Esercizi di programmazione in linguaggio Python - Costrutto iterazione

prof. Roberto Fuligni

1. [**valida1**] ★[1] Scrivere un programma che richieda l'inserimento da tastiera di un numero intero pari. Nel caso in cui il numero non sia pari, il programma deve segnalare l'errore e ripetere l'inserimento.
2. [**valida2**] – Ripetere l'esercizio precedente, nell'ipotesi che il numero da inserire debba essere di tre cifre e dispari.
3. [**sommanum**] ★ Scrivere un programma che richieda da tastiera l'inserimento di un numero intero *NUM*. Successivamente, il programma richiede all'utente l'immissione di *NUM* numeri float (uno alla volta) sommandoli all'interno di un apposito accumulatore. Al termine, il programma mostra il valore finale dell'accumulatore.
4. [**potenze1**] ★ Scrivere un programma che visualizzi il valore di tutte le prime *n* potenze di 2 con *n* richiesto all'utente e compreso tra 2 e 14. È richiesta la validazione del numero *n*.
5. [**potenze2**] – Scrivere un programma che verifichi se un numero inserito da tastiera è una potenza del 2 e, se lo è, determini l'esponente.
6. [**contamu11**] – Date due costanti intere A e B, scrivere un programma che conti tutti i numeri multipli di 11 compresi tra A e B (estremi inclusi), mostrando a video il risultato finale. Verificare che, scegliendo A= 23 e B = 78, il conteggio è pari a ?.
7. [**sequenza1**] ★ Un programma richiede l'inserimento di una sequenza di numeri interi positivi la cui lunghezza non è nota a priori. L'utente inserisce i numeri della sequenza uno alla volta e per segnalare la fine dell'inserimento immette il numero 0 (dato "tappo"). Il programma elabora i valori della sequenza contando i numeri pari e quelli dispari. Al termine, i due conteggi sono mostrati sullo schermo.
8. [**quadrato1**] – Scrivere un programma visualizzi il quadrato dei primi 24 numeri naturali.
9. [**coppie1**] – Scrivere un programma che, data la costante intera N, permetta di caricare *N* coppie di numeri reali, calcoli la somma di ogni coppia e la visualizzi. Eseguire il programma prima con N = 3, poi con N = 5.
10. [**coppie2**] – Scrivere un programma che, date *n* coppie di numeri reali, conti le coppie formate da valori opposti.
11. [**coppie3**] – Scrivere un programma che, date *n* coppie di numeri reali, conti quelle che generano un prodotto negativo, positivo o uguale a zero senza eseguire le moltiplicazioni.
12. [**maxmin**] – Scrivere un programma che determini il maggiore e il minore tra gli *n* numeri immessi dall'utente.
13. [**media1**] – Scrivere un programma che calcoli e visualizzi la media di *n* numeri reali immessi dall'utente.
14. [**media2**] ★ Dati *n* numeri interi, scrivere un programma che calcoli la media aritmetica dei valori pari e quella dei valori dispari.

[1] Nelle pagine successive sono indicate le soluzioni degli esercizi che riportano il simbolo ★.

15. [**seriearmonica**] – Scrivere un programma che, richiesto all'utente un numero intero *m*, calcoli e visualizzi la somma:

$$\sum_{k=1}^{m} \frac{1}{k}$$

dei primi *m* elementi della serie armonica. Verificare che la somma della serie, arrestata al quinto elemento, è pari a circa 2.2833.

16. [**rettangolo1**] – Scrivere un programma che visualizzi un rettangolo la cui cornice sia costituita da caratteri asterisco, la parte interna da caratteri Q e dove i numeri di righe e di colonne del rettangolo siano decisi dall'utente (ciascuno di questi numeri non deve essere inferiore a 3).

Per esempio, se il numero delle righe è uguale a 5 e il numero di colonne a 21, sul video deve apparire:

```
*********************
*QQQQQQQQQQQQQQQQQQQ*
*QQQQQQQQQQQQQQQQQQQ*
*QQQQQQQQQQQQQQQQQQQ*
*********************
```

[**rettangolo2**] - Ripetere l'esercizio precedente, visualizzando però il rettangolo un numero di volte scelto dall'utente.

17. [**divisori**] ★ Scrivere un programma che, richiesto un numero intero, visualizzi tutti i suoi divisori (Suggerimento: ricordiamo che a divide b solo se *b mod a = 0*).

18. [**numprimo**] ★ Scrivere un programma che, richiesto un numero intero, stabilisca se questo è primo (Suggerimento: sfruttare il precedente esercizio).
Utilizzare il programma per rispondere alla seguente domanda: quali dei seguenti numeri sono primi?

96 553 15 983 567 2 086 749 377

(Risposta: il primo e il terzo)

19. [**cap**] ★ Scrivere un programma che, dati in input *n* codici di avviamento postale, conti quelli che hanno le prime due cifre uguali a 28 e comunichi il risultato (si considerino CAP formati da numeri interi a 5 cifre, es. Novara: 28100).

20. [**imc**] ★ Conoscendo il peso P e l'altezza H di una persona, è possibile calcolare l'indice di massa corporea (IMC) mediante la formula:

$$IMC = \frac{P}{H^2}$$

Scrivere un programma che: richieda l'inserimento del peso (in chilogrammi) e l'altezza (in metri) di N persone (dove N è un numero inserito da tastiera) , visualizzando l'indice IMC corrispondente; conti il numero di persone aventi IMC superiore a una certa soglia S prefissata (utilizzare il valore S = 41.5 kg/m^2).

