



ELEMENTI DI INFORMATICA

DOCENTE: FRANCESCO MARRA

INGEGNERIA CHIMICA

INGEGNERIA ELETTRICA

SCIENZE ED INGEGNERIA DEI MATERIALI

INGEGNERIA GESTIONALE DELLA LOGISTICA E DELLA PRODUZIONE

INGEGNERIA NAVALE

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

STRUTTURA DI CONTROLLO



AGENDA

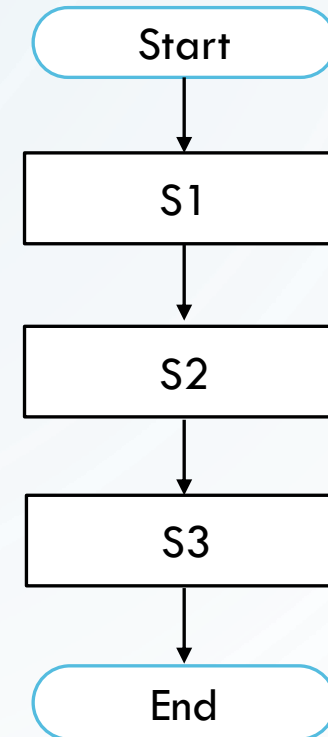
- Strutture di controllo
 - Sequenziali
 - Selettive
 - if
 - switch
 - Iterative
 - while
 - do while
 - for
 - Non strutturate

SULL'IMPORTANZA DELLE STRUTTURE

- **Teorema di Böhm-Jacopini (1966)**

- Qualunque algoritmo può essere implementato utilizzando **tre** sole strutture
 - Sequenza
 - Selezione
 - Iterazione o ciclo

Sequenza

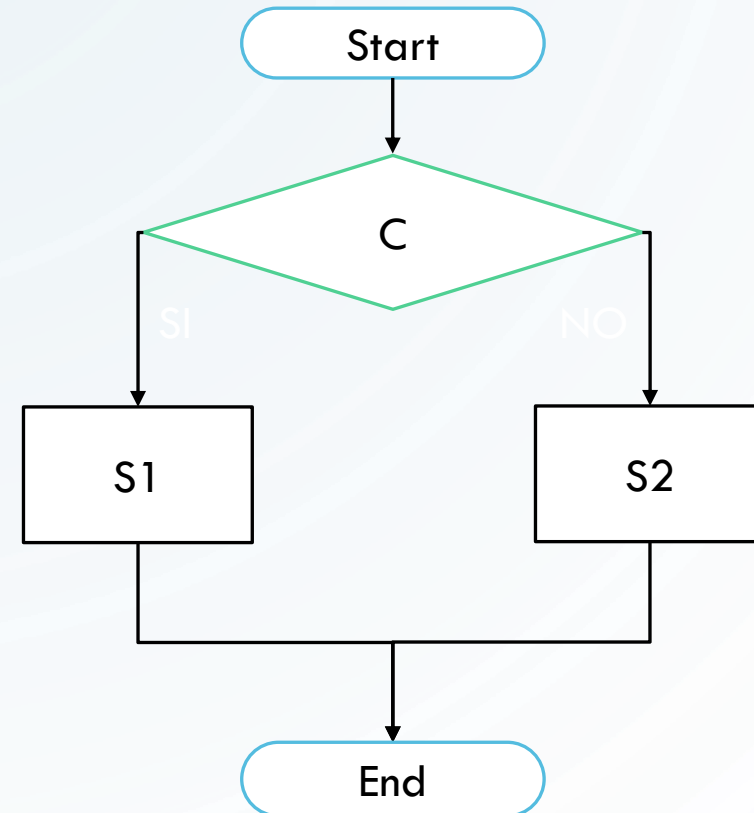


SULL'IMPORTANZA DELLE STRUTTURE

- **Teorema di Böhm-Jacopini**
(1966)

- Qualunque algoritmo può essere implementato utilizzando **tre** sole strutture
 - Sequenza
 - Selezione
 - Iterazione o ciclo

Selezione

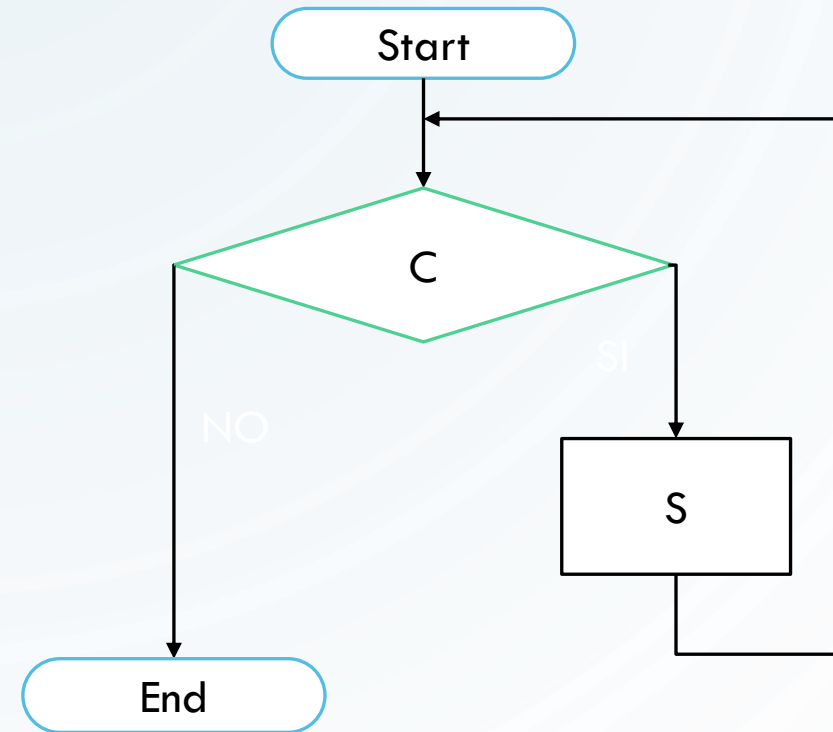


SULL'IMPORTANZA DELLE STRUTTURE

- **Teorema di Böhm-Jacopini**
(1966)

