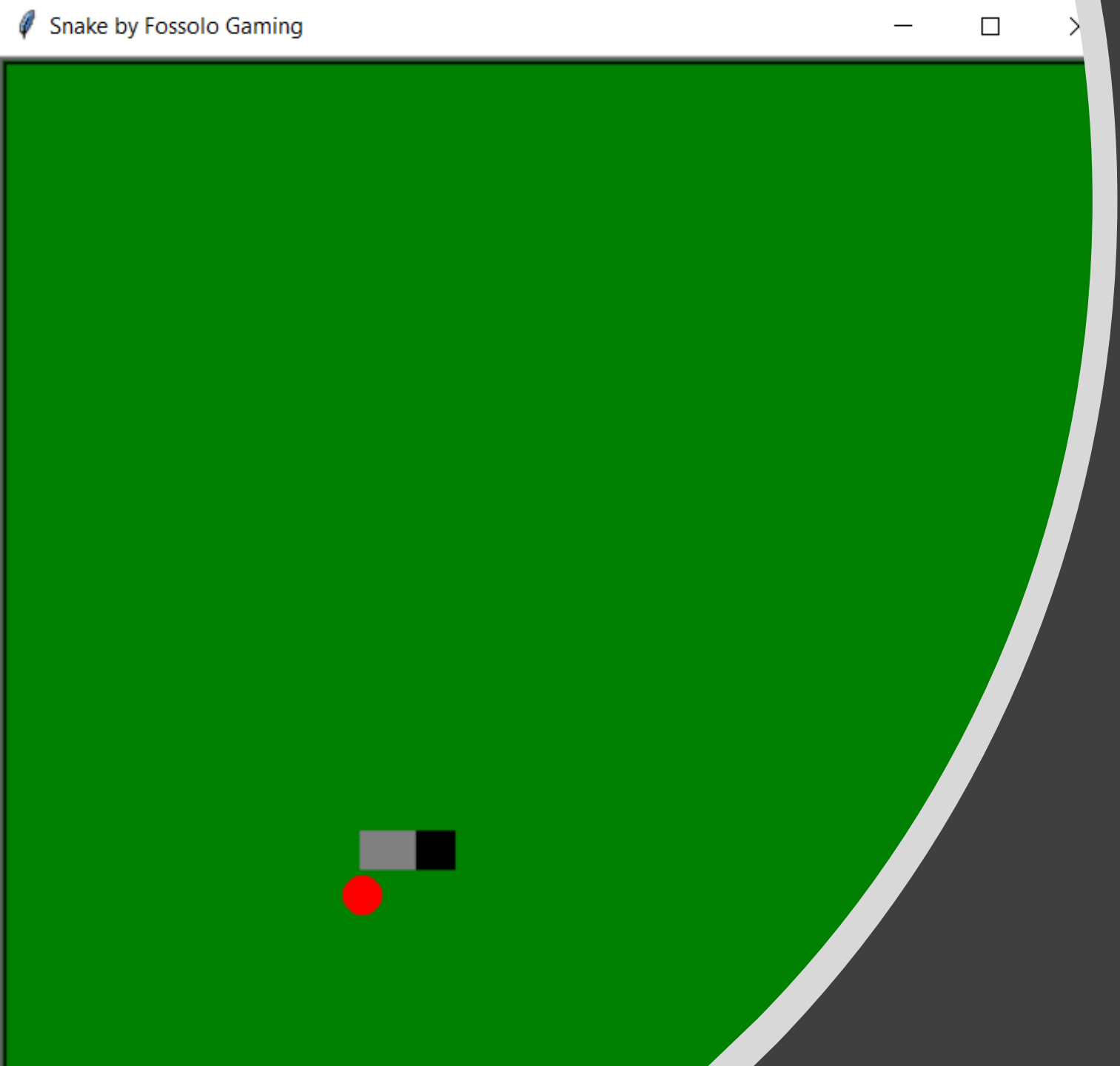
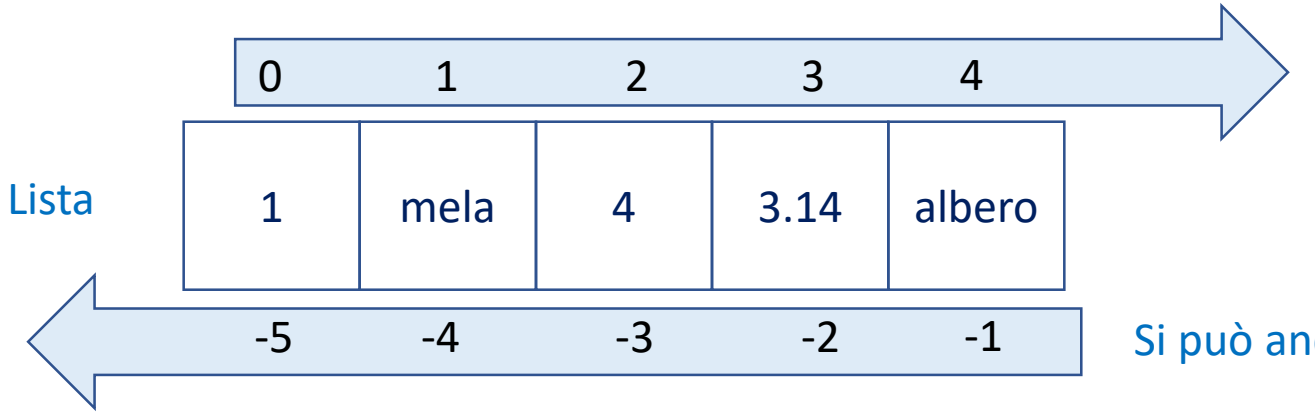


