

# Come si comporta il seguente programma?

```
text_anni = input("quanti anni hai?")
eta = int(text_anni)
print(text_anni * 2)
print(eta * 2)
print(type(text_anni), type(eta))
input()
print('e ora?')
y = 'FINE'
print(y)
```

*inserisco 20*

*quindi text\_anni ha valore '20'*

*mentre eta ha valore 20*

*2020*

*40*

*<class 'str'> <class 'int'>*

*si ferma in attesa della pressione di  
Enter/Return*

*e ora?*

*FINE*

## Esercizi

```
name = input ("your name: ")
age = input ("your age: ")
year = input("your birth year: ")
future_age = int(year)+10
print("The age of ", name, "in 2033 will be ", future_age)
```

**dov'è l'errore?**

```
n = input()    # inserisce 3
```

```
x = n+'4'
```

```
y = int(n)+3
```

```
print(x, y)    cosa stampa?
```

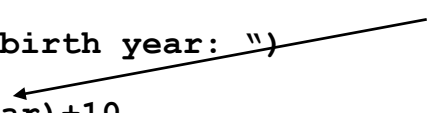
```

name = input ("your name: ")
age = input ("your age: ")
year = input("your birth year: ")
future_age = int(year)+10
print("The age of ", name, "in 2033 will be ", future_age)

```

dov'è l'errore?

age non year !



```

n = input()    # inserisce 3
x = n+'4'
y = int(n)+3
print(x, y)    # stampa 34 e 6

```

# Errori

- **syntax errors**: le regole sintattiche sono rigide, l'interprete **non riconosce le parole** del linguaggio.
  - `print("ciao") pront('ciao') a= 1 " 2`
  - **prima di iniziare ad eseguire** il programma, l'interprete controlla se ci sono errori di sintassi, e se ne trova non inizia nemmeno.
- **runtime errors** (*errore in esecuzione*): l'esecuzione è iniziata ma l'interprete trova un'**istruzione che non riesce ad eseguire correttamente**
  - `print(new_var)` `NameError: name 'new_var' is not defined`
  - `'cielo'+ 2` `TypeError: can only concatenate str (not "int") to str`
  - **l'esecuzione si interrompe** in questo punto
- **errori di semantica** (o *di logica*): hanno a che fare con il **senso** del programma. Il programma viene eseguito completamente ma **non fa la cosa giusta**.
  - Il programma esegue le istruzioni indicate, ma l'algoritmo/ la logica risolutiva è errata, oppure è disallineato rispetto alla specifica

# Debugging: come trovare gli errori?

- leggere ed **interpretare i messaggi d'errore** dell'interprete.  
**Attenzione:** a volte il problema non sta nella riga indicata nel messaggio, ma un po' prima
- riguardare linea per linea, immaginando l'effetto sul programma
- **inserire in punti critici delle istruzioni print**
  - per controllare il valore delle variabili in quel punto,
  - o per controllare che un certo punto del codice sorgente sia effettivamente raggiunto durante l'esecuzione
- debugger tools (es. Python Tutor)

## Espressioni ed Istruzioni

- Un'**istruzione o comando** è una porzione di codice che la macchina **esegue** e che ha qualche **effetto**
- Un'**espressione** è una porzione di codice che **trova** e restituisce il **valore** di qualcosa. **Non è un errore, è un'espressione di valore False** **valuta**, cioè

```
print('Calcolo:', 4+8)
print(4 >= 8)
print( not (4 >= 8) )
```



```
Calcolo: 12
False
True
```

l'interprete

1. **valuta i valori delle espressioni** passate come parametro alla print,
2. esegue la funzione **print** usando come **argomenti i valori calcolati**

# Variabili: Valori e Assegnamento

Cosa stampa?

```
x = 10
y = x + 1
z = (x==8)
s = 'ciao'
print (z, s*2)
```

x	10
y	11
z	False
s	'ciao'

False      ciaociao

l'istruzione di assegnamento

- è sempre della forma **variable = espressione**
- viene eseguita nel modo seguente:
  1. si **valuta l'espressione a destra** di = e si ottiene un **valore**
  2. questo valore viene **assegnato alla variabile a sinistra** di =