- Qualunque algoritmo può essere implementato utilizzando **tre** sole strutture
 - Sequenza
 - Selezione
 - Iterazione o ciclo

Iterazione

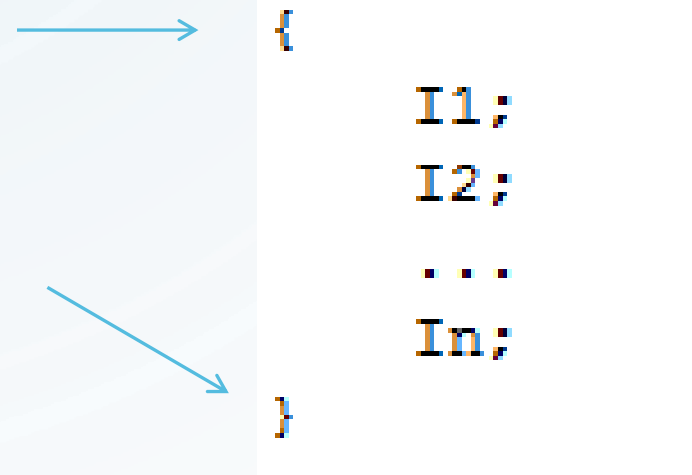


A decorative graphic on the left side of the slide, consisting of a network of thin, light-blue lines and small circles, resembling a circuit board or a neural network structure.

STRUTTURE DI CONTROLLO SEQUENZIALI

BLOCCO

- Elemento base per la costruzione di un programma
- Formato dalla sequenza S di n istruzioni
 - $I_1; I_2; \dots; I_n$
- L'inizio del blocco è indicato da una parentesi graffa aperta
- La fine del blocco è indicata da una parentesi graffa chiusa
- Le istruzioni vengono eseguite secondo l'ordine di lettura
 - Dall'alto verso il basso



The diagram illustrates the mapping from the list notation $I_1; I_2; \dots; I_n$ to the code block notation. A blue arrow points from the list notation to the opening curly brace '{' of the code block. Another blue arrow points from the list notation to the closing curly brace '}' of the code block. The code block contains the instructions I1;, I2;, ..., In; in a monospaced font.

```
{  
    I1;  
    I2;  
    ...  
    In;  
}
```


BLOCCO

- La presenza del carattere di terminazione «;» consente di scrivere le istruzioni anche su di uno stesso rigo

```
{  
    cout << "ciao ";  
    cout << "a ";  
    cout << "tutti";  
}
```



```
{  
    cout << "ciao "; cout << "a "; cout << "tutti";  
}
```

- La prima forma favorisce la lettura

BLOCCO

- Un blocco può contenerne altri
- Incolonnamento
 - Evidenzia la presenza di blocchi innestati
 - Migliora la leggibilità del codice

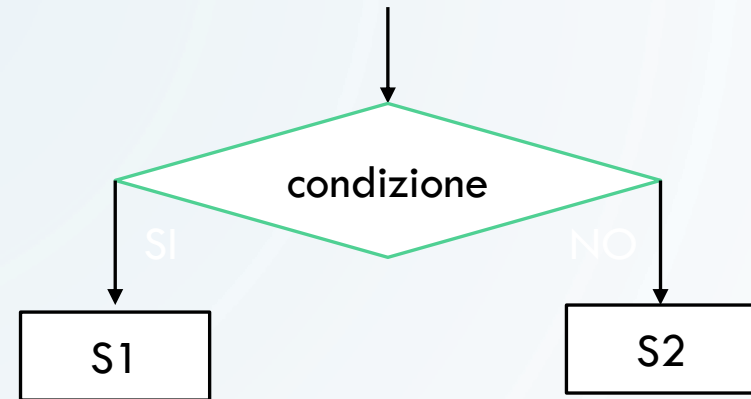
```
{  
  Blocco1-I1;  
  Blocco1-I2;  
  {  
    Blocco2-I1;  
    Blocco2-I2;  
    {  
      Blocco3-I1;  
      Blocco3-I2;  
    }  
    Blocco2-I3;  
  }  
  Blocco1-I3;  
}
```

A decorative graphic on the left side of the slide, consisting of a network of thin, light-blue lines and small circles, resembling a circuit board or a neural network, extending vertically from the top to the bottom.