Liste ed esempio di Snake con Python turtle

Roberto SEMPRINI

Le liste con Python

Le **Liste** sono un tipo di variabili. Si può vedere come un insieme di scatole che contengono altre variabili



Indice di posizione nella lista. Comincia sempre da «0»

Si può anche percorrere al contrario

Sintassi: **elenco = [1, "mela", 4, 3.14, "albero"]** ← La lista va racchiusa tra parentesi quadre

Nome della lista

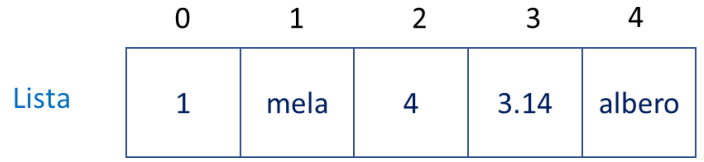
assegnazione

Le stringhe di caratteri vanno tra doppi apici

Lunghezza della lista **len(elenco)**

```
>>> elenco = [1, "mela", 4, 3.14, "albero"]
>>> elenco
[1, 'mela', 4, 3.14, 'albero']
>>> len(elenco)
5
```

Le liste con Python



Accedere agli elementi della lista

```
>>> elenco
[1, 'mela', 4, 3.14, 'albero']
```

Accesso all'intera lista

```
>>> elenco[0]
1
>>> elenco[-1]
'albero'
>>> elenco[-5]
1
```

Accesso ai singoli elementi della lista, specificando l'indice di posizione tra parentesi quadre

Questo è il secondo elemento

```
>>> elenco[1:3]
['mela', 4]
```

range degli indici da 1 a 3

Accesso ad un sottoinsieme (intervallo di elementi)

Nota: vale il principio del ciclo **for** con il **range**

Il ciclo si ferma prima di arivare all'ultima posizione del range !

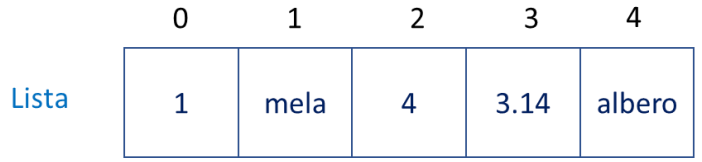
```
>>> elenco[1:]
['mela', 4, 3.14, 'albero']
```

Accesso ad un sottoinsieme con intervallo posizioni aperto

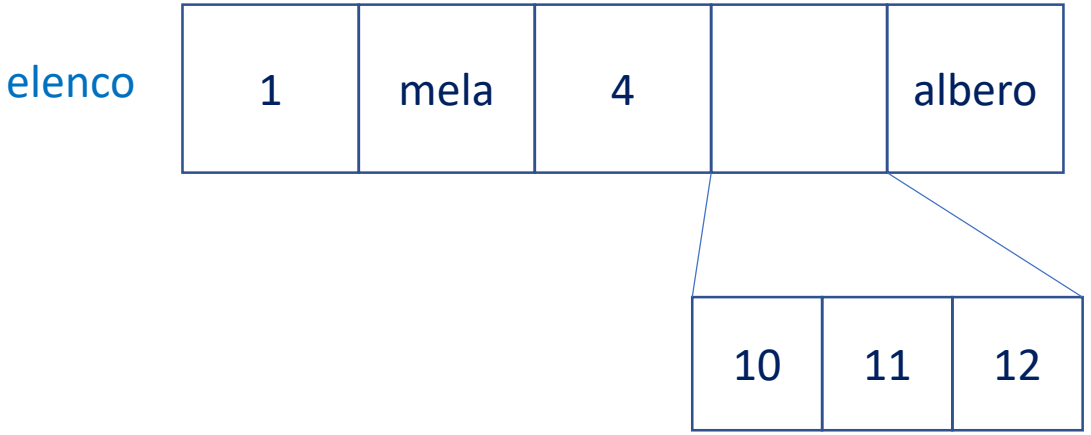
Accesso agli elementi della lista con ciclo **for**

```
>>> for elemento in elenco:
        print(elemento)

1
mela
4
3.14
albero
```

Le liste con Python



Una lista può a sua volta contenere come suo elemento un’altra lista

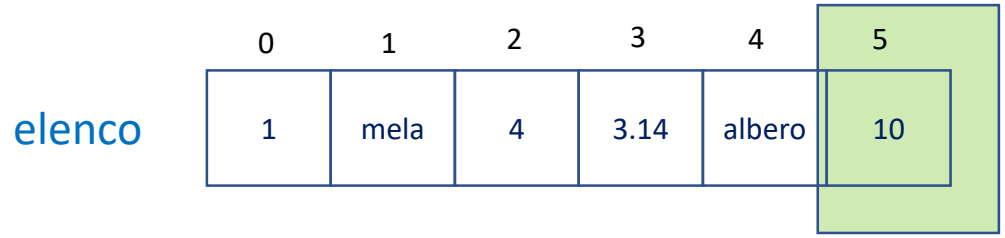


elenco = [1, "mela", 4, [10, 11, 12], "albero"]

