

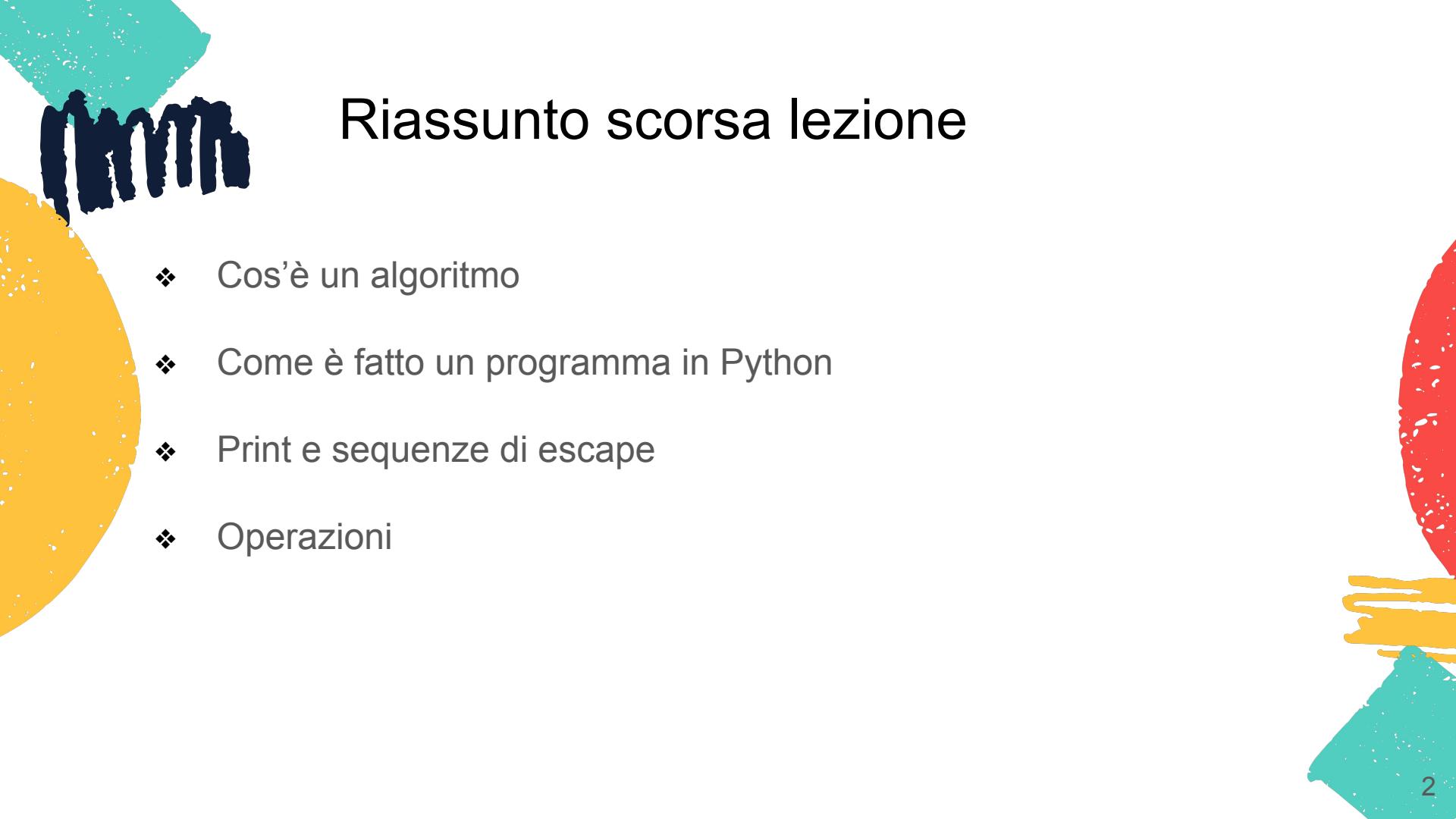
# CORSO PYTHON

Seconda parte

Variabili, input e decisioni

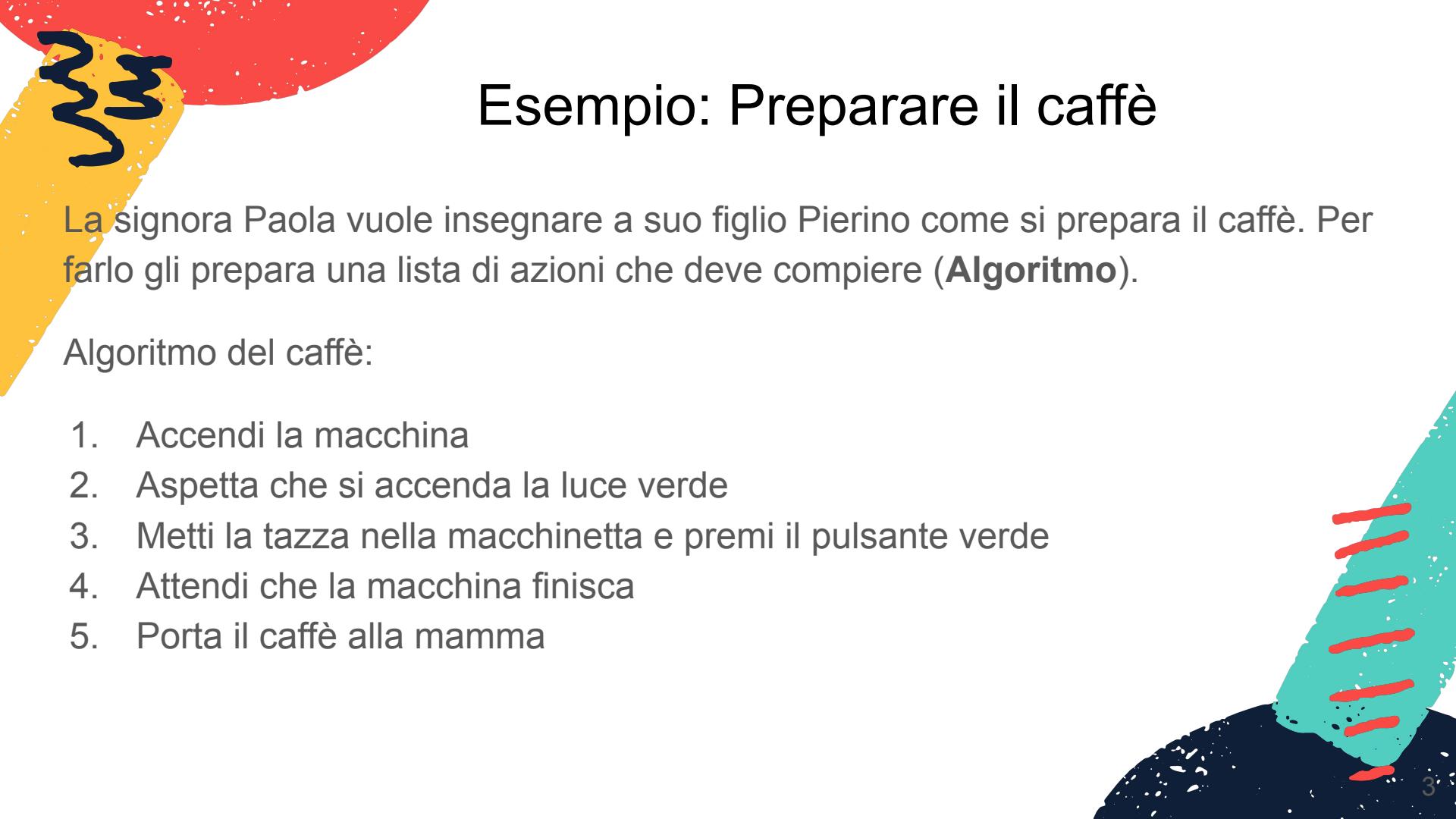
Carlo Tacchella





# Riassunto scorsa lezione

- ❖ Cos'è un algoritmo
- ❖ Come è fatto un programma in Python
- ❖ Print e sequenze di escape
- ❖ Operazioni



# Esempio: Preparare il caffè

La signora Paola vuole insegnare a suo figlio Pierino come si prepara il caffè. Per farlo gli prepara una lista di azioni che deve compiere (**Algoritmo**).

