

Indice

INTRODUZIONE	7
RICHIAMI	9
1) Dal problema al programma e simboli utilizzati nei flowchart	9
2) Un po' di matematica.....	11
3) Date.....	17
4) Il linguaggio JAVA.....	18
5) Il linguaggio PYTHON.....	26
6) ChatGPT	39
SEQUENZA	45
Sequenza: esercizi risolti.....	45
1) Dati in input 3 numeri interi, calcolarne il prodotto.....	46
2) In input un numero intero, stampare i numeri interi precedente e successivo.....	49
3) Dati in input il numero n e il tasso percentuale t , calcolare la percentuale p di n	51
4) Ripartizione diretta: dividere un valore n in tre parti direttamente proporzionali ai tre valori a, b, c	54
5) In input un numero intero, dire di quante centinaia, decine, unità è composto.....	58
6) Determinare il minor numero di banconote da 50, 20, 10 euro necessario per formare un valore.....	61
7) Dato un tempo in secondi, calcolare il valore corrispondente espresso in ore.....	64
8) Dato un tempo in ore, minuti e secondi, calcolare il valore corrispondente in secondi.....	67
9) Calcolo di un determinato polinomio per un valore x , numero reale, dato in input (es. $2x^2+3x+1$).....	70
10) Arrotondare un numero reale positivo N alla M -esima cifra decimale.....	72
11) Dati due numeri interi, scambiare i loro valori.....	75
12) Dati tre numeri interi, ruotare il loro valore ($a \rightarrow b \rightarrow c \rightarrow a$).....	78
Sequenza: esercizi da risolvere.....	81
SELEZIONE	83
Selezione: esercizi risolti.....	83
1) Dato in input un numero intero, dire se è positivo o negativo.....	84
2) Dati in input due numeri interi positivi, dire se sono divisibili tra di loro.....	87
3) Dati in input tre numeri reali positivi, verificare se possono essere le misure dei lati di un triangolo.....	91
4) Dati in input due numeri interi, trovare il minore.....	98
5) Dati in input tre numeri interi, stamparli in ordine crescente.....	101
6) Dati in input quattro numeri interi, trovare i due che danno la somma massima.....	109
7) Dati quattro numeri interi, dire se sono in ordine crescente.....	116
8) Dati in input tre numeri interi, stampare quelli con lo stesso segno in maggioranza.....	119
9) Dato un valore da 1 a 12, corrispondente a un mese, scrivere il numero dei giorni del mese.....	124
10) Stabilire se un anno è bisestile. In input anno.....	127
11) Inserita una data, stampare la data del giorno successivo.....	130
12) Inserita una data, stampare il numero d'ordine del giorno nell'anno.....	136
13) Dati in input i coefficienti di un'equazione di II grado, risolvere l'equazione.....	142
Selezione: esercizi da risolvere.....	147
ITERAZIONE	148
Iterazione: esercizi risolti.....	149
1) Input di N numeri interi; per ogni numero inserito calcolare il quadrato e il cubo.....	151
2) Input di N numeri interi; sommare e contare i numeri positivi e i numeri negativi:.....	154
3) Input di una frase. Contare il numero di vocali e il numero di consonanti presenti nella frase.....	159
4) Contare quante volte compaiono due caratteri uguali consecutivi in una sequenza di caratteri.....	163
5) In input una sequenza di N numeri interi; verificare se la sequenza è ordinata in modo crescente.....	167

6) In input N coppie (a, b) di numeri interi positivi; verificare la divisibilità di ogni coppia.	172
7) Sequenza di numeri: minimo e il valore immediatamente superiore al minimo.	178
8) Calcolare la somma di due numeri interi positivi, sommando un'unità alla volta.	185
9) Calcolare il prodotto di due numeri interi positivi o negativi, procedendo per addizioni successive.	188
10) Calcolare la potenza n-esima di un numero, procedendo per moltiplicazioni successive.	193
11) Calcolare tutte le potenze da 1 a P di un numero N.	197
12) Calcolare il fattoriale di un numero N ($N! = 1*2*3* \dots *N$).	201
13) Sommare i numeri compresi tra N1 ed N2 ($N1 \leq N \leq N2$).	204
14) Calcolare i primi K multipli di un numero N.	208
15) Trovare i divisori interi di un numero N.	211
16) Stampare i quadrati e i cubi dei primi N numeri naturali divisibili per 3.	215
17) Stampare i primi N numeri che sono quadrati perfetti.	221
18) Dato N, intero positivo, calcolare il quadrato di N, sommando i primi N numeri dispari.	224
19) Data la successione 1, 2, 4, 8, ..., trovare il primo termine maggiore o uguale del valore N.	228
20) Trovare la radice quadrata intera di un numero N, intero positivo.	231
21) Stampare i numeri di Fibonacci minori di N.	237
22) Calcolare la somma delle cifre di un numero N.	241
23) Calcolare il M.C.D. di due numeri a e b.	245
24) Semplificare una frazione o dire se è irriducibile.	249
25) Dato un numero N, intero positivo, verificare se è un numero triangolare.	254
26) Dato un numero N, intero positivo, verificare se è un numero primo.	258
27) Calcolo di un'espressione aritmetica.	262
Iterazione: esercizi da risolvere.	269

ITERAZIONE CON L'UTILIZZO DEL CICLO FOR 272

Iterazione For: esercizi risolti	272
1) Trovare tutti i numeri naturali I, J, K tali che $I*J*K = I+J+K$	273
2) Stampare i numeri interi negativi tra -1 e un valore dato N negativo:	277
3) Stampare un quadrato di asterischi: N asterischi per ogni lato. In input N.	281
4) Stampare il triangolo di Floyd.	288
Iterazione For: esercizi da risolvere.	291

Introduzione

“**Usus magister est optimus**” (“la pratica è il miglior maestro”, Cicerone).

“**A pratica ruppa a grammatica**” (“la pratica rompe la teoria”, proverbio calabrese).

“**Se ascolto dimentico, se vedo ricordo, se faccio capisco**” (massima attribuita a Confucio).

Con questa raccolta di esercizi vorrei mettere a disposizione di chi si avvicina allo studio dell’informatica del materiale che lo possa aiutare a **fare per capire**.