Modificare liste con Python

	0	1	2	3	4
Lista	1	mela	4	3.14	albero

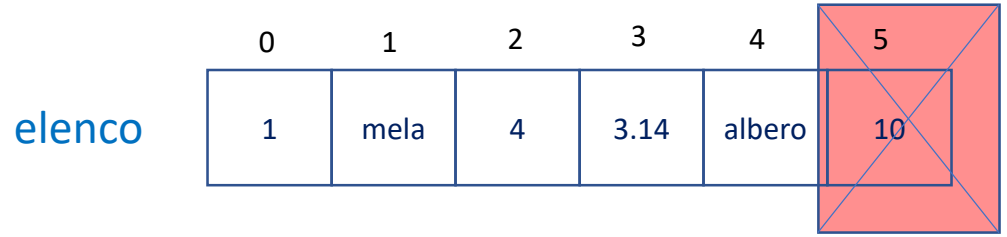
Aggiungere un elemento alla fine (**append**)



elenco.**append**(10)

```
>>> elenco.append(10)
>>> elenco
[1, 'mela', 4, 3.14, 'albero', 10]
```

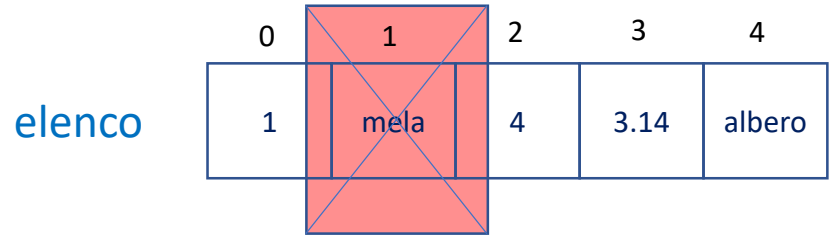
Rimuovere l'ultimo elemento della lista (**pop**)



elenco.**pop**()

```
>>> elenco.pop()
10
>>> elenco
[1, 'mela', 4, 3.14, 'albero']
```

Rimozione di un elemento qualunque della lista (**del**)



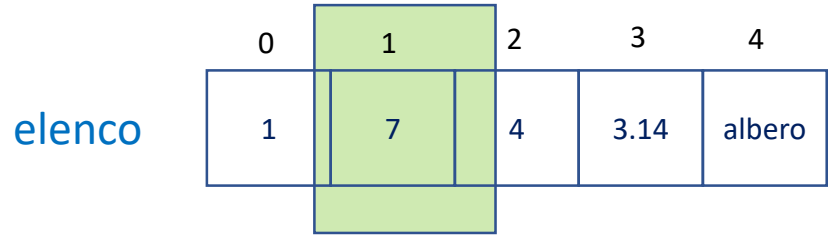
del elenco[1]

```
>>> del elenco[1]
>>>
>>> elenco
[1, 4, 3.14, 'albero']
```

Modificare liste con Python

	0	1	2	3	4
Lista	1	mela	4	3.14	albero

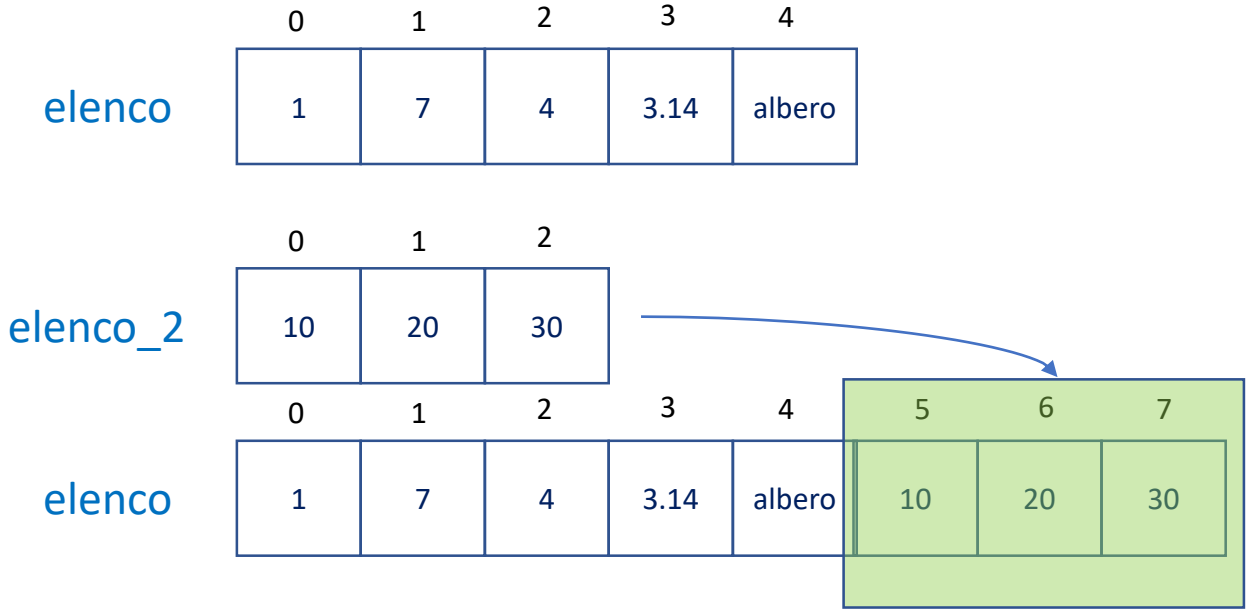
Inserire un elemento all'indice specificato (**insert**)



elenco.**insert**(1, 7)

```
>>> elenco.insert(1,7)
>>> elenco
[1, 7, 4, 3.14, 'albero']
```

