



**Università degli Studi di Palermo**

***Dipartimento di Ingegneria Chimica,  
Gestionale, Informatica, Meccanica***

# **Lezioni di Informatica di Base per la Facoltà di Lettere e Filosofia**

Anno Accademico 2012/20103

Docente: ing. Salvatore Sorce

## **Algoritmi**



## Introduzione

- L'informatica è una tra le discipline scientifiche più giovani e stimolanti
  
- Evoluzione continua:
  - Realtà virtuale
  - Telemedicina
  - Monitoraggio ambientale
  - Produzione multimediale
  - Elaborazione di immagini e suoni
  - Elaborazione / riconoscimento automatico del testo
  - ...
  
- Non sempre intuitivamente si riesce a comprendere gli argomenti che sono oggetto dell'informatica



## Malintesi

➤ Primo malinteso

*L'informatica è lo studio dei calcolatori*

➤ Secondo malinteso:

*L'informatica è lo studio di come scrivere programmi per calcolatori*

➤ Terzo malinteso:

*L'informatica è lo studio degli utilizzi e delle applicazioni dei calcolatori e del software*

➤ Fellows and Parberry, Computing Research News, 1993:

*L'informatica non riguarda i calcolatori, così come l'astronomia con i telescopi, la biologia con i microscopi e la chimica con le provette. La scienza non riguarda i dispositivi: riguarda il modo in cui li utilizziamo e ciò che scopriamo utilizzandoli*



## La definizione di Informatica

- Il concetto centrale nell'informatica è il concetto di *algoritmo*
- Algoritmo (da Abu Ja'far Muhammad ibn-Musa Al-Khowarizmi, 780-850 a.c.):
  - Una procedura per risolvere matematicamente un problema in un numero finito di passi, che spesso comprende ripetizioni di una operazione. In generale: un metodo passo-passo per eseguire un dato compito.
- Gibbs&Tucker, Communications of the ACM, 1986  
Informatica: lo studio degli algoritmi, che comprende:
  - Le loro proprietà formali e matematiche
  - Le loro realizzazioni hardware
  - Le loro realizzazioni linguistiche
  - Le loro applicazioni



## Definizione Formale di Algoritmo

- Un insieme ben ordinato di operazioni non ambigue ed effettivamente calcolabili che, eseguito, produce un risultato e termina in una quantità finita di tempo.
  - Insieme ben ordinato
  - Operazioni non ambigue e calcolabili
  - Produce un risultato
  - Termina in una quantità finita di tempo



## Insieme ben ordinato

- Ordinamento delle operazioni da eseguire chiaro e non ambiguo
- Il controllo deve procedere senza ambiguità da una operazione alla successiva
  - Le operazioni sono elencate come passi numerati
  - In assenza di altra indicazione (operazioni condizionali o iterative) il controllo passa sempre al passo successivo

### Preparazione di una torta di ciliegie

1. Prepara la base
2. Prepara il ripieno di ciliegie
3. Versa il ripieno sulla base
4. Cuoci in forno a 200°C per 45 minuti



## Operazioni non ambigue e calcolabili

- Tutti i passi devono essere chiari per l'agente.
  - I passi 1 e 2 potrebbero essere chiari solo ad un pasticciere professionista.
- Una operazione non-ambigua è detta una operazione ***primitiva***
- Partendo da una prima versione dell'algoritmo, occorre verificare che tutte le operazioni coinvolte siano primitive

### Preparazione di una torta di ciliegie

1. Prepara la base
2. Prepara il ripieno di ciliegie
3. Versa il ripieno sulla base
4. Cuoci in forno a 200°C per 45 minuti



## Operazioni non ambigue e calcolabili

- Per un non professionista, il passo 1 va scomposto in un insieme di sottopassi più semplici

### ➤ Preparazione di una torta di ciliegie

#### 1. Prepara la base

- 1.1. *Prendi 1/3 tazza di farina*
- 1.2. *Setaccia la farina*
- 1.3. *In una terrina, miscela farina, 1/2 tazza di burro e 1/4 tazza di acqua*
- 1.4. *Spiana il composto in due basi di torta da circa 23 cm*

#### 2. Prepara il ripieno di ciliegie

#### 3. Versa il ripieno sulla base

#### 4. Cuoci in forno a 200°C per 45 minuti





## Operazioni non ambigue e calcolabili

- Il passo 1 viene scomposto in un insieme di sottopassi più semplici
- Il passo 2 viene scomposto in una serie di sottopassi più semplici
- Tuttavia, il passo 1.2 potrebbe essere non adatto ad essere eseguito per esempio da un bambino

