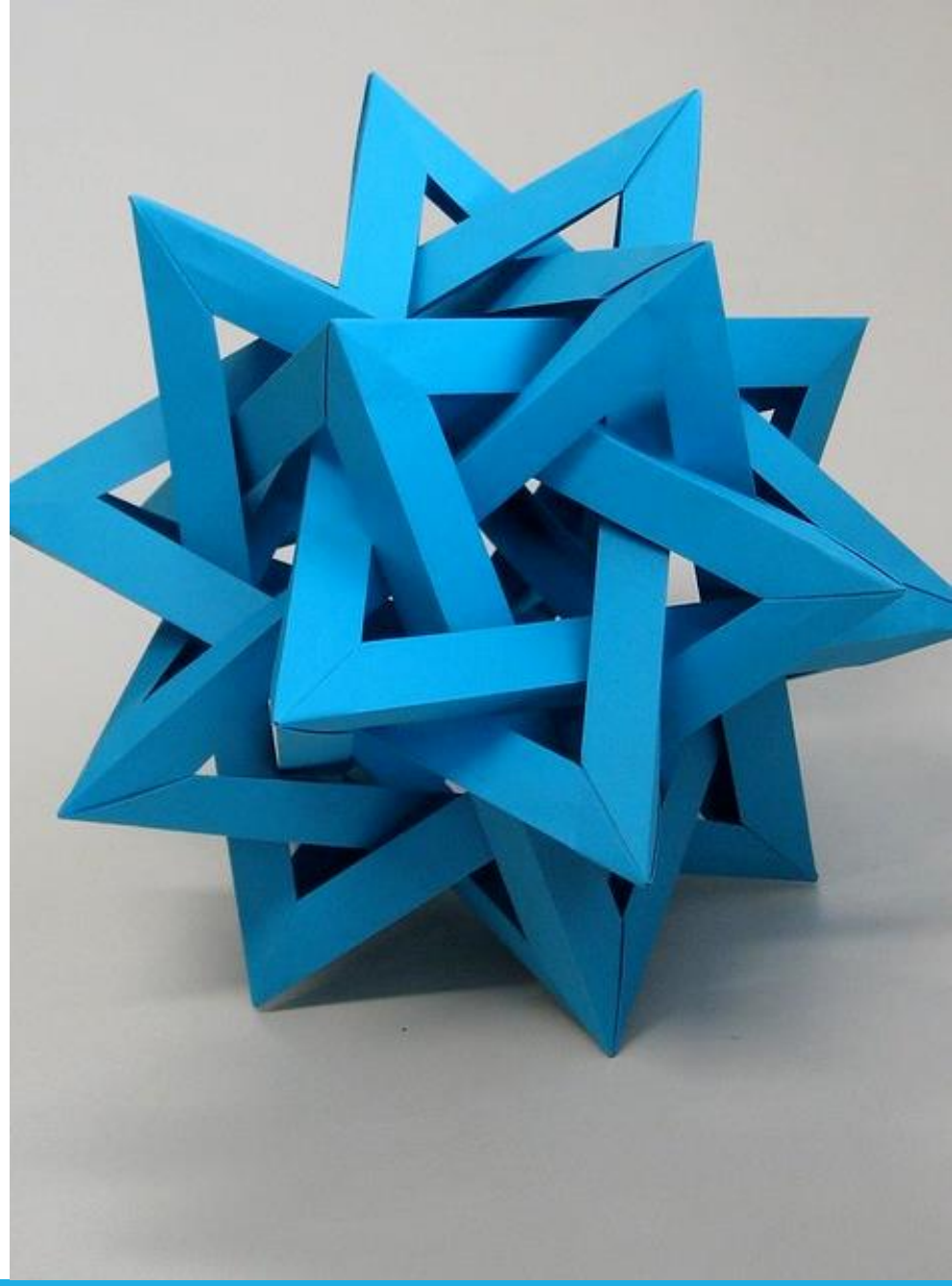
POLITECNICO DI TORINO

Dipartimento di Automatica e Informatica

Laboratorio 4

CICLI – WHILE E FOR



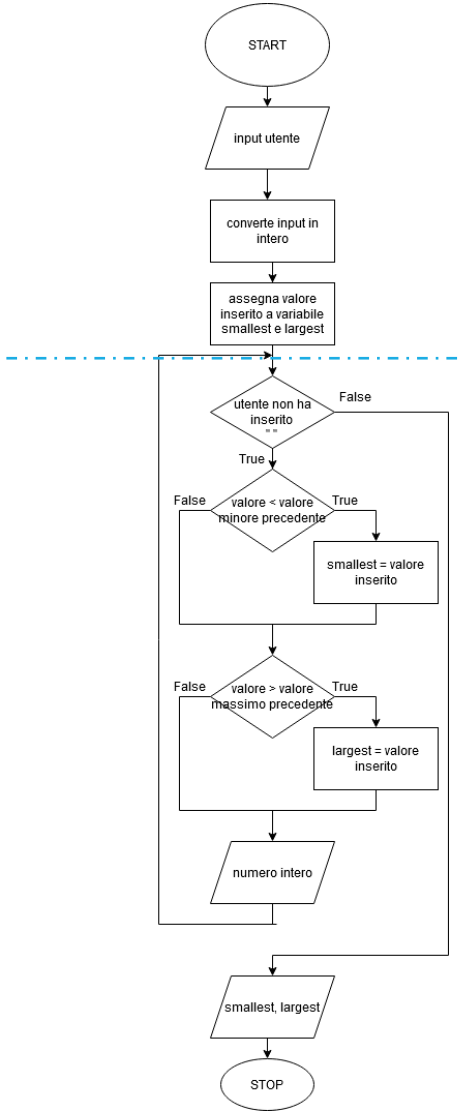
This Photo by Unknown Author is licensed under CC BY-SA

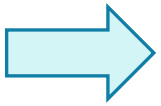
Esercizio 1

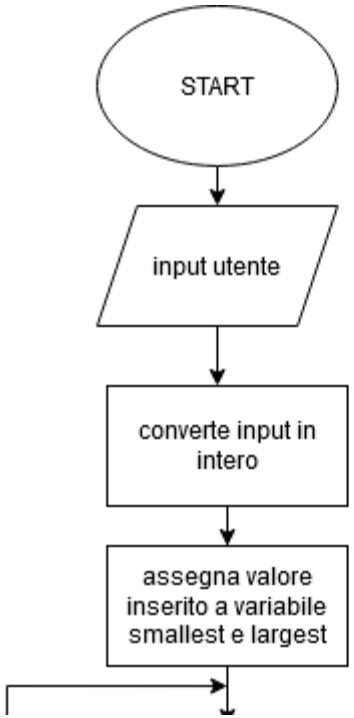
Esercizio 1. Scrivete programmi che leggano una sequenza di numeri interi (la sequenza termina quando viene inserita la *stringa vuota*) che visualizzino quanto segue:

a. il valore minimo e il valore massimo tra quelli acquisiti
b. il numero di valori pari e il numero di valori dispari tra quelli acquisiti;
c. le somme parziali di tutti i numeri acquisiti, calcolate e visualizzate dopo ogni nuova acquisizione (se, ad esempio, i valori in ingresso sono 1 7 2 9, il programma visualizzerà 1 8 10 19);
d. i valori adiacenti duplicati (se, ad esempio, i valori acquisiti sono 1 3 3 4 5 5 6 6 6 3, il programma visualizzerà 3 5 6). [P4.2]

