python

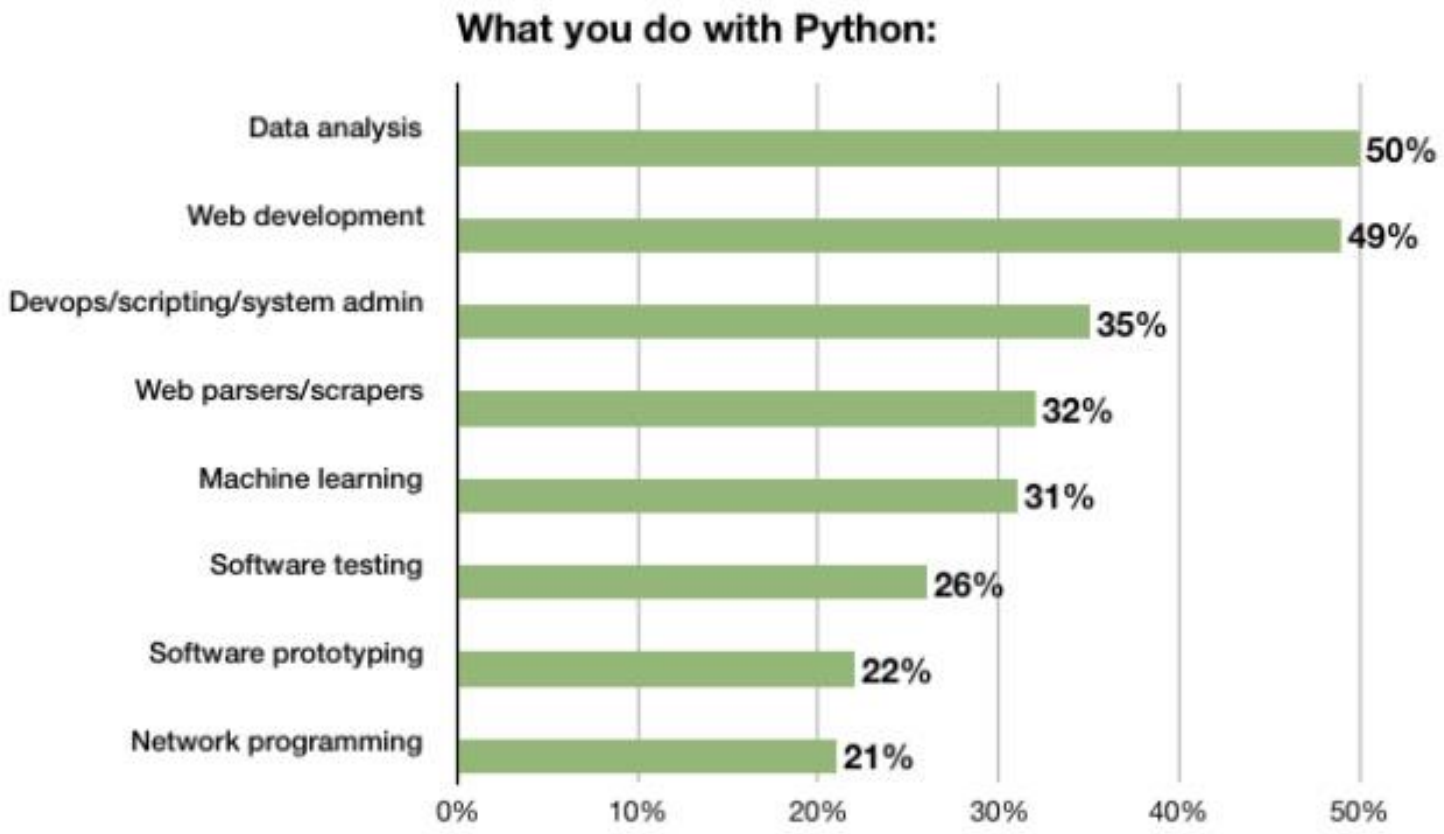
- python è un linguaggio ideato da Guido van Rossum nel ***1989***
- è molto utilizzato come linguaggio di ***scripting***
- è un linguaggio *compilato e* ***interpretato***
- è dotato di un ***ambiente interattivo*** in cui dare comandi e ottenere immediatamente risultati
- caratteristica (quasi) unica: l'***indentazione*** è ***obbligatoria***