Algoritmo del caffè:

1. Accendi la macchina
2. Aspetta che si accenda la luce verde
3. Metti la tazza nella macchinetta e premi il pulsante verde
4. Attendi che la macchina finisca
5. Porta il caffè alla mamma



## Esempio: helloWord.py

```
#Questo programma stampa a video la frase Hello, World!  
print ('Hello, World!')
```

# Operazioni

In Python possiamo usare tutte le operazioni aritmetiche principali

Somma             $2+3=5$

Sottrazione       $3-2=1$

Moltiplicazione     $4*2=8$

Divisione           $4/2=2$

Modulo (resto)     $5\%3=2$

Potenza             $3^{**}2=9$

# tipi

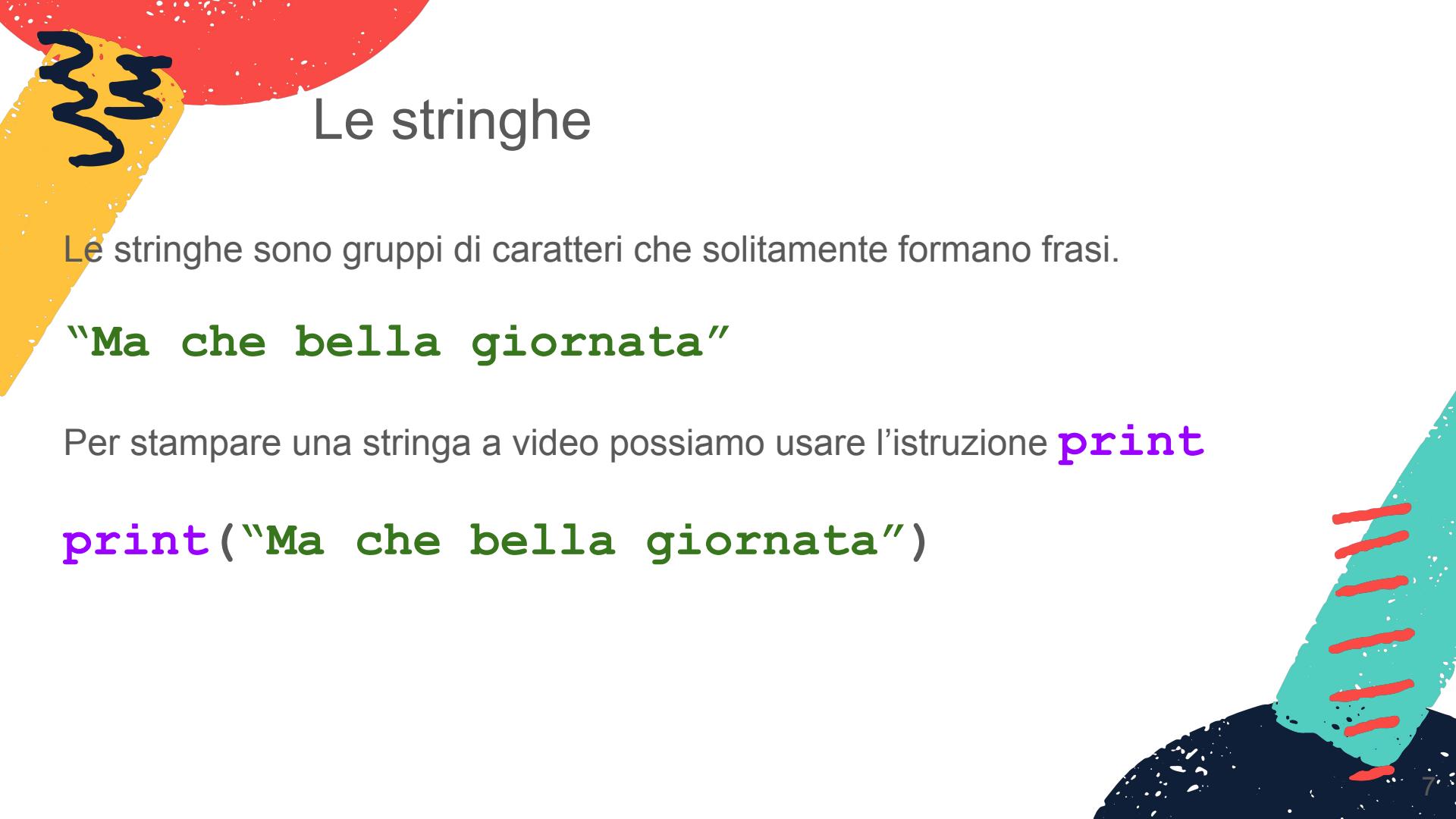
A ogni oggetto di Python è assegnato un tipo.