21. [**diecipari**] ★ Scrivere un programma che stampi i dieci numeri pari successivi a numero intero *n* dato in input.

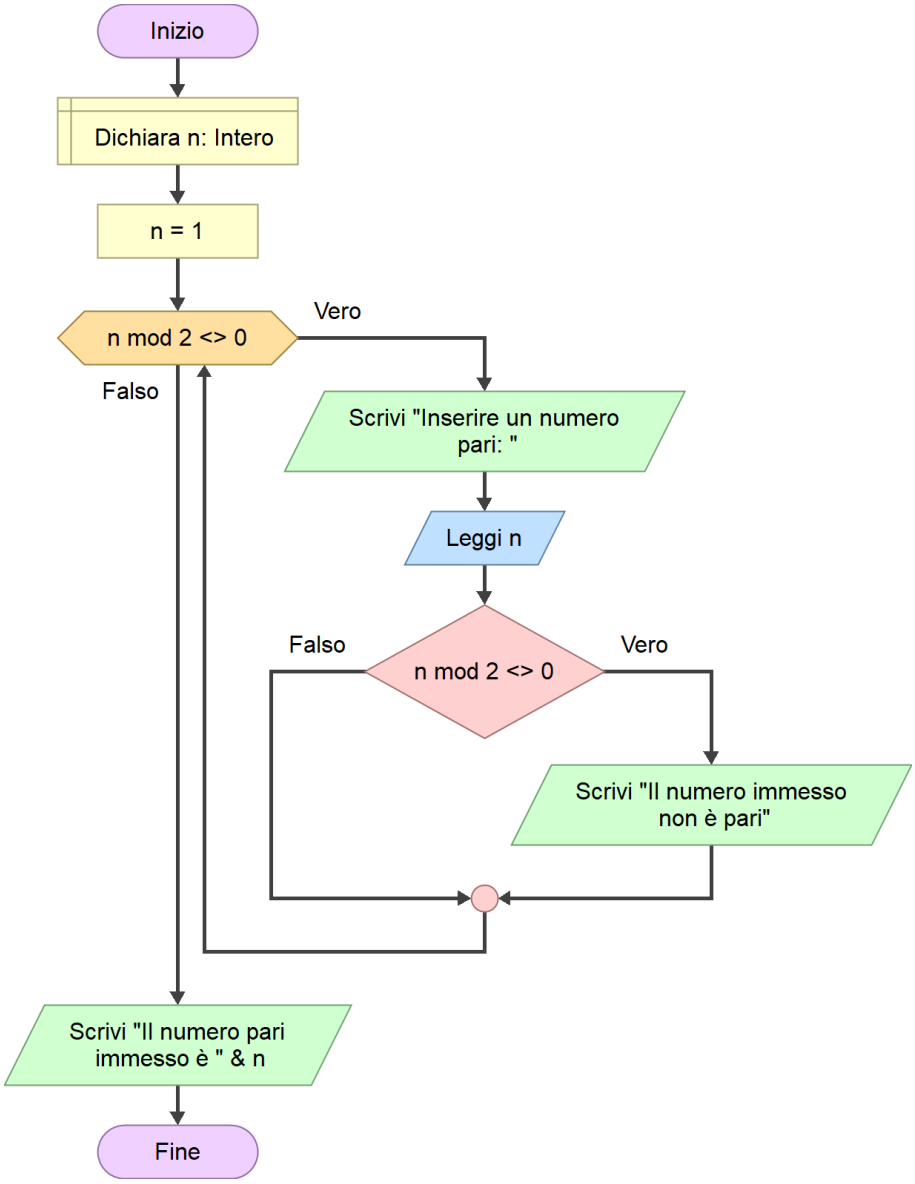
22. [**frazioni**] ★ Scrivere un programma che, date in input una alla volta N frazioni, riconosca quelle irriducibili.

23. [**assicurazioni**] ★ All'inizio in un nuovo anno, un'agenzia deve adeguare gli importi delle assicurazioni delle automobili sulla base degli incidenti registrati nell'anno precedente: se un'automobile non ha subito incidenti, l'importo è ridotto del 4%; in caso contrario, l'importo è aumentato del 12%.
    Scrivere un programma che, richiesti in input l'importo e il numero di incidenti dell'anno precedente di N automobili, determini e visualizzi il nuovo importo da pagare per ciascuna assicurazione. Il programma deve inoltre stampare il totale degli importi previsti per il nuovo anno.
24. [**coppieord**] - Scrivere un programma che visualizzi tutte le coppie ordinate di numeri naturali la cui somma è 20.
25. [**tabellaprimi**] ★ Scrivere un programma che visualizzi la tabella dei numeri primi compresi tra 2 e N.
26. [**fattori**] - Scrivere un programma che scomponga un numero in fattori primi.