STRUTTURE DI CONTROLLO SELETTIVE

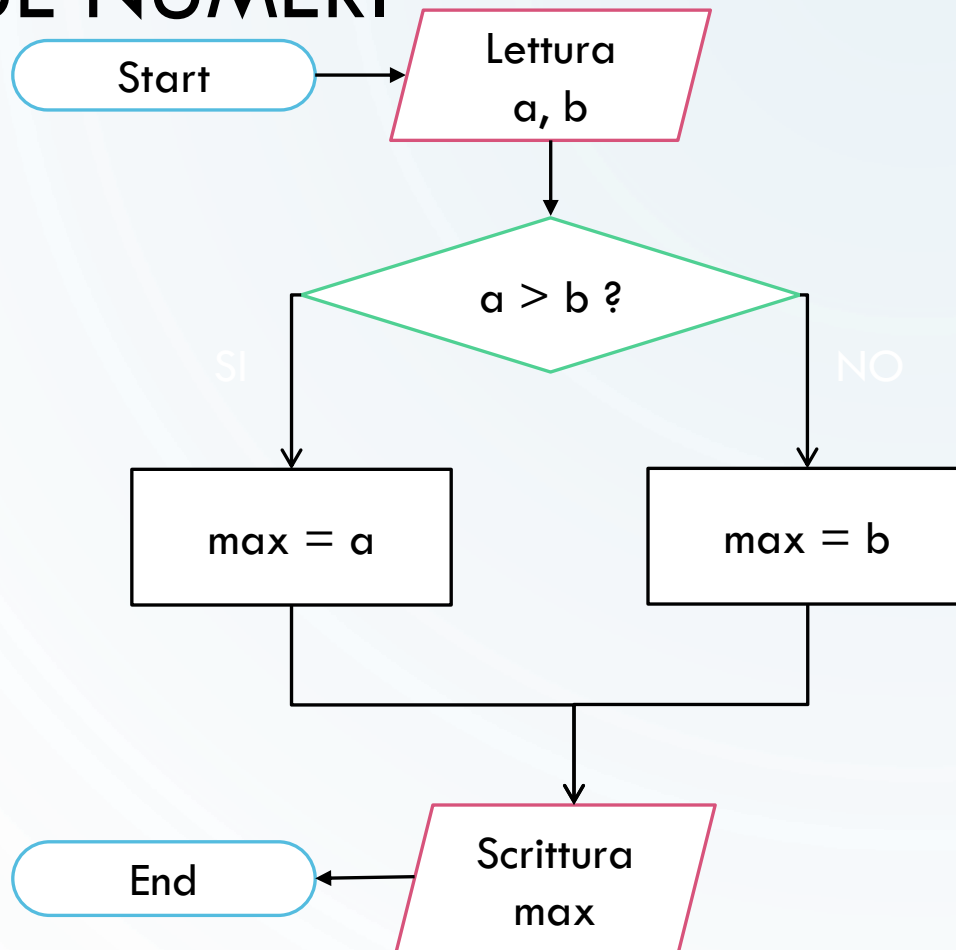
IF ... ELSE

- Con `if` ... `else` si effettua la scelta tra due blocchi di istruzioni
- La condizione è una espressione di tipo logico
- Il ramo `else` può non esistere



```
if (condizione)
{
    S1;
}
else
{
    S2;
}
```

ESEMPIO: CODIFICA IN C/C++ DEL MASSIMO TRA DUE NUMERI



```
/* Massimo tra 2 numeri interi */
#include<iostream>
using namespace std;

int main()
{
    int a,b,max;

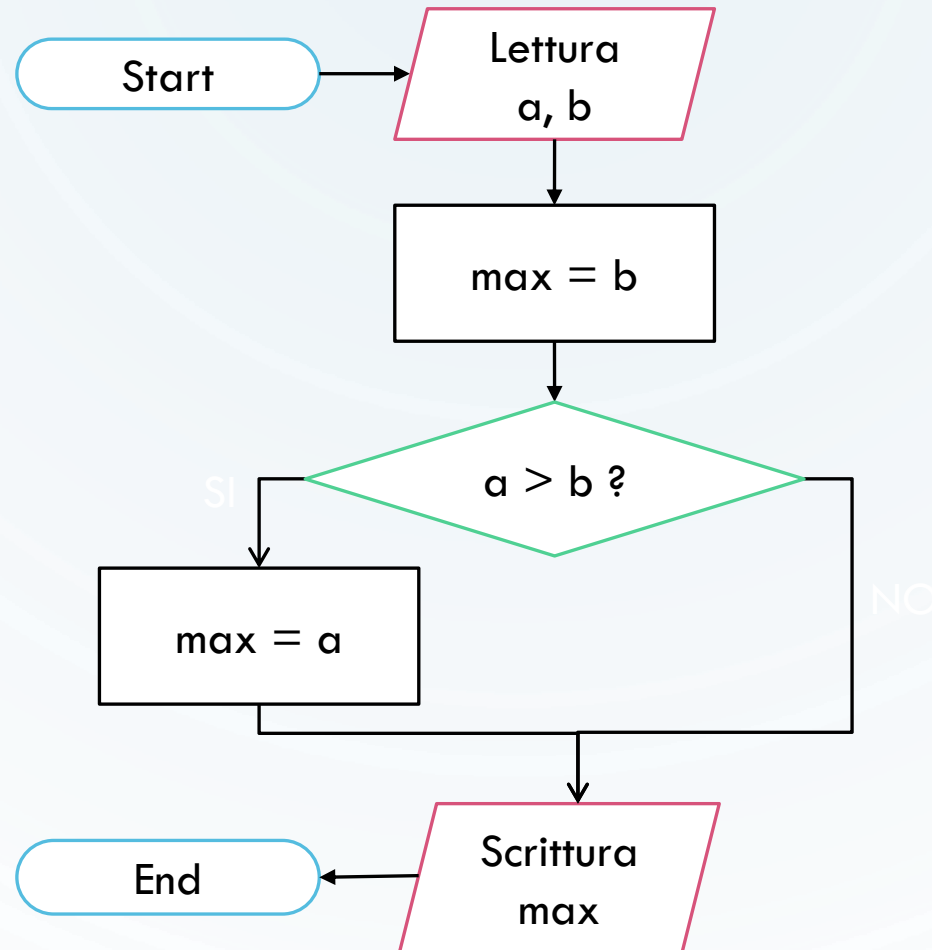
    cout<<"Questo programma permette di calcolare il massimo tra 2 numeri interi.\n\n";
    cout<<"Inserisci il primo numero: ";
    cin>>a;
    cout<<"Inserisci il secondo numero: ";
    cin>>b;

    if(a>b)
    {
        max=a;
    }
    else
    {
        max=b;
    }

    cout<<"Il massimo e': "<<max;

    cout<<endl<<endl;
    system("pause");
}
```

ESERCIZIO: CODIFICARE IN C/C++ IL MASSIMO TRA DUE NUMERI (VARIANTE)



ESERCIZIO: CODIFICARE IN C/C++ IL MASSIMO TRA DUE NUMERI (VARIANTE)