I tipi principali di Python sono

- `str` è il tipo delle stringhe come '`Hello World`'
- `int` è il tipo dei numeri interi come `1 12 435`
- `float` è il tipo dei numeri con la virgola come `3.1415926`
- `bool` è il tipo delle espressioni booleane e di `True False`

Se volete sapere qual è il tipo di un oggetto, potete usare il comando `type`

Per esempio `type('Hello World')` ritorna `<class 'str'>`



# Le stringhe

Le stringhe sono gruppi di caratteri che solitamente formano frasi.

**"Ma che bella giornata"**

Per stampare una stringa a video possiamo usare l'istruzione **print**

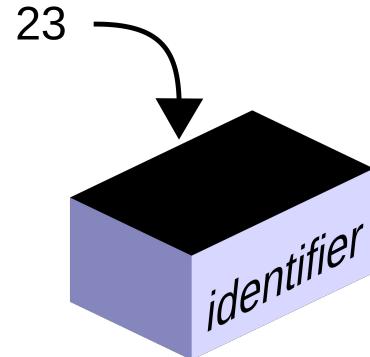
```
print("Ma che bella giornata")
```



# Variabili

In Python, possiamo utilizzare le variabili per memorizzare informazioni e poi riutilizzarle.

Una variabile ha un nome e memorizza un valore; potete immaginare una variabile come un cassetto nel quale possiamo inserire o prendere un oggetto.



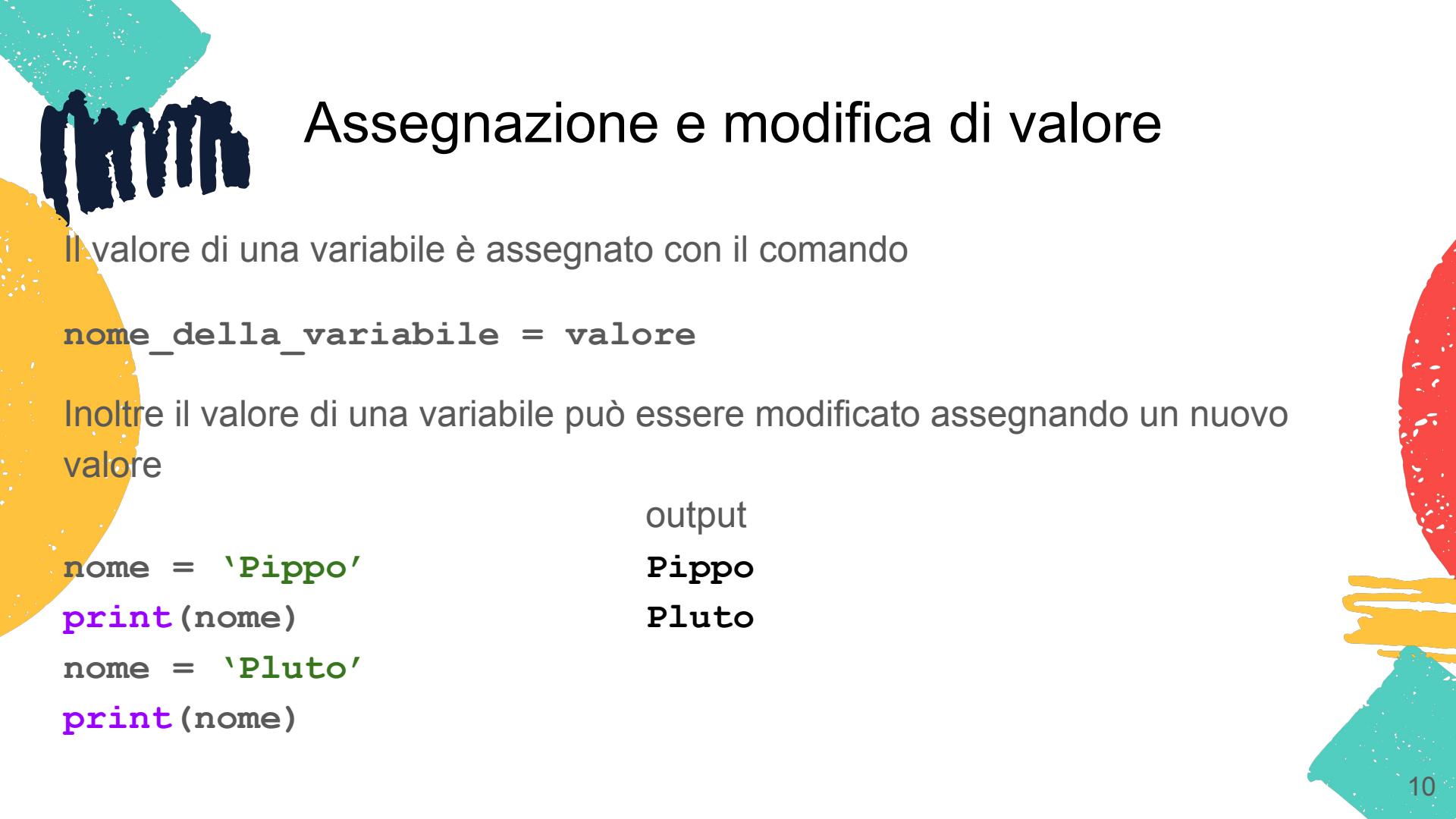
# Esempio

```
variabile = 23
```

```
nome = "Fabio"
```

```
print(nome, "ha", variabile, "anni.")
```