- ***free***
  - è possibile utilizzarlo e distribuirlo senza restrizioni di copyright (open-source)
- ***facile*** da usare
  - linguaggio di alto livello (semplice e potente)
  - sintassi facile da imparare
- ***portabile***
  - linguaggio interpretato, il codice può essere eseguito su qualsiasi piattaforma purché abbia l'interprete Python installato (Unix, Linux, Windows, macOS, Android, iOS …)
- ***multi-paradigma***
  - supporta sia la programmazione procedurale, la programmazione ad oggetti e diversi elementi della programmazione funzionale

chi usa python

- Python è uno dei linguaggi più utilizzati
  - https://www.tiobe.com/tiobe-index/
- è utilizzato in molte grandi aziende del mondo informatico (e non informatico)
  - la NASA usa Python per lo sviluppo di sistemi di controllo;
  - Yahoo! ha sviluppato in Python alcuni servizi di internet;
  - Google, Youtube e RedHat usano Python

campi di applicazione

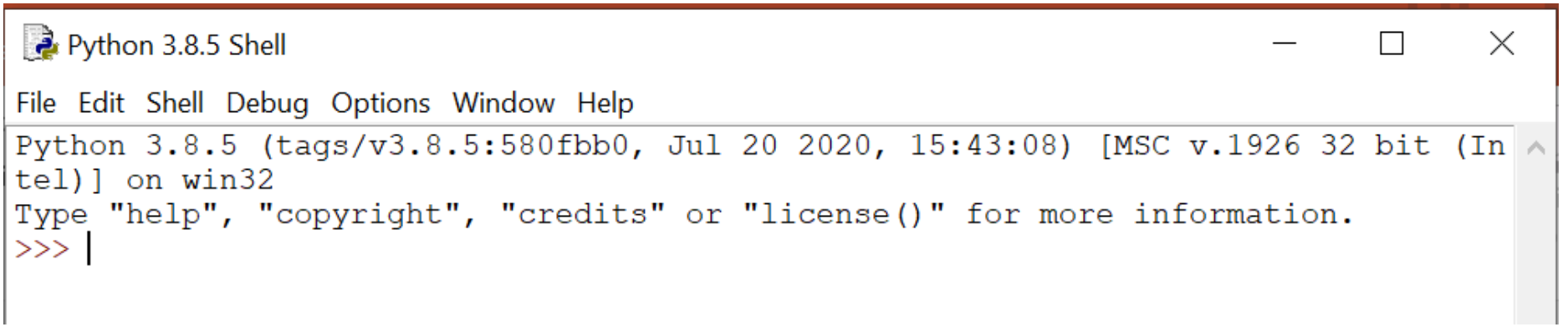


- https://www.python.org/
  - interprete Python
  - ambiente di sviluppo ***idle***

- oltre a idle è utile utilizzare un editor semplice ma più completo
  - https://www.geany.org/
- per un primo approccio:
  - playground (http://www.ce.unipr.it/brython/)


Download the latest version for Windows
Download Python 3.8.5
Looking for Python with a different OS? Python for Windows,
Linux/UNIX, Mac OS X, Other

IDLE


Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>

- un ***tipo di dato*** specifica
  - un insieme di valori
  - un insieme di operazioni ammesse su questi valori
- **`int, float, bool, str`**
- operazioni aritmetiche, logiche, confronti

```
>>> print(100)
100
>>> print("CISITA Parma")
CISITA Parma
>>> print(3.14)
3.14
>>> type(100)
<class 'int'>
>>> type("CISITA Parma")
<class 'str'>
>>> type(3.14)
<class 'float'>
>>>
```

```
>>> 10-3
7
>>> 5+3
8
>>> 10/3
3.3333333333333335
>>> 10//4
2
>>> 5*4
20
>>> 2**5
32
```

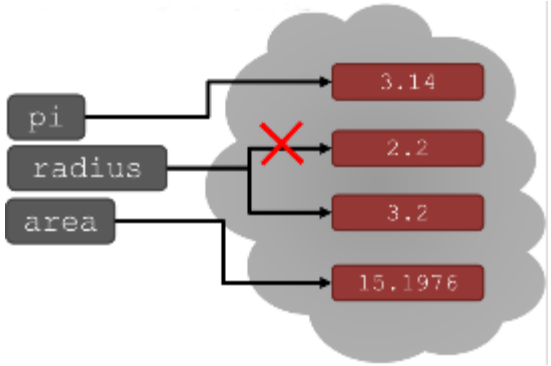
```
>>> 5>4
True
>>> 5<5
False
>>> 5<=5
True
>>> 3>2 and 4>5
False
>>> 3>2 or 4>5
True
>>> not 3>2
False
```