Soluzioni di alcuni esercizi

Esercizio n. 1 (valida1)

Diagramma a blocchi



Pseudocodifica

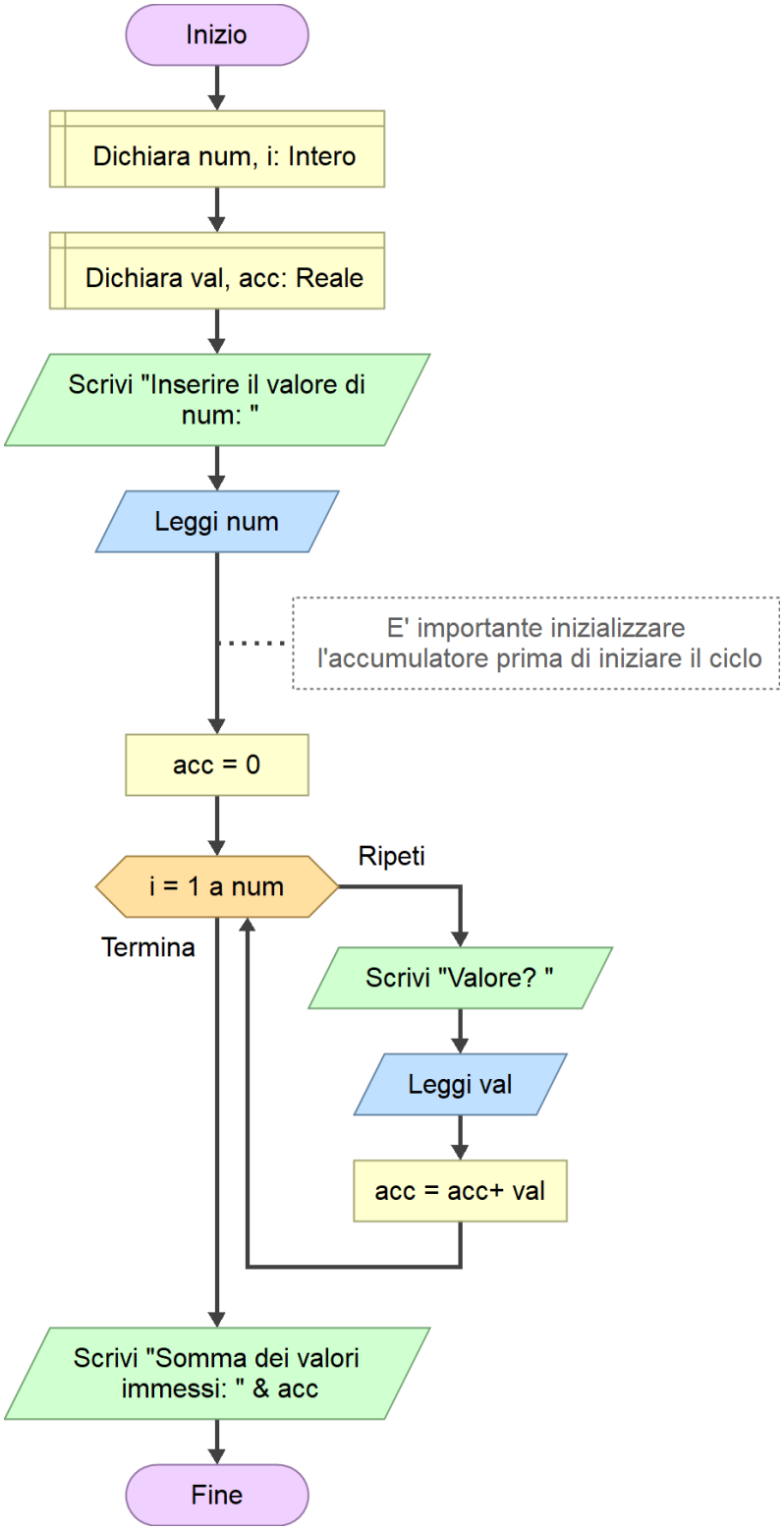
```
ALGORITMO valida1
VARIABILI
    n: INTERO

INIZIO

    n <- 1

    # n MOD 2 restituisce il resto della divisione tra n e 2.
    # se n è pari, il resto è pari zero; se n è dispari, il resto è diverso
    # da zero (in particolare, vale 1)

    MENTRE (n MOD 2) <> 0 RIPETI
        SCRIVI "Inserire un numero pari: "
        LEGGI n

        SE (n MOD 2) <> 0 ALLORA
            SCRIVI "Il numero inserito non è pari"
        FINE SE
    FINE MENTRE

    SCRIVI "Il numero pari inserito è ", pari
FINE
```

Programma

```
#       valida1.py
#
#       Scrivere un programma che richieda l'inserimento da tastiera di un
#       numero intero pari. Nel caso in cui il numero non sia pari,
#       il programma deve segnalare l'errore e ripetere l'inserimento.

# Si assegna a n un valore iniziale dispari per "forzare" l'esecuzione
# del corpo del ciclo

n = 1

while (n % 2) != 0:
    # Corpo del ciclo while
    n = int(input("Inserire un numero pari: "))

    if (n % 2) != 0:
        print("Il numero inserito non è pari")


print("Il numero pari inserito è: ", n)
```

Esercizio n. 3 (sommanum)