Gli esercizi sono raggruppati per **argomento: Sequenza, Selezione, ..., File, Database**, e raccolti in quattro parti:

- **I parte: Sequenza, Selezione, Iterazione**
- **II parte: Funzioni, Array**
- **III parte: Liste e altre strutture dati, Classi e oggetti**
- **IV parte: File, Database**

Per ogni **argomento** sono proposti:

- **Esercizi risolti**
- **Esercizi da risolvere**

Spesso, in un esercizio risolto, si trova il rimando a un gruppo di esercizi da risolvere; ad es.

Sequenza: esercizi risolti

1) Dati in input tre numeri, calcolarne il prodotto.

(Esercizi da risolvere: 1-8)

2) ...

Il gruppo di esercizi da risolvere (1-8) è “simile” all’es. 1) risolto, che può essere utilizzato come guida.

Per ogni **esercizio risolto** sono riportati:

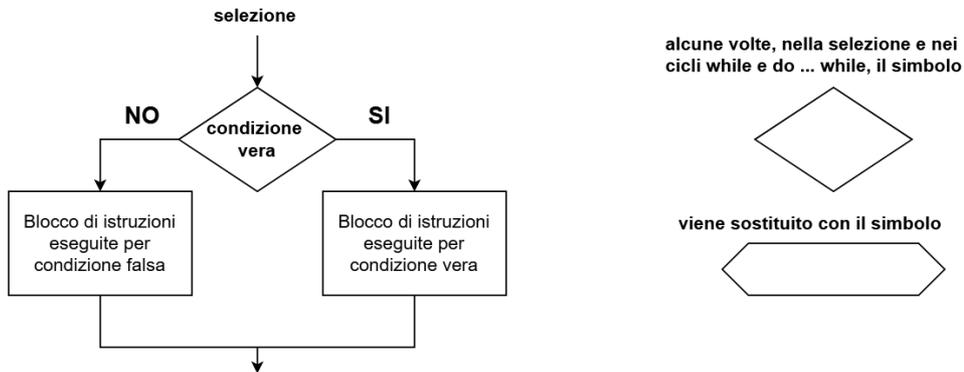
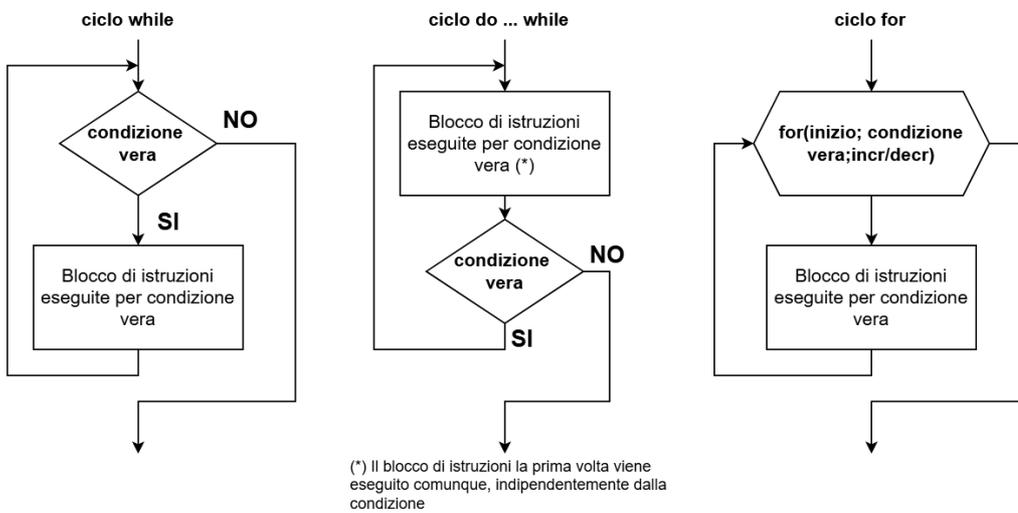
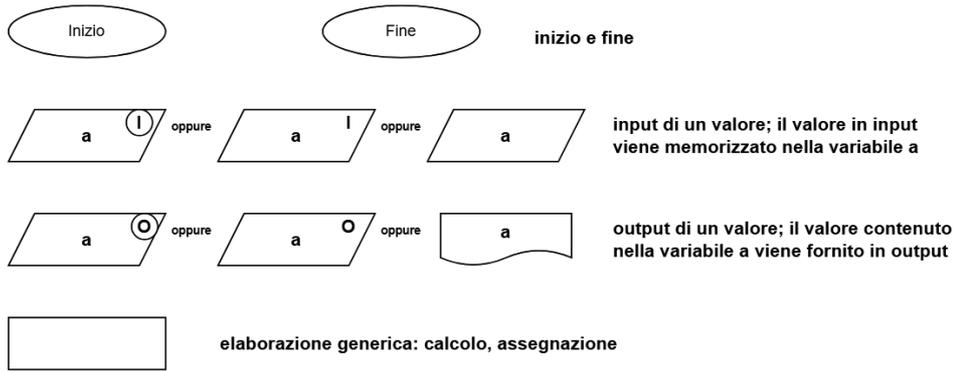
- **Esempio di input e output** (dal programma Python): serve a mettere in luce i dati che bisogna fornire al programma (**input**) e cosa si ottiene (**output**).
- **Variabili**: in una tabella sono elencate le variabili utilizzate nel programma, distinguendo per:
 - **Input**: variabili per l’input dei dati;
 - **Elaborazione**: variabili né di input né di output, usate solo nel corso dell’elaborazione;
 - **Output**: variabili per l’output dei dati; di solito tutto ciò che deve essere mandato in output è memorizzato, in modo opportuno, in una sola variabile di nome *risultato*.

Per ogni **variabile** è indicato il tipo:

int: numeri interi; **double**: numeri reali; **String**: stringhe (insieme di caratteri alfanumerici); **boolean**: valore *True* o *False*.

- **Flowchart** (diagramma di flusso): l'analisi del problema è schematizzata tramite il flowchart. Nel primo esercizio, a titolo esemplificativo, ci sono anche una breve analisi descrittiva del problema e la “**trace-table**”.
- **Codifica Java**: programma in Java.
- **Codifica Python**: programma in Python.
- **ChatGPT**: soluzione data da ChatGPT.