### ➤ Preparazione di una torta di ciliegie

1. Prepara la base
  - 1.1. Prendi  $\frac{1}{3}$  tazza di farina
  - 1.2. Setaccia la farina
  - 1.3. In una terrina, miscela farina,  $\frac{1}{2}$  tazza di burro e  $\frac{1}{4}$  tazza di acqua
  - 1.4. Spiana il composto in due basi di torta da circa 23 cm
2. Prepara il ripieno di ciliegie
  - 2.1. *Versa in una terrina 100 g di ripieno di ciliegia*
  - 2.2. *Aggiungi un pizzico di noce moscata e cannella*
  - 2.3. *Mescola*
3. Versa il ripieno sulla base
4. Cuoci in forno a 200°C per 45 minuti



## Operazioni non ambigue e calcolabili

- Il passo 1.2 può essere ulteriormente decomposto in operazioni più elementari

### ➤ Preparazione di una torta di ciliegie

1. Prepara la base
  - 1.1. Prendi  $\frac{1}{3}$  tazza di farina
  - 1.2. Setaccia la farina**
    - 1.2.1. *Prendi un setaccio e mettilo su una terrina da due quarti di litro*
    - 1.2.2. *Versa la farina nel setaccio e gira la manovella*
    - 1.2.3. *Lascia cadere la farina nella terrina*
  - 1.3. In una terrina, miscela farina,  $\frac{1}{2}$  tazza di burro e  $\frac{1}{4}$  tazza di acqua
  - 1.4. Spiana il composto in due basi di torta da circa 23 cm
2. Prepara il ripieno di ciliegie
  - 2.1. Versa in una terrina 100 g di ripieno di ciliegia
  - 2.2. Aggiungi un pizzico di noce moscata e cannella
  - 2.3. Mescola
3. Versa il ripieno sulla base
4. Cuoci in forno a 200°C per 45 minuti



## Operazioni non ambigue e calcolabili

- In sintesi, è fondamentale arrivare ad una descrizione dell'algoritmo in primitive eseguibili da un agente di calcolo senza necessità di ulteriori istruzioni.
- Le operazioni devono poi essere effettivamente calcolabili
  - *Stampare la lista di tutti i numeri primi*
  - *Somma 1 al valore corrente di  $x$*

### ➤ Preparazione di una torta di ciliegie

1. Prepara la base
  - 1.1. Prendi  $\frac{1}{3}$  tazza di farina
  - 1.2. Setaccia la farina
    - 1.2.1. Prendi un setaccio e mettilo su una terrina da due quarti di litro
    - 1.2.2. Versa la farina nel setaccio e gira la manovella
    - 1.2.3. Lascia cadere la farina nella terrina
  - 1.3. In una terrina, miscela farina,  $\frac{1}{2}$  tazza di burro e  $\frac{1}{4}$  tazza di acqua
  - 1.4. Spiana il composto in due basi di torta da circa 23 cm
2. Prepara il ripieno di ciliegie
  - 2.1. Versa in una terrina 100 g di ripieno di ciliegia
  - 2.2. Aggiungi un pizzico di noce moscata e cannella
  - 2.3. Mescola
3. Versa il ripieno sulla base
4. Cuoci in forno a  $200^{\circ}\text{C}$  per 45 minuti



## Produce un risultato

- Gli algoritmi risolvono problemi
- Per comprendere se una soluzione algoritmica è corretta, il suo risultato deve potere essere confrontato con quello atteso.
- Se un risultato non è producibile, l'algoritmo deve produrre un messaggio di errore, attivare un allarme, o fornire una approssimazione del risultato corretto

### ➤ **Preparazione di una torta di ciliegie**

1. Prepara la base
2. Prepara il ripieno di ciliegie
3. Versa il ripieno sulla base
4. Cuoci in forno a 200°C per 45 minuti

### ➤ Risultato:

- la torta

### ➤ **Programmazione di un VCR**

### ➤ Risultato:

- il nastro col programma televisivo registrato



## Termina in una quantità finita di tempo

- Il risultato deve essere prodotto dopo l'esecuzione di un numero finito di operazioni
- E' tipico causare cicli infiniti quando la condizione presente nelle operazioni iterative non si verifica mai
- Esempio:
  1. Poni  $n=0$
  2. Ripeti i passi 3 e 4 mentre  $n < 3$
  3. Stampa "Ciao!"
  4.  $n = n - 1$
  5. Fine
- Risultato:
  - Ciao!Ciao!Ciao!Ciao!Ciao!...

- **Fare lo shampoo - I**
  1. Inumidisci i capelli
  2. Insapona
  3. Risciacqua
  4. Ripeti
- Risultato
  - Il cliente si stanca, finisce l'acqua, finisce lo shampoo?
- **Fare lo shampoo - II**
  1. Inumidisci i capelli
  2. Ripeti due volte i passi 3 e 4
  3. Insapona
  4. Risciacqua
  5. Stop. Lo shampoo è fatto



## Termina in una quantità finita di tempo

- Il risultato deve essere prodotto dopo l'esecuzione di un numero finito di operazioni
- E' tipico causare cicli infiniti quando la condizione presente nelle operazioni iterative non si verifica mai
- Esempio:
  1. Poni  $n=0$
  2. Ripeti i passi 3 e 4 mentre  $n < 3$
  3. Stampa "Ciao!"
  4.  $n = n + 1$
  5. Fine
- Risultato:
  - Ciao!Ciao!Ciao!