## Esercizi - Soluzioni

3. Scrivere un'espressione booleana che ha valore **True** se e solo se la variabile x ha un valore che **sta** nell'intervallo di numeri [0,...,10]

`(x >= 0) and (x<=10)`

4. Scrivere un'espressione booleana che ha valore **True** se e solo se x **non sta** nell'intervallo di numeri [0,...,10]

`(x<0) or (x>10)`      `not((x>=0) and (x<=10))`

5. Scrivere un'espressione booleana che coinvolge 3 variabili x,y,z e ha valore **True** se e solo se x **sta** nell'intervallo di numeri [y,...,z]

`(x>=y) and (x<=z)`

# Esercizio

controllare la  
soluzione usando  
python

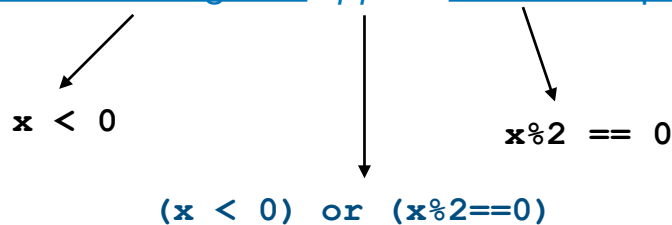
Scrivere un'espressione booleana che è **Falsa** se e solo se  $x$  è una variabile di valore un numero negativo oppure un numero pari

`not ((x<0) or (x%2==0))`

`(not(x<0)) and (not(x%2==0))`

`(x >= 0) and (x%2 != 0)`

Scrivere un'espressione booleana che è **Vera** se e solo se  $x$  è una variabile di valore un numero negativo oppure un numero pari



# Esercizio

controllare la  
soluzione usando  
python

Scrivere un'espressione booleana che è **Falsa** se e solo se  $x$  è una variabile di valore un numero negativo oppure un numero pari

`not ((x<0) or (x%2==0))`

`(x >= 0) and (x%2 != 0)`

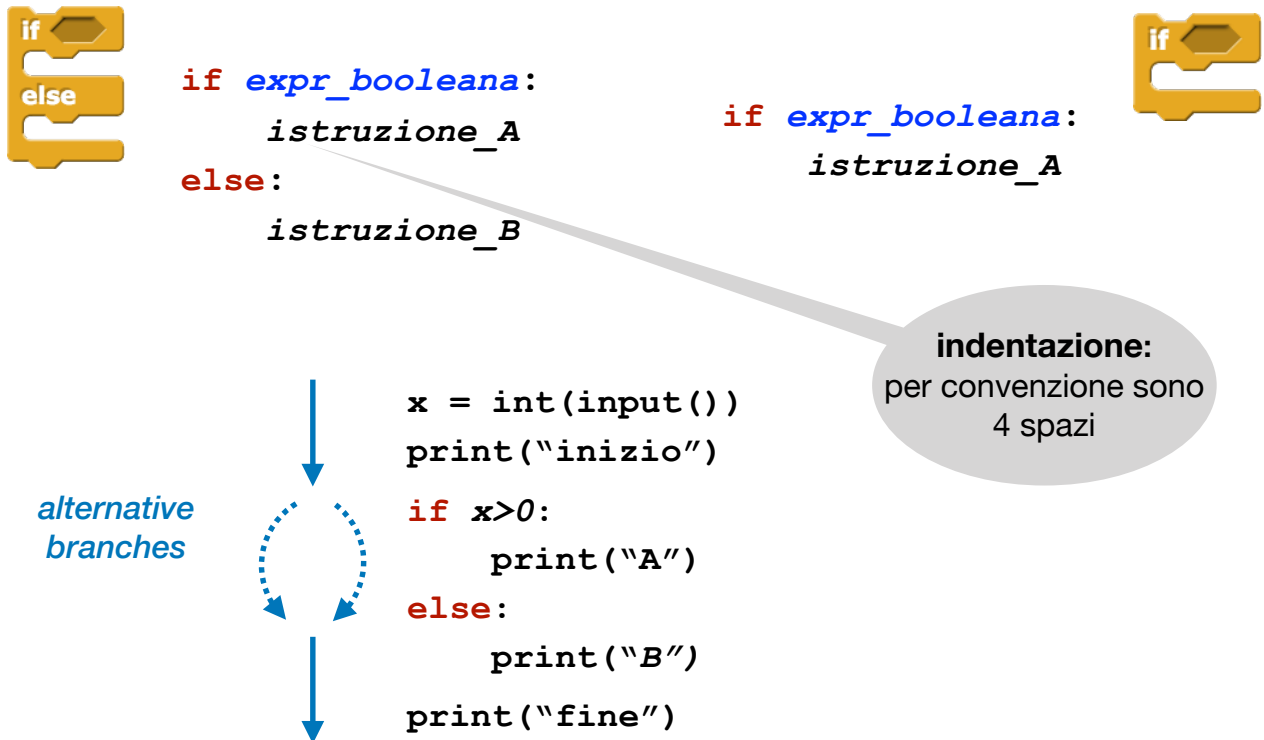
## Esempio di programma per controllare la soluzione:

```
print('Due espressioni False se e solo se inserisco un  
numero negativo oppure pari')  
  
x = int(input('inserisci un numero: '))  
  
expr1 = not((x<0) or (x%2==0))  
expr2 = (x >=0) and (x%2 != 0)  
  
print('La prima expr booleana ha valore ', expr1)  
print('La seconda expr booleana ha valore ', expr2)
```