- ***int*** per numeri interi
- ***float*** per i numeri reali
  - operazioni di base `+`, `-`, `*`, `/`
  - divisione intera `//`, resto della divisione intera `%`, potenza `**`
  - confronti: `<`, `<=`, `>`, `>=`, `==`, `!=`
- ***bool*** per valori booleani: `True`, `False`
  - operazioni: `and`, `or`, `not`
  - i confronti danno un risultato booleano

```
>>> -2 // 3 # divisione intera
-1
>>> -2 % 3   # resto divisione
1
>>> 2 ** 1000  # potenza
10715086071862673209484250490600018105614048117055336074437503883703510511249361224931983788156958581275946729175531468251871452856923140435984577574698574803934567774824230985421074605062371141877954182153046474983581941267398767559165543946077062914571196477686542167660429831652624386837205668069376
```

assegnamento (assegnazione)

- una ***variabile*** può essere definita come un contenitore per memorizzare un risultato
- **assegnamento**, operatore **=**
  - a sinistra una variabile
  - a destra un valore (o una espressione)
- non confondere il confronto di uguaglianza **==** con l'assegnamento **=**




```
>>> pi = 3.14  # Assignment
>>> radius = 2.2
>>> area = pi * (radius ** 2)
>>> area
15.1976
>>> radius = radius + 1
# guess radius... and area!
```

variabili

- una ***variabile*** è definita da
  - un ***nome*** (etichetta - identificatore)
  - associato ad un ***valore*** (oggetto)
- un oggetto assegnato a più variabili: ***non viene copiato, ma riceve più etichette***
- il ***tipo*** della variabile ***dipende*** dal ***valore*** attualmente assegnato *(tipizzazione dinamica)*
- una variabile ***non*** deve essere ***dichiarata***, ma ***deve essere inizializzata***

```
>>> x = 100
>>> DELTA = 5   # Constants: UPPER_CASE
>>> x = x + DELTA  # Variables:
lower_case
>>> next_position = x
# Use explicative names!
```

nei linguaggi a tipizzazione dinamica le variabili hanno tipi che possono cambiare durante l'esecuzione di un programma, di solito a causa di assegnamenti
i linguaggi a tipizzazione dinamica sono spesso interpretati
linguaggi a tipizzazione dinamica: JavaScript, PHP, Prolog, Python, Ruby …

- **str**
  - sequenze di caratteri Unicode
  - primi 128 codici Unicode == ASCII
  - a capo: '\n' (10, o 13-10...)
  - tabulazione: '\t' (9)

```
>>> str1 = "Monty Python's "
>>> str2 = 'Flying Circus'
>>> result = str1 + str2
>>> result
"Monty Python's Flying
Circus"
```

Unicode era stato originariamente pensato come una codifica a 16 bit per codificare 65.535 (2^16 -1) caratteri.
Ora lo standard Unicode prevede una codifica fino a 21 bit e supporta un repertorio di codici numerici che possono rappresentare circa un milione di caratteri

tabella ASCII

A. Ferrari

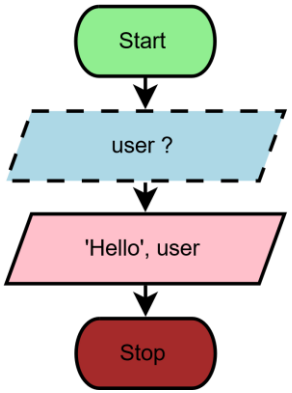
Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

- il confronto tra stringhe segue l'ordine lessicografico
- operatori di confronto: **<, <=, >, >=, ==, !=**

```
>>> 'first' < 'second'
True
>>> 'Second' <  'first'
True
>>> chr(90)
'Z'
>>> ord('Z')
90
```