- **Fare lo shampoo - I**
  1. Inumidisci i capelli
  2. Insapona
  3. Risciacqua
  4. Ripeti
- Risultato
  - Il cliente si stanca, finisce l'acqua, finisce lo shampoo?
- **Fare lo shampoo - II**
  1. Inumidisci i capelli
  2. Ripeti due volte i passi 3 e 4
  3. Insapona
  4. Risciacqua
  5. Stop. Lo shampoo è fatto



## Rappresentazione di un algoritmo

- Linguaggio naturale
- Pseudocodice
- Diagrammi di flusso
- Linguaggio di programmazione formale



## Rappresentazione di un algoritmo

➤ Esempio:

**Somma di due numeri interi  $A$  e  $B$   
composti da  $m$  cifre**

$$A = a_{m-1}a_{m-2}\dots a_1a_0, B = b_{m-1}b_{m-2}\dots b_1b_0$$

$$C = A + B = c_m c_{m-1} c_{m-2} \dots c_1 c_0$$

$$A = \quad a_{m-1} \ a_{m-2} \ \dots \ a_0 \ +$$

$$B = \quad b_{m-1} \ b_{m-2} \ \dots \ b_0 \ =$$

---

$$C = \ c_m \ c_{m-1} \ c_{m-2} \ \dots \ c_0$$





## Rappresentazione di un algoritmo

### ➤ Linguaggio naturale

- Metti "in colonna" in due numeri e poni inizialmente il valore del riporto a 0. Cominciando dalla colonna più a destra (quella delle unità), somma le cifre in colonna più il riporto per ottenere la corrispondente cifra nel risultato. Se il risultato è maggiore di dieci, la cifra del risultato si ottiene sottraendo dieci e ponendo il riporto alla colonna successiva pari a 1 e passa alla colonna successiva, altrimenti passa direttamente alla colonna successiva. Ripeti queste operazioni per tutte le colonne fino all'ultima. Quando sono finite le cifre da sommare poni la cifra più a sinistra del risultato uguale all'ultimo riporto e stampa il risultato finale. Dopo la stampa, l'algoritmo è finito. Fermati.



