " ;
    cin>>i ;

    if (i >= a && i <= b)
        cout<<i<<" è in ["<<a<<" , "<<b<<" ]\n";

    return 0 ;
}
```



- La precedente soluzione contiene un errore
  - Provate ad individuarlo prima di guardare la risposta nella slide successiva

- $i$  è erroneamente definita costante

# Esercizio 1/2

---

- Scrivere un programma che definisca ed inizializzi due costanti intere  $a$  e  $b$ , poi legga in ingresso un numero intero e scriva un messaggio se il numero è compreso nell'intervallo  $[a, b]$

*Inserisci un numero intero: 5*

*5 è in [1, 100]*

- Non utilizzare l'operatore  $\&\&$

# Esercizio 2/2

---

```
int main()
{
    const int a = 1, b = 100 ;

    cout<<"Inserisci un numero intero: " ;
    int i ; cin>>i ;

    if (!(i < a || i > b) )
        cout<<i<<" è in ["<<a<<" , "<<b<<" ]\n";

    return 0 ;
}
```

# Esercizio 1/3

---

- Scrivere un programma che definisca ed inizializzi due costanti intere  $a$  e  $b$ , poi legga in ingresso due numeri interi: *controllo* ed  $i$
- L'intero *controllo* si utilizza per controllare il comportamento del programma
- In particolare,
  - se *controllo* è diverso da 0,
    - il programma scrive un messaggio se  $i$  non è compreso nell'intervallo  $[a, b]$
  - altrimenti non stampa nulla

- Esempio:

*Inserisci il valore per controllo: 1*

*Inserisci un numero intero: 0*

*0 non è in [1, 100]*

# Esercizio 3/3

---

```
int main()
{
    const int a = 1, b = 100 ;
    int i, controllo ;

    cout<<"Inserisci un numero intero: " ;
    cin>>i ;

    cin>>controllo ;

    if (controllo != 0 && (i < a || i > b) )
        cout<<i<<" non è in ["<<a<<" , "<<b<<"]\n";

    return 0 ;
}
```

# Compiti per casa

---

- *multiplo.cc*
- *tre\_ordinati.cc*