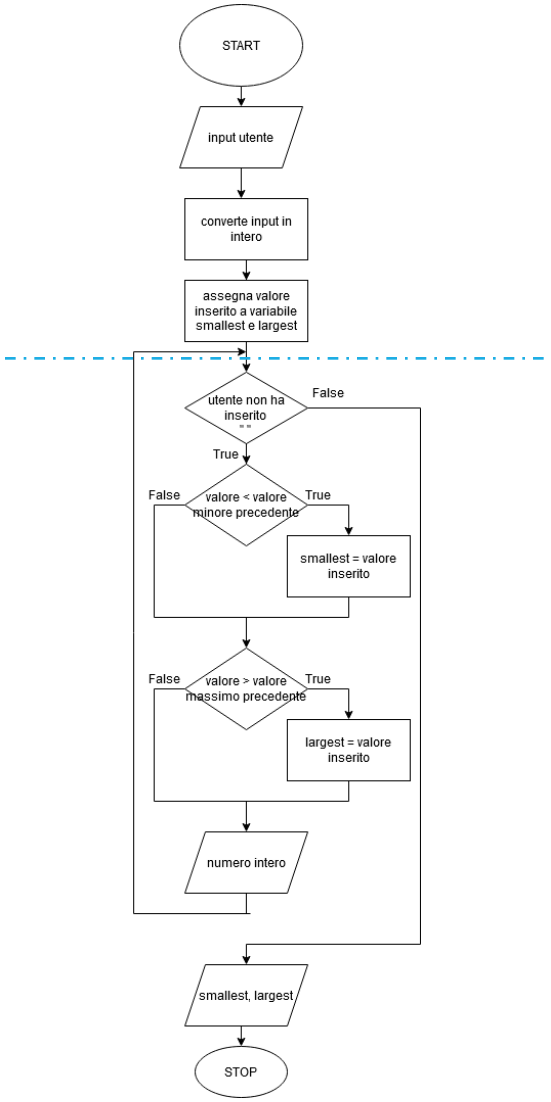
Esercizio 1 – *il flowchart* ***a.***

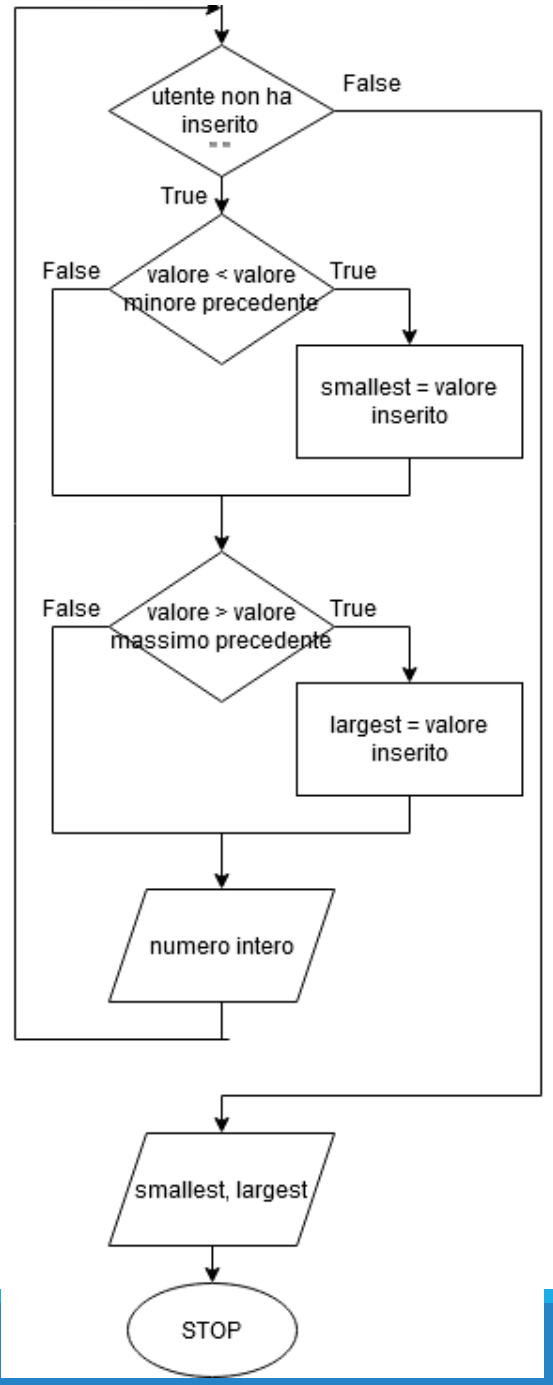






Esercizio 1 – *il flowchart* ***a.***

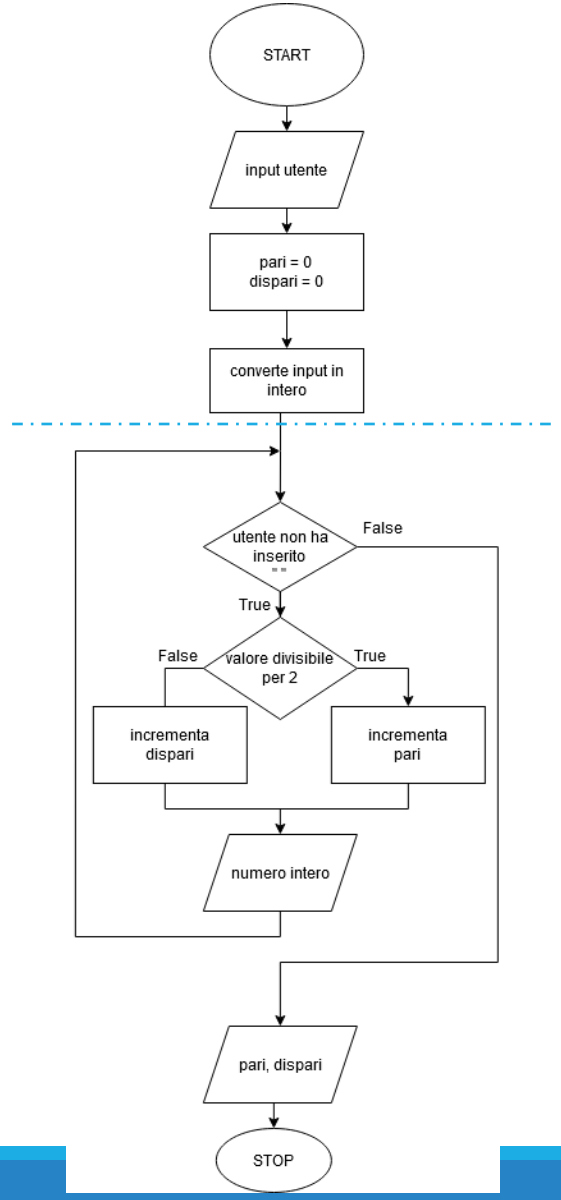


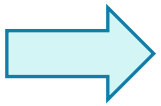


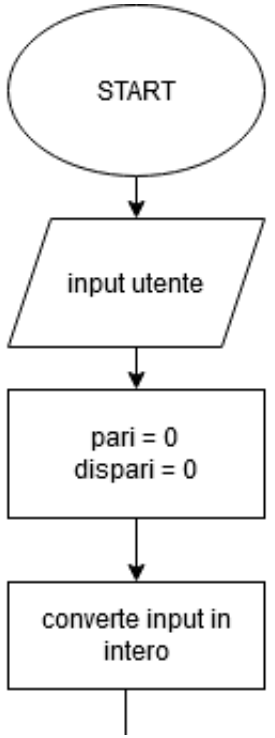
Esercizio 1 – *il codice Python* ***a.***

```
## LAB4_es1
#

# Legge l'input dell'utente
inputStr = input("Inserire un numero intero (invio per uscire): ")

value = int(inputStr) # conversione in intero
smallest = value # assegna il valore iniziale alla variabile per il minimo
largest = value

# Finchè utente non inserire riga vuota: converte input in intero
# e determina se è da considerare come nuovo valore minore o maggiore
while inputStr != "" :
   value = int(inputStr)

   if (value < smallest) :
      smallest = value
   if (value > largest) :
      largest = value

   inputStr = input("Inserire un numero intero (invio per uscire): ")

# Mostra il risultato
print("Il valore minore è", smallest, "e il valore maggiore è", largest)
```