Concatenare due liste (+) oppure (extend)

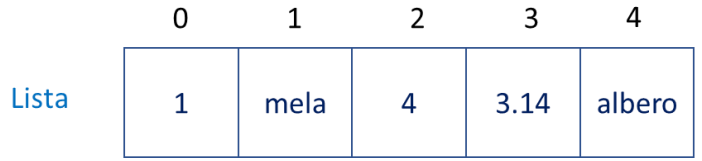


elenco+elenco_2

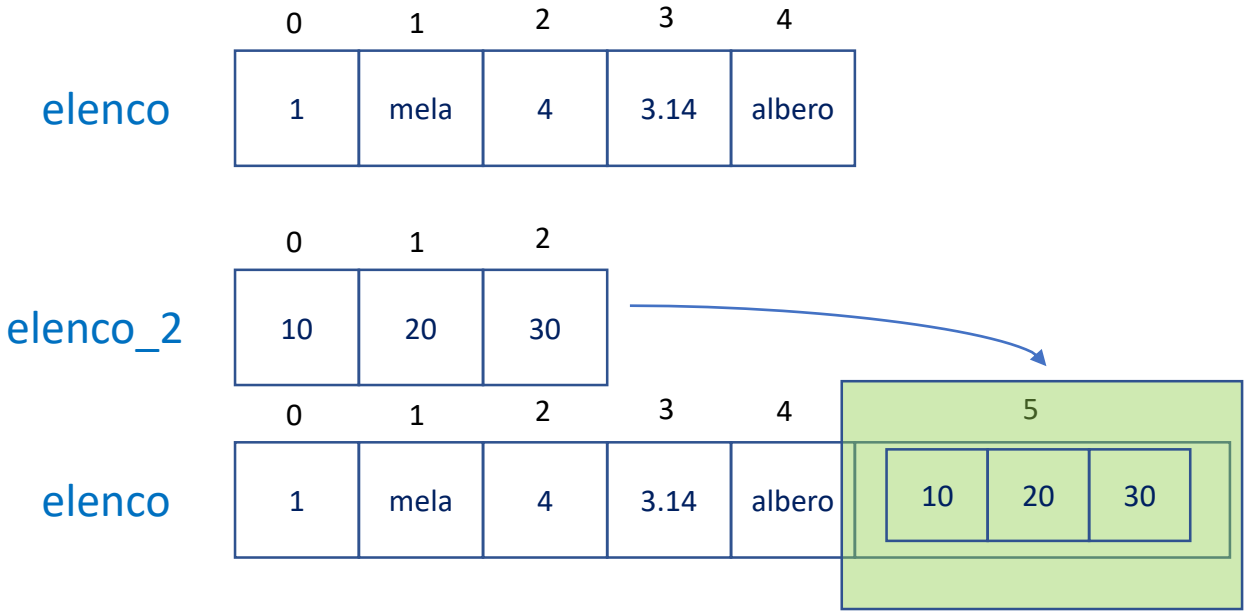
elenco.**extend**(elenco_2)

```
>>> elenco_2=[10, 20, 30]
>>> elenco+elenco_2
[1, 7, 4, 3.14, 'albero', 10, 20, 30]
```

Modificare liste con Python



Concatenare due liste (**append**)



elenco.**append**(elenco_2)

```
>>> elenco
[1, 'mela', 4, 3.14, 'albero', 30]
>>> elenco.append(elenco_2)
>>> elenco
[1, 'mela', 4, 3.14, 'albero', 30, [10, 20, 30]]
```

Lista di oggetti Turtle (Snake)

Creare e spostare uno snake

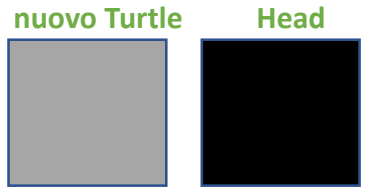
- Snake fatto con una testa (head) ed un corpo
- Ogni rettangolino è un oggetto Turtle
- Testa (head) -> Turtle nero
- Corpo (body) -> lista di tanti oggetti Turtle grigi

- Spostare la *head* e la lista *body* dentro lo Screen

Lista di oggetti Turtle (Snake)