Simboli utilizzati nei flowchart



L'istruzione *for (inizio; condizione vera; incremento/decremento)* ricalca l'istruzione C/Java.

Nei flowchart molto spesso è utilizzata anche l'istruzione *for* Python:

for indice in range (inizio, fine, incremento/decremento)

Ad es.

for i in range(0, n, 1) # equivale all'istruzione Java: **for (i=0; i<n; i++)**

2) Un po' di matematica

- Insiemi numerici

(<https://www.youmath.it/domande-a-risposte/view/6844-insiemi-numerici.html>)

- **Numeri naturali (insieme N):** $\{0, 1, 2, 3, 4, \dots\}$ numeri interi non negativi; numeri che si ottengono partendo da 0 e aggiungendo un'unità.
- **Numeri interi relativi (insieme Z):** $\{\dots, -4, -3, -2, -1, 0, +1, +2, +3, +4, \dots\}$ numeri interi con segno.
- **Numeri razionali relativi (insieme Q):** $\{\dots, -2, -(3/2), \dots, -1, -(1/2), \dots, 0, \dots, +(1/2), 1, \dots, +(3/2), 2, \dots\}$ numeri espressi tramite una frazione preceduti da segno positivo (+), negativo (-) o nullo (solo lo zero); generalizzando, è l'insieme dei valori (a/b) con a e b due qualsiasi numeri interi relativi, con b diverso da 0.
- **Numeri irrazionali (insieme I):** numeri che non possono essere espressi tramite una frazione, ad es. $\sqrt{2}, \pi, -\sqrt{3}, \dots$
- **Numeri reali (insieme R):** è l'insieme formato dall'unione dei numeri razionali relativi e dei numeri irrazionali.
- **Numeri complessi (insieme C):** insieme i cui elementi sono nella forma $a + ib$, con a e b numeri reali e i unità immaginaria ($\sqrt{-1}$).

- Divisione intera

Nella **divisione intera** il **quoziente** è sempre un numero intero, pertanto:

dividendo = (divisore*quoziente) + resto; ad es.

9 **divisione** per 2 -> quoziente 4,5 e resto 0

9 **divisione intera** per 2 -> quoziente 4 e resto 1; $9 = (4*2) + 1$

simbolo per la **divisione intera**

- Visual Basic: \ (barra rovesciata);
- Python: // (doppia barra);
- Java e C: non hanno un simbolo particolare; ma se dividendo e divisore sono interi il risultato della divisione sarà un intero.

- Operatore modulo (%) (simbolo percentuale)

L'operatore **modulo (%)** dà come risultato il **resto** della **divisione intera** tra i suoi operandi, che devono essere di tipo intero.

a e b : due numeri interi, con b diverso da 0

$c = a \% b$: in c il **resto** della **divisione intera** di a per b .

- Numero pari e numero dispari

- n è un **numero pari**: $n \% 2 = 0$
- n è un **numero dispari**: $n \% 2 = 1$

4) Il linguaggio JAVA

(https://www.vallesoftware.it/wp-content/uploads/2021/01/Appunti_JAVA.pdf)

a) Un programma Java

- Disposizione a formato libero; non ci sono regole sugli spazi e le interruzioni di riga: si possono scrivere più istruzioni su una riga, così come si può scrivere ogni parola/simbolo di una istruzione su una riga diversa;
- ogni istruzione deve terminare con il simbolo ';' (punto e virgola);
- commenti:
 - // (doppia barra) per commenti che seguono un'istruzione o commenti che occupano una sola riga;
 - /* ... */ (barra+asterisco ... asterisco+barra) racchiudono un commento che può occupare più righe;
- ogni programma Java si compone di una o più classi;
- un programma Java può essere un applet o un'applicazione;
- in un'applicazione ci deve essere una classe che contiene il metodo *main()*, punto di inizio e fine di ogni programma;
- Java distingue fra maiuscolo e minuscolo: la variabile di nome *a* è diversa dalla variabile di nome *A*;
- ambiente di sviluppo (IDE, Integrated Development Environment): un valido strumento per scrivere programmi Java è l'ambiente di sviluppo NetBeans (<https://netbeans.apache.org/>).

Esempio di programma Java

```
public class EsempioPgmJava {
    public static void main(String[] args) {
        int valore = Integer.parseInt(JOptionPane.showInputDialog("Inserire un valore"));
        String risultato = "Il valore " + valore;
        if (valore >= 0) {
            risultato += " è positivo";
        }
        else {
            risultato += " è negativo";
        }
        JOptionPane.showMessageDialog(null, risultato);
        System.exit(0);
    }
}
```

5) Il linguaggio PYTHON

a) Un programma Python

Un programma Python è composto da un certo numero di **righe logiche**.

- **righe logiche**: una riga logica è costituita da una o più **righe fisiche**.
- **righe fisiche**: una riga fisica termina dove si trova il terminatore di riga (in Windows è la sequenza ASCII: *CRLF* (return seguito da linefeed)); in pratica la riga fisica termina quando si preme il tasto *Invio*.

- Commenti

- **#** (cancellato) per commenti che seguono un'istruzione o commenti che occupano una sola riga;
- **""" ... """** (triple virgolette ...triple virgolette) commento su più righe (da Python 3).

- **Unione esplicita di righe**: due o più righe fisiche possono essere unite in righe logiche utilizzando il carattere backslash **"\"** (barra rovesciata) , per es.

```
if (1900 < anno < 2100) and (1 <= mese <= 12) \
    and (1 <= giorno <= 31) and (0 <= ora < 24) \
    and (0 <= minuti < 60) and (0 <= secondi < 60):    #Controllo data valida
    print("data corretta")
```

una linea che termina con un backslash **"\"** non può contenere un commento; e non si può utilizzare un backslash **"\"** per scrivere un commento su più righe. Notare che in Python è possibile fare il seguente confronto:

```
(1900 < anno < 2100)
```

mentre in Java (e in altri linguaggi) bisogna scomporlo nel doppio confronto:

```
(anno > 1900 && anno < 2100)
```

- **Unione implicita di righe**: le espressioni in parentesi tonde, quadre o graffe possono essere divise su più righe fisiche senza utilizzare il backslash **"\"**. per es.

```
nomi_mesi = ["Gennaio", "Febbraio", "Marzo",          # Questi sono i
             "Aprile", "Maggio", "Giugno",          # nomi, in italiano,
             "Luglio", "Agosto", "Settembre",      # dei mesi
             "Ottobre", "Novembre", "Dicembre"]    # dell'anno
```

Le righe che continuano implicitamente possono avere dei commenti. L'indentazione delle righe di continuazione non è importante. Sono ammesse righe di continuazione vuote.