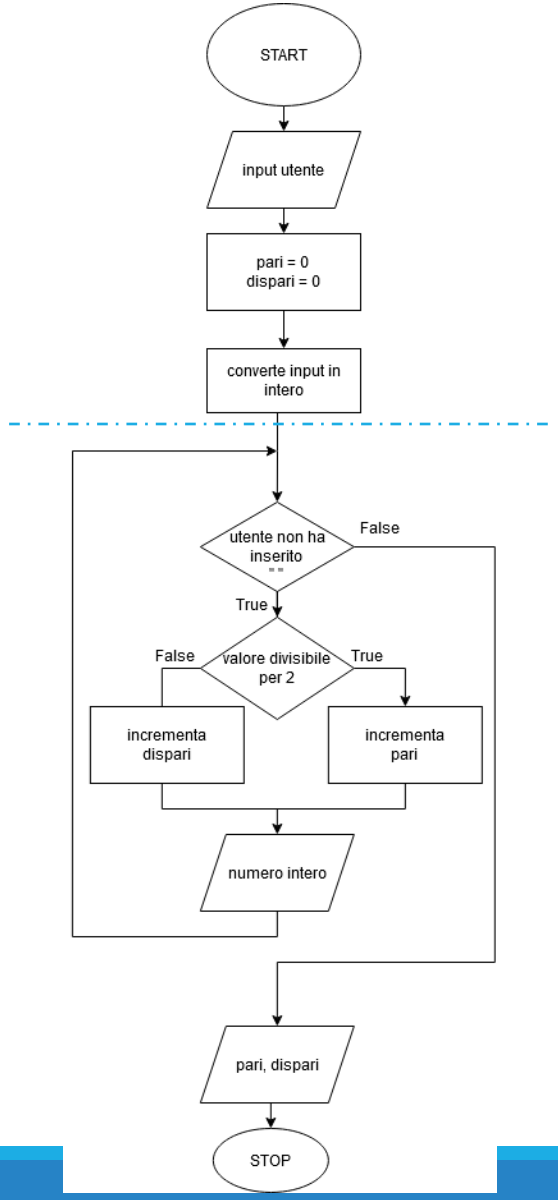
Esercizio 1 – *il flowchart* ***b.***

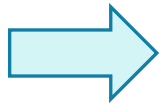


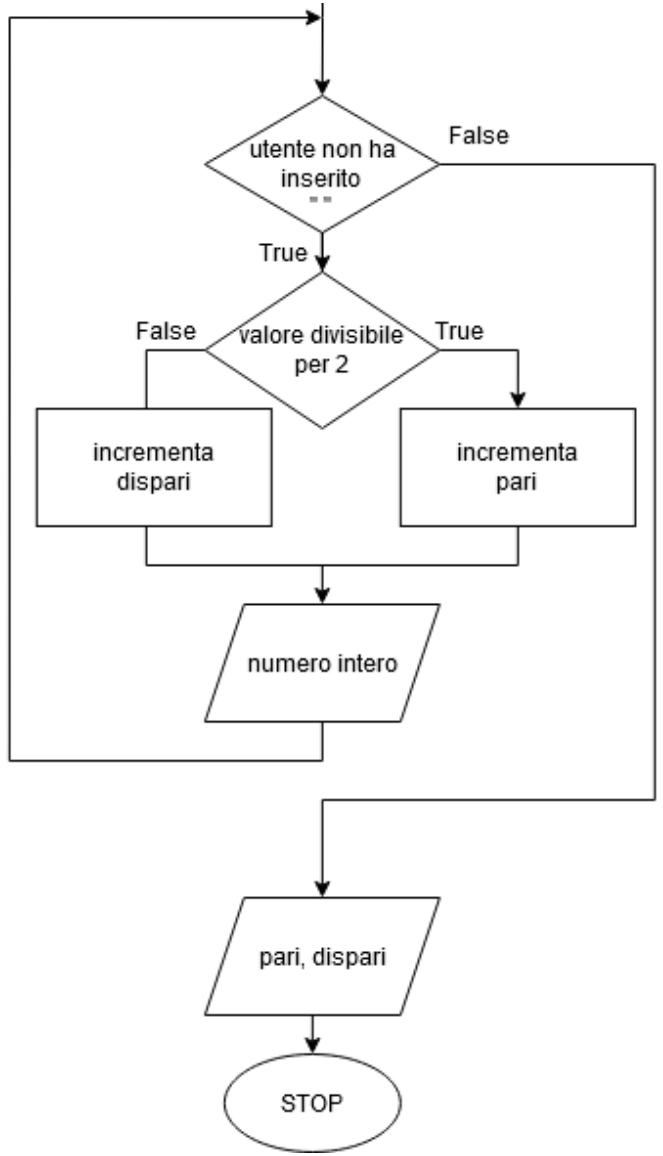




Esercizio 1 – *il flowchart* ***b.***







Esercizio 1 – *il codice Python **b**.*

```
## LAB4_es1
#

# Legge input da utente
inputStr = input("Inserire un intero (invio per uscire): ")
pari = 0
dispari = 0

while inputStr != "" :
   # conversione in intero del valore in input
   value = int(inputStr)
   if value % 2 == 0 :
      pari = pari + 1
   else :
      dispari = dispari + 1

   inputStr = input("Inserire un intero (invio per uscire): ")

# Mostra i risultati
print("L'utente ha inserito", pari, "valori pari e", dispari, "valori dispari.")
```

Esercizio 1 – *il codice Python* ***c.***

```python
## LAB4_es1

# Legge input utente
inputStr = input("Inserire un intero (invio per uscire): ")