## Rappresentazione di un algoritmo

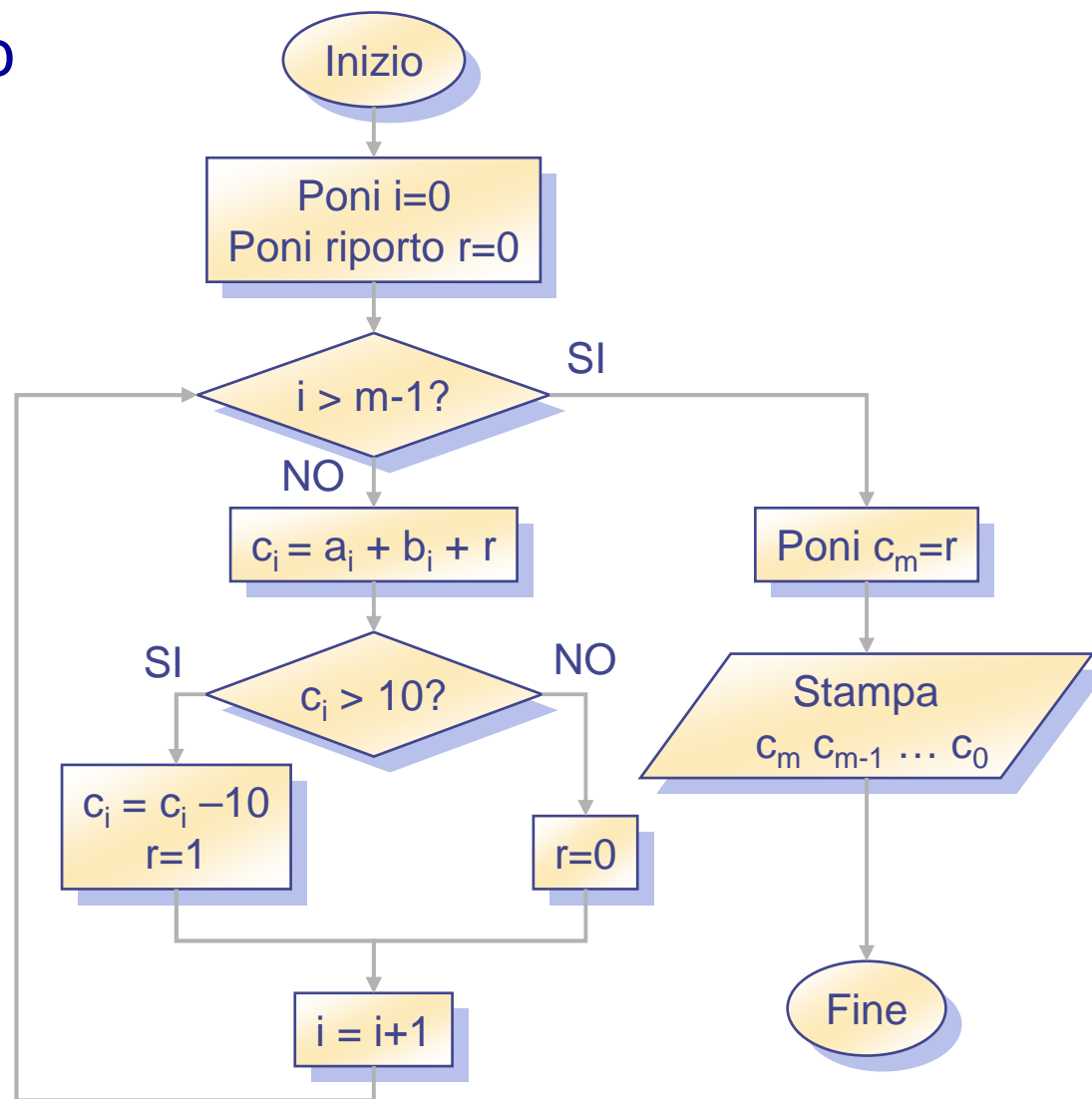
### ➤ Pseudocodice

1. Poni il valore della variabile *riporto* a 0
2. Poni il valore della variabile *i* a 0
3. Ripeti i passi da 4 a 6 fino a che il valore di *i* diventa maggiore di  $m-1$
4. Somma  $a_i$  e  $b_i$  al valore corrente del *riporto* per ottenere  $c_i$
5. Se il valore ottenuto per  $c_i$  è maggiore o uguale di 10 allora calcola il nuovo  $c_i$ , sottraendo 10 dal valore corrente di  $c_i$   
poni *riporto*=1  
altrimenti poni *riporto* uguale a 0
6. Aggiungi 1 ad *i*.