- **Maiuscolo e minuscolo**: Python distingue fra maiuscolo e minuscolo: la variabile di nome **a** è diversa dalla variabile di nome **A**.

6) ChatGPT

Per ogni esercizio svolto è riportata la soluzione proposta da ChatGPT.

Essa si compone di:

- a) pseudocodice: codifica con l'utilizzo di un codice descrittivo;
- b) codifica Python;
- c) spiegazione;
- d) esempi.

È utile per:

- suggerimento di soluzioni alternative;
- approfondimento dell'analisi del problema;
- confronto e verifica della propria soluzione.

In ogni caso, ChatGPT, e strumenti simili, si dovrebbero utilizzare solo dopo aver prodotto una propria soluzione del problema.

ChatGPT, nella soluzione degli esercizi, spesso

- utilizza **funzioni, liste, tuple, set, dizionari**;
- evidenzia la **complessità** del programma.

Per un aiuto nella comprensione di questi argomenti, di seguito sono fornite alcune nozioni di base su di essi.

a) **Funzioni** (https://www.w3schools.com/python/python_functions.asp)

Una **funzione** può essere definita come un insieme di dati e istruzioni, cui è associato un nome. L'uso delle funzioni consente di:

- applicare concretamente la metodologia TOP-DOWN;
- scrivere una volta sola le istruzioni che realizzano operazioni che ricorrono frequentemente.

Una funzione è caratterizzata da:

- **definizione della funzione:** comprende

- **l'intestazione:** nome della funzione ed eventuali dati da elaborare;
- **corpo:** blocco di istruzioni eseguite dalla funzione;

- **chiamata della funzione:** quando si vogliono far eseguire le istruzioni contenute nella funzione, viene "**chiamata la funzione**", specificando il nome della funzione e fornendo a essa, eventualmente, i dati di cui necessita (**argomenti** o **parametri attuali**). La funzione può restituire al programma chiamante un valore (**valore di ritorno**; in Python, può restituire più di un **valore di ritorno**). I "contenitori" dei parametri attuali, le variabili cioè destinate, nella funzione, a contenere i dati passati al momento della chiamata, sono detti **parametri formali** (o semplicemente **parametri**).

Definizione della funzione (Python)

```
def nome_funzione ( [param_form1, param_form2,...] ):      # intestazione
.....
.....                                                    # corpo
[return (valore_ritorno1, valore_ritorno2,...) ]
```

I **parametri formali** e il **return** sono opzionali, cioè la funzione può non ricevere alcun dato e può non restituire niente.

2) In input un numero intero, stampare i numeri interi precedente e successivo.

Esempio

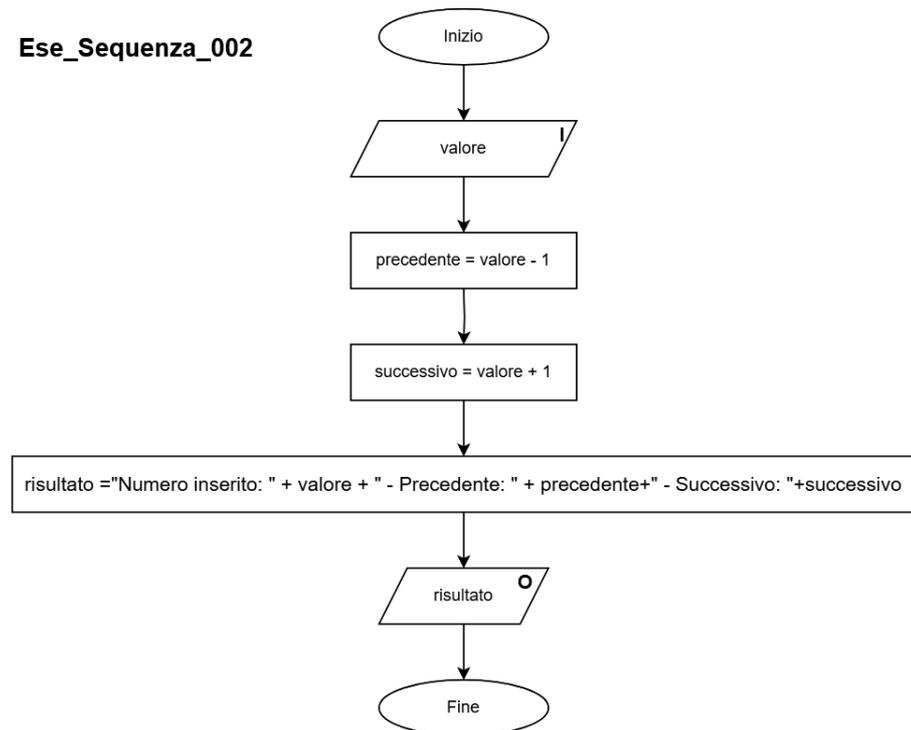
Inserire il numero: 129

Numero inserito: 129 - Precedente: 128 - Successivo: 130

Variabili

Input	Elaborazione	Output
valore: int	precedente: int	risultato: String
	successivo: int	

Ese_Sequenza_002



Codifica Java

```
import javax.swing.JOptionPane;
public class Es2PrecedenteSuccessivo {
    public static void main(String[] args) {
        int valore = Integer.parseInt(JOptionPane.showInputDialog("Inserire un numero intero"));
        int precedente = valore-1;
        int successivo = valore+1;
        String risultato = "Numero inserito: " + valore + " - Precedente: " + precedente;
        risultato=risultato+" - Successivo: "+successivo;
        JOptionPane.showMessageDialog(null, risultato);
        System.exit(0);
    }
}
```