Diagramma a blocchi



Pseudocodifica

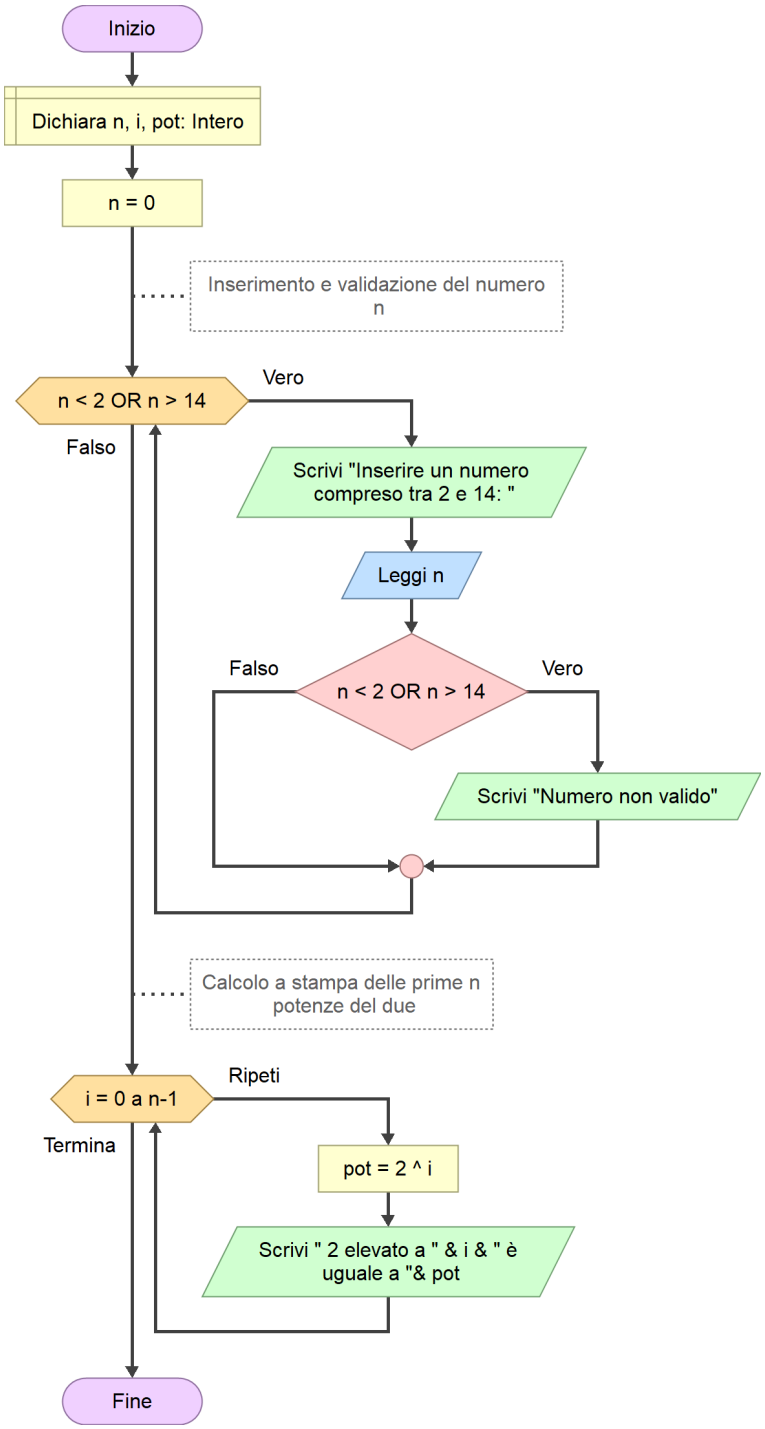
```
ALGORITMO sommanum
VARIABILI
    num, i: INTERO
    val, acc: REALE

INIZIO
    SCRIVI "Inserire il valore di NUM: "
    LEGGI num

    # E' importante inizializzare l'accumulatore prima di iniziare il ciclo
    acc = 0

    PER i = 1 FINO A num RIPETI
        SCRIVI "Valore? "
        LEGGI val
        acc <- acc + val
    FINE PER

    SCRIVI "somma dei valori accumulati: ", acc
FINE
```

Programma

```
#       sommanum.py
#
#       Scrivere un programma che richieda da tastiera l'inserimento di
#       un numero intero num. Successivamente, il programma richiede all'utente
#       l'immissione di n numeri float (uno alla volta), sommandoli
#       all'interno di un apposito accumulatore.
#       Al termine, il programma mostra il valore finale dell'accumulatore.

num = int(input("Inserire il valore di NUM: "))
acc = 0

# Ripete il ciclo per num volte, assegnando ogni volta alla variabile i
# un numero crescente da 0 a num-1

for i in range(num):
    val = float(input("Valore? "))
    acc = acc + val

print(f"Somma dei valori immessi: {acc}")
```

Esercizio n. 4 (potenze1)

