UNIVERSITÀ
DI TORINO

DIPARTIMENTO DI
INFORMATICA

CORSO DI LAUREA
TRIENNALE

Programmazione I
Laboratorio

Elvio Amparore (turno 1)
Alessia Antelmi (turno 2)

Lezione n° 4

Iterazioni

A.A. 2024/25
Canale A

Iterazioni con il for

Quando usare il for:

- quando il numero di iterazioni è noto sin dall'inizio. Tipicamente i cicli for avranno una forma del tipo:

```
for (int i=0; i<n; i++) {
    // codice che usa ma non cambia i
    ...
}
```

oppure:

```
for (int i=n-1; i>=0; i--) {
    // codice che usa ma non cambia i
    ...
}
```

Iterazioni con il while

Quando usare il while:

- quando la condizione di terminazione si determina solo durante le iterazioni stesse.
- quando non serve inizializzare una variabile di iterazione

```
bool continua = true;
while (continua) {
   ...
   if (condizione)
      continua = false;
}
```

```
int i = 0;
...
while (i < n) {
   ...
   i = i + k;
}
```

Dichiarazione del ciclo for

Come cambia la variabile di controllo i in questi cicli for?

```
for (int i = 1; i <= 100; ++i) { … }

for (int i = 100; i >= 1; --i) { … }


for (int i = 7; i <= 77; i += 7) { … }

for (int i = 20; i >= 2; i -= 7) { … }

for (size_t i = 0; i < 10; ++i) { … }

for (ptrdiff_t i = 100; i > 0; --i) { … }
```

Esempio: sommatoria (1 / 2)

Scriviamo un ciclo che calcola la somma dei numeri pari da 2 a 100:

```
// inizializza accumulatore della somma
int sum = 0;

// scorre tutti i numeri da 0 a 100
for (int number = 0; number <= 100; ++number) {
   // accumula in sum i valori pari
   if (number % 2 == 0)
      sum += number;
}

printf("La somma è %d\n", sum);
```

Esempio: sommatoria (2 / 2)

Scriviamo un ciclo che calcola la somma dei numeri pari da 2 a 100:

```
// inizializza accumulatore della somma
int sum = 0;

// scorre tutti i numeri pari da 2 a 100
for (int number = 2; number <= 100; number += 2) {
    sum += number;
}

printf("La somma è %d\n", sum);
```

Esercizi iterativi per iniziare

Scrivere un programma **iterazioni_n.c** che implementa i seguenti punti:

1. Dato un intero positivo **n** letto da input con scanf, si stampino a video tutti gli interi da **0** a **n** e poi da **n** a **0**. **DOMANDA**: Quale ciclo è più appropriato?
2. Successivamente si stampino a video tutti gli interi dispari da **0** a **n**
3. Infine si stampi a video il fattoriale di **n**.

Si ricorda che il fattoriale di **n** è pari a: 1 * 2 * … * (**n**-1) * **n**.

Iterazioni su sequenze

- Ipotizziamo di leggere una sequenza (con scanf) per sommarne tutti i numeri, finché non leggiamo il numero 0 (che funge da terminatore della sequenza).
- Non sappiamo quindi quanti numeri leggere, finché non leggiamo e troviamo lo 0.
- In questo caso, usiamo il ciclo while.

Iterazioni su sequenze (1 / 2)

```
// inizializza accumulatore della somma
int sum = 0;
bool continue = true;

while (continue) {
    int number;
    // legge il prossimo numero della sequenza
    scanf("%d", &number);
    if (number == 0)
        continua = false; // sequenza terminata
    else
        sum += number;
}

printf("La somma è %d\n", sum);
```

Iterazioni su sequenze (2 / 2)

```
// inizializza accumulatore della somma
int sum = 0;
int number;

while (1 == scanf("%d", &number) && number != 0) {
    sum += number;
}

printf("La somma è %d\n", sum);
```

NOTA: scanf legge un numeri di *specificatori di conversione* (ciascuno preceduto dal %) dalla stringa di formato, e "**ritorna**" il numero di variabili riceventi lette con successo.

Quindi se abbiamo una stringa di formato "%d", **scanf** ritorna 1 se e solo se ha letto un numero intero. In ogni altro caso (non abbiamo un numero, l'input è terminato, ...) non ritorna 1.

NOTA: stiamo usando la **valutazione lazy**.

Media di una sequenza

Scrivere un programma **media_seq.c** che:

- Legge dall'input una sequenza di numeri interi, terminata dal numero **0** (che assumiamo non appartenga alla sequenza letta).
- Quando l'utente inserisce il numero **0**, il programma interrompe la lettura della sequenza, e stampa la media (come numero intero) di tutti i numeri letti.
- Esempio

Sequenza in input: **4 8 10 2 0** Output: **6**

SUGGERIMENTO: ci serviranno due variabili di supporto:

- un accumulatore che somma i numeri letti;
- un contatore di quanti numeri sono stati letti.