Output >>> Fabio ha 23 anni.



# Assegnazione e modifica di valore

Il valore di una variabile è assegnato con il comando

```
nome_della_variabile = valore
```

Inoltre il valore di una variabile può essere modificato assegnando un nuovo valore

```
nome = 'Pippo'  
print(nome)  
nome = 'Pluto'  
print(nome)
```

output

Pippo

Pluto

# I nomi delle variabili

Potete scegliere voi il nome delle variabili che utilizzerete, ma il nome di ogni variabile deve iniziare con una lettera e non potete utilizzare le seguenti parole perché sono utilizzate da Python

False, None, True, and, as, assert, break, class, continue,  
def, del, elif, else, except, finally, for, from, global,  
if, import, in, is, lambda, nonlocal, not, or, pass, raise,  
return, try, while, with, yield

Inoltre dovete fare attenzione a **maiuscole e minuscole** nei nomi, perché i nomi delle variabili sono *case sensitive*.



# Esempio

- |                 |    |
|-----------------|----|
| 1. nome         | SI |
| 2. as           | NO |
| 3. As           | SI |
| 4. Cognome_1    | SI |
| 5. 1_cognome    | NO |
| 6. nome cognome | NO |
| 7. nome_cognome | SI |
- 

# Input

Vorrei creare un programma che chiede all'utente di inserire il proprio nome e cognome. In Python questo è possibile grazie al comando:

```
input("Inserisci il tuo nome e cognome: ")
```

Quando viene eseguita questa istruzione il programma si ferma e aspetta che qualcuno inserisca i dati; il programma ripartirà quando verrà premuto il tasto invio

Per salvare i dati inseriti da tastiera possiamo usare le variabili

```
Nominativo = input("Inserisci il tuo nome e cognome: ")
```

# Esempio

```
anni = input('Inserisci i tuoi anni: ')
print('Hai', anni, 'anni!')
```



# Espressioni Booleane

Python è in grado di capire se alcune asserzioni sono vere (`True`) o false (`False`). Queste asserzioni che si chiamano **espressioni booleane** sono utilizzate per prendere decisioni (come vedremo tra poco). Ogni espressione booleana ha **uno e un solo valore di verità** (vero o falso).

I principali operatori sono

- `==`       `!=`       Uguale e diverso
- `>`       `<`       Strettamente maggiore e minore
- `>=`      `<=`      Maggiore o uguale e Minore o uguale
- `and`      `or`       And e Or per lavorare



# Esempi espressioni booleane

- `1==2` **False**
  - `1==1` **True**
  - `'molto' != 'poco'` **True**
  - `'ciao' == 'ciao'` **True**
  - `2 > 3` **False**
  - `2 < 3` **True**
  - `2 <= 2` **True**
  - `2 == 2 and 3==4` **False**
  - `2 == '2'` **False**
- 

# L'operatore di cast

A volte in Python può essere necessario cambiare il tipo di un oggetto.

Per esempio se abbiamo un valore di **float** (ossia un numero con la virgola) possiamo tenere solo il suo valore intero cambiando il suo tipo con **int**.