Diagramma a blocchi

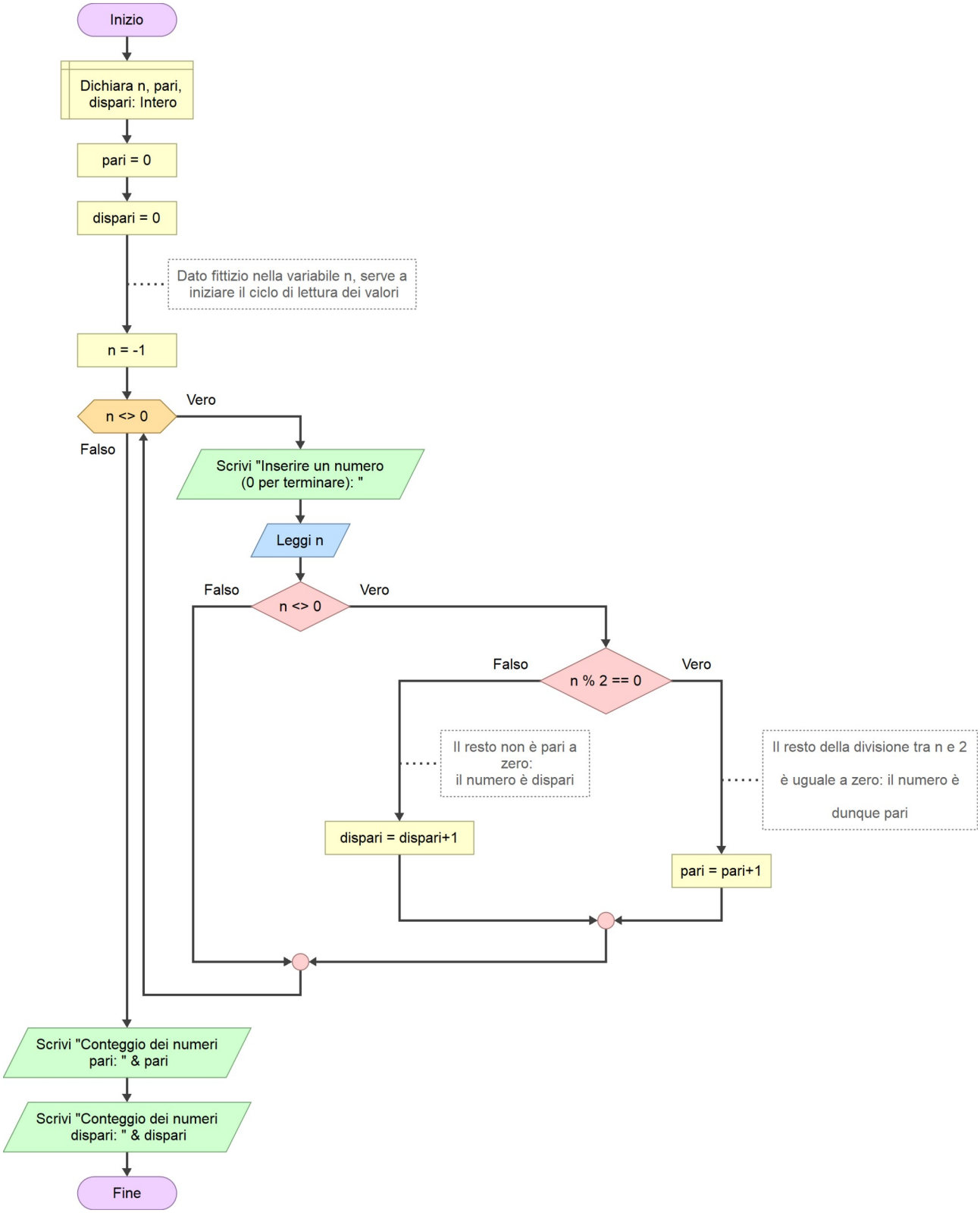


Pseudocodifica

```
#       potenze1.py
#
#       Scrivere un programma che visualizzi il valore di tutte le
#       prime n potenze di 2 con n richiesto all'utente e compreso tra 2 e 14.
#       È richiesta la validazione del numero n.

# Inserimento di un dato fittizio  diverso da zero nella variabile n
# Serve per iniziare il ciclo di lettura

n = 0

# Inserimento e controllo (validazione) del numero n

while (n < 2) or (n > 14):
    n = int(input("Inserire un numero compreso tra 2 e 14: "))
    if (n < 2) or (n > 14):
        print("Numero non valido")

# calcolo e stampa delle prime n potenze del due
for i in range(n):
    pot = 2 ** i
    print(f"due elevato a {i} è uguale a {pot}")
```

Esercizio n. 7 (sequenza1)

Diagramma a blocchi



Programma

```
#       sequenza1.py
#
#       Un programma richiede l'inserimento di una sequenza di numeri
#       interi positivi la cui lunghezza non è nota a priori.
#       L'utente inserisce i numeri della sequenza uno alla volta e per
#       segnalare la fine dell'inserimento immette il numero 0 (dato "tappo").
#       Il programma elabora i valori della sequenza contando i numeri pari e
#       quelli dispari. Al termine, i due conteggi sono mostrati sullo schermo.

# Inizializzazione dei contatori

pari = 0
dispari = 0

# Inserimento di un dato fittizio  diverso da zero nella variabile n
# Serve per iniziare il ciclo di lettura

n = -1

while n != 0:
    n = int(input("Inserire un numero (0 per terminare): "))
    if n != 0:
        # Il numero immesso non è il dato tappo: si aggiornano i contatori

        if n % 2 == 0:
            # Il resto della divisione tra n e 2 è uguale a zero:
            # Il numero è dunzuqe pari
            pari = pari+1
        else:
            dispari = dispari + 1

# Stampa finale dei contenuti dei contatori
print(f"Conteggio dei numeri pari: {pari}")
print(f"Conteggio dei numeri dispari: {dispari}")
```

Esercizio n. 14 (media2)

Pseudocodifica

```
ALGORITMO media2
COSTANTI
    N: INTERO = 3

VARIABILI
    conta_pari, conta_dispari: INTERO
    somma_pari, somma_dispari: INTERO
    media_pari, media_dispari: REALE
    num: INTERO
    i: INTERO

INIZIO
    conta_pari ← 0
    somma_pari ← 0
    somma_dispari ← 0

    PER i = 0 FINO A N-1 RIPETI
        SCRIVI "Inserire il ”, i+1, "° valore: "
        LEGGI num
        SE num MOD 2 = 0 ALLORA
            # Il numero immesso è pari
            conta_pari ← conta_pari + 1
            somma_pari ← somma_pari + num
        ALTRIMENTI
            # Il numero è dispari
            somma_dispari ← somma_dispari + num
        FINE SE
    FINE PER

    # Il conteggio dei valori dispari è ricavato per differenza
    conta_dispari ← N – conta_pari

    media_pari ← somma_pari / conta_pari
    media_dispari ← somma_dispari / conta_dispari

    SCRIVI "I valori pari inseriti sono ‟, conta_pari, "; la media è ", media_pari
    SCRIVI "I valori dispari inseriti sono ‟, conta_dispari, "; la media è ",
media_dispari
FINE
```