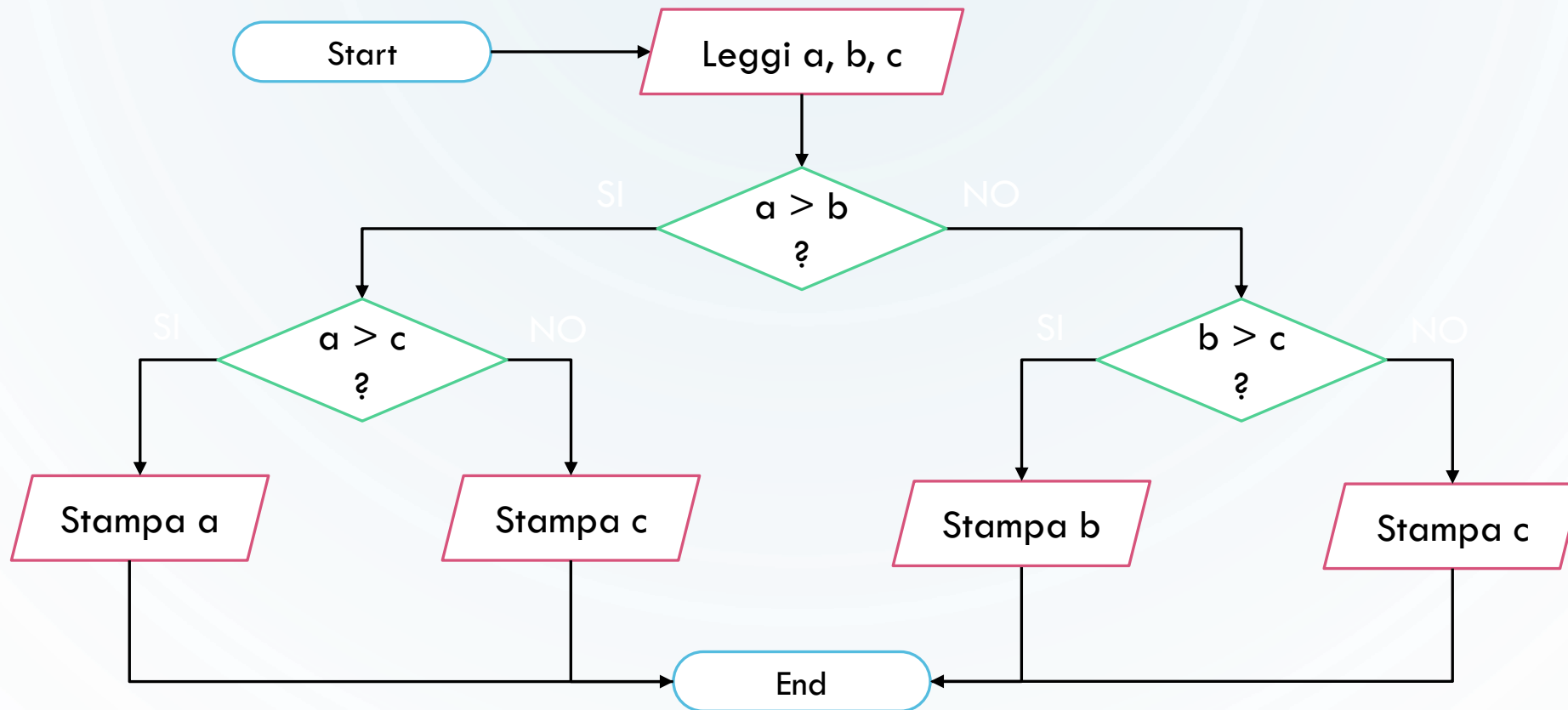
```
/* Massimo tra 2 numeri interi, variante */
#include<iostream>
using namespace std;

int main()
{
    int a,b,max;

    cout<<"Questo programma permette di calcolare il massimo tra 2 numeri interi.\n\n";
    cout<<"Inserisci il primo numero: ";
    cin>>a;
    cout<<"Inserisci il secondo numero: ";
    cin>>b;
    max=a;
    if(b>a)
    {
        max=b;
    }
    cout<<"Il massimo e': "<<max;

    cout<<endl<<endl;
    system("pause");
}
```

ESERCIZIO: CODIFICARE IN C/C++ IL MASSIMO TRA TRE NUMERI



ESERCIZIO: CODIFICARE IN C/C++ IL MASSIMO TRA TRE NUMERI

```
/* Massimo tra 3 numeri interi */
#include<iostream>
using namespace std;

int main()
{
    int a,b,c;

    cout<<"Questo programma permette di calcolare il massimo tra 3 numeri interi.\n\n";
    cout<<"Inserisci il primo numero: ";
    cin>>a;
    cout<<"Inserisci il secondo numero: ";
    cin>>b;
    cout<<"Inserisci il terzo numero: ";
    cin>>c;
    if(a>b)
    {
        if(a>c)
        {
            cout<<"Il massimo e' "<<a;
        }
        else
        {
            cout<<"Il massimo e' "<<c;
        }
    }
    else
    {
        if(b>c)
        {
            cout<<"Il massimo e' "<<b;
        }
        else
        {
            cout<<"Il massimo e' "<<c;
        }
    }

    cout<<endl<<endl;
    system("pause");
}
```

IF MULTIPLI

- In generale, un'istruzione `if` può prevedere `zero`, `uno`, o `più` rami `else` a seconda dei rami di esecuzione possibili tra cui si vuole scegliere
 - se in un ramo di esecuzione bisogna eseguire più di un'istruzione, esse devono essere racchiuse all'interno di un blocco `{ }`

```
if(espressione)
    istruzione
else if(espressione)
    istruzione
...
else
    istruzione
```