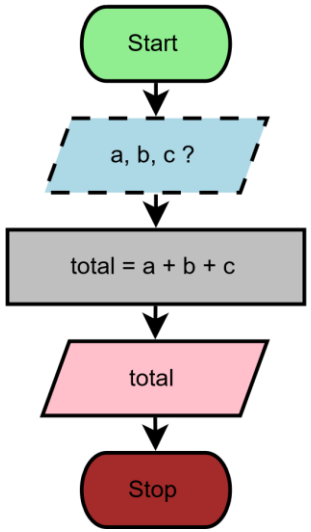
- **input**
  - ***legge*** una riga di ***testo*** (dallo standard input – ***tastiera***)
  - e la inserisce in una variabile
  - prima mostra un messaggio
- **print**
  - ***scrive*** una serie di valori sullo standard outpt (***schermo***)
  - tra i valori (parametri) viene inserito uno spazio

```
user = input("What's your name? ")

print("Hello,", user)
```



somma di 3 valori numerici

```
a = float(input("Insert 1st val: "))
b = float(input("Insert 2nd val: "))
c = float(input("Insert 3rd val: "))

total = a + b + c

print("The sum is", total)
```



- oltre alla possibilità di interagire direttamente con l'ambiente python ***IDLE*** permette di scrivere ed eseguire ***programmi*** (moduli)
- **`file -> new file`**
- un programma sorgente python è un semplice file di testo con estensione ***.py***
- è possibile utilizzare un qualsiasi editor testuale per scrivere codice python

1. Dati due valori interi, fornire come risultato il quadrato della loro differenza
2. Dato un valore che rappresenta il perimetro di un quadrato, fornire come risultato il valore del lato e dell'area.
3. *Col termine gallone (abbreviato gal) si identificano alcune unità di misura di capacità utilizzate generalmente per i liquidi. Il gallone non fa parte del Sistema Internazionale di unità di misura ma è tuttora comunemente utilizzato in paesi come Stati Uniti e Gran Bretagna.*
   - *In Gran Bretagna si utilizza il gallone imperiale (gal Imp) o gallone inglese:*
     - *1 gal Imp = 4,54609 Litri*
   - *Negli Stati Uniti si usa il gallone americano (gal US):*
     - *1 gal US = 3,785411784 Litri*
   - Realizzare un programma che risolve il seguente problema: si riceve in input la quantità di liquido (espressa in litri) presente in un recipiente, si fornisce poi in output la quantità equivalente espressa in galloni inglesi e in galloni americani

- python è ricco di ***moduli*** (librerie di funzioni e costanti) che possono essere utilizzate nei propri programmi
- **`import <nome libreria>`**
- **`<nome libreria>.<nome funzione>(<parametri>)`**
- ***math*** offre una serie di funzioni e di costanti di tipo *matematico*
- ***random*** per la generazione di numeri *pseudocasuali*
- Python Standard Library: http://docs.python.org/3/library/

```
>>> import math
>>> print(math.sqrt(2))
1.4142135623730951
>>> print(math.pi)
3.141592653589793
```

1. La sezione aurea o rapporto aureo o numero aureo o costante di Fidia o proporzione divina, nell'ambito delle arti figurative e della matematica, denota il numero irrazionale ottenuto dalla seguente formula $\frac{1+\sqrt{5}}{2}$. Scrivere un programma che visualizza il valore della sezione aurea
2. Il tetraedro regolare è un solido con 4 facce ognuna delle quali è un triangolo equilatero. Il tetraedro regolare è uno dei cinque solidi platonici, cioè uno dei poliedri regolari e le sue facce sono triangoli equilateri. Esso presenta un angolo diedro di 70° 32'. Il volume del tetraedro si ottiene applicando la formula $\frac{1}{12}a^3\sqrt{2}$ dove a è la lunghezza dello spigolo di base. Scrivere un programma che richiede la lunghezza dello spigolo di base di un tetraedro e visualizza il volume.
3. Scrivere un programma che chiede all'utente il raggio di base e l'altezza di un cilindro poi calcola e visualizza il volume (V =πr2h)
4. Scrivere un programma che riceve in input la temperatura misurata in gradi Fahrenheit e la fornisce in output convertita in gradi Celsius (https://it.wikipedia.org/wiki/Grado_Fahrenheit)

algoritmi in python 3

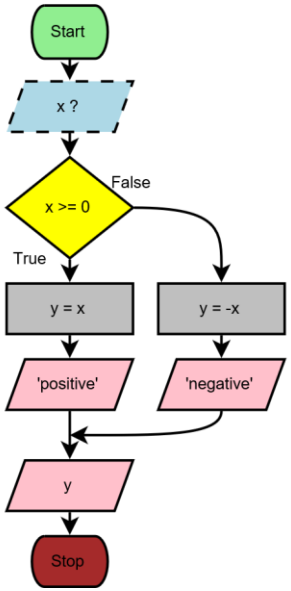
strutture di controllo

- ***indentazione*** del corpo di **`if`** o **`else`**
- ***necessaria*** (per sintassi), non opzionale!
- clausola ***else***: opzionale
  - eseguita solo se la condizione non è verificata

```
age = int(input("Age? "))

if age < 14:
    print("You're too young for driving a scooter...")
    print("But not for learning Python!")
```

