

Presentazioni per la lezione frontale con la LIM

Per ogni capitolo del corso è disponibile una sintesi illustrata degli argomenti in forma di presentazione per la LIM.

Le presentazioni, progettate come ausilio per la lezione frontale, sono disponibili in tre versioni:

- versione compatta nel formato **.pdf** di Adobe Acrobat;
- versione interattiva nel formato PowerPoint **.ppt** di Microsoft Office;
- versione interattiva nel formato Impress **.odp** di LibreOffice.

Le versioni interattive – che presentano ciascun argomento aggiungendo via via informazioni sulle slide - si possono editare aprendole con le rispettive applicazioni.

In ogni capitolo la prima slide è un indice formato da **titoli attivi**: con un clic del mouse si può passare direttamente a ogni singola lezione, senza dover «sfogliare» tutta la presentazione.

In ogni slide il **pulsante INDICE** in alto a destra consente di ritornare alla slide con l'indice del capitolo.

capitolo 14 – I cicli iterativi

1. Il ciclo **for**
2. Strutture iterative annidate
3. Il ciclo **while**
4. Il ciclo **do**

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

14.3 Il ciclo **while**

Indice

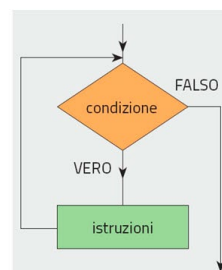
Il ciclo **while** è utile quando il numero delle iterazioni non è noto *a priori*.

sintassi:

```
inizializzazione di var;  
while (condizione su var)  
{ istruzioni da eseguire finché la condizione è vera; }
```

qui niente punto e virgola

diagramma di flusso:



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

Nella **versione .pdf** della presentazione ogni slide appare a schermo già completa di tutti i contenuti.

Nelle **versioni interattive** invece i contenuti appaiono gradualmente. Qui per esempio appare per prima cosa il **diagramma di flusso**...

... poi ogni successivo **click del mouse** mette in evidenza e spiega un elemento specifico del diagramma.

14.3 Il ciclo while

Indice

Il **ciclo while** è utile quando il numero delle iterazioni non è noto *a priori*.

Per qualsiasi ciclo **for** si può scrivere un ciclo **while** equivalente.

Esempio:

ciclo **for**

```
int conteggio;
for (conteggio=0; conteggio<=10; conteggio++)
{
printf("\n %d\n", conteggio);
}
```

ciclo **while**

```
int conteggio=0;
while (conteggio<=10)
{
printf("\n %d\n", conteggio);
conteggio++;
}
```

ma con la struttura **while** si possono trattare anche quei casi in cui il numero delle iterazioni non è prestabilito all'inizio del ciclo

Tibone, Progettare e programmare © Zanichelli editore 2018

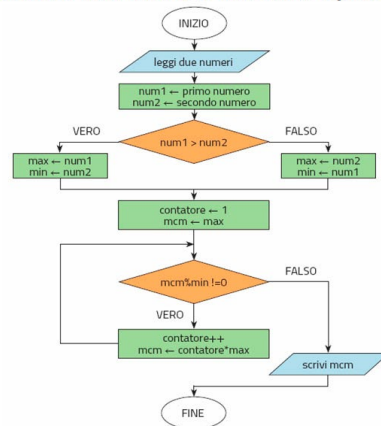
ZANICHELLI

14.3 Il ciclo while

Indice

Il **ciclo while** è utile quando il numero delle iterazioni non è noto *a priori*.

Esempio: calcolo del **minimo comune multiplo (mcm)** di due numeri dati



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

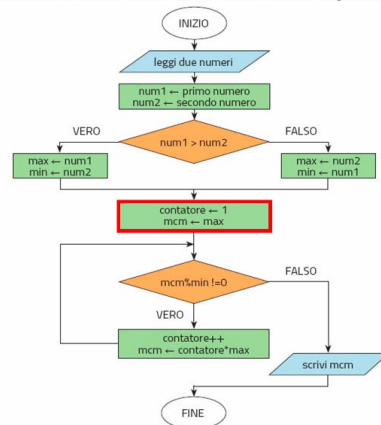
14.3 Il ciclo while

Indice

Il **ciclo while** è utile quando il numero delle iterazioni non è noto *a priori*.

Esempio: calcolo del **minimo comune multiplo (mcm)** di due numeri dati

1. si trova il maggiore dei due numeri e lo si chiama **mcm** + si inizia con **contatore = 1**



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

Questo permette all'insegnante di iniziare con una discussione generale dell'algoritmo...

... per poi illustrarne passo passo le caratteristiche più importanti.

A ogni clic del mouse sullo schermo si aggiunge una quantità limitata di nuove informazioni...