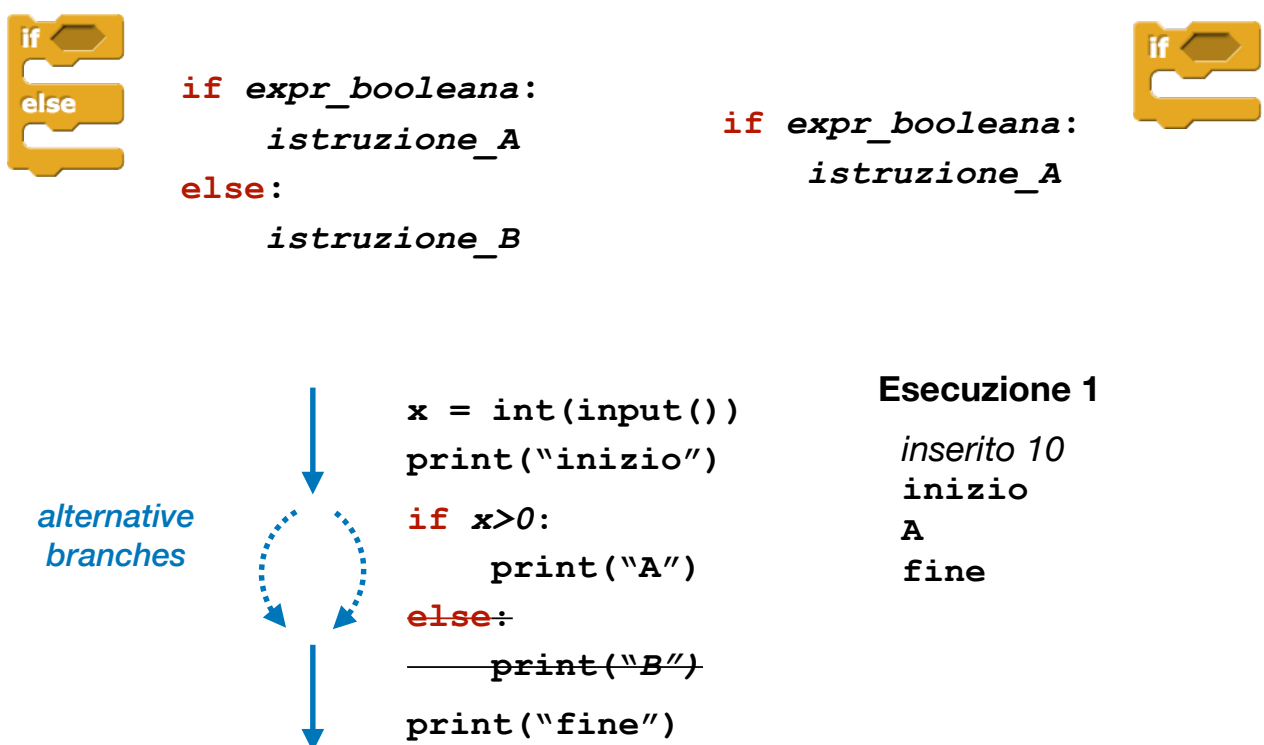
### test con

- n. negativo e pari
- n. negativo e dispari
- n. positivo e pari
- n. positivo e dispari
- zero

# Istruzioni condizionali



# Istruzioni condizionali



# Istruzioni condizionali



```
if expr_booleana:
    istruzione_A
else:
    istruzione_B
```



```
if expr_booleana:
```

questa riga si esegue sempre:  
la condizione (expr booleana)  
va valutata

alternative  
branches



```
x = int(input())
print("inizio")
if x>0:
    print("A")
else:
    print("B")
print("fine")
```

Esecuzione 2

inserito -4  
inizio  
B  
fine

che output produce questo programma?

```
1 a = 3
2 b = 4
3 if a+1==b:
4     print("yes")
5
6 if b%2 == 0:
7     print("b pari")
8 else:
9     print("b dispari")
10
11 print("buona giornata")
```

istruzione 1

istruzione 2

istruzione 3

istruzione 4

istruzione 5

yes

b pari

buona giornata

**indentazione:**  
per convenzione sono  
4 spazi

attenzione ai :  
altrimenti è  
errore di sintassi

che output produce questo programma?

```
1 a = 3
2 b = 4
3 if a+5==b:
4     print("yes")
5
6 if b%2 == 0:
7     print("b pari")
8 else:
9     print("b dispari")
10
11 print("buona giornata")
```

istruzione 1  
istruzione 2  
istruzione 3  
istruzione 4  
istruzione 5

b pari  
buona giornata

che output produce questo programma?

```
a = 3
b = a
if (a==b or a > b):
    print("pippo")
    print("hello")
else:
    print("pluto")
```

**blocco** di istruzioni:  
identificato da  
allineamento **indentato**

pippo  
hello



```
if expr_booleana:
    blocco_istruzioni_A
else:
    blocco_istruzioni_B
```



```
if expr_booleana:
    blocco_istruzioni
```