- nel corpo di if o else: è possibile inserire qualsiasi istruzione
  - anche altri blocchi if o altre strutture di controllo annidate!

```
x = float(input("insert a value: "))

if x >= 0:
    y = x
    print(x, "is positive")
else:
    y = -x
    print(x, "is negative")

print("abs =", y)
```

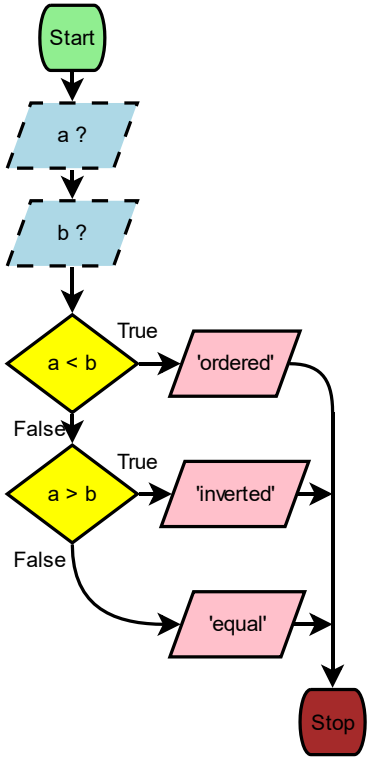


- **`and, or, not`** ***connettivi logici*** per espressioni booleane

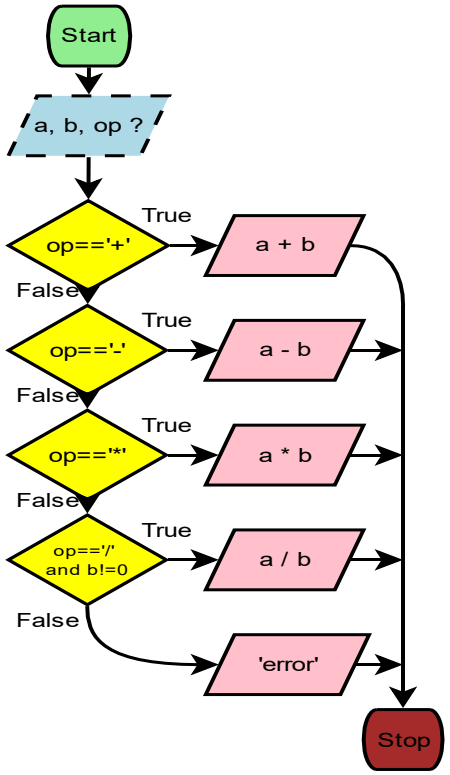
```
birth_year = int(input("Birth year? "))
birth_month = int(input("Birth month? "))
birth_day = int(input("Birth day? "))
current_year = int(input("Current year? "))
current_month = int(input("Current month? "))
current_day = int(input("Current day? "))

if (current_month > birth_month
    or (current_month == birth_month and current_day >= birth_day)):
    age = current_year - birth_year
else:
    age = current_year - birth_year - 1

print("Your age is", age)
```

- **elif**: clausola ***else*** che contiene un altro ***if***
- in Python non è presente il costrutto switch (e nemmeno do-while)
- es. confronto fra stringhe

```
a = input("First word? ")
b = input("Second word? ")

if a < b:
    print("The words are ordered")
elif a > b:
    print("The words are inverted")
else:
    print("The words are equal")
```