# Aggiunge ogni valore al totale, mostra la somma parziale ad ogni
# acquisizione
totale = 0
while inputStr != "" :
    value = int(inputStr)
    totale = totale + value
    print("La somma parziale è", totale)
    inputStr = input("Inserire un intero (invio per uscire): ")
```

Esercizio 1 – *il codice Python **d**.*

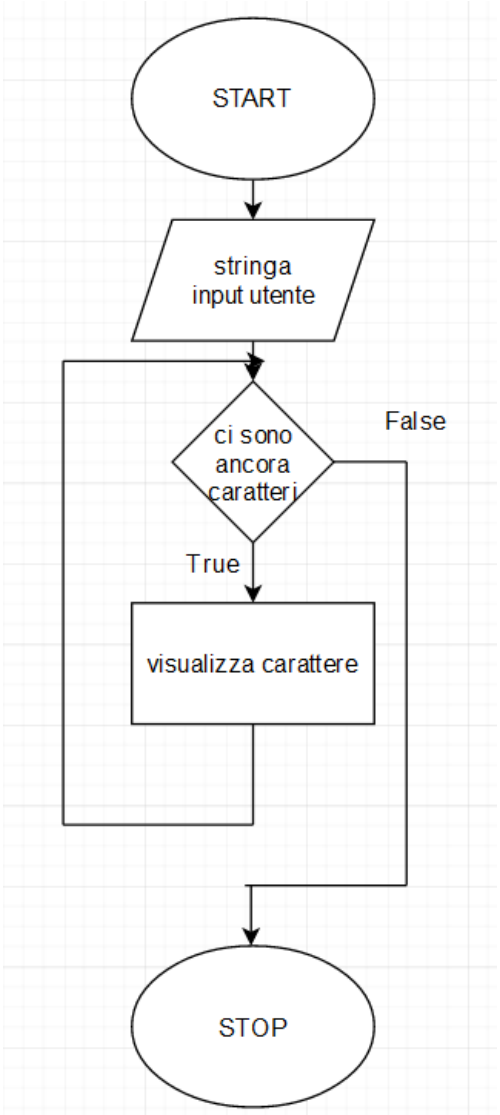
```
## LAB4_es1

# Legge input utente e salva in come valore iniziale in prev
inputStr = input("Inserire un intero (invio per uscire): ")
prev = inputStr

# Controlla ogni valore inserito dall'utente
printed = False
while inputStr != "" :
    inputStr = input("Inserire un intero (invio per uscire): ")
    # Visualizza i numeri adiacenti duplicati
    if prev == inputStr and printed == False :
        # print(inputStr, "è un duplicato.")
        printed = True
        print(inputStr)
    if prev != inputStr :
        printed = False
    prev = inputStr
```