Per cambiare il tipo di una variabile con il tipo desiderato scriviamo semplicemente:

```
variabile2 = tipo_desiderato(variabile)
```

variabile2 può essere variabile

# L'operatore di cast

Per convertire un valore decimale in valore intero scriviamo ad esempio

```
val = 123.45  
val = int(val)
```

L'istruzione `input` per esempio non riesce a leggere numeri, ma legge solo stringhe. Per esempio

```
variabile = input("Inserisci un numero: ")  
if (variabile == 10):  
    print("Hai inserito 10!")  
  
## L'espressione var == 10 non sarà mai vera ##
```

# L'operatore di cast

Dobbiamo quindi prima dire a Python di convertire la stringa in un numero

```
variabile = input("Inserisci un numero: ")  
variabile = int(variabile)  
if (variabile == 10):  
    print("Hai inserito 10!")
```

Ora il programma esegue correttamente



# Prendere decisioni

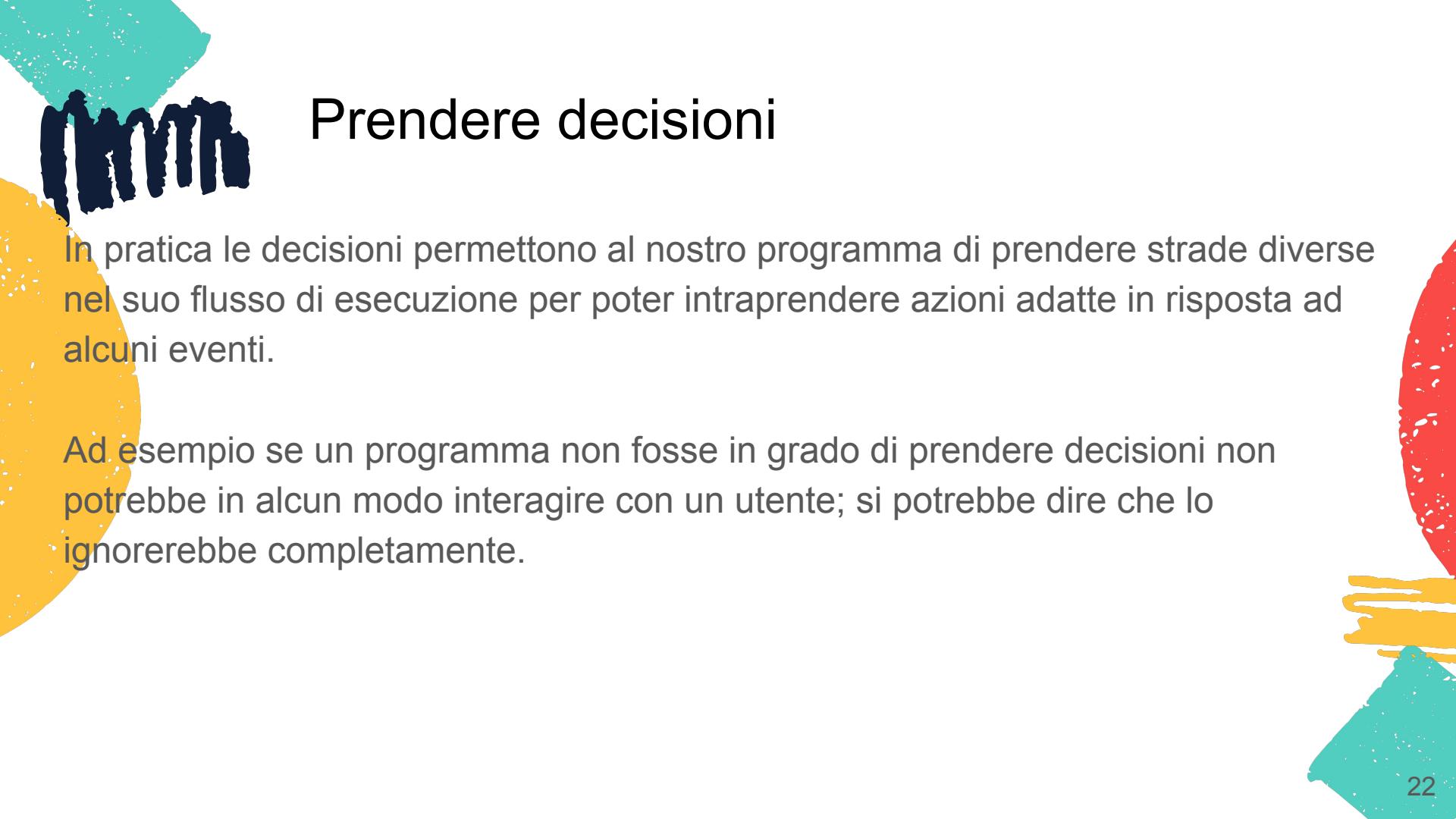
Spesso negli algoritmi è necessario effettuare alcuni controlli per poter decidere quali istruzioni eseguire.