7. Poni la cifra più a sinistra del risultato,  $c_i$ , uguale a *riporto*
8. Stampa il risultato finale,  $c_m c_{m-1} c_{m-2} \dots c_1 c_0$
9. Fermati.



## Rappresentazione di un algoritmo

### ➤ Diagramma di flusso





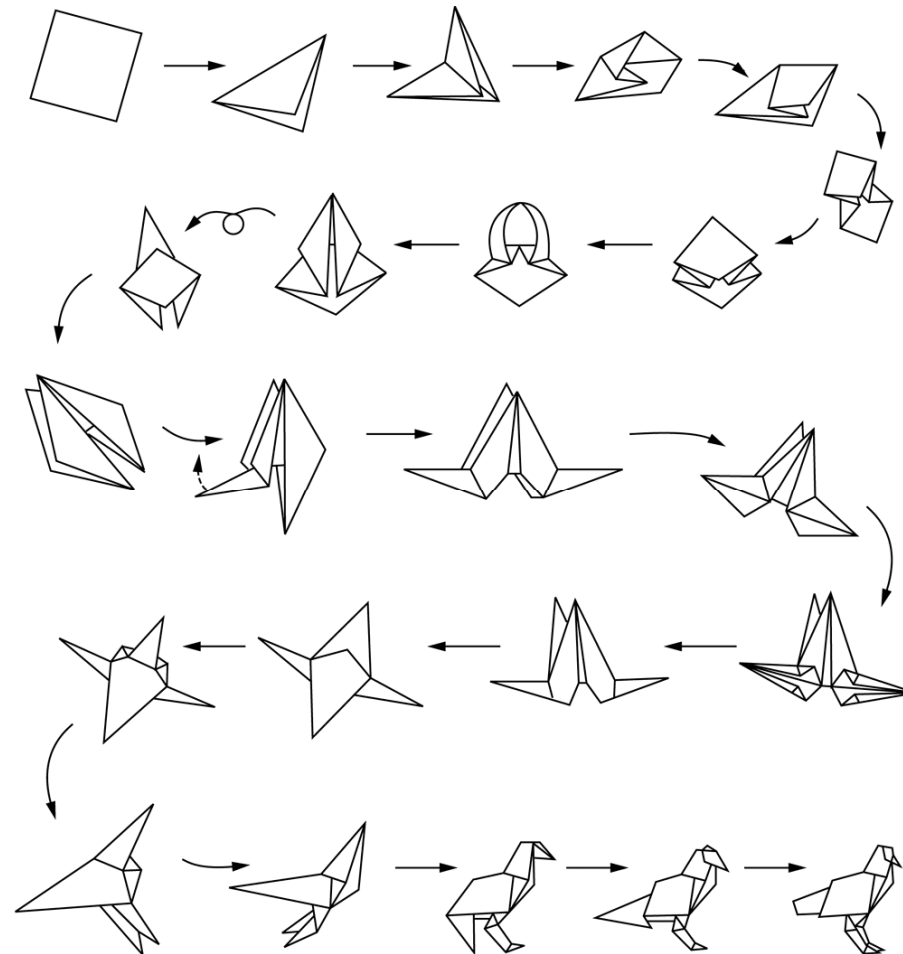
## Rappresentazione di un algoritmo

### ➤ Linguaggio di programmazione formale (C++)

```
1. int a[3], b[3], c[3];
2. int m, i, riporto;
3. riporto=0;
4. i=0;
5. while(i<m) {
6.     c[i]=a[i]+b[i]+riporto;
7.     if(c[i]>10)
8.         {c[i]-=10;
9.           r=1;}
10.    else r=0;
11.    i++;
12. }
13. c[m]=r;
14. printf("%d %d %d\n", c[2],c[1],c[0]);
```