Codifica Python

```
#Ese_Sequenza_002_PrecedenteSuccessivo
#
valore_str = input("Inserire il numero: ")
valore = int(valore_str)
precedente = valore - 1
successivo = valore + 1
risultato = "Numero inserito: " + str(valore) + " - Precedente: " + str(precedente)
risultato += " - Successivo: " + str(successivo)

print()
print(risultato)
print()
print("FINE")
```

ChatGPT

Ecco l'algoritmo in pseudocodice per stampare il numero intero precedente e il numero intero successivo di un dato numero intero:

Pseudocodice:

1. ****Inizializzazione****:
 - Leggi in input il numero intero (N).
2. ****Calcolo del numero precedente e successivo****:
 - Calcola il numero precedente: (precedente = N - 1).
 - Calcola il numero successivo: (successivo = N + 1).
3. ****Output****:
 - Stampa il numero precedente e il numero successivo.

Codice Python corrispondente:

```
```python
Leggi in input il numero intero N
N = int(input("Inserisci un numero intero: "))

Calcola il numero precedente e successivo
precedente = N - 1
successivo = N + 1

Stampa i risultati
print(f"Il numero precedente di {N} è {precedente}")
print(f"Il numero successivo di {N} è {successivo}")
```
```

Spiegazione:

1. ****Input****: L'utente inserisce un numero intero (N).
2. ****Calcolo****: Il programma calcola:
 - Il numero precedente come (N - 1).
 - Il numero successivo come (N + 1).
3. ****Output****: Stampa il numero precedente e successivo.

Esempio:

Se l'input è:
- (N = 10)

L'output sarà:

```
```
Il numero precedente di 10 è 9
Il numero successivo di 10 è 11
```
```

10) Stabilire se un anno è bisestile. In input anno.

Esempio

Inserire l'anno: 2024
L'anno 2024 è bisestile

Inserire l'anno: 2018
L'anno 2018 non è bisestile

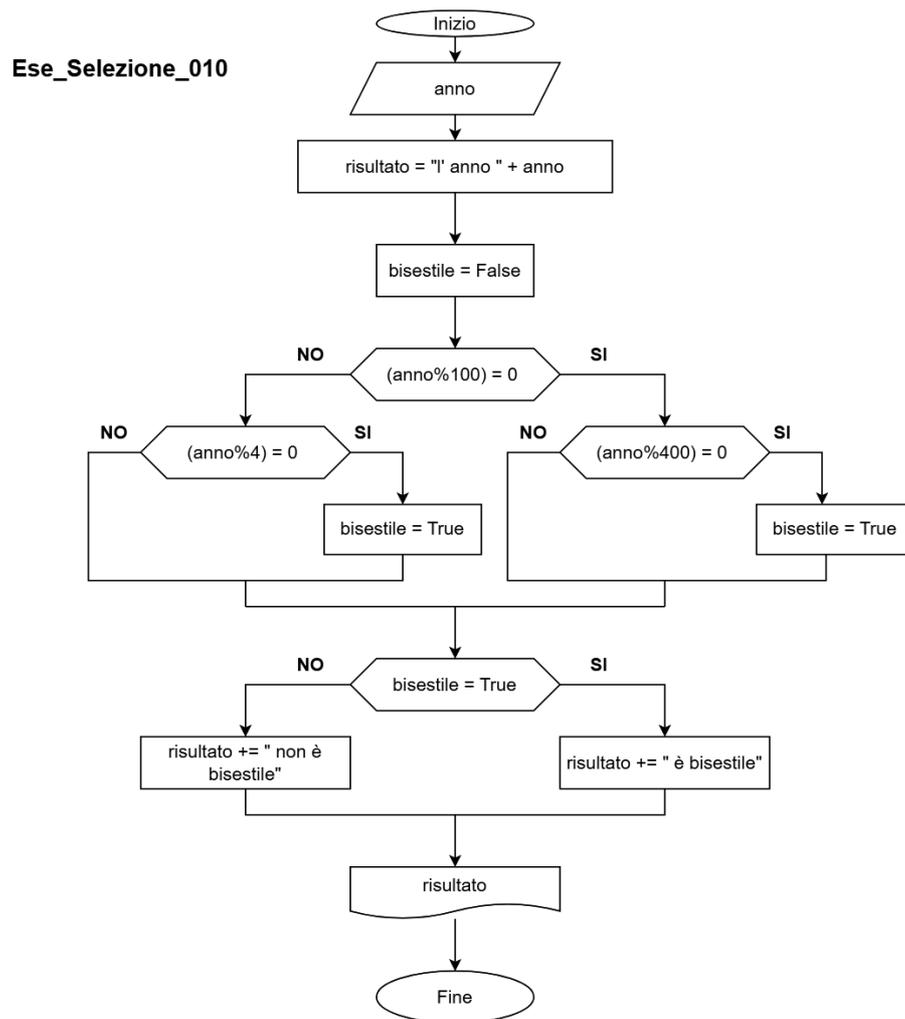
Inserire l'anno: 2000
L'anno 2000 è bisestile

Inserire l'anno: 1900
L'anno 1900 non è bisestile

Variabili

| Input | Elaborazione | Output |
|-----------|--------------------|-------------------|
| anno: int | bisestile: boolean | risultato: String |

Gli anni di inizio secolo: 1800, 1900, etc. (divisibili per 100) sono bisestili se divisibili per 400; tutti gli altri anni (non divisibili per 100) sono bisestili se divisibili per 4.