Ad esempio nell'algoritmo per la preparazione del caffè potremmo dover controllare che il contenitore dell'acqua sia sufficientemente carico; se lo è possiamo continuare con le normali operazioni, altrimenti prima dobbiamo riempirlo d'acqua.



# Esempio preparazione caffè

1. Accendi la macchina
2. Aspetta che si accenda la luce verde
3. Controlla che ci sia abbastanza acqua nel serbatoio  
Se non c'è abbastanza acqua, caricalo
4. Metti la tazzina nella macchinetta e premi il pulsante verde
5. Attendi che la macchina finisca
6. Porta il caffè alla mamma



# Prendere decisioni

In pratica le decisioni permettono al nostro programma di prendere strade diverse nel suo flusso di esecuzione per poter intraprendere azioni adatte in risposta ad alcuni eventi.

Ad esempio se un programma non fosse in grado di prendere decisioni non potrebbe in alcun modo interagire con un utente; si potrebbe dire che lo ignorerebbe completamente.

# Il comando if

In Python il comando utilizzato per prendere decisioni è il comando **if**  
La sintassi per il comando **if** è la seguente:

```
if (espressione booleana) :  
    comandi  
else:  
    altri comandi
```

# Esempio

```
tentativo = input ('Inserisci la password: ')  
  
if (tentativo=='python123'):  
    print('Password corretta!')  
else:  
    print('Password errata!')
```

# Controllare se due numeri sono pari

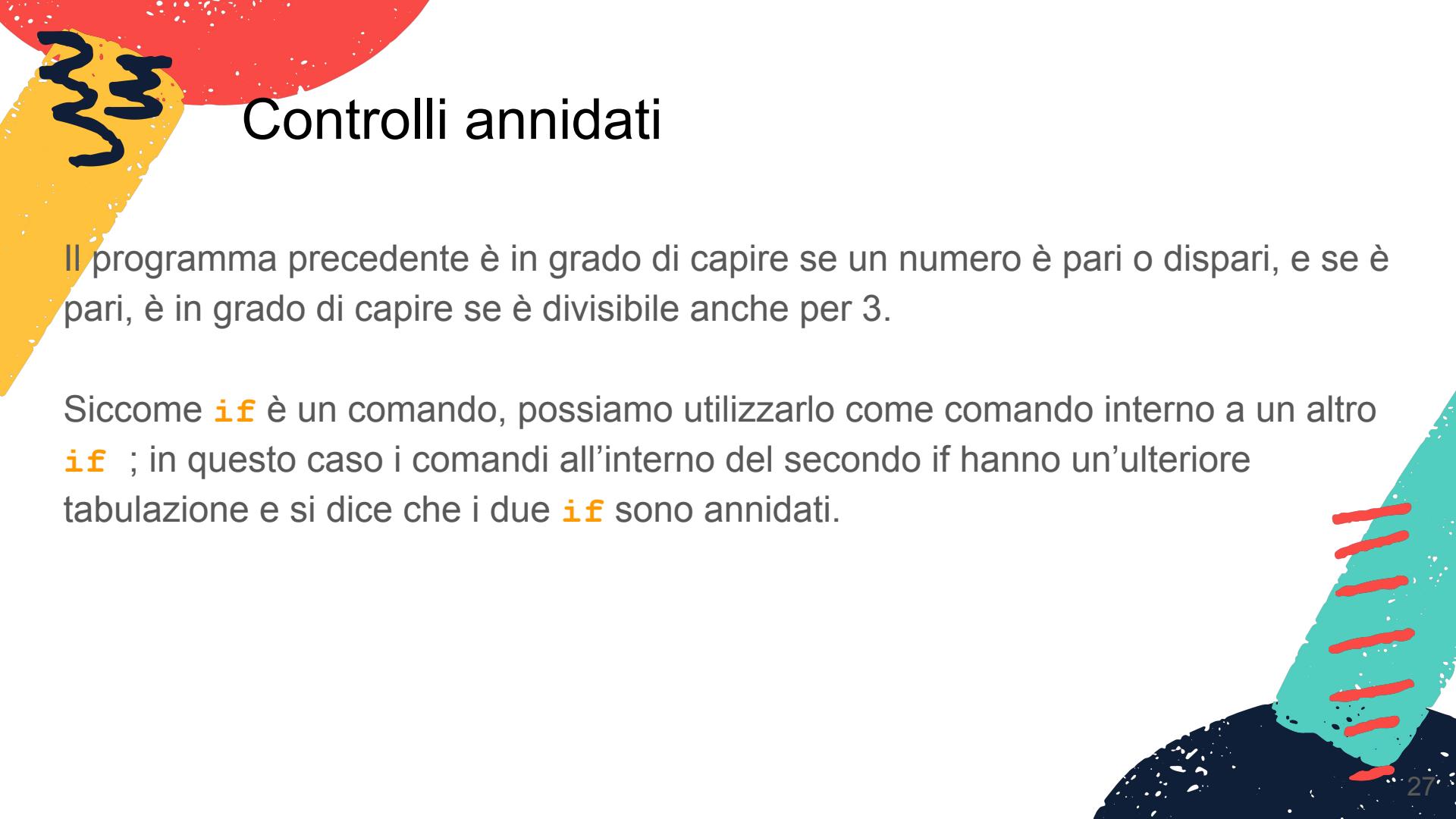
Vogliamo scrivere un programma che controlla se due numeri inseriti dall'utente sono pari.

```
primo = input('Inserisci il primo numero: ')
secondo = input('Ora inserisci il secondo: ')
primo = int(primo)
secondo = int(secondo)
if (primo%2==0 and secondo%2==0):
    print(primo, ' e ', secondo, ' sono entrambi pari')
```