Per cui, alla fine del ciclo, possiamo calcolare:

media = accumulatore_somma / contatore

Media di una sequenza

Nel codice iniziale, trovate una cartella **input_media_seq** con all'interno diverse sequenze di numeri, una per file.

Possiamo eseguire il nostro programma **media_seq** in due modi:

- **Interattivo**: lanciamo **media_seq** e digitiamo a mano l'input
- **Non-interattivo**: usiamo il contenuto di un file come input per il programma **media_seq**.

Detto *input01.txt* il file che contiene la sequenza in input, usiamo:

- Unix: ./**media_seq** < *input01.txt*
- Windows: **media_seq.exe** < *input01.txt*

Divisori Interi

Sulla pagina Moodle trovate un esercizio con nome

Lab04-Es1 Divisori Interi

Completate il programma, assicurandovi che passi tutti i test proposti.

Quale ciclo è più adatto per il problema proposto?

Numeri di Floyd

Scrivere un programma **floyd.c** che prende in input un numero **n** e stampa le prime **n** righe del *triangolo di Floyd*.

Il triangolo di Floyd è costituito dai numeri naturali scritti in modo consecutivo, per riempire le righe con 1,2,3,... valori.

Ad esempio per **n**=5 il programma deve stampare:

```
1
2  3
4  5  6
7  8  9  10
11  12  13  14  15
```

NOTA: I numeri di Floyd non c'entrano nulla con la logica di Floyd!!

Quadrato

Scrivere un programma **quadrato.c** che legge in input un numero **n** e stampa a video un quadrato di **n***n caratteri tra '*', '\' e ':' seguendo questo pattern (esempio per **n**=6):

```
\:::::
*\::::
**\:::
***\::
****\:
*****\
```

Suggerimento: fare due cicli annidati, uno esterno per le righe, ed uno o più cicli interni per i caratteri sulla stessa una riga.

Quantificazione universale

Supponiamo che stiamo scrivendo un programma con un ciclo che legge una sequenza di interi, terminata dallo **0**.

Vogliamo rispondere a questa domanda:

Tutti i numeri della sequenza
soddisfano una condizione *Cond*?

Come possiamo fare?

```
bool tutti = true;
while (1==scanf("%d", &n) && n!=0) {
    if (! Cond valutata su n )
        tutti = false;
}
```

Quantificazione esistenziale

Supponiamo che stiamo scrivendo un programma con un ciclo che legge una sequenza di interi, terminata dallo **0**.

Vogliamo rispondere a questa domanda:

Esiste almeno un numero della sequenza
che soddisfa una condizione *Cond*?

Come possiamo fare?

```
bool esiste = false;
while (1==scanf("%d", &n) && n!=0) {
    if (Cond valutata su n)
        esiste = true;
}
```

Cicli con accumulatori booleani

- Se invece non dobbiamo leggere tutti gli elementi della sequenza (ad esempio perché sono già in memoria), possiamo aggiungere la variabile `tutti(esiste)` che accumula la condizione *Cond* anche nella condizione del while.
- In questo modo il ciclo termina non appena la variabile che accumula la condizione booleana diventa falsa (universale) o vera (esistenziale).
- Vedremo in un prossimo laboratorio questo schema.

Esercizi su condizioni

Sulla pagina Moodle trovate un esercizio con nome

Lab04-Es2 Condizioni su sequenza

Completate il programma, determinando correttamente tutte le condizioni esiste/per-ogni che vengono richieste.

Fibonacci

Scrivere un programma **fibonacci.c** che legge in input un numero **k** e stampa a video i primi **k** numeri della successione di Fibonacci. Ad, esempio la stampa attesa per **k = 10** è:

```
0  1  1  2  3  5  8  13  21  34
```

Per semplicità assumere che **k** >= 2.

Si ricorda che la successione di Fibonacci parte da due numeri 0 e 1, ed ogni elemento successivo della successione è ottenuto come somma dei due elementi precedenti.

Suggerimento: partite da due numeri **n=0** ed **m=1** e stampateli. Ad ogni iterazione si procede ad aggiornare le variabili:

$$n' = m, \quad m' = n + m$$

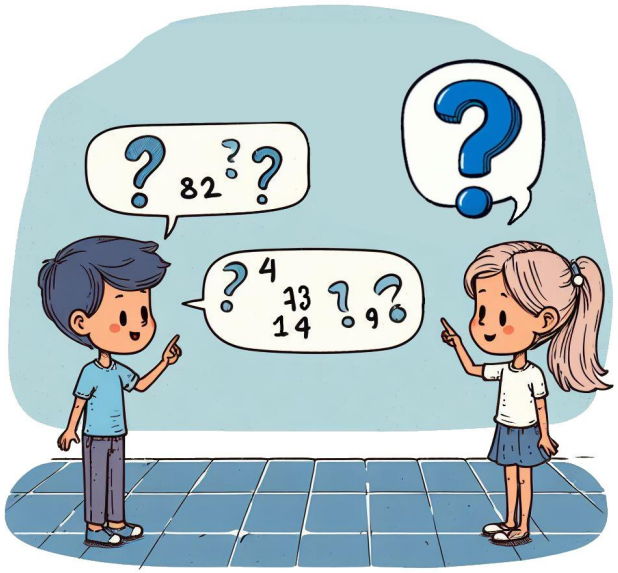
e a stampare **m'**.