# Un esempio di algoritmo – L'origami





## Il flusso di esecuzione

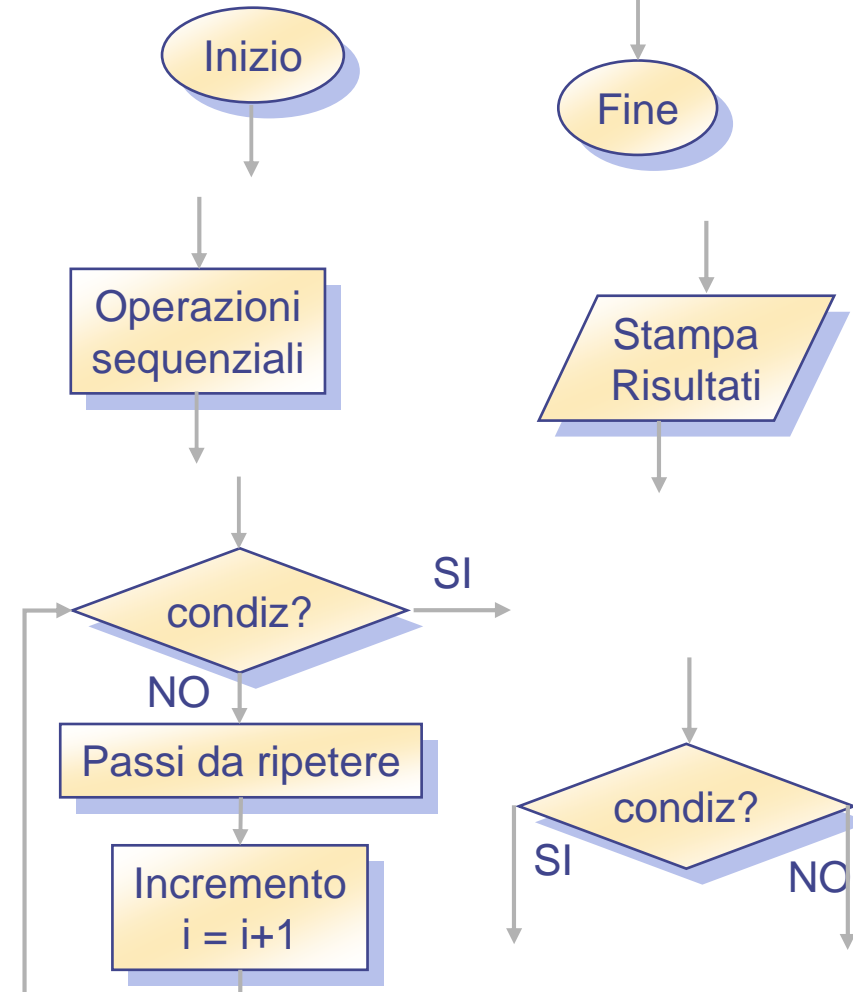
Il flusso di esecuzione procede da un'istruzione a quella successiva fino a quando non viene raggiunta la fine del programma



# Algoritmi e diagrammi di flusso

- **Pseudocodice**
- Operazioni sequenziali
  - Realizzano un solo compito ben definito
  - Il controllo passa all'operazione successiva quando il compito è finito
  - Frase dichiarativa
  - Operazioni di I/O
- Operazioni condizionali
  - Selezionano l'operazione successiva sulla base di una domanda
- Operazioni iterative
  - Eseguono un ciclo di istruzioni fino a quando la condizione di uscita è verificata

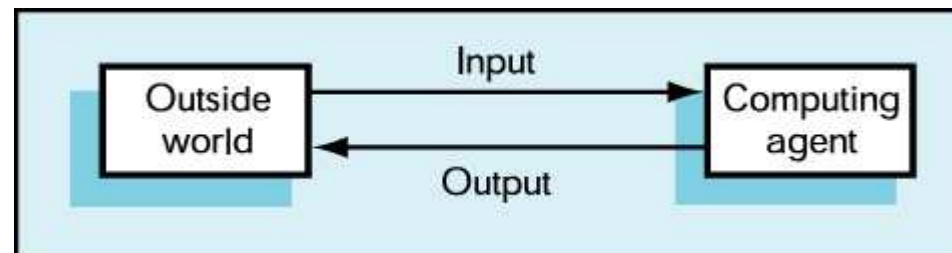
- **Diagrammi di flusso**





## Operazioni sequenziali

- Elaborazione, Ingresso, Uscita
  - Operazioni di ingresso
    - ◆ Es.: acquisisci il valore per “variabile”
  - Operazioni di uscita
    - ◆ Es.: stampa il valore per “variabile”, descrizione della variabile
  - Operazioni di Elaborazione
    - ◆ Es.: eseguire un calcolo per il valore di “variabile”