Esercizio 2

Esercizio 2. Scrivete programmi che leggano una riga di dati in ingresso sotto forma di stringa e visualizzino quanto segue:

a. le sole lettere maiuscole della stringa;
b. a partire dalla seconda lettera della stringa, una lettera viene visualizzata e l'altra no, alternativamente;
c. la stringa con tutte le vocali sostituita da un carattere di sottolineatura (*underscore*);
d. il numero di cifre presenti nella stringa;
e. le posizioni di tutte le vocali presenti nella stringa. [P4.3]

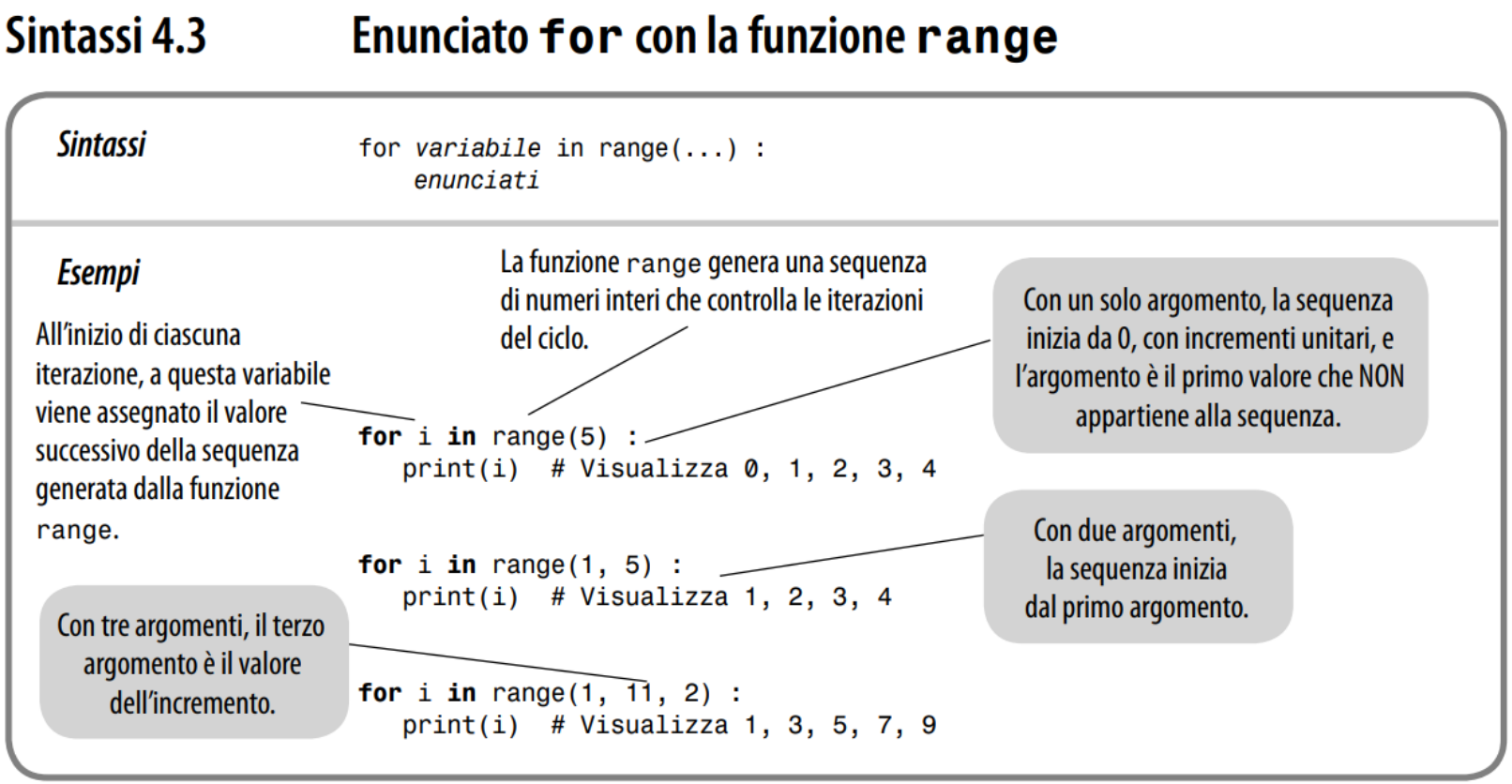
Esercizio 2 – *il flowchart* ***a.***



Esercizio 2 – *il codice Python **a**.*

```python
## LAB4_es2

# Legge l'input
inputStr = input("Inserire una stringa: ")

# Trova e mostra le lettere maiuscole
for ch in inputStr :
    if ch >= "A" and ch <= "Z" :
        print(ch, end=" ")
```