Domanda: Per fare questo aggiornamento serve una variabile temporanea?

Fai indovinare un numero!

Partendo dal codice iniziale di **fai_indovinare_numero.c**, leggere con attenzione il codice e la richiesta, ed implementare il gioco.

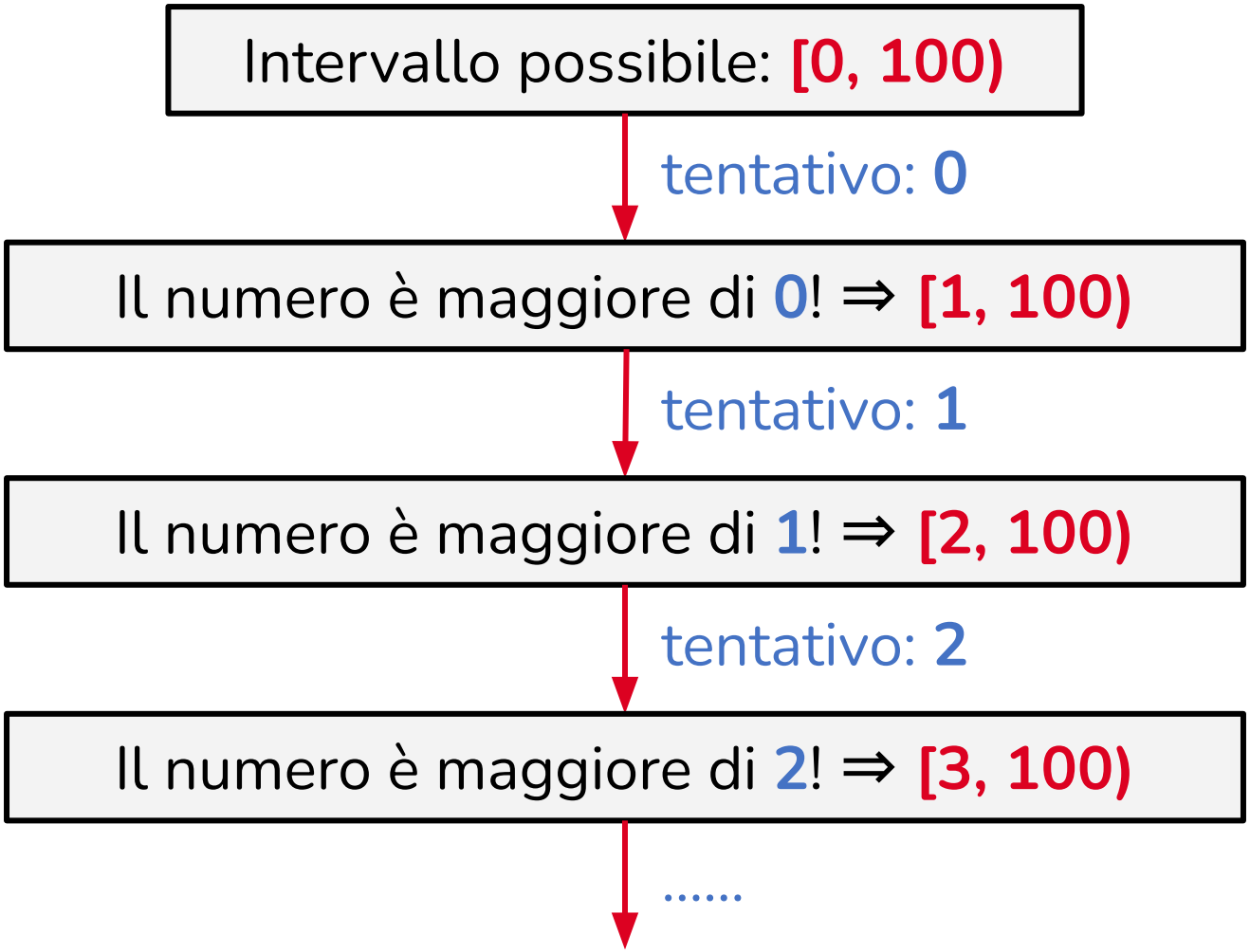
Quale strategia usa una persona per indovinare il numero casuale scelto dal programma C?