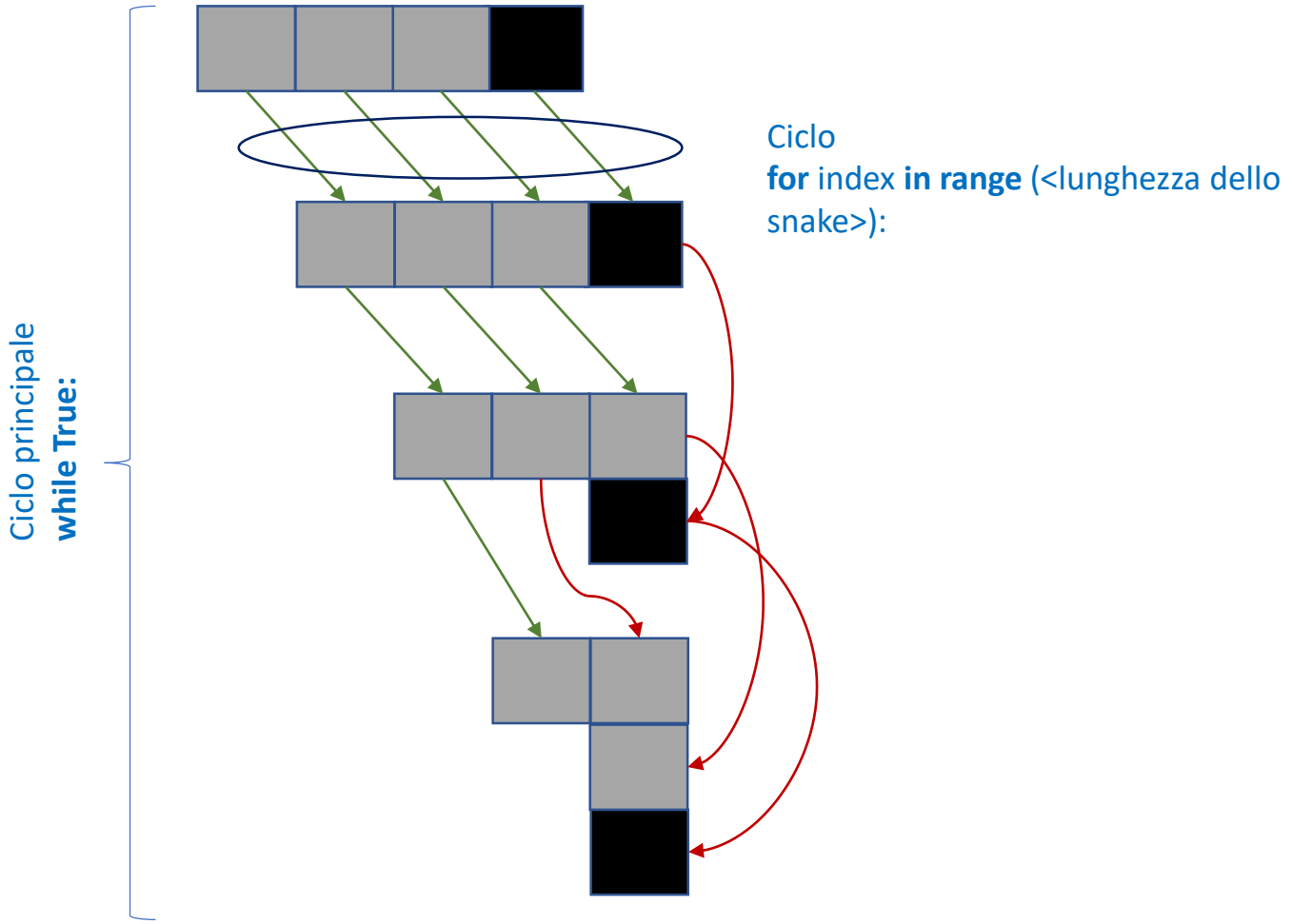
Creare e spostare uno snake

Ogni volta che accade un evento (la testa dello snake mangia del cibo) si crea un altro pezzo del body dello snake -> creare una nuova istanza di Turtle



Il nuovo Turtle va appeso al resto del body dello snake per creare un unico body

Ad ogni ciclo, la testa si sposta, ed ogni pezzo che segue deve prendere il posto di quello che prima gli stava davanti



Snake

Passaggi della programmazione

- Creare uno Screen
- Creare un oggetto Turtle per la head dello snake
- Creare una lista vuota per il body dello snake

- Creare il ciclo del programma, dove ad ogni ciclo
  - Si crea un nuovo pezzo del body
  - Si fa crescere la lista del body
  - Si sposta la head
  - Ogni elemento della lista del body segue la head

Snake

Passaggi della programmazione

- Creare uno Screen
- Creare un oggetto Turtle per la head dello snake
- Creare una lista vuota per il body dello snake

```
# -------------
# Liste e creazione Snake
# ---------------

import turtle
import time #serve per la funzione "delay"

delay = 0.3 #ritardo di 0.3 sec

# ------------------------------------
# set up the screen
# ------------------------------------
win = turtle.Screen()
win.title ("Snake by Fossolo Gaming")
win.bgcolor("green")
win.setup(width=600, height=600)



# ------------------------------------
# Creare la testa dello snake
# ogni forma della penna Turtle ha dimensioni standard 20x20 pixels
# ------------------------------------

head=turtle.Turtle()
head.shape("square")
head.color("black")
head.penup()
head.goto(0,0)


# ------------------------------------
# Creazione lista per i pezzi del corpo dello snake
# inizialmente vuota
# ------------------------------------

body_snake =[]
```

Altre funzioni per lo Snake

Animazione della finestra

La **finestra** è un oggetto che viene generato durante l'esecuzione del programma. **Non è il programma in corso di esecuzione**

Per creare un'**animazione** serve ripetere più volte un insieme di istruzioni, ad esempio usando il ciclo continuo con l'istruzione **while**

Il ciclo **while** ripete le istruzioni all'interno del suo corpo.