Codifica Java

```
import javax.swing.JOptionPane;
public class Es010Bisestile {
    public static void main(String[] args) {
        int anno = Integer.parseInt(JOptionPane.showInputDialog("Inserire l'anno"));
        String risultato = "L'anno " + anno;
        boolean bisestile = false;
        if (anno % 100 == 0) {
            if (anno % 400 == 0){
                bisestile = true;
            }
        } else {
            if (anno % 4 == 0){
                bisestile = true;
            }
        }
        if (bisestile){
            risultato = risultato + " è bisestile";
        }
        else{
            risultato = risultato + " non è bisestile";
        }
        JOptionPane.showMessageDialog(null, risultato);
        System.exit(0);
    }
}
```

Codifica Python

```
#Ese_Selezione_010 - AnnoBisestile
#
anno = int(input("Inserire l'anno: "))
risultato = "L'anno " + str(anno)
bisestile = False
if (anno % 100 == 0):
    if (anno % 400 == 0):
        bisestile = True
else:
    if (anno % 4 == 0):
        bisestile = True
if (bisestile):
    risultato += " è bisestile"
else:
    risultato += " non è bisestile"

print(risultato)
print("\nFINE")
```

ChatGPT

Ecco l'algoritmo in pseudocodice per determinare se un anno è bisestile:

Condizioni per un anno bisestile:

- Un anno è bisestile se:
 - È divisibile per 4 **e** non divisibile per 100, **oppure**
 - È divisibile per 400.

Pseudocodice:

1. **Inizializzazione**:
 - Leggi in input il valore dell'anno (A).

2. ****Verifica anno bisestile****:
- Se ($A \% 400 = 0$), stampa "L'anno è bisestile".
 - Altrimenti, se ($A \% 100 = 0$), stampa "L'anno non è bisestile".
 - Altrimenti, se ($A \% 4 = 0$), stampa "L'anno è bisestile".
 - Altrimenti, stampa "L'anno non è bisestile".

3. ****Output****:
- Stampa se l'anno è bisestile o no.

Codice Python corrispondente:

```
```python
Leggi in input il valore dell'anno
A = int(input("Inserisci l'anno: "))
```

```
Verifica se l'anno è bisestile
if A % 400 == 0:
 print("L'anno è bisestile")
elif A % 100 == 0:
 print("L'anno non è bisestile")
elif A % 4 == 0:
 print("L'anno è bisestile")
else:
 print("L'anno non è bisestile")
```
```

Spiegazione:

- Un anno è bisestile se è divisibile per 400.
- Se non è divisibile per 400, ma è divisibile per 100, allora ****non**** è bisestile.
- Se non è divisibile per 100, ma è divisibile per 4, allora è bisestile.
- Se non soddisfa nessuna di queste condizioni, non è bisestile.

Esempio 1:

Se l'input è:
- ($A = 2024$)

L'output sarà:

```
```
```

L'anno è bisestile

```
```
```

Esempio 2:

Se l'input è:
- ($A = 1900$)

L'output sarà:

```
```
```

L'anno non è bisestile

```
```
```

Esempio 3:

Se l'input è:
- ($A = 2000$)

L'output sarà:

```
```
```

L'anno è bisestile

```
```
```

9) Calcolare il prodotto di due numeri interi positivi o negativi, procedendo per addizioni successive.

In input i due numeri.

Esempio

valore a: 12
valore b: 3

Calcolo prodotto di 12 x 3
prodotto senza segno: 0
12
24
36

prodotto con segno: 36

valore a: 12
valore b: -3

Calcolo prodotto di 12 x -3
prodotto senza segno: 0
12
24
36

prodotto con segno: -36

valore a: -12
valore b: -3

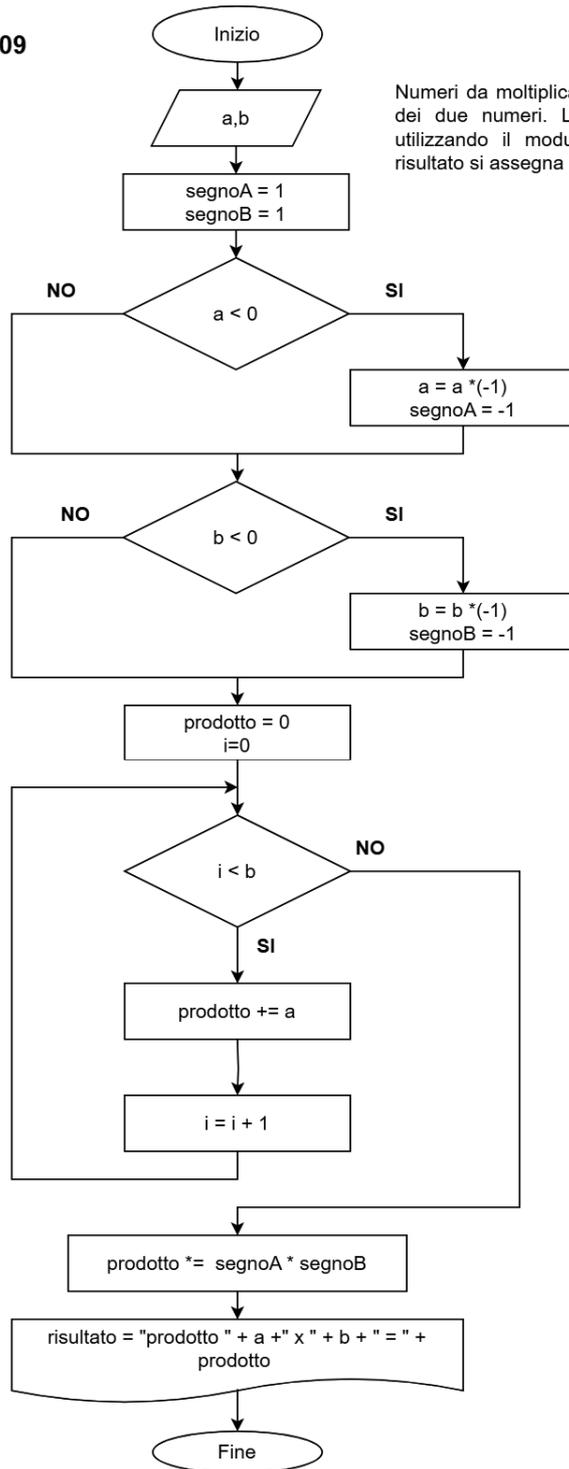
Calcolo prodotto di -12 x -3
prodotto senza segno: 0
12
24
36

prodotto con segno: 36

Variabili

| Input | Elaborazione | Output |
|--------|---------------|-------------------|
| a: int | i: int | risultato: String |
| b: int | prodotto: int | |
| | segnoA: int | |
| | segnoB: int | |

Ese_Iterazione_009



Numeri da moltiplicare. Si controlla il segno dei due numeri. L'operazione viene fatta utilizzando il modulo dei due numeri. Al risultato si assegna il segno corretto.