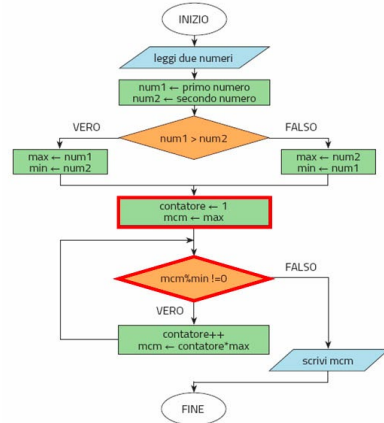
14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del **minimo comune multiplo (mcm)** di due numeri dati

1. si trova il maggiore dei due numeri e lo si chiama **mcm** + si inizia con **contatore = 1**
2. si controlla che **mcm** non sia multiplo del numero minore



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

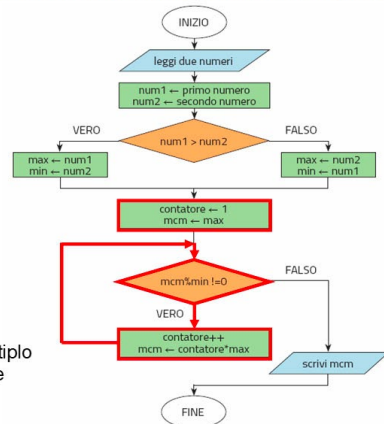
14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del **minimo comune multiplo (mcm)** di due numeri dati

1. si trova il maggiore dei due numeri e lo si chiama **mcm** + si inizia con **contatore = 1**
2. si controlla che **mcm** non sia multiplo del numero minore
3. se non lo è, si incrementa **contatore** e si riprova con il multiplo successivo del numero maggiore



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del **minimo comune multiplo (mcm)** di due numeri dati

l'algoritmo avrà sicuramente fine: infatti nel peggiore dei casi, quello in cui i due numeri di input sono primi tra loro, quando **contatore** diventa uguale al numero minore si avrà **mcm % min == 0**



Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

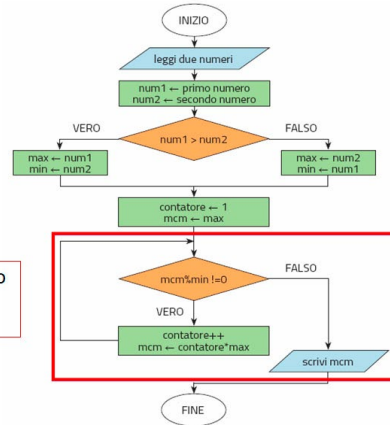
... e ciò rende più semplice per gli allievi seguire lo sviluppo del ragionamento.

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati



il numero delle iterazioni del ciclo è imprevedibile: dipenderà dai due numeri inseriti dall'utente

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

Un discorso analogo vale per la presentazione dei codici sorgente.

Si può mostrare il codice come riferimento per gli allievi mentre programmano...

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati

```
mcm-while.c
1 #include<stdio.h>
2 int main(void)
3 {
4     int numero1, numero2, max, min, contatore, mcm;
5     /*la variabile contatore servirà per iterare nel ciclo; la inizializziamo a 1*/
6     contatore=1;
7     printf("\n Inserisci il primo numero: ");
8     scanf("%d", &numero1);
9     printf("\n Inserisci il secondo numero: ");
10    scanf("%d", &numero2);
11    /*scopriamo quale tra i due numeri di input è più grande*/
12    if (numero1 > numero2)
13        {max = numero1;
14         min = numero2;}
15    else
16        {max = numero2;
17         min = numero1;}
18    /*inizializziamo la variabile mcm*/
19    mcm =contatore*max;
20    /*si itera se (e poi fintantoché) mcm non risulta divisibile per min*/
21    while (mcm%min != 0)
22        {
23            contatore++;
24            mcm =contatore*max;
25        }
26    printf("\n Il minimo comune multiplo tra %d e %d e' %d\n", numero1, numero2, mcm);
27 }
```

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

... e con l'avanzamento al clic del mouse si possono evidenziare i singoli costrutti...