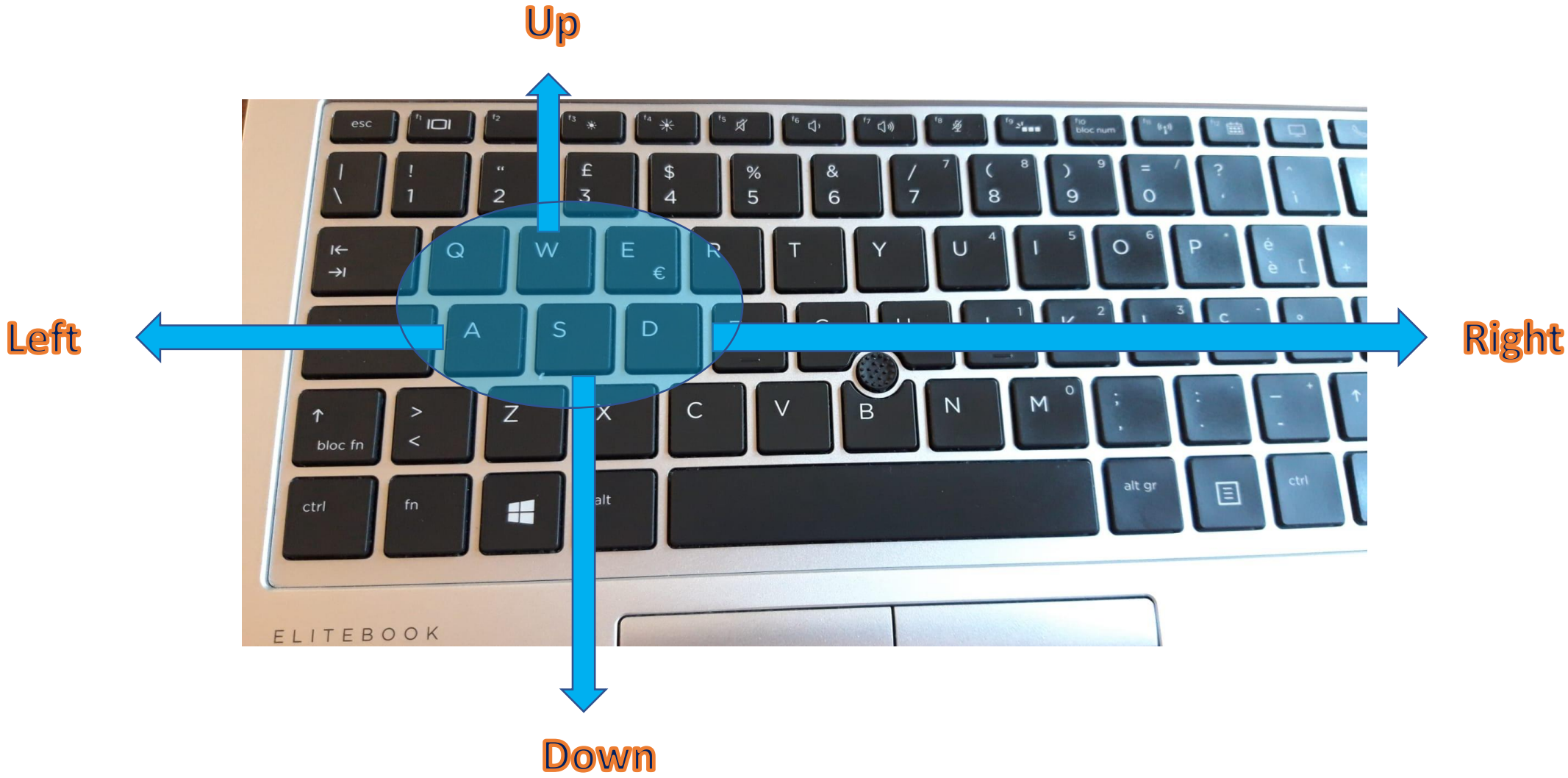
All'interno del ciclo **while è necessario richiedere l'aggiornamento della finestra**, altrimenti il programma continua a girare e la finestra rimane bloccata, non riuscendo più ad avere il controllo sulla finestra
-> python not responding

```
#Animazione finestra Turtle

import turtle
win = turtle.Screen()

while True:
    win.update()
```

Uso del metodo **update** della libreria turtle, all'interno del ciclo **while**

Altre funzioni per lo Snake

Acquisizione eventi da tastiera



Altre funzioni per lo Snake

Acquisizione eventi da tastiera

```
import turtle
win = turtle.Screen()


# -----------------------------------
# Keyboard bindings
# -----------------------------------

win.listen()
win.onkeypress(go_up, "w")
win.onkeypress(go_down, "s")
win.onkeypress(go_left, "a")
win.onkeypress(go_right, "d")
```

Il metodo **listen** predispone la finestra di turtle ad ascoltare eventi da tastiera

Il metodo **onkeypress** associa l'evento di pressione di un tasto ad una funzione