ESERCIZIO

- Scrivere un programma per la conversione delle lettere da minuscolo a maiuscolo, e viceversa
 - richiedere all'utente di inserire una lettera
 - controllare che la lettera inserita sia minuscola, o maiuscola
- Nota
 - convertire in maiuscolo una lettera minuscola
 - sottrarre 32 alla lettera da convertire
 - convertire in minuscolo una lettera maiuscola
 - aggiungere 32 alla lettera da convertire

SOLUZIONE:

CONVERSIONE TRA MINUSCOLO E MAIUSCOLO

```
/* Conversione da lettera minuscola a maiuscola e viceversa */  
  
#include<iostream>  
using namespace std;  
  
int main()  
{  
    char c;  
  
    cout<<"Questo programma permette di trasformare una lettera maiuscola in minuscola e viceversa.\n";  
    cout<<"Inserisci carattere: ";  
    cin>>c;  
    if(c>=65 && c<=90)  
    {  
        c+=32;  
        cout<<"Carattere trasformato: "<<c;  
    }  
    else if(c>=97&&c<=122)  
    {  
        c-=32;  
        cout<<"Carattere trasformato: "<<c;  
    }  
    else  
        cout<<"Il carattere inserito non e' una lettera."  
  
    cout<<endl<<endl;  
    system("pause");  
}
```

NOTA:

Se un blocco si compone di una sola istruzione, le parentesi graffe possono essere omesse

IF MULTIPLI

- In alcuni casi, l'istruzione `if` può diventare complicata e lunga da scrivere
- Ad esempio, consideriamo un programma che legga un numero intero $n \in [1, 7]$, e stampi a video il giorno della settimana corrispondente
- Come lo impostereste utilizzando solo strutture if-else?

IF MULTIPLI

```
/* Questo programma scrive il giorno della settimana corrispondente ad un numero intero */
#include<iostream>
using namespace std;

int main()
{
    int n;


    cout<<"Inserisci un numero intero tra 1 e 7: ";
    cin>>n;
    if(n==1)
        cout<<"Giorno corrispondente: Lunedì";
    else if(n==2)
        cout<<"Giorno corrispondente: Martedì";
    else if(n==3)
        cout<<"Giorno corrispondente: Mercoledì";
    else if(n==4)
        cout<<"Giorno corrispondente: Giovedì";
    else if(n==5)
        cout<<"Giorno corrispondente: Venerdì";
    else if(n==6)
        cout<<"Giorno corrispondente: Sabato";
    else if(n==7)
        cout<<"Giorno corrispondente: Domenica";
    else
        cout<<"Il numero inserito non e' compreso tra 1 e 7";

    cout<<endl<<endl;
    system("pause");
}
```

SWITCH

- Con il costrutto `switch` si sceglie l'istruzione di inizio della esecuzione in una sequenza di istruzioni
- La scelta avviene:
 - Calcolando il valore dell'espressione *selettore*
 - Confrontando tale valore con una serie di valori costanti indicati con la parola chiave `case`
- Può comprendere una condizione finale di `default`
 - viene eseguita quando il valore del selettore è diverso dalle costanti riportate nelle frasi `case`

```
switch (selettore)
{
    case cost1: S1;
    case cost2: S2;
    ...
    case costn: Sn;
    default: Sfinale;
}
```



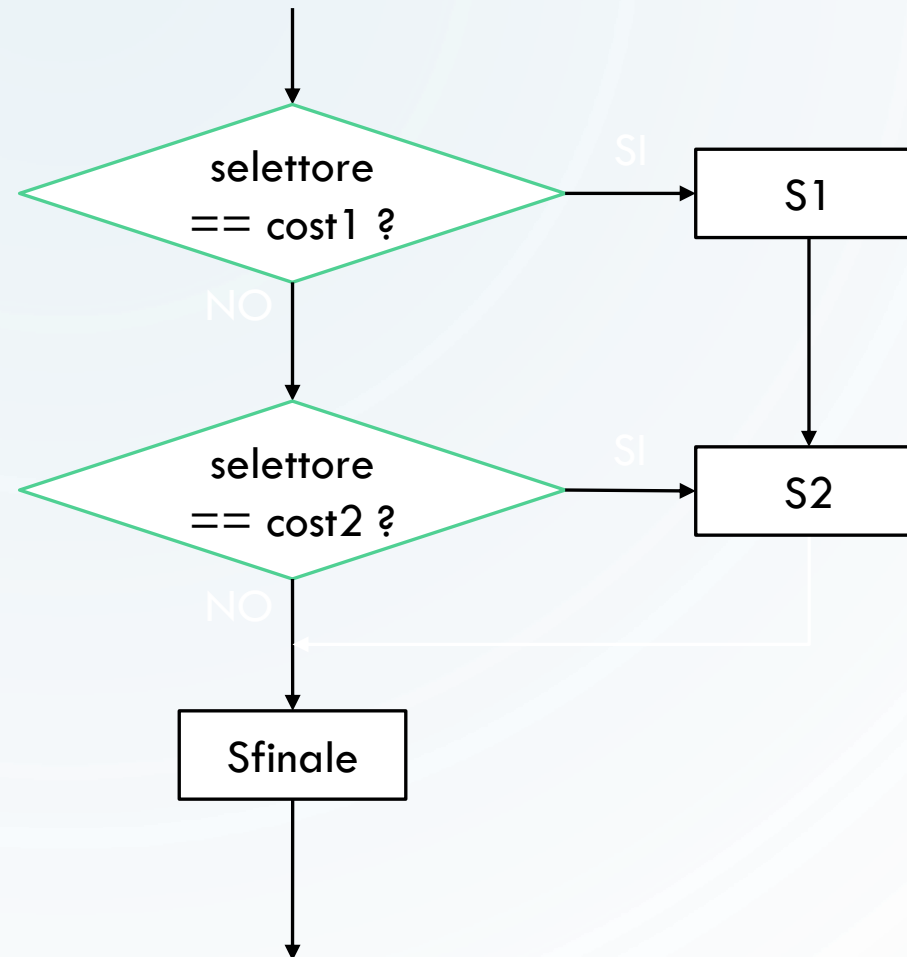
SWITCH

- La struttura `switch` diventa un selettore di blocchi se
 - tutte le sequenze etichettate con le frasi `case` si concludono con l'istruzione `break`
 - che produce come effetto l'uscita dallo switch

```
switch (selettore)
{
    case cost1: S1;
                break;
    case cost2: S2;
                break;
    ...
    case costn: Sn;
                break;
    default:    Sfinale;
}
```