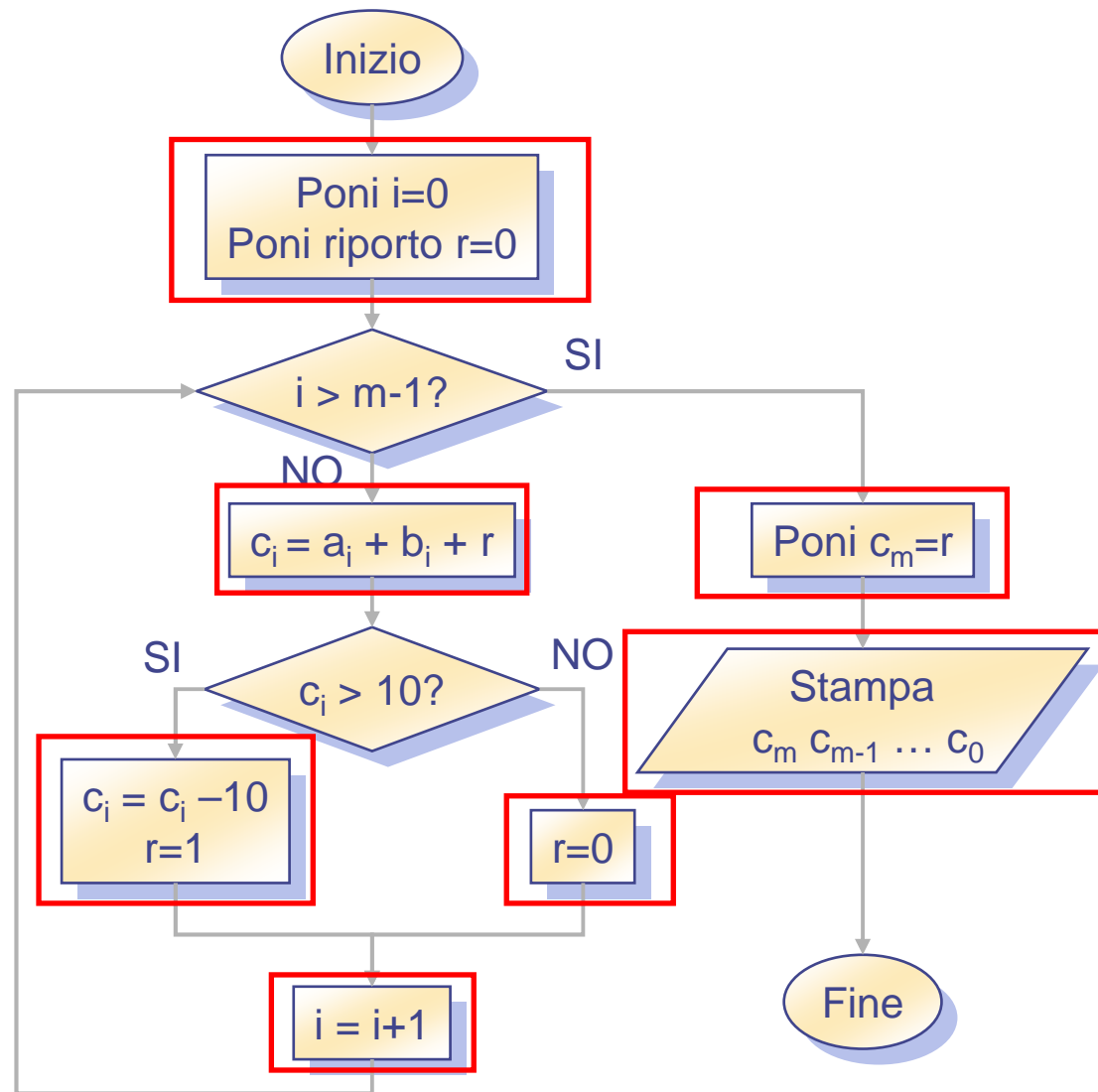






## Rappresentazione di un algoritmo

Operazioni sequenziali





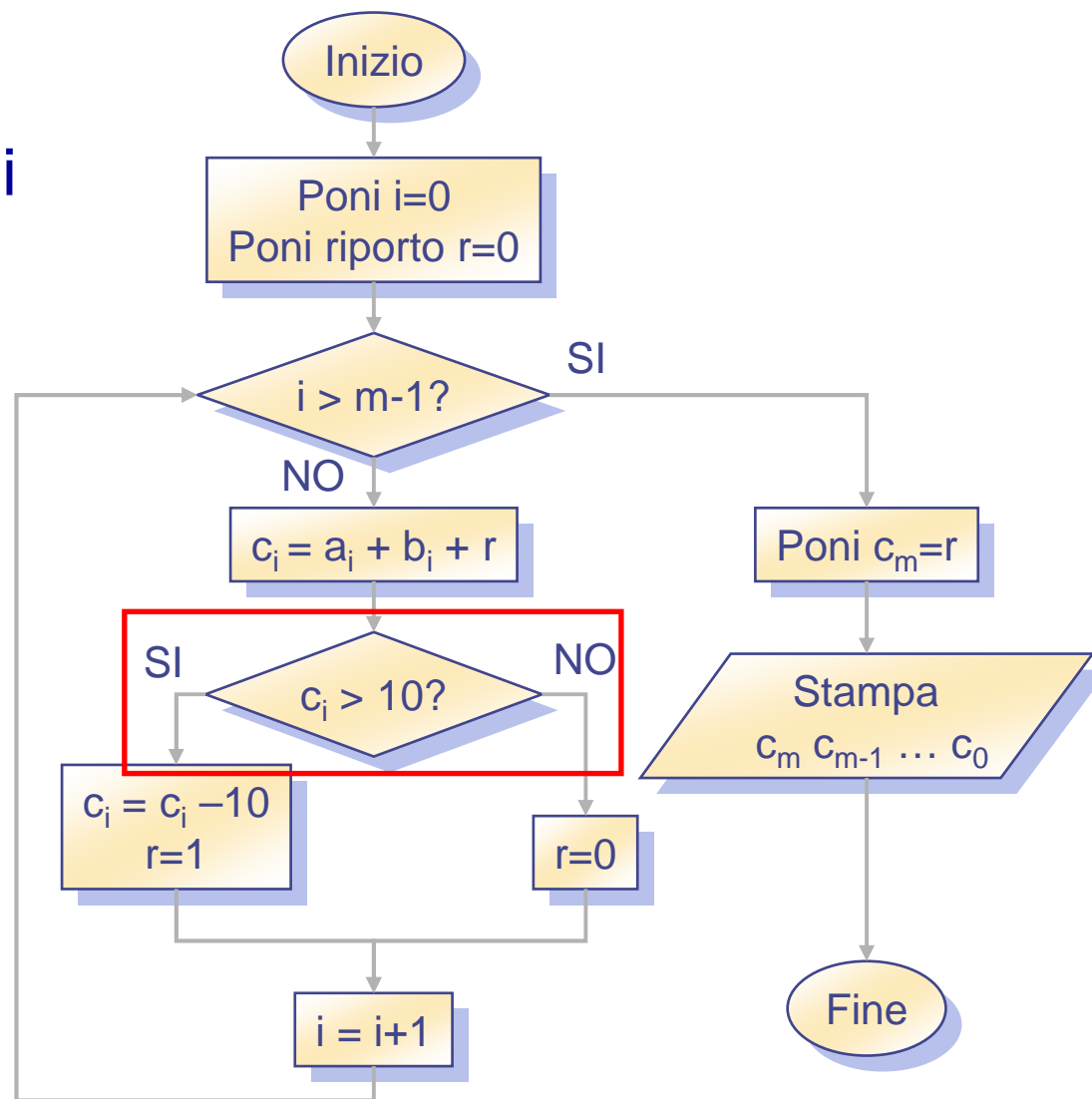
## Operazioni condizionali

- Se “condizione” è vera/falsa allora
    - Prima serie di istruzioni
  - Altrimenti
    - Seconda serie di istruzioni
- 
- Alterano il flusso di esecuzione dell’algoritmo