Carattere del tasto della keyboard

Funzione da definire nel programma

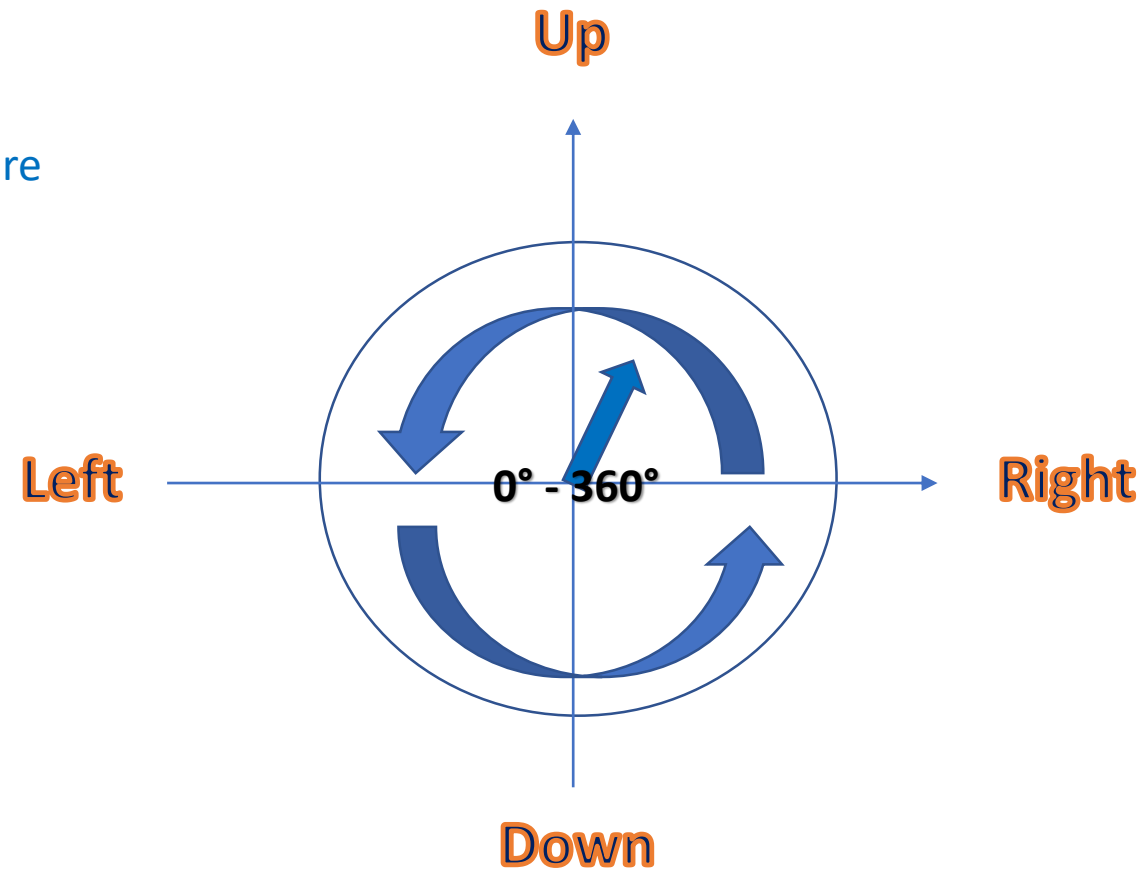
Altre funzioni per lo Snake

Movimenti della Turtle

direction è un attributo della tartaruga, permette di memorizzare o di cambiare la sua direzione relativa al sistema di riferimento assoluto.

<tartaruga>.direction = <angolo>

La direzione assoluta ha angolo 0 verso destra e considera come verso positivo di rotazione il verso antiorario.
Valori predefiniti "**Up**", "**Down**", "**Left**", "**Right**", "**Stop**",



Altre funzioni per lo Snake

Movimenti della Turtle

```
#Animazione finestra Turtle

import turtle
win = turtle.Screen()
head = turtle.Turtle()



# ------------------------------------
# Keyboard bindings
# ------------------------------------

win.listen()
win.onkeypress(go_up, "w")
win.onkeypress(go_down, "s")
win.onkeypress(go_left, "a")
win.onkeypress(go_right, "d")
```

Definizione della funzione che associ tasto all'attributo di **direction** della tartaruga

```
def go_up():
    head.direction = "up"

def go_down():
    head.direction = "down"

def go_left():
    head.direction = "left"

def go_right():
    head.direction = "right"
```

Stessa definizione, ma con il vincolo c la direzione non si può rigirare su di s

```
def go_up():
    if head.direction != "down":
        head.direction = "up"
def go_down():
    if head.direction != "up":
        head.direction = "down"
def go_left():
    if head.direction != "right":
        head.direction = "left"
def go_right():
    if head.direction != "left":
        head.direction = "right"
```