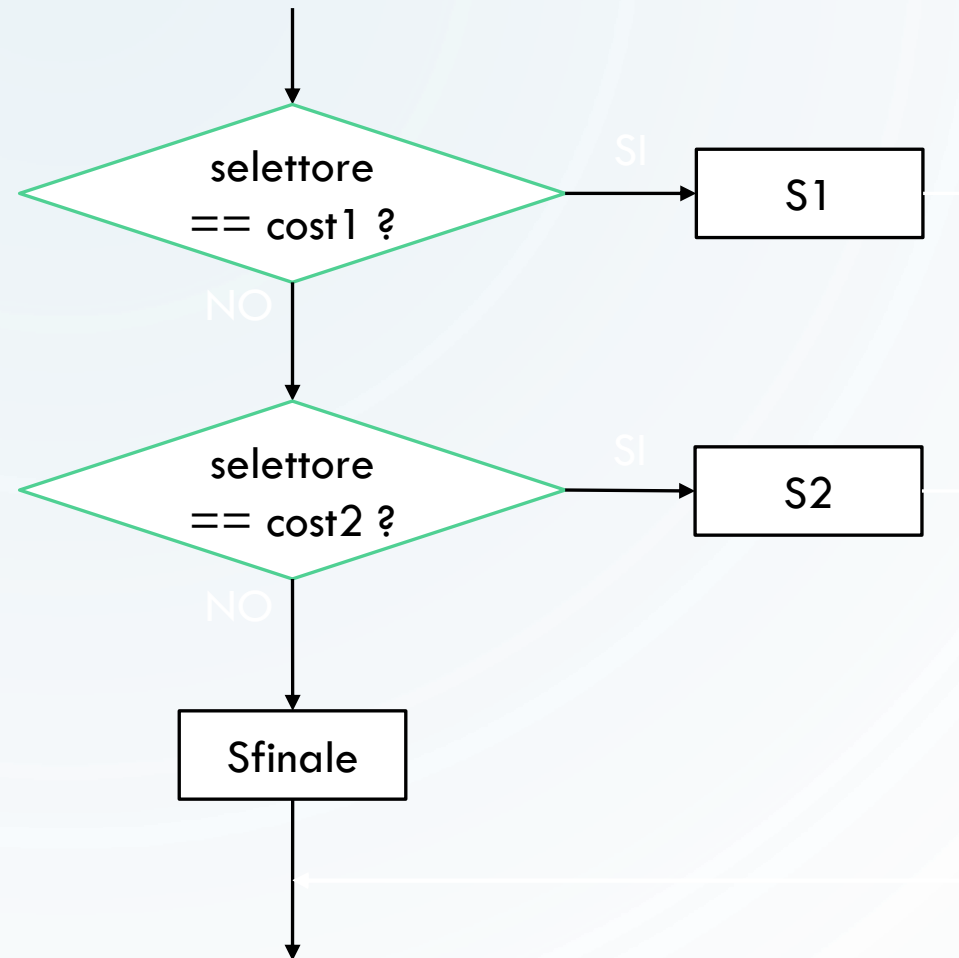

SWITCH

```
switch (selettore)
{
    case cost1: S1;
                break;
    case cost2: S2;
                break;
    default:    Sfinale;
}
```



SWITCH

```
switch (selettore)
{
    case cost1: S1;
                break;
    case cost2: S2;
                break;
    default:    Sfinale;
}
```



ESERCIZIO

- Scrivere un programma che legga un numero intero $n \in [1, 7]$, e stampi a video il giorno della settimana corrispondente
 - Utilizzando la struttura switch

ESERCIZIO

```
/* Questo programma scrive il giorno della settimana corrispondente ad un numero intero */
#include<iostream>
using namespace std;

int main()
{
    int n;

    cout<<"Inserisci un numero intero tra 1 e 7: ";
    cin>>n;
    switch(n)
    {
        case 1:
            cout<<"Giorno corrispondente: Lunedì";
            break;
        case 2:
            cout<<"Giorno corrispondente: Martedì";
            break;
        case 3:
            cout<<"Giorno corrispondente: Mercoledì";
            break;
        case 4:
            cout<<"Giorno corrispondente: Giovedì";
            break;
        case 5:
            cout<<"Giorno corrispondente: Venerdì";
            break;
        case 6:
            cout<<"Giorno corrispondente: Sabato";
            break;
        case 7:
            cout<<"Giorno corrispondente: Domenica";
            break;
        default:
            cout<<"Il numero inserito non e' compreso tra 1 e 7";
    }

    cout<<endl<<endl;
    system("pause");
}
```

A decorative graphic on the left side of the slide, consisting of a network of thin, light-blue lines and small circles, resembling a circuit board or a neural network structure.

