Costrutti condizionali e iterazioni

- Permettono di controllare il flusso di esecuzione del programma
- Condizioni decidono quale codice eseguire
- Iterazioni ripetono blocchi di codice
- Fondamentali per automatizzare operazioni

Istruzioni condizionali

- **IF**: esegue codice se la condizione è vera
- **ELSE**: esegue codice alternativo se falsa
- **ELIF / ELSE IF**: condizioni multiple
- Esempio Python:

```
if x > 0:
    print("Positivo")
else:
    print("Non positivo")
```

Cicli

- Permettono di ripetere istruzioni
- **FOR**: ciclo con numero definito di iterazioni
- **WHILE**: ciclo fino a quando la condizione è vera
- Utilizzati per elaborazione di elenchi e calcoli ripetuti

Ciclo FOR

- Itera su range o sequenze
- Esempio Python:

```
for i in range(5):
    print(i)
```

- Utile per contatori e liste
- Ciclo deterministico

Ciclo WHILE

- Continua finché la condizione è vera
- Esempio Python:

```
x = 0
while x < 5:
    print(x)
    x += 1
```

- Utile per condizioni non predeterminate
- Ciclo condizionale

Cicli annidati

- Cicli dentro cicli
- Permettono elaborazioni complesse (matrici, tabelle)
- Necessario attenzione alla complessità
- Strumento potente in algoritmi avanzati

Controllo dei cicli

- **BREAK**: interrompe il ciclo
- **CONTINUE**: salta all’iterazione successiva
- Migliora flessibilità dei programmi
- Utile per condizioni particolari

Importanza dei costrutti

- Permettono programmi intelligenti e dinamici
- Automazione di operazioni complesse
- Base di tutti gli algoritmi pratici
- Fondamentali per sviluppatori