Associazione tasti alle funzioni sopra definite

Altre funzioni per lo Snake

Movimenti della Turtle

Incremento delle coordinate nella direzione selezionata di 20 pixels

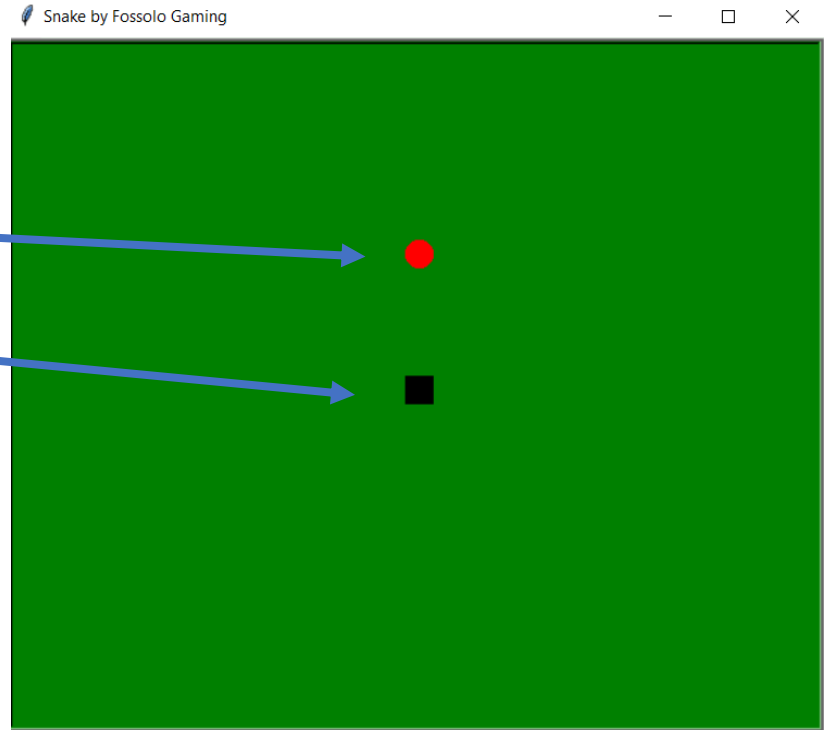
```
def move():
    if head.direction=="up":
        y=head.ycor()
        head.sety(y+20)

    if head.direction=="down":
        y=head.ycor()
        head.sety(y-20)

    if head.direction=="left":
        x=head.xcor()
        head.setx(x-20)

    if head.direction=="right":
        x=head.xcor()
        head.setx(x+20)
```

Altre funzioni per lo Snake

Cattura del cibo

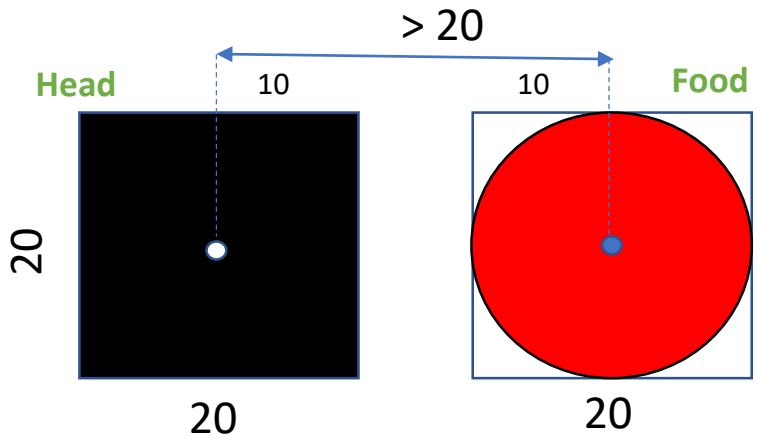
2^ Turtle: Food

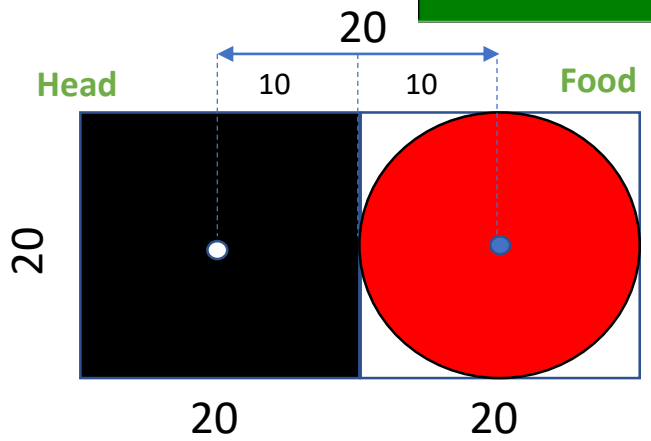
1^ Turtle: Head dello snake



Criterio per determinare se la testa dello snake arriva al cibo ?

Dimensioni base di ciascun Turtle: 20 x 20 pixels
Si toccano se distanza tra i centri < 20

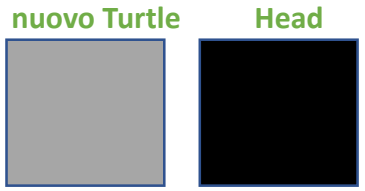




Altre funzioni per lo Snake

Come far crescere il corpo dello snake

Ogni volta che accade un evento (la testa dello snake mangia del cibo) si crea un altro pezzo del body dello snake -> creare una nuova istanza di Turtle



Il nuovo Turtle va appeso al resto del body dello snake per creare un unico body

Ad ogni ciclo, la testa si sposta, ed ogni pezzo che segue deve prendere il posto di quello che prima gli stava davanti