Sia in Java che in Python, per i due valori da moltiplicare si può utilizzare il metodo *abs()* per avere il valore assoluto.

Codifica Java

```
public class Ese_009_ProdottoAddizioniSuccessive {
    public static void main(String[] args) {
        int a=0;
        int b=0;
        int segnoA=0;
        int segnoB=0;
        int i=0;
        int prodotto=0;
        String risultato = "";
        a=Integer.parseInt(JOptionPane.showInputDialog("Inserire il primo fattore: "));
        b=Integer.parseInt(JOptionPane.showInputDialog("Inserire il secondo fattore: "));
        risultato = "Prodotto: " + a + " x " + b + " = ";
        segnoA = 1;
        segnoB = 1;
        if (a < 0){
            a *= (-1);
            segnoA = -1;
        }
        if (b < 0){
            b *= (-1);
            segnoB = -1;
        }
        i=0;
        while (i < b){
            prodotto += a;
            i += 1;
        }
        prodotto *= (segnoA * segnoB);
        risultato += prodotto;
        JOptionPane.showMessageDialog(null,risultato);
        System.exit(0);
    }
}
```

Codifica Python

```
#Ese_Iterazione_009_ProdottoAddizioniSuccessive
a = int(input("valore a: "))
b = int(input("valore b: "))
risultato = "Calcolo prodotto di " + format(a, "d") + " x " + format(b, "d") + "\n"
segnoA = 1
segnoB = 1
if (a < 0):
    a *= (-1)
    segnoA = -1

if (b < 0):
    b *= (-1)
    segnoB = -1

prodotto = 0
i=0
risultato += "prodotto senza segno:".ljust(24)
risultato += format(prodotto, ">d")+ "\n"
```

```

while i < b:
    prodotto += a
    risultato += "".ljust(23) + format(prodotto, ">d")+ "\n"
    i += 1

prodotto *= (segnoA * segnoB)
risultato += "\nprodotto con segno:".ljust(24) + format(prodotto, ">d")
print()
print (risultato)
print("\nFINE")

```

ChatGPT

Per calcolare il prodotto di due numeri interi (positivi o negativi) utilizzando addizioni successive, possiamo basarci su un concetto base: moltiplicare ($a * b$) significa sommare (a), (b) volte. Se uno o entrambi i numeri sono negativi, dobbiamo tenere conto del segno finale del risultato.

Algoritmo in linguaggio naturale:

1. Prendere in input due numeri interi (a) e (b).
2. Determinare il segno del risultato (positivo o negativo) in base ai segni di (a) e (b).
 - Se uno solo dei due numeri è negativo, il risultato sarà negativo.
3. Considerare i valori assoluti di (a) e (b) per il calcolo.
4. Sommare (a) per (b) volte usando addizioni successive.
5. Restituire il risultato, applicando il segno calcolato in precedenza.

Implementazione in Python:

```

```python
Funzione per calcolare il prodotto di due numeri tramite addizioni successive
def prodotto_addizioni_successive(a, b):
 # Determina il segno del risultato
 segno = 1
 if (a < 0 and b > 0) or (a > 0 and b < 0):
 segno = -1

 # Considera i valori assoluti dei due numeri
 a = abs(a)
 b = abs(b)

 # Inizializza il prodotto a 0
 prodotto = 0

 # Somma 'a' per 'b' volte
 for _ in range(b):
 prodotto += a

 # Applica il segno al risultato
 return prodotto * segno

Input dell'utente
a = int(input("Inserisci il primo numero intero (positivo o negativo): "))
b = int(input("Inserisci il secondo numero intero (positivo o negativo): "))

Chiamata alla funzione e visualizzazione del risultato
risultato = prodotto_addizioni_successive(a, b)
print(f"Il prodotto di {a} e {b}, calcolato con addizioni successive, è: {risultato}")
```

```

Spiegazione del codice:

1. ****Input:**** L'utente inserisce due numeri interi (a) e (b), che possono essere positivi o negativi.

2. **Determinazione del segno:**
 - Se uno solo dei due numeri è negativo, il risultato sarà negativo.
 - Altrimenti, se entrambi sono positivi o entrambi sono negativi, il risultato sarà positivo.
3. **Calcolo del prodotto:**
 - Usiamo il valore assoluto di (a) e (b) per il calcolo del prodotto.
 - Il ciclo `for` somma (a), (b) volte (considerando il valore assoluto di (b)).
4. **Risultato:** Il prodotto calcolato viene moltiplicato per il segno determinato inizialmente e restituito.

Esempi di output:

1. **Caso con numeri positivi:**

...

Inserisci il primo numero intero (positivo o negativo): 4
 Inserisci il secondo numero intero (positivo o negativo): 3
 Il prodotto di 4 e 3, calcolato con addizioni successive, è: 12

...

2. **Caso con un numero negativo:**

...

Inserisci il primo numero intero (positivo o negativo): -4
 Inserisci il secondo numero intero (positivo o negativo): 3
 Il prodotto di -4 e 3, calcolato con addizioni successive, è: -12

...

3. **Caso con entrambi i numeri negativi:**

...

Inserisci il primo numero intero (positivo o negativo): -5
 Inserisci il secondo numero intero (positivo o negativo): -2
 Il prodotto di -5 e -2, calcolato con addizioni successive, è: 10

...

Considerazioni:

- **Efficienza:** L'algoritmo somma (a) per (b) volte, quindi la complessità è (O(b)). Questo metodo può risultare inefficiente per numeri molto grandi, ma rispetta il requisito di usare addizioni successive.
- **Gestione del segno:** Il segno viene gestito separatamente rispetto al calcolo delle addizioni.
- **Caso speciale:** Se (a) o (b) è 0, il ciclo non somma nulla e il risultato sarà correttamente 0.

Questo algoritmo funziona per qualsiasi combinazione di numeri interi positivi o negativi e utilizza addizioni successive per calcolare il prodotto.

4) Stampare il triangolo di Floyd.