# Esempio

- Scrivere un programma che chiede di **inserire due numeri** e **stampa il più grande** dei due.

```
x = int(input("primo int:"))
y = int(input("secondo int:"))

if x > y:
    print(x)
else:
    print(y)
```

Se sono **inseriti due numeri uguali**  
stampa uno dei due... quale?

# Esempio

- **Specifica:** scrivere un programma che chiede di **inserire due numeri** e **stampa il più grande** dei due. **Se sono uguali il programma deve stampare la stringa uguali**

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x==y:
    print('uguali')
```

```
if x > y:
    print(x)
else:
    print(y)
```

se inserisco due numeri uguali,  
stampa la stringa **uguali** ma  
ANCHE il numero

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x==y:
    print('uguali')
else:
```

```
    if x > y:
        print(x)
    else:
        print(y)
```

se inserisco due numeri uguali,  
stampa **SOLO** la stringa **uguali**

# Esempio

- **Specifica:** scrivere un programma che chiede di **inserire due numeri** e **stampa il più grande** dei due. **Se sono uguali il programma deve stampare la stringa uguali**

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x > y:
    print(x)
else:
    if x == y:
        print('uguali')
    else:
        print(y)
```

come si comporta?  
che istruzioni sono?

tracciare il flusso di controllo di  
qualche esecuzione

# Esempio

- **Specifica:** scrivere un programma che chiede di **inserire due numeri** e **stampa il più grande** dei due. **Se sono uguali il programma deve stampare la stringa uguali**

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x > y:
    print(x)
```

```
if y > x:
    print(y)
```

```
if x==y:
    print('uguali')
```