Programma

```
#       media2.py
#
#       Dati n numeri interi, scrivere un programma che calcoli la media
#       aritmetica dei valori pari e quella dei valori dispari.

N = 3

conta_pari = 0
somma_pari = 0
somma_dispari = 0

for i in range(N):
    print("Inserire il ", i+1, "° valore:")
    num = int(input(""))
    if num % 2 == 0:
        # Il numero immesso è pari
        conta_pari = conta_pari + 1
        somma_pari = somma_pari + num
    else:
        # Il numero è dispari
        somma_dispari = somma_dispari + num

# Il conteggio dei valori dispari è ricavato per differenza
conta_dispari = N - conta_pari

media_pari = somma_pari / conta_pari
media_dispari = somma_dispari / conta_dispari

print(f"I valori pari inseriti sono {conta_pari}; la media è {media_pari}")
print(f"I valori dispari inseriti sono {conta_dispari}; la media è {media_dispari}")
```

Esercizio n. 17 (divisori)

Programma

```
#       divisori.py
#
#       Scrivere un programma che, richiesto un numero intero, visualizzi tutti
#       i suoi divisori (Suggerimento: ricordiamo che a divide b solo se b mod a = 0).

num = int(input("Inserire un numero intero positivo: "))

print("I divisori del numero", num, "sono:")

for i in range(1, num+1):
    if num % i == 0:
        # i è un divisore di num
        print(i)
```

Esercizio n. 18 (numprimo)

Programma

```
#       numprimo.py
#
#       Scrivere un programma che, richiesto un numero intero, stabilisca se
#       questo è primo.
#       Utilizzare il programma per rispondere alla seguente domanda:
#       quali dei seguenti numeri sono primi?
#           96553    15983567    2086749377


num = int(input("Inserire un numero intero positivo: "))

conta_divisori = 0

for i in range(1, num+1):
    if num % i == 0:
        # i è un divisore di num
        conta_divisori = conta_divisori + 1


if conta_divisori > 2:
    print("Il numero", num, "NON è primo")
else:
    print("Il numero", num, "è primo")
```

Esercizio n. 19 (cap)

Programma

```
#       cap.py
#
#        Scrivere un programma che, dati in input n codici di avviamento postale,
#        conti quelli che hanno le prime due cifre uguali a 28 e comunichi
#        il risultato (si considerino CAP formati da numeri interi a 5 cifre,
#        es. Novara: 28100).

N = 5
conta28 = 0

for i in range(N):
    cap = int(input("Inserire un C.A.P. di cinque cifre: "))

    # Determina le prime due cifre del CAP dividendolo per 1000 (divisione intera)
    # (esempio: 28100 // 1000 = 28)

    inizio_cap = cap // 1000
    if inizio_cap == 28:
        conta28 = conta28 + 1

print("\nI C.A.P. che iniziano con la coppia di cifre '28' sono", conta28)
```

Esercizio n. 20 (imc)

Pseudocodifica

```
ALGORITMO imc
COSTANTI
    S: REALE = 41.5            # Soglia massima dell'IMC (kg/m^2)
VARIABILI
    n, i, conta: INTERO
    p, h, imc: REALE

INIZIO
    SCRIVI "Numero di persone da inserire: "
    LEGGI n

    # Inizializza il contatore delle persone con imc oltre la soglia
    conta = 0

    # Si contano le iterazioni a partire da zero (quindi fino a n-1)
    PER i = 0 FINO A n-1 RIPETI
        SCRIVI "Persona n. ", i+1
        SCRIVI "      Peso (kg): "
        LEGGI p
        SCRIVI "    Altezza (m): "
        LEGGI h
        imc <- p / (h^2)
        SCRIVI "   IMC (kg/m^2): ", imc"

        SE imc > S ALLORA
            conta <- conta + 1
        FINE SE
    FINE PER

    SCRIVI "Numero di persone che hanno superato la soglia: ", conta
FINE
```

Programma

```
#       imc.py
#
#       Calcolo dell'indice di massa corporea (IMC) di N persone
#       e determinazione del numero di persone avente IMC superiore a
#       una data soglia prefissata.

S = 41.5    # Soglia massima dell'IMC (kg/m^2)

n = int(input("Numero di persone da inserire: "))

conta = 0   # Inizializza il contatore delle persone con imc oltre la soglia

for i in range(n):
    print("Persona n. ", i+1)
    p = float(input("      Peso (kg): "))
    h = float(input("    Altezza (m): "))
    imc = p / (h**2)
    print(f"   IMC (kg/m^2): {imc:.2f}\n")     # \n crea una riga vuota dopo il messaggio
    if imc > S:
        conta = conta + 1

print("Numero di persone che hanno superato la soglia: ", conta)
```

Esercizio n. 21 (diecipari)

Programma

```
#       diecipari.py
#
#        Scrivere un programma che stampi i dieci numeri pari successivi a
#        numero intero n dato in input.


n = int(input("Inserire un numero intero positivo: "))

print("I dieci numeri pari succesivi a ", n, "sono:")

if n % 2 == 0:
    # Il numero n è pari, il primo successivo è n + 2
    succ = n + 2
else:
    # il numero non è pari, il primo successivo è n + 1
    succ = n + 1

for i in range(10):
    # Stampa il numero pari memorizzato nella variabile succ...
    print(succ)
    # ... e determina il prossimo numero da stampare
    succ = succ + 2
```