E' un triangolo rettangolo formato da numeri naturali consecutivi, in modo tale che sulla riga i -esima ci siano i numeri. In input il numero di righe da stampare N , intero positivo.

```

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

Esempio

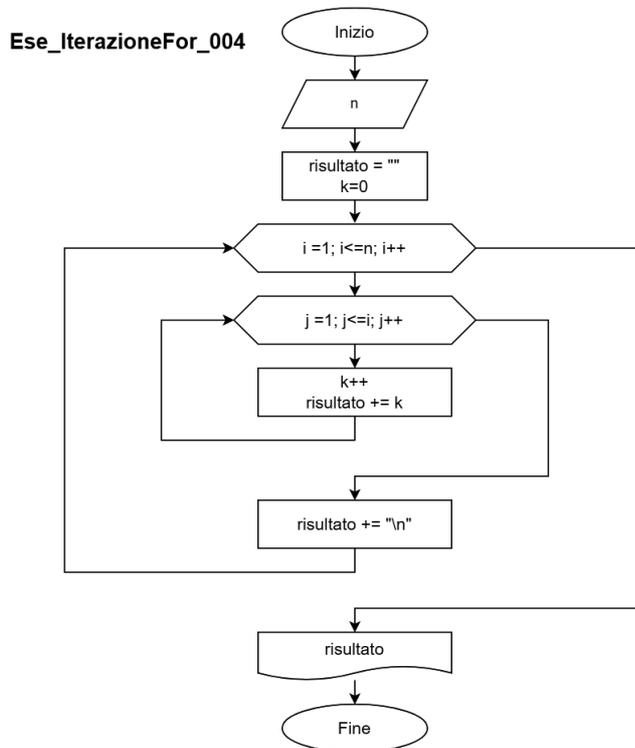
Triangolo di Floyd di N righe (n) = 6
 Triangolo di Floyd di 6 righe

```

1
2   3
4   5   6
7   8   9   10
11  12  13  14  15
16  17  18  19  20  21
    
```

Variabili

| Input | Elaborazione | Output |
|--------|--------------|-------------------|
| n: int | i: int | risultato: String |
| | j: int | |
| | k: int | |



Codifica Java

```
public class Ese_For_004 {
    public static void main(String[] args) {
        int n=0;
        int i=0;
        int j=0;
        int k=0;

        String risultato = "";
        n=Integer.parseInt(JOptionPane.showInputDialog("Triangolo di Floyd: numero max di righe N;
        (n) = "));
        risultato = "Triangolo di Floyd di " + n + " righe " + "\n";

        for (i =1;i<=n;i++){
            for (j=1;j<=i;j++){
                k++;
                risultato += String.format("% 6d",k);
            }
            risultato += "\n";
        }
        JOptionPane.showMessageDialog(null,risultato);
        System.exit(0);
    }
}
```

Codifica Python

```
#Ese_IterazioneFor_004_Triangolo di Floyd
print()
n = int(input("Triangolo di Floyd di N righe (n) = "))
print()
risultato = "Triangolo di Floyd di " + str(n) + " righe " + "\n"
k = 0
for i in range(1,n+1,1):
    for j in range(1,i+1,1):
        k += 1
        risultato += format(k, ">6d")
    risultato += "\n"

print(risultato)
print("\nFINE")
```

ChatGPT

Il **Triangolo di Floyd** è un triangolo formato da numeri consecutivi che vengono stampati riga per riga. Ogni riga ha un numero crescente di elementi, con la prima riga che contiene un numero, la seconda riga due numeri, e così via. L'algoritmo per generare un triangolo di Floyd con (N) righe può essere strutturato utilizzando un ciclo che incrementa i numeri riga dopo riga.

Strategia dell'algoritmo:

1. **Numero iniziale**: Si parte con il numero 1.
2. **Ciclo esterno**: Per ogni riga da 1 a (N), stampare un numero crescente di elementi.
3. **Incremento del numero**: Dopo ogni stampa, incrementare il numero per generare il prossimo numero della sequenza.
4. **Righe**: Ogni riga contiene (i) numeri, dove (i) è l'indice della riga corrente.

Pseudocodice:

```
``pseudo
funzione stampa_triangolo_floyd(N):
    numero = 1
```

```

per i da 1 a N:
  per j da 1 a i:
    stampa numero, senza andare a capo
    numero = numero + 1
  vai a capo
...

```

Implementazione in Python:

Ecco come implementare questo algoritmo in Python:

```

```python
def stampa_triangolo_floyd(N):
 numero = 1
 # Ciclo per ogni riga
 for i in range(1, N + 1):
 # Stampa i numeri per la riga i-esima
 for j in range(1, i + 1):
 print(numero, end=" ")
 numero += 1
 # Vai a capo alla fine di ogni riga
 print()

```

# Esempio di utilizzo

```

N = 5 # Numero di righe
stampa_triangolo_floyd(N)
...

```

### Spiegazione del codice:

1. **\*\*Variabile `numero`\*\***: Inizialmente, la variabile `numero` è impostata su 1. Questa variabile verrà incrementata di 1 dopo ogni stampa, per mantenere la sequenza numerica consecutiva.
2. **\*\*Ciclo esterno `for i in range(1, N + 1)`\*\***: Questo ciclo scorre su ogni riga del triangolo. Ogni riga (i) contiene esattamente (i) numeri.
3. **\*\*Ciclo interno `for j in range(1, i + 1)`\*\***: Per ogni riga ( i ), vengono stampati (i) numeri consecutivi.
4. **\*\*`print(numero, end=" ")`\*\***: Stampa il numero senza andare a capo, ma separato da uno spazio.
5. **\*\*`print()`\*\***: Dopo aver stampato tutti i numeri di una riga, eseguiamo un `print()` vuoto per andare a capo prima della riga successiva.

### Esempio di esecuzione:

Se l'input (N) è 5, il programma stamperà:

```

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

```

Se l'input ( N ) è 4, l'output sarà:

```

...
1
2 3
4 5 6
7 8 9 10
...

```

### Considerazioni:

- Il triangolo di Floyd continua ad aggiungere numeri consecutivi riga per riga.
- L'algoritmo utilizza un ciclo annidato: il ciclo esterno controlla le righe e il ciclo interno stampa il numero corretto di elementi per ciascuna riga.