## Rappresentazione di un algoritmo

Operazioni condizionali





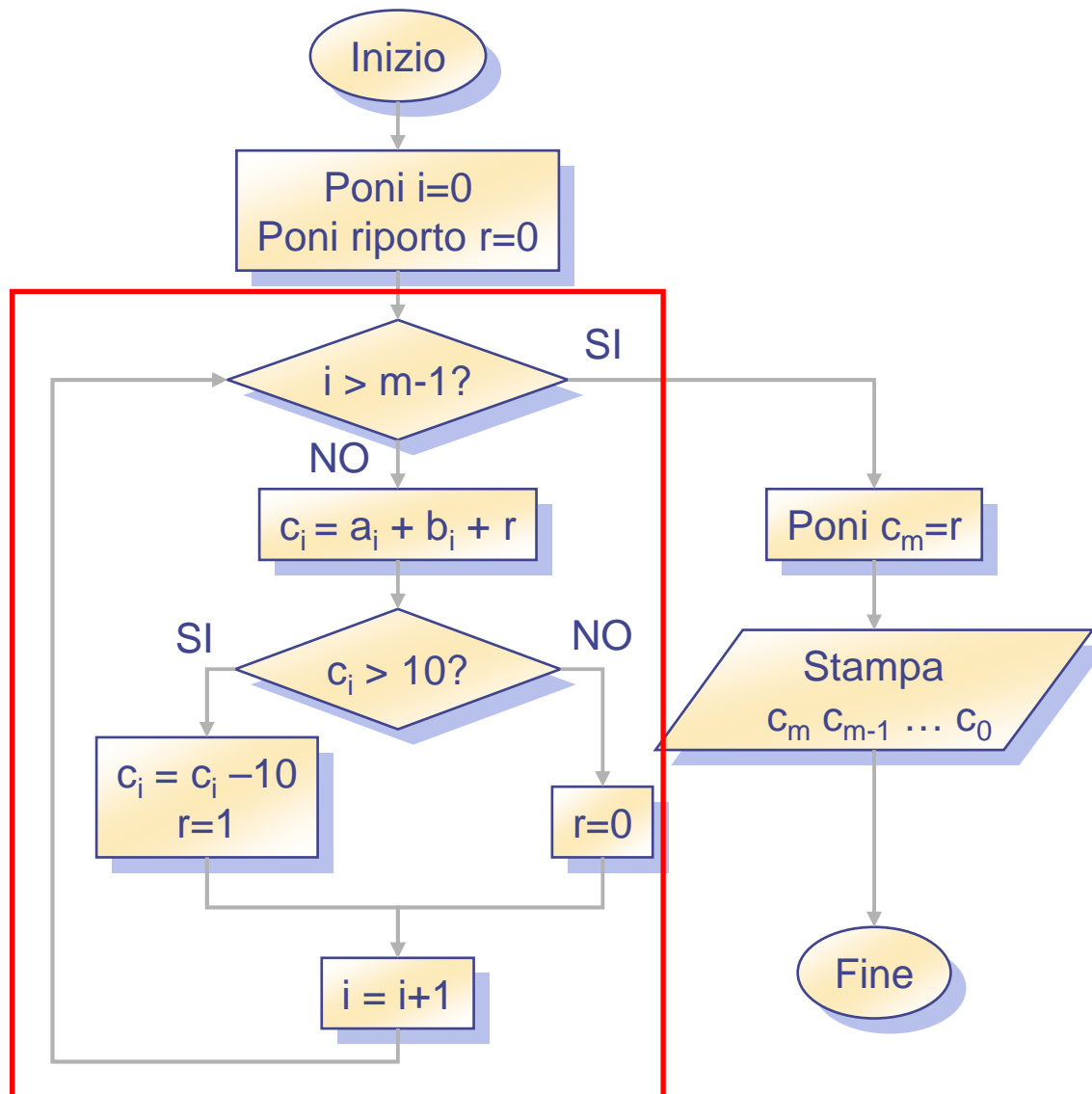
## Operazioni iterative

- Ciclo a condizione iniziale:
  - Finchè "condizione" è vera/falsa ripeti:
    - ◆ Istruzione 1
    - ◆ Istruzione 2
    - ◆ ...
    - ◆ Istruzione N
  
- Ciclo a condizione finale:
  - Ripeti:
    - ◆ Istruzione 1
    - ◆ Istruzione 2
    - ◆ ...
    - ◆ Istruzione N
  - Finchè "condizione" è vera/falsa



## Rappresentazione di un algoritmo

Operazioni iterative





## Attributi di un algoritmo

- Correttezza
  - Un algoritmo non deve soltanto produrre un risultato, ma deve produrre un risultato corretto
  - Un algoritmo deve produrre un risultato utile
- Facilità di comprensione
  - Necessità di adattare un algoritmo per una varietà di scenari possibili.
  - Importante per garantire la manutenibilità dei programmi
- Eleganza
  - Spesso in antitesi con facilità di comprensione
  - Es.: Somma dei primi 100 numeri



## Attributi di un algoritmo - II

- **Efficienza**
  - Tempo di calcolo e spazio in memoria sono quantità limitate
  - L'efficienza di un algoritmo misura la sua capacità di utilizzare bene le risorse del calcolatore sui cui gira, in termini di tempo di calcolo e memoria utilizzata
- **Efficienza nell'uso dello spazio**
  - Quantità di informazioni da memorizzare per svolgere il compito in aggiunta ai dati di ingresso
  - Tanto più inefficiente quanto più memoria aggiuntiva è richiesta
- **Efficienza nell'uso del tempo di calcolo**
  - Benchmarking: fissare i parametri della misura, ovvero l'insieme dei dati di ingresso, la macchina specifica, il particolare profilo di uso dell'algoritmo



## Attributi di un algoritmo - III

- Efficienza nell'uso del tempo di calcolo
  - Indicazione della **quantità di lavoro** richiesta dalla natura dell'algoritmo
  - Tale quantità di lavoro dipende dal numero di passi richiesti per eseguire il compito
  - Il confronto tra due algoritmi va operato sulla base del numero di passi e non del tempo di esecuzione su una particolare macchina.





# Domande?