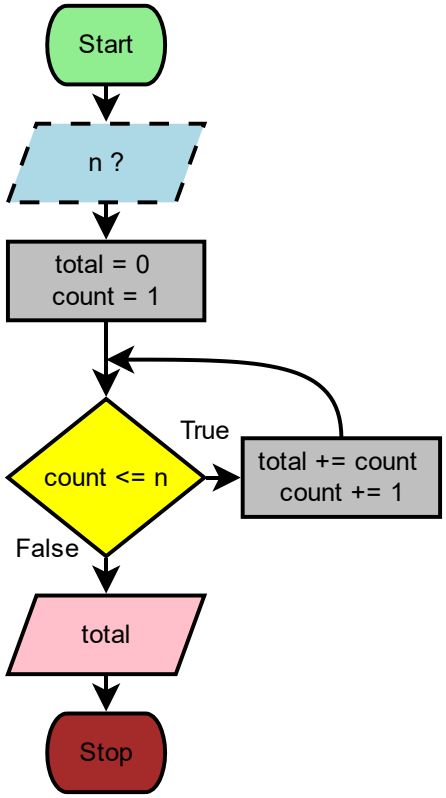
```
a = float(input())
b = float(input())
op = input()

if op == '+':
    print(a + b)
elif op == '-':
    print(a - b)
elif op == '*':
    print(a * b)
elif op == '/' and b != 0:
    print(a / b)
else:
    print("Operation not allowed")
```



- Sapendo che in un parcheggio ogni ora di sosta costa € 1.2 per le prime 3 ore e € 2 per le successive, scrivere un programma che richiede il numero di ore di parcheggio e visualizza il totale da pagare.
- Si riceve in input un valore che rappresenta il voto ottenuto da uno studente. Se il voto è minore o uguale a 13 si visualizza "non ammesso", se maggiore di 13 e minore di 18 "ammesso con riserva", se maggiore di 18 "ammesso".
- Dati tre valori che rappresentano le misure dei lati di un triangolo controllare se si tratta effettivamente di un triangolo (https://it.wikipedia.org/wiki/Disuguaglianza_triangolare) e, in caso positivo determinare se si tratta di un triangolo scaleno, isoscele o equilatero
- Si ricevono in input due numeri interi positivi determinare se il maggiore dei due è multiplo del minore

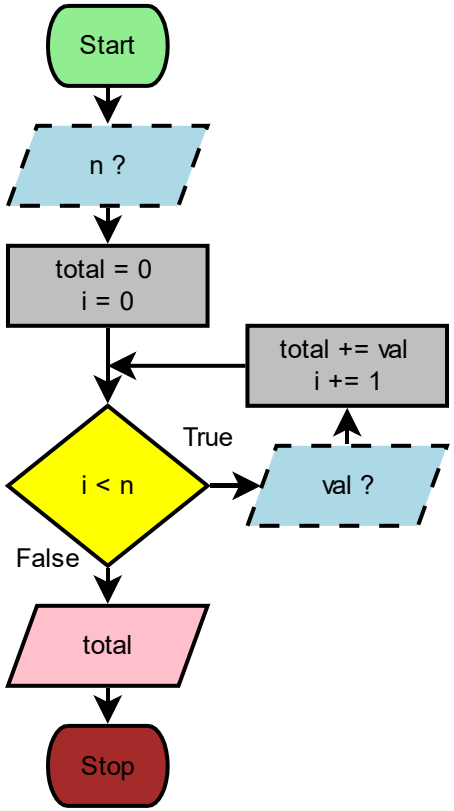
- ***condizione*** booleana di ***permanenza*** nel ciclo
- controllo preliminare (precondizione)
  - *possibile che il corpo non sia mai eseguito*

```
# Sum of the numbers from 1 to n
total = 0
count = 1
n = int(input("n? "))

while count <= n:
    total = total + count
    count = count + 1

print("The sum is", total)
```

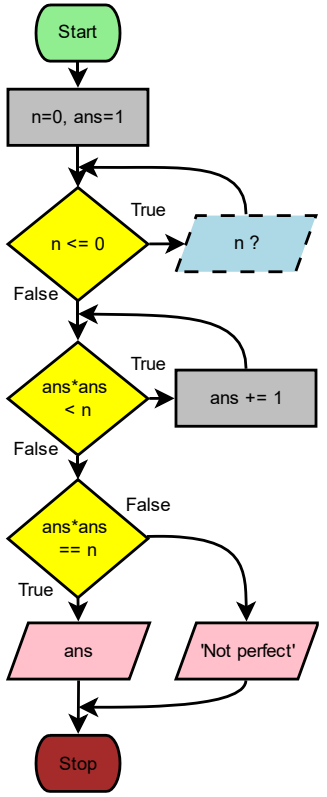


somma n valori in input

```
n = int(input("How many values? "))
total = 0
i = 0

while i < n:
    val = int(input("Val? "))

    total += val   # total = total + val
    i += 1         # i = i + 1

print("The sum is", total)
```