STRUTTURE DI CONTROLLO ITERATIVE

WHILE

- Impone che l'esecuzione del blocco di istruzioni sia ripetuta fino a quando la condizione non diventa FALSE

Struttura iterativa
A CONTROLLO INIZIALE

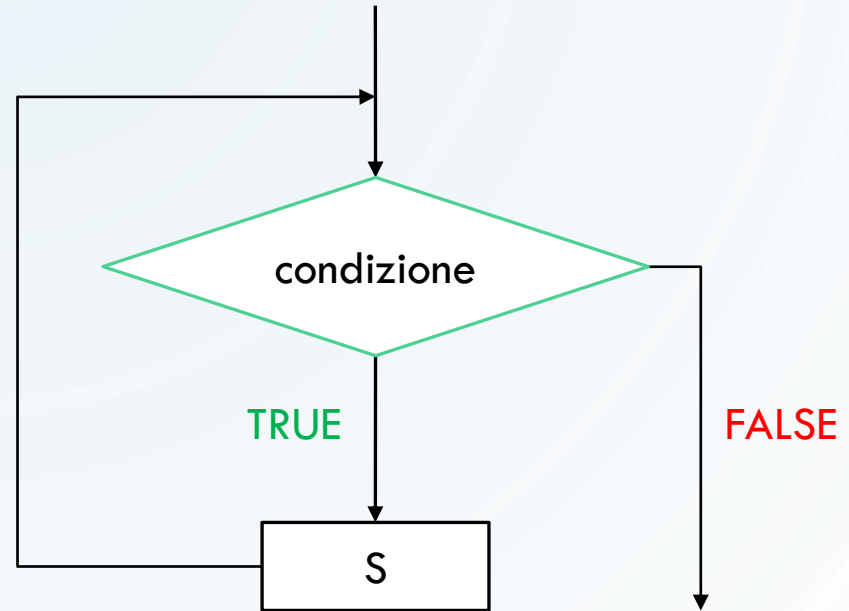
Numero di esecuzioni:
 $[0, n]$

- Viene calcolata la *condizione*
 - Se **FALSE** la sequenza S non viene eseguita
 - Se **TRUE** si esegue S
 - al suo termine si ricalcola la condizione e si riesegue S se è ancora **TRUE**
 - Si continua fino a che la condizione non diventa **FALSE**

```
while (condizione)
{
    S;
}
```

WHILE

```
while (condizione)
{
    S;
}
```



DO WHILE

- Come il while, ma la condizione viene scritta dopo la sequenza S
 - L'esecuzione del blocco avviene almeno una volta
- Si esegue la sequenza S
- Si calcola la *condizione*
 - Se **TRUE** si riesegue S
 - Si continua fino a che la condizione non diventa **FALSE**

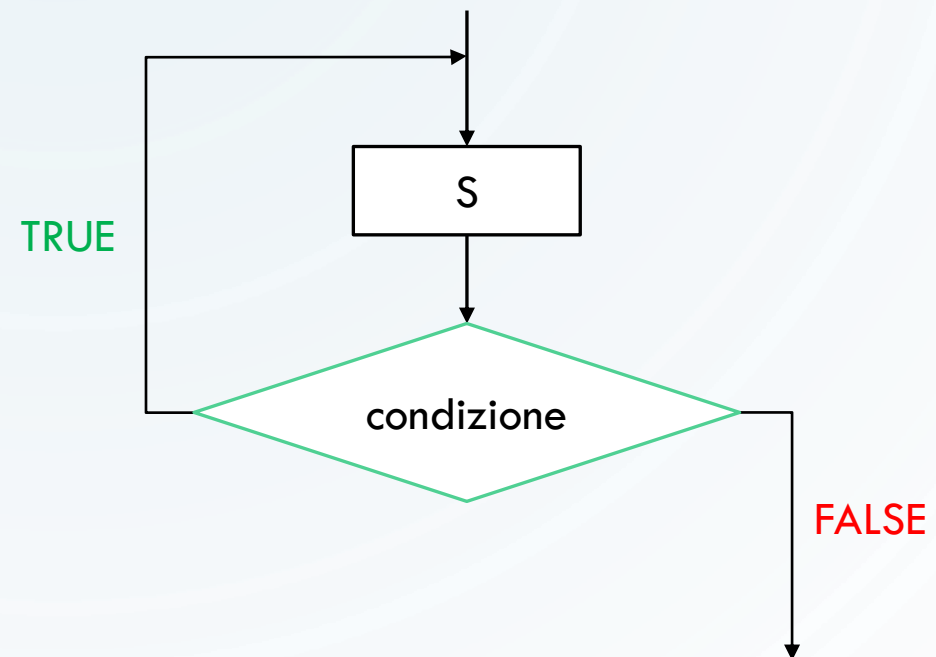
Struttura iterativa
A CONTROLLO FINALE

Numero di esecuzioni:
 $[1, n]$

```
do
{
    S;
}
while (condizione)
```



DO WHILE

```
do  
{  
    S;  
}  
while (condizione)
```





ESERCIZI

- Trovare il minimo tra 10 numeri
 - Utilizzando un ciclo while
 - Trovare il fattoriale di un numero
 - Utilizzando un ciclo while
- 