Strategia di gioco con iterazioni

ESERCIZIO EXTRA

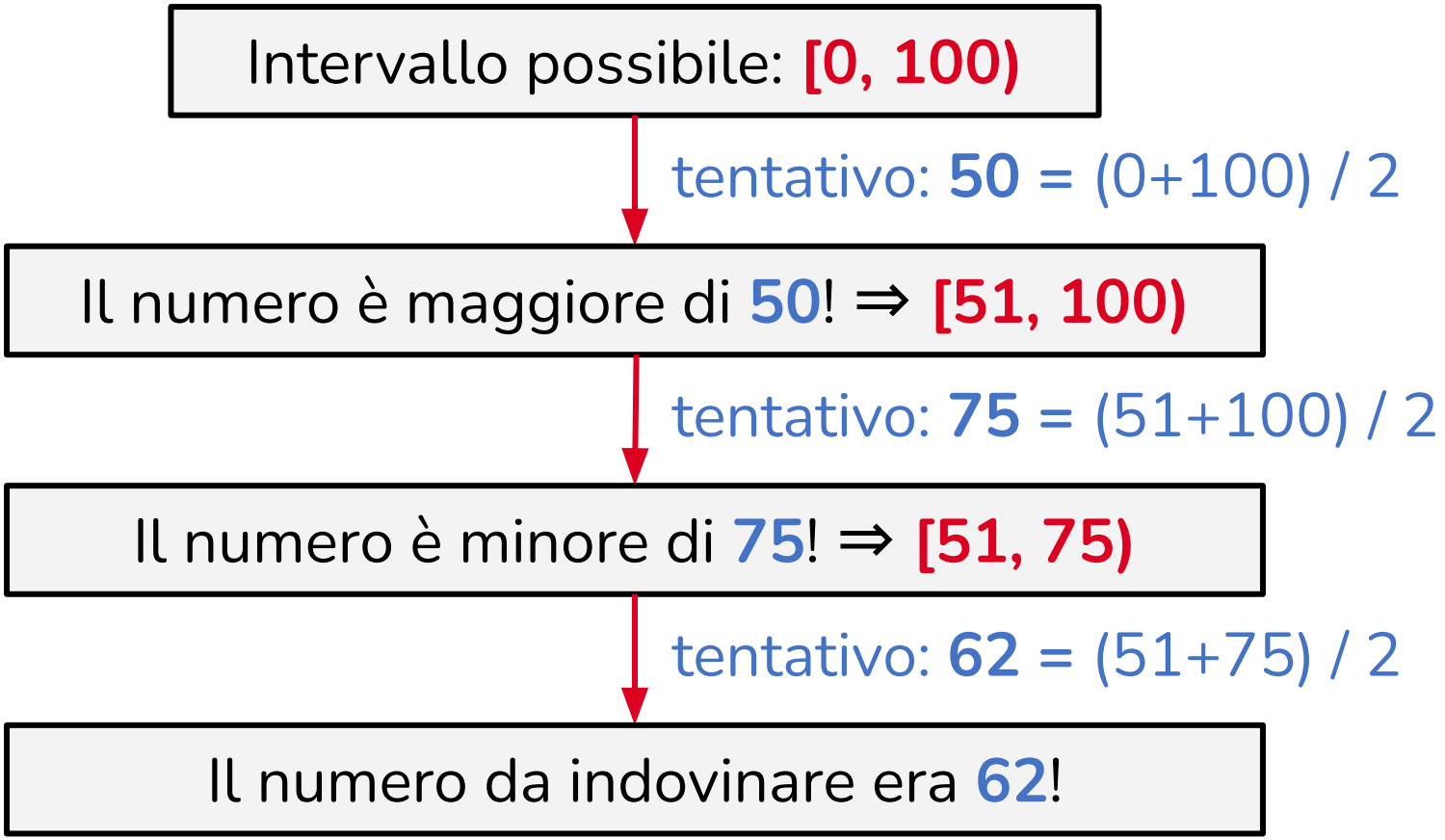
Sappiamo che il numero da indovinare è compreso tra **0** incluso e **100** escluso. Supponiamo sia **62**. Quindi:



Strategia di gioco con dimezzamenti

ESERCIZIO EXTRA

Sappiamo che il numero da indovinare è compreso tra **0** incluso e **100** escluso. Supponiamo sia **62**. Quindi:



Indovina tu il numero!

ESERCIZIO EXTRA

Adesso scrivete l'algoritmo che indovina il numero casuale che ha scelto il computer!

Sulla pagina Moodle trovate un esercizio con nome

Lab04-Es3 Indovina tu il numero!

Completate il programma, assicurandovi che indovini tutti i valori nel numero massimo di tentativi concessi.

Quale ciclo è più adatto per il problema proposto?