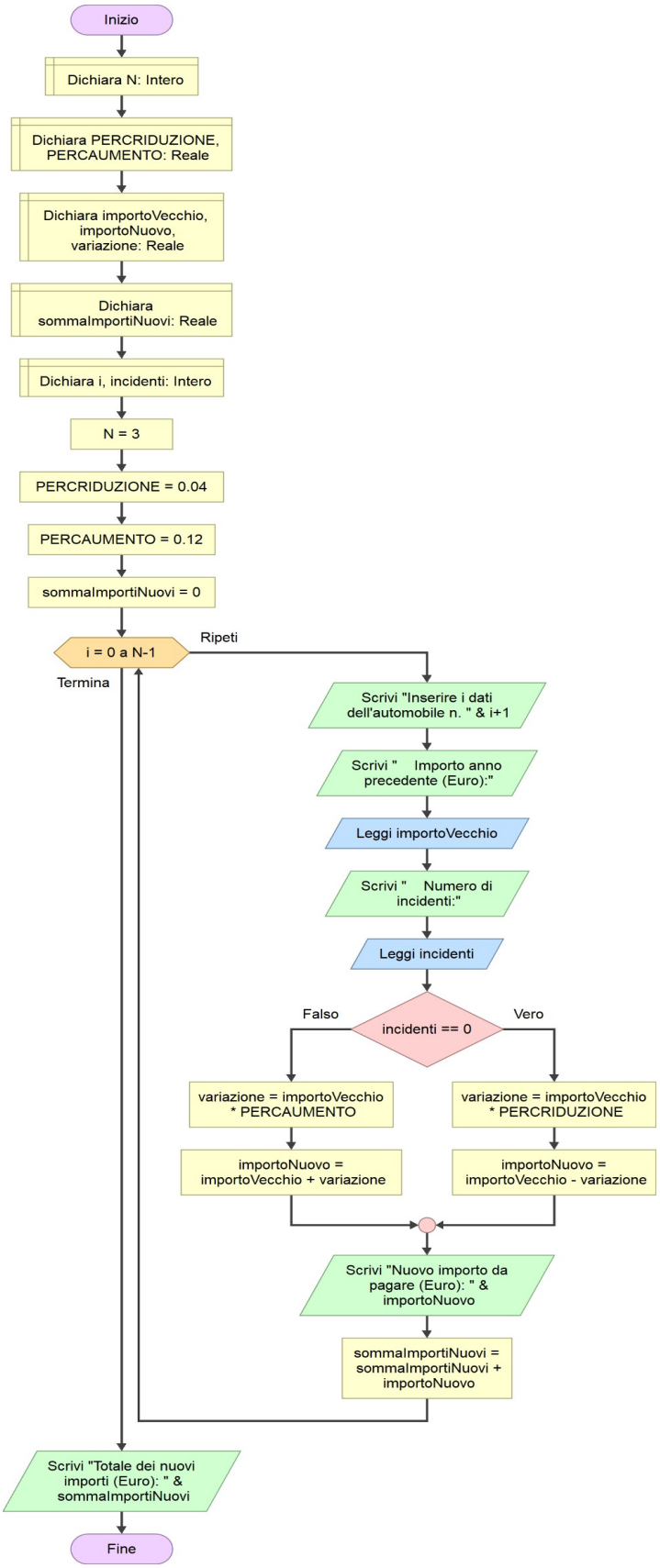
Esercizio n. 22 (frazioni)

Programma

```
#       frazioni.py
#
#        Scrivere un programma che, date in input una alla volta N frazioni,
#        riconosca quelle irriducibili.

import math           # Libreria contenente la funzione gcd per il calcolo
                      # del massimo comun divisore

N = 3       # Numero di frazioni da inserire

for i in range(N):
    print("Inserire la frazione n. ", i+1)
    num = int(input("    Numeratore: "))
    den = int(input("    Denominatore: "))
    mcd = math.gcd(num, den)

    # Una frazione è irriducibile se MCD(num, den) è uguale a uno
    if mcd == 1:
        print(f"\nLa frazione {num}/{den} è irriducibile\n")
    else:
        # La frazione è riducibile, si calcolano numeratore e denominatore
        # della frazione ridotta ai minimi termini

        num2 = num // mcd
        den2 = den // mcd
        print(f"\nLa frazione {num}/{den} è riducibile a {num2}/{den2}\n")
```

Esercizio n. 23 (assicurazioni)