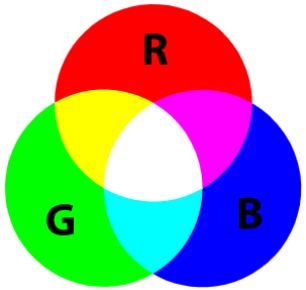


```
n = 0
while n <= 0:
    n = int(input("Positive val? "))

ans = 1
while ans * ans < n:
    ans += 1

if ans * ans == n:
    print("Square root:", ans)
else:
    print("Not a perfect square")
```



- Scrivere un programma che richiede in input un numero intero positivo (se il numero non è positivo richiederlo nuovamente) e visualizza tutti i suoi divisori.
- Scrivere un programma che riceve in input il valore n compreso fra 2 e 20 e visualizza il valore di tutte le n potenze di 2.
- Si riceve come dato d'ingresso una sequenza di numeri interi, i numeri sono al massimo 10, non è conosciuta a priori la lunghezza di questa sequenza che termina o al raggiungimento del decimo valore o quando viene inserito il valore 0. Al termine dell'input si visualizzi la media aritmetica dei valori inseriti (non si consideri il valore 0 finale).
- Scrivere un programma che riceve in input un valore intero compreso fra 2 e 10 e stampa la "tabellina" di quel numero (prodotti da 1 a 10)

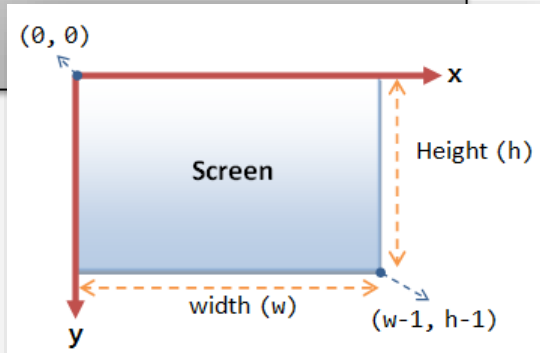
- ***sequenza immutabile*** di valori, anche di tipo diverso
  - spesso tra parentesi, per separarla da altri valori
  - utili anche per grafica:
    - Color: (red, green, blue)
      Ogni componente nel range 0..255
    - Point: (x, y)
    - Size: (width, height)
    - Rect: (left, top, width, height)




```
center = (150, 100)
color = (10, 10, 200)  # ~ blue
size = (640, 480)
rectangle = (150, 100, 200, 200)  # square
```

- ***transpiler*** (*source-to-source compiler*): converte codice Python in JavaScript
  - sia il traduttore che il codice generato girano nel browser
  - playground http://www.ce.unipr.it/brython/
- useremo un modulo ad-hoc: g2d
  - genera html e lancia web-server locale
  - definisce funzioni di disegno
- per l'esecuzione locale, copiare nella carella di lavoro il file g2d.py, da:
- http://albertoferrari.github.io/oop_Python/materiale/g2d.zip

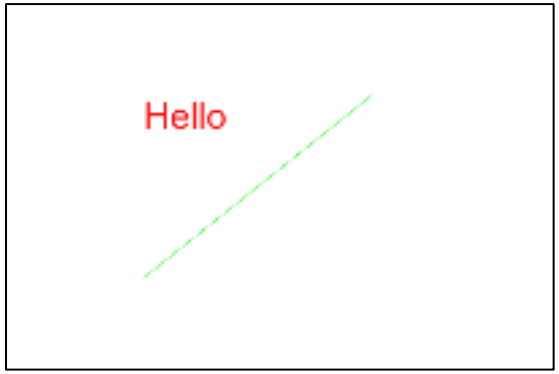
```
# importazione del modulo
import g2d
# Creazione del canvas, larghezza = 600 pixel, altezza = 400 pixel
g2d.init_canvas((600, 400))
# Rettangolo giallo, posizione angolo sinistra = 150, top = 100
# larghezza larghezza = 250, altezza = 200
# red=255 (max), green=255 (max), blue=0 (min)
g2d.set_color((255, 255, 0))          # impostazione colore
g2d.fill_rect((150, 100, 250, 200))   # disegno rettangolo
# cerchio blu, centro = (400, 300), raggio = 20
g2d.set_color((0, 0, 255))        # impostazione colore
g2d.fill_circle((400, 300), 20) # disegno del cerchio
# Ciclo di gestione eventi
g2d.main_loop()
```