ESERCIZI

```
/* Questo programma permette di calcolare il minimo tra 10 numeri */
#include<iostream>
using namespace std;

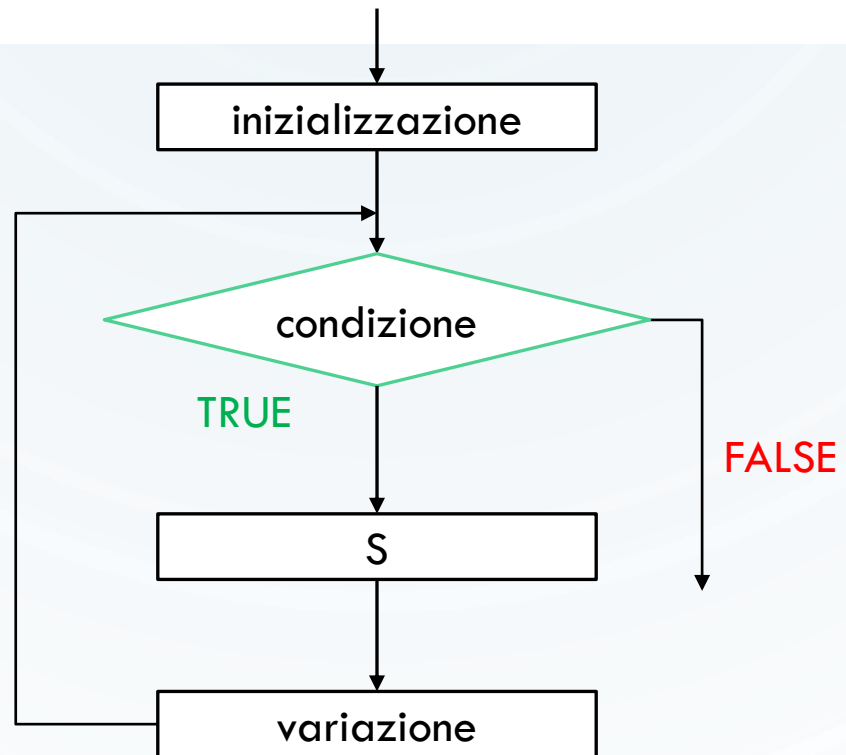
int main()
{
    const int n=10;
    float r,min;
    int i;

    cout<<"Inserisci un numero: ";
    cin>>r;
    min=r;
    i=2;
    while(i<=10)
    {
        cout<<"Inserisci un numero: ";
        cin>>r;
        if(r<min)
            min=r;
        i++;
    }
    cout<<"Il minimo e': "<<min;

    cout<<endl<<endl;
    system("pause");
}
```

FOR

```
for (inizializzazioni; condizione; variazioni)
{
    S;
}
```



FOR

- Il ciclo `for` prescrive
 - L'esecuzione delle istruzioni di **inizializzazione**
 - Il calcolo della **condizione**
 - L'esecuzione della sequenza S se è vera la condizione
 - In caso contrario l'esecuzione termina
 - L'esecuzione delle istruzioni di **variazione** al termine di S
 - La rivalutazione della condizione con il ripetersi dei passi precedenti fino alla determinazione della falsità della **condizione**

```
for (inizializzazioni; condizione; variazioni)
{
    S;
}
```

FOR


- Le istruzioni di **inizializzazione** e di **variazione** non sono obbligatorie
 - mentre la **condizione** è obbligatoria
- Se inizializzazione e variazione non vengono espresse, i cicli `for` e `while` sono equivalenti

```
for (inizializzazioni; condizione; variazioni)  
{  
    S;  
}
```

- In generale, i cicli `for` vengono utilizzati quando il numero delle operazioni richieste è ben definito



ESERCIZI

- Trovare il massimo tra 10 numeri
 - Utilizzando un ciclo for
 - Trovare il fattoriale di un numero
 - Utilizzando un ciclo for
- 



STRUTTURE DI CONTROLLO IN SEQUENZA / INNESTATE

- Struttura di controllo **innestata**
 - Contenuta totalmente in un'altra
- Struttura di controllo **in sequenza**
 - Il suo punto di ingresso è collegato al punto di uscita di un'altra struttura

```
if (condizioneA)
{
    if (condizioneB)
        istruzione1;
    else
        istruzione2;
}
else
    istruzione3;
```

```
if (condizioneA)
    istruzione1;
else
    istruzione2;
    if (condizioneB)
        istruzione3;
    else
        istruzione4;
```


A decorative graphic on the left side of the slide, consisting of a network of thin, light-blue lines and small circles, resembling a circuit board or a neural network diagram. The lines are vertical and horizontal, with some diagonal connections, and the circles are placed at various points along these lines.

ISTRUZIONI NON STRUTTURATE

GOTO / BREAK / CONTINUE

- Le istruzioni **non strutturate** sono istruzioni di salto che possono violare i principi della programmazione strutturata
- **Sono da evitare**
- `goto`
 - Provoca il trasferimento incondizionato del flusso di controllo del programma all'istruzione identificata dall'etichetta `<label>`

```
goto <label>;
```

```
label: istruzione;
```

GOTO / BREAK / CONTINUE

- Le istruzioni **non strutturate** sono istruzioni di salto che possono violare i principi della programmazione strutturata
- **Sono da evitare**
- **break**
 - Comporta l'uscita da `while`, `do-while`, `switch`, `for`
 - È normalmente usata solo per lo `switch`

GOTO / BREAK / CONTINUE

- Le istruzioni **non strutturate** sono istruzioni di salto che possono violare i principi della programmazione strutturata
- Sono da evitare
- `continue`
 - Può essere usata nel
 - `while` e `do-while`, in cui è equivalente al salto alla verifica della condizione
 - `for`, in cui è equivalente al salto alla variazione delle condizioni del ciclo

ESERCIZI RIEPILOGATIVI

- Utilizzando un ciclo for e un ciclo while:
- Scrivere un programma che trova il fattoriale di un numero n . Il programma chiede all'utente di inserire n , e ripete la richiesta finché l'utente non inserisce un numero intero non negativo.

ESERCIZI RIEPILOGATIVI

- Scrivere un programma che calcola la potenza ad esponente intero di un numero reale positivo, r^k .
- L'utente inserisce r (il programma controlla che sia positivo) ed inserisce k (che può essere positivo, nullo o negativo).
- N.B. non è concesso l'utilizzo della libreria *math*

ESERCIZI RIEPILOGATIVI

- Scrivere un programma che calcola l'integrale definito tra x_{\min} e x_{\max} , della funzione $f(x)=2x^2+4x-1$.
- L'utente inserisce gli estremi
- Si utilizzi la definizione di integrale come somma di aree, con $\Delta x=0.01$:
 - integrale=0
 - $x=x_{\min}+\Delta x/2$
 - Finché $x < x_{\max}$:
 - integrale aumenta di $\Delta x * f(x)$
 - x aumenta di Δx

DOMANDE, DUBBI, PERPLESSITÀ