Diagramma di flusso

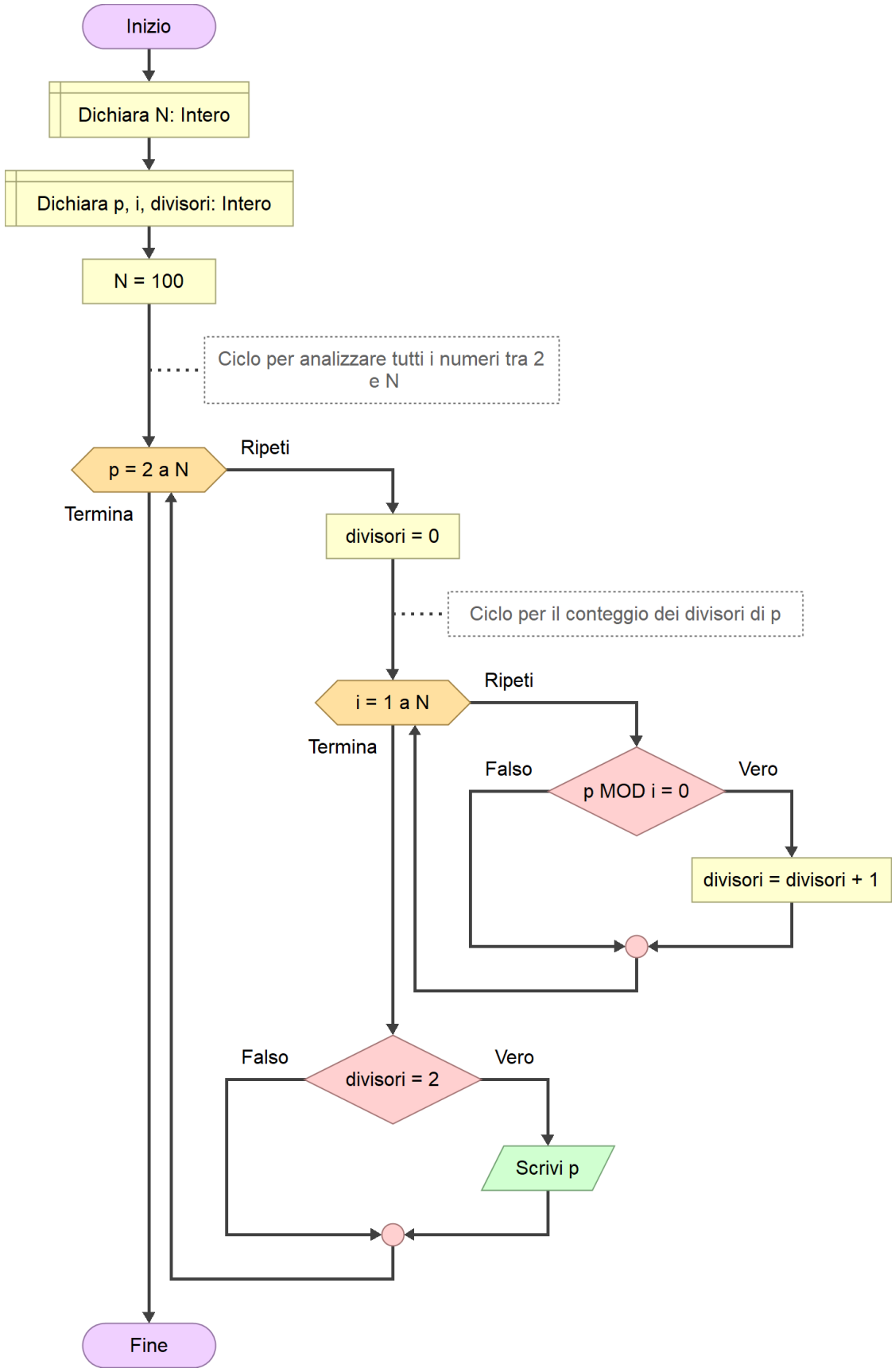


Programma

```
#       assicurazioni.py
#
#       All'inizio in un nuovo anno, un'agenzia deve adeguare gli importi delle
#       assicurazioni delle automobili sulla base degli incidenti registrati
#       nell'anno precedente: se un'automobile non  ha subito incidenti, l'importo è
#       ridotto del 4%; in caso contrario, l'importo è aumentato del 12%.
#       Scrivere un programma che, richiesti in input l'importo e il numero
#       di incidenti dell'anno precedente di N automobili, determini e
#       visualizzi il nuovo importo da pagare per ciascuna assicurazione.
#       Il programma deve inoltre stampare il totale degli importi previsti
#       per il nuovo anno.

# Costanti del programma
N = 3                   # Numero di automobili da elaborare
PERC_RIDUZIONE = 0.04   # Percentuale di riduzione dell'importo
PERC_AUMENTO = 0.12     # Percentuale di aumento dell'importo

somma_nuovi_importi = 0

for i in range(N):
    print("Inserire i dati dell'automobile n.", i+1)
    importo_vecchio = float(input("    Importo dell'anno precedente (Euro): "))
    incidenti = int(input("Numero di incidenti: "))

    if (incidenti == 0):
        # L'automobile non ha subito incidenti, l'importo da pagare deve essere ridotto
        variazione = importo_vecchio * PERC_RIDUZIONE
        importo_nuovo = importo_vecchio - variazione
    else:
        # A causa degli incidenti dell'anno precedente, l'importo deve essere aumentato
        variazione = importo_vecchio * PERC_AUMENTO
        importo_nuovo = importo_vecchio + variazione

    print(f"\nNuovo importo da pagare (Euro): {importo_nuovo:.2f}\n")

    # Si aggiorna l'accumulatore dei nuovi importi
    somma_nuovi_importi = somma_nuovi_importi + importo_nuovo


# Terminato il calcolo dei singoli importi, si stampa la somma finale
print(f"\nSomma dei nuovi importi (Euro): {somma_nuovi_importi:.2f}")
```

Esercizio n. 25 (tabellaprimi)

Diagramma di flusso



Programma

```
#       tabellaprimi.py
#
#       Scrivere un programma che visualizzi la tabella dei numeri primi
#       compresi tra 2 e N.

N = 100

# Per ogni numero p compresso nell'intervallo tra 2 e N, si eseguono le seguenti operazioni:
#      * Conteggio di tutti i numeri, tra 1 e p, che sono divisori di p
#      * Se il conteggio dei divisori è uguale a due, il numero p è primo, quindi si stampa p

for p in range(2, N+1):
    # E' importante azzerare il contatore dei divisori
    # ogni volta che si elabora un nuovo numero
    divisori = 0

    # Ciclo for annidato
    for i in range(1, p+1):
        if p % i == 0:
            divisori = divisori + 1

    if divisori == 2:
        print(p)
```