Esercizio 2 – *b.*



Esercizio 2 – *il codice Python **b**.*

```
# Read input from the user.
input_str = input("Enter a string: ")

# Display every second letter.
for i in range(1, len(input_str), 2):
    print(input_str[i], end='')
```

Esempio:

`inputStr = ‘Laboratorio’`

L	a	b	o	r	a	t	o	r	i	o
0	1	2	3	4	5	6	7	8	9	10

Esercizio 2 – *il codice Python **c**.*

```
# Read input from the user.
input_str = input("Enter a string: ")

# Check each character.  If it is a vowel display an underscore instead of
# the character.
for ch in input_str:
    if ch in "aeiouAEIOU":  # or: ch.lower() in "aeiou"
        print("_", end="")
    else:
        print(ch, end="")

# Or using replace()
# for ch in "aeiouAEIOU":
#   input_str = input_str.replace(ch,"_")
# print(input_str)

print()
```

Esercizio 2 – *il codice Python **d**.*

```
## LAB4_es2
#
# Input dell'utente
inputStr = input("Inserire una stringa: ")

# Controlla ogni carattere, se è un numero lo conta
count = 0
for ch in inputStr :
    if ch >= "0" and ch <= "9" :
        count = count + 1

print("La stringa contiene", count, "numeri.")
```

Esercizio 2 – *il codice Python* ***e.***

```
## LAB4_es2
#
# Input dell'utente
inputStr = input("Inserire una stringa: ")

# Mostra la posizione di ogni vocale presente nella stringa
print("Le vocali sono in posizione: ", end="")
for i in range(0, len(inputStr)) :
    if inputStr[i] in "aeiouAEIOU" :
        print(i, end=" ")

print()
```

Esercizio 3

Esercizio 3. Scrivete un programma che legga un numero intero, *n*, e visualizzi usando asterischi, un quadrato e un rombo pieni il cui lato abbia lunghezza *n*. Se, ad esempio, l'utente fornisce il numero 4, il programma deve visualizzare:

```
****
****
****
****

   *
  ***
 *****
*******
 *****
  ***
   *
```

[P4.22]

Esercizio 3 – *il codice Python*

```
## LAB4_es3
#
# Read the side length from the user.
sideLength = int(input("Inserire la lunghezza del lato: "))
print("Rombo di lato %d: " % sideLength)
# Visualizza il rombo
for y in range(1, sideLength) :
   print(" " * (sideLength - y) + "*" * (y * 2 - 1))
for y in range(sideLength, 0, -1) :
   print(" " * (sideLength - y) + "*" * (y * 2 - 1))

print("Quadrato di lato %d: " % sideLength)
# Visualizza il quadrato
for y in range(1, sideLength) :
   print("*" * sideLength)
```

Esercizio 4

Esercizio 4. Scrivete un programma che legga una parola e visualizzi:

a. la parola al contrario. Se, ad esempio, l'utente fornisce la stringa "`Ciao`", il programma deve visualizzare `oaiC`; [P4.9]
b. le lettere maiuscole partendo dal fondo. Se, ad esempio, l'utente fornisce la stringa "CiAo", il programma deve visualizzare `AC`.

Esercizio 4 – *il codice Python **a**.*

```python
# Read input from the user.
input_str = input("Enter a word: ")

# String length
str_len = len(input_str)

# Display the result.
print(f"{input_str} reversed is {input_str[str_len::-1]}") # with string slicing

print()
```

Esercizio 4 – *il codice Python **b**.*

```
# Read input from the user.
input_str = input("Enter a word: ")

for ch in "aeiou":
  input_str = input_str.replace(ch, '')

# String length
str_len = len(input_str)

# Reverse
input_str[str_len::-1]

# Display the result.
print(input_str)
```

Esercizio 5

Esercizio 5. *Numeri primi*. Scrivete un programma che chieda all'utente un numero intero e, poi, visualizzi tutti i numeri primi minori o uguali a tale numero. Se, ad esempio, l'utente fornisce il numero 20, il programma deve visualizzare:

2
3
5
7
11
13
17
19

[P4.17]

Esercizio 5 – *il codice Python*

```python
## LAB4_es5
#
from math import sqrt

# Input dell'utente
limit = int(input("Enter an integer: "))

# Visualizza i numeri primi minori O uguali al numero inserito
for num in range(2, limit + 1) :
   isPrime = True
   for factor in range(2, round(sqrt(num)) + 1) :
      if num % factor == 0 :
         isPrime = False
   # Visualizza il numero se è primo
   if isPrime :
      print(num)
```