The 2D Screen Coordinates: The origin is located at the top-left corner, with x-axis pointing left and y-axis pointing down.

linee e testi

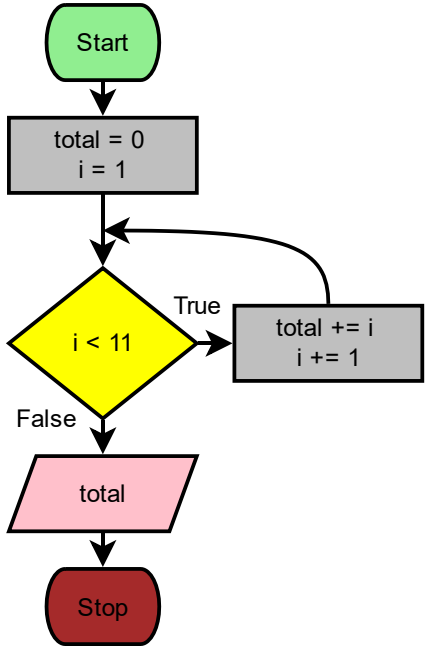
```
import g2d
g2d.init_canvas((600, 400))

g2d.set_color((0, 255, 0))
g2d.draw_line((150, 100), (400, 300))    # pt1, pt2

g2d.set_color((255, 0, 0))
g2d.draw_text("Hello", (150, 100), 40) # text,left-top,font-size

g2d.main_loop()
```



intervallo di valori: range

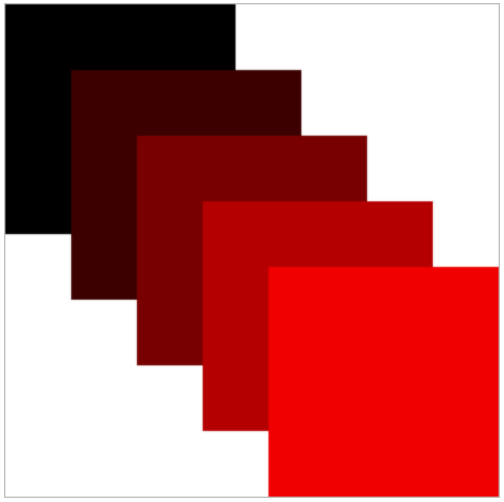
- range: intervallo di valori aperto a destra
  - estremo inferiore incluso
  - estremo superiore escluso
  - iterabile con un ciclo for

```
# Add up numbers from 1 to 10

total = 0
for i in range(1, 11):
    total += i
print(total)

# total = 0;  i = 1
# while i < 11:
#     total += i;  i += 1
```



sequenza di quadrati

```
import g2d

g2d.init_canvas((300, 300))

for i in range(5):  ## range(0, 5)
    x = i * 40
    y = x
    red = i * 60
    g2d.set_color((red, 0, 0))
    g2d.fill_rect((x, y, 100, 100))
g2d.main_loop()
```



- esempio di utilizzo della funzione randint per la generazione di numeri pseudocasuali

```
import random

# estrazione casuale di 5 valori compresi fra 10 e 30

for x in range(1,6):
    n = random.randint(10,30)
    print("estrazione numero",x,"valore estratto",n)
```

algoritmi in python 3

esercizi

- 1.1 cerchio
  - chiedere all'utente il valore del raggio r di un cerchio
  - mostrare il valore dell'area e della circonferenza
  - se r è negativo, però, mostrare un messaggio d'errore

- 1.2 minore e maggiore
  - generare e stampare tre numeri interi casuali: a, b, c
  - ciascuno compreso tra 1 e 6
  - determinare e mostrare qual è il minore dei tre

controllare prima di tutto se a è minore degli altri due
altrimenti controllare se b è minore di c
altrimenti ...

- 1.3 quadrati casuali
  - chiedere all'utente un numero n
  - disegnare n quadrati
    - tutti con lato di 100 pixel
    - ciascuno in posizione casuale
    - ciascuno con un colore casuale

cominciare a disegnare un solo quadrato grigio, in posizione casuale

- 1.4 numero segreto
  - generare all'inizio del programma un numero "segreto" a caso tra 1 e 90
  - chiedere ripetutamente all'utente di immettere un numero, finché non indovina quello generato
  - ad ogni tentativo, dire se il numero immesso è maggiore o minore del numero segreto