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x > y:
    print(x)
```

```
if y > x:
    print(y)
else:
    print('uguali')
```

come si comportano ?  
sono programmi **corretti** ? come lo verifico?

# Esempio

Che differenza c'è tra questi due programmi?

- **confrontare il comportamento** quando si inseriscono due numeri **uguali**
- **confrontare il comportamento** quando si inseriscono due numeri **diversi**

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x==y:
    print('uguali')
```

```
if x > y:
    print(x)
else:
    print(y)
```

**Fare attenzione  
all'indentazione**

```
x = int(input("primo int:"))
y = int(input("secondo int:"))
```

```
if x==y:
    print('uguali')
```

```
if x > y:
    print(x)
else:
    print(y)
```

se inseriti due numeri  
**diversi** non fa nulla

**che output produce questo programma?**

```
a = 3
```

```
b = a
```

```
if (a==b or a > b):
```

```
    print("pippo")
```

```
    print("hello")
```

```
    if b%2==0:
```

```
        print("pari")
```

```
    else:
```

```
        print("dispari")
```

```
else:
```

```
    print("pluto")
```

```
pippo
```

```
hello
```

```
dispari
```

in un blocco di istruzioni si può  
inserire qualsiasi istruzione valida,  
anche un'altra istruzione if-else

# condizioni innestate

```
if x == y:
    print("x e y sono uguali")
else:
    if x < y:
        print("x minore di y")
    else:
        print("x maggiore di y")
```

equivalente a questa istruzione

```
if x < y:
    print("x minore di y")
elif x > y:
    print("x maggiore di y")
else:
    print("x e y sono uguali")
```

## più di due rami (chained conditionals)

```
scelta = input('scegli a,b o c')
if scelta == 'a':
    print('prima scelta')
elif scelta == 'b':
    print('seconda scelta')
elif scelta == 'c':
    print('terza scelta')
else:
    print('scelta errata')
```

le condizioni vengono **controllate in ordine**:

- se la prima è falsa, viene controllata la seconda e così via
- appena una condizione è vera, si esegue il ramo corrispondente e l'istruzione termina
- ci possono essere tanti rami **elif** (else if) e al più un ramo **else** alla fine
- se ci sono più condizioni vere, viene eseguita sempre solo la prima

# Esercizi

1. Scrivere un programma che legge da input 3 interi e scrive in output se sono tutti e tre uguali, oppure se sono ordinati in ordine crescente, oppure se sono ordinati in ordine decrescente, oppure se non è vero nessuno dei casi precedenti.
2. Scrivere un programma che legge da input 4 interi, li memorizza in 4 variabili, e stampa la loro somma, la media, il valore minimo e il massimo.
3. Dati tre bastoncini, non sempre è possibile riuscire a sistemarli in modo da formare un triangolo. Ad esempio se uno è lungo 12cm e gli altri due sono lunghi 1cm non si riesce a disporli a triangolo. Esiste la seguente regola:  
*date 3 lunghezze, se una qualsiasi è maggiore della somma delle altre due, allora non è possibile formare un triangolo che abbia per lati quelle lunghezze*  
Scrivere un programma che dati in input tre interi, controlla se possono essere le lunghezze dei lati di un triangolo.

## Imparare a leggere il codice

- Descrivere a parole *come si comporta* il seguente codice

```
x = int(input('Inserisci un numero: '))
y = int(input('Inserisci un numero: '))
z = int(input('Inserisci un numero: '))

if x<y and y<z:
    print('GIALLO')
else:
    if x>y and y>z:
        print('BLU')
    else:
        print('ROSSO')
```

- Descrivere a parole *la specifica* del seguente codice

Il programma prende in input 3 numeri, e

- se *sono 3 numeri crescenti* stampa GIALLO
- se *sono 3 numeri decrescenti* stampa BLU
- in *ogni altro caso* stampa ROSSO