Esercizio 6

Esercizio 6. Scrivete un programma che legga una parola e visualizzi tutte le sue sottostringhe, ordinate per lunghezza crescente. Se, ad esempio, l'utente fornisce la stringa "rum", il programma deve visualizzare

r

u

m

ru

um

rum

[P4.12]

Esercizio 6 – *il codice Python*

```
## LAB4_es6

# Input utente
word = input("Inserire una parola: ")

# Visualizza le sottostringhe disposte per lunghezza
for length in range(1, len(word) + 1) :
    for pos in range(0, len(word) - length + 1) :
        print(word[pos : pos + length])
```

Esercizio 7

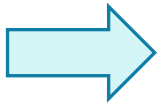
Esercizio 7. *Il gioco di Nim*. Si tratta di un gioco molto noto, con un certo numero di varianti: quella qui descritta ha una strategia vincente davvero interessante. Due giocatori prelevano alternativamente biglie da un mucchietto. Ad ogni mossa, un giocatore sceglie quante biglie prendere: almeno una e al massimo metà delle biglie disponibili. Poi è il turno dell'altro giocatore. Il giocatore che prende l'ultima biglia perde la partita.

Scrivete un programma che consenta all'utente di giocare contro il computer. Generate un numero intero compreso tra 10 e 100 e usatelo come dimensione iniziale del mucchietto di biglie. Generate un numero intero, 0 o 1, e utilizzatelo per decidere se sarà l'utente o il computer a giocare per primo. Generate un numero intero, 0 o 1, e usatelo per decidere se il computer giocherà in modo intelligente o stupido: giocando in modo stupido, ad ogni sua mossa il computer semplicemente preleva dal mucchietto un numero di biglie casuale (ma valido, cioè compreso tra 1 e n/2, se nel mucchietto sono rimaste n biglie); in modalità intelligente, invece, preleva un numero di biglie tale che il numero di quelle che rimangono nel mucchio sia una potenza di due diminuita di un'unità, cioè 3, 7, 15, 31 o 63. Quest'ultima è sempre una mossa valida, tranne quando la dimensione del mucchio è proprio uguale a una potenza di due diminuita di un'unità: in tal caso il computer fa una mossa scelta a caso (ovviamente tra quelle valide). Come potrete verificare sperimentalmente, il computer non può essere battuto quando gioca in modalità intelligente e fa la prima mossa, a meno che la dimensione iniziale del mucchio non sia 15, 31 o 63. Analogamente, un giocatore umano che faccia la prima mossa e conosca la strategia qui descritta è in grado di battere il calcolatore.

[P4.23]

Esercizio 7 – *il codice Python*

Consultare la documentazione!



```python
## LAB4_es7
#

from random import randint
from math import log

# Definisce
HUMAN = 0
COMPUTER = 1
SMART = 0
DUMB = 1

# Mucchietto iniziale di biglie
pile = randint(10, 100)
# Turno iniziale
turn = randint(0, 1)
# Strategia
strategy = randint(0, 1)
```

Esercizio 7 – *il codice Python*

```
# Svolgimento del gioco
while pile > 0 :
    if turn == COMPUTER :
        if pile == 1 :
            # Prende ultima biglia
            take = 1
        elif strategy == DUMB :
            # Prende un numero variabile, consentito, di biglie dal mucchio
            take = randint(1, pile // 2)
        elif pile == 3 or pile == 7 or pile == 15 or pile == 31 or pile == 63 :
            # Il computer gioca in modalità intelligente, ma non può fare una mossa intelligent
            # Prende un numero variabile, consentito, di biglie dal mucchio
            take = randint(1, pile // 2)
        else :
            # Il computer gioca in modalità intelligente e può fare una mossa intelligente
            # Prende un numero di biglie in modo che il mucchio ne conterrò una potenza di 2, meno 1
            take = pile - 2 ** int(log(pile, 2)) + 1

        pile = pile - take
        print("Il computer ha preso %d biglie, lasciandone %d." % (take, pile))
        print()
        turn = HUMAN
    elif turn == HUMAN :
        print("E' il tuo turno.   Nel mucchio ci sono", pile, "biglie.")

        # Chiede all'utente quante biglie prenderà
        take = int(input("Quante biglie prendi? "))
        while take < 1 or take > max(pile // 2, 1) :
            print("Mossa non consentita.")
            take = int(input("Quante biglie prendi? "))

        pile = pile - take
        print("Ora nel mucchio ci sono", pile, "biglie.")
        print()
        turn = COMPUTER

# Determina il vincitore
if turn == COMPUTER :
    print("Ha vinto il computer!")
else :
    print("Hai vinto!")
```