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati

```
mcm-while.c
1 #include<stdio.h>
2 int main(void)
3 {
4     int numero1, numero2, max, min, contatore, mcm;
5     /*la variabile contatore servirà per iterare nel ciclo; la inizializziamo a 1*/
6     contatore=1;
7     printf("\n Inserisci il primo numero: ");
8     scanf("%d", &numero1);
9     printf("\n Inserisci il secondo numero: ");
10    scanf("%d", &numero2);
11    /*scopriamo quale tra i due numeri di input è più grande*/
12    if (numero1 > numero2)
13        {max = numero1;
14         min = numero2;}
15    else
16        {max = numero2;
17         min = numero1;}
18    /*inizializziamo la variabile mcm*/
19    mcm =contatore*max;
20    /*si itera se (e poi fintantoché) mcm non risulta divisibile per min*/
21    while (mcm%min != 0)
22        {
23            contatore++;
24            mcm =contatore*max;
25        }
26    printf("\n Il minimo comune multiplo tra %d e %d e' %d\n", numero1, numero2, mcm);
27 }
```

struttura di selezione per trovare il numero maggiore e il minore

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

LEZIONI
Presentazioni per la
lezione frontale con la LIM

... oppure le **connessioni** tra le diverse parti del codice sorgente.

Questo tipo di sequenza aiuta a illustrare in modo efficace la logica del funzionamento dei programmi...

... ed è utile anche come promemoria dei messaggi più importanti da trasmettere agli allievi.

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati

```
mcm-while.c
1 #include<stdio.h>
2 int main(void)
3 {
4     int numero1, numero2, max, min, contatore, mcm;
5     /*la variabile contatore servirà per iterare nel ciclo; la inizializziamo a 1*/
6     contatore=1;
7     printf("\n Inserisci il primo numero: ");
8     scanf("%d", &numero1);
9     printf("\n Inserisci il secondo numero: ");
10    scanf("%d", &numero2);
11    /*scopriamo quale tra i due numeri di input è più grande*/
12    if (numero1 > numero2)
13        (max = numero1;
14         min = numero2);
15    else
16        (max = numero2;
17         min = numero1);
18    /*inizializziamo la variabile mcm*/
19    mcm = contatore*max;
20    /*ci ripeteremo finché mcm non risulta divisibile per min*/
21    while (mcm%min != 0)
22    {
23        contatore++;
24        mcm = contatore*max;
25    }
26    printf("\n Il minimo comune multiplo tra %d e %d e' %d\n", numero1, numero2, mcm);
27 }
```

inizializzazione di mcm

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati

```
mcm-while.c
1 #include<stdio.h>
2 int main(void)
3 {
4     int numero1, numero2, max, min, contatore, mcm;
5     /*la variabile contatore servirà per iterare nel ciclo; la inizializziamo a 1*/
6     contatore=1;
7     printf("\n Inserisci il primo numero: ");
8     scanf("%d", &numero1);
9     printf("\n Inserisci il secondo numero: ");
10    scanf("%d", &numero2);
11    /*scopriamo quale tra i due numeri di input è più grande*/
12    if (numero1 > numero2)
13        (max = numero1;
14         min = numero2);
15    else
16        (max = numero2;
17         min = numero1);
18    /*inizializziamo la variabile mcm*/
19    mcm = contatore*max;
20    /*ci ripeteremo finché mcm non risulta divisibile per min*/
21    while (mcm%min != 0)
22    {
23        contatore++;
24        mcm = contatore*max;
25    }
26    printf("\n Il minimo comune multiplo tra %d e %d e' %d\n", numero1, numero2, mcm);
27 }
```

ciclo che si ripete fintantoché mcm non è multiplo di min

il ciclo si ripeterà per un massimo di min volte, poi terminerà e verrà stampato il risultato

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

14.3 Il ciclo while

Indice

Il ciclo while è utile quando il numero delle iterazioni non è noto a priori.

Esempio: calcolo del minimo comune multiplo (mcm) di due numeri dati

```
mcm-while.c
1 #include<stdio.h>
2 int main(void)
3 {
4     int numero1, numero2, max, min, contatore, mcm;
5     /*la variabile contatore servirà per iterare nel ciclo; la inizializziamo a 1*/
6     contatore=1;
7     printf("\n Inserisci il primo numero: ");
8     scanf("%d", &numero1);
9     printf("\n Inserisci il secondo numero: ");
10    scanf("%d", &numero2);
11    /*scopriamo quale tra i due numeri di input è più grande*/
12    if (numero1 > numero2)
13        (max = numero1;
14         min = numero2);
15    else
16        (max = numero2;
17         min = numero1);
18    /*inizializziamo la variabile mcm*/
19    mcm = contatore*max;
20    /*ci ripeteremo finché mcm non risulta divisibile per min*/
21    while (mcm%min != 0)
22    {
23        contatore++;
24        mcm = contatore*max;
25    }
26    printf("\n Il minimo comune multiplo tra %d e %d e' %d\n", numero1, numero2, mcm);
27 }
```

NB: se i due numeri di input sono multipli uno dell'altro, si avrà subito $mcm \% min = 0$ e le istruzioni del ciclo non verranno mai eseguite

Tibone, Progettare e programmare © Zanichelli editore 2018

ZANICHELLI

Presentazioni per la lezione frontale con la LIM
LEZIONI