# Esercizio

## Imparare a leggere il codice

- Descrivere a parole *la specifica* del seguente codice

```
x = int(input('Inserisci un numero: '))
y = int(input('Inserisci un numero: '))
z = int(input('Inserisci un numero: '))

if x%2!=0:
    if y%2!=0 and z%2!=0:
        print('GIALLO')
    else:
        print('ROSSO')
else:
    if y%2==0 and z%2==0:
        print('BLU')
    else:
        print('ROSSO')
```

# Esercizio

- Descrivere a parole la specifica del seguente codice, indicando in quali casi stampa Rosso, in quali casi stampa Blu, e in quali casi stampa Giallo

```
n = int(input('Inserisci un numero '))
if n > 5:
    print("Rosso")
else:
    if n > 10:
        print("Blu")
    else:
        print("Giallo")
```

- Descrivere a parole la specifica del seguente codice, indicando in quali casi stampa Rosso, in quali casi stampa Blu, e in quali casi stampa Giallo

```
n = int(input('Inserisci un numero '))
if n > 5:
    print("Rosso")
else:
    if n < 5:
        print("Blu")
    else:
        print("Giallo")
```

# Esercizio: Anni bisestili

Un anno è **bisestile** (*leap* in inglese) se:

- è divisibile per 4, ma non anche divisibile per 100  
(es. 1996 e 2020 non 1900)
- con un'eccezione: se è divisibile per 400, allora è bisestile  
(es. anno 2000)

Si considerino i seguenti programmi Python, che stampano **anno bisestile** se la variabile **year** ha un valore che corrisponde ad un anno bisestile, altrimenti stampano **anno normale**.

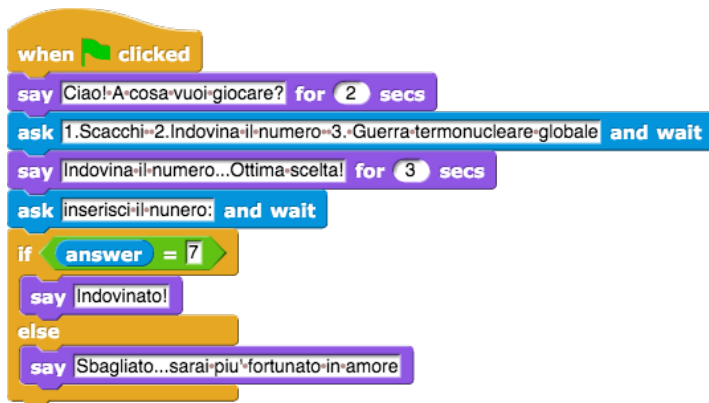
Sono tutti programmi **corretti**, che hanno la **stessa specifica**.

```
if year%400 ==0 or (year%4==0 and (not year%100==0)):
    print('anno bisestile')
else:
    print('anno normale')
```

```
if year%400 ==0:
    print('anno bisestile')
else:
    if year%100==0:
        print('anno normale')
    else:
        if year%4==0:
            print('anno bisestile')
        else:
            print('anno normale')
```

```
if year%4 ==0:
    if not year%100==0:
        print('anno bisestile')
    else:
        if year%400==0:
            print('anno bisestile')
        else:
            print('anno normale')
else:
    print('anno normale')
```

Discutere **quale versione** è la più **leggibile**, quella più **elegante**, quella più **facile da ricordare**, e quella in cui è più **facile individuare errori**.



## Esercizio:

modificare il programma in modo che tenga conto di quale gioco ha scelto l'utente  
....con creatività

```
print('Ciao! a cosa vuoi giocare?')
input('1.Scacchi 2.Indovina il numero 3.Guerra termonucleare globale')
print('Hai scelto: Indovina il numero Ottima scelta!')
answer = int(input('inserisci il numero: '))
if answer == 7 :
    print('Indovinato!')
else:
    print('Sbagliato...sarai piu fortunato in amore')
```

# Esercizio

- Scrivere un programma che calcola e stampa quanti secondi ci sono in 3 ore e quanti in 12 ore.
- Scrivere un programma che calcola e stampa quanti giorni sono trascorsi dal 1 gennaio 2023 ad oggi 30 novembre 2023.

ci sono tante diverse soluzioni,  
provate a confrontarle