# Controlli annidati

In alcuni casi se un controllo preliminare ci ha dato risposta positiva, potremmo voler svolgere ulteriori controlli. Facciamo subito un esempio

```
num = input('Inserisci un numero: ')
num = int(num)
if (num%2==0):
    if (num%3==0):
        print(num,' è un multiplo di 6')
    else:
        print(num,' è pari')
else:
    print(num,' è dispari')
```



# Controlli annidati

Il programma precedente è in grado di capire se un numero è pari o dispari, e se è pari, è in grado di capire se è divisibile anche per 3.

Siccome **if** è un comando, possiamo utilizzarlo come comando interno a un altro **if** ; in questo caso i comandi all'interno del secondo if hanno un'ulteriore tabulazione e si dice che i due **if** sono annidati.



## Il comando elif

In Python potremmo voler ulteriormente controllare una condizione presente nel ramo **else**. Facciamo un esempio:

```
if (num%2==0) :  
    print("Il numero è pari")  
else:  
    if (num==5) :  
        print("Il numero è dispari ed è cinque")  
    else:  
        print("Il numero è dispari ma non è cinque")
```



# Il comando elif

Tuttavia scrivere così il codice può essere difficile e pesante, utilizziamo allora il comando **elif** che sostituisce la coppia

**else:**

**if** (condizione):

Il codice precedente quindi può essere riscritto come:

```
if (num%2==0) :  
    print("Il numero è pari")  
elif(num==5) :  
    print("Il numero è dispari ed è cinque")  
else:  
    print("Il numero è dispari ma non è cinque")
```

# Errori di sintassi:

Il parser riporta la riga sbagliata e mostra una piccola «freccia» che indica il primo punto in cui l'errore è stato rilevato. L'errore è causato (o almeno rilevato) dall'elemento che *precede* la freccia: nell'esempio qui sopra, l'errore è rilevato nella funzione print(), perché mancano i «due punti» (':') prima.

```
>>> while True print('Hello world')
      File "<stdin>", line 1
        while True print('Hello world')
                    ^
SyntaxError: invalid syntax
```

# Eccezioni

Anche quando un'istruzione o un'espressione sono corretti dal punto di vista sintattico, possono provocare un errore quando sono *eseguiti*. L'ultima riga del messaggio d'errore ci dice che cosa è successo. Gli oggetti-eccezioni possono avere diversi tipi, e la prima parte del messaggio riporta il tipo: negli esempi [ZeroDivisionError](#), [NameError](#) e [TypeError](#).

```
>>> 10 * (1/0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
>>> 4 + spam*3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'spam' is not defined
>>> '2' + 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object to str implicitly
```

# LA PRESENZA DI BUG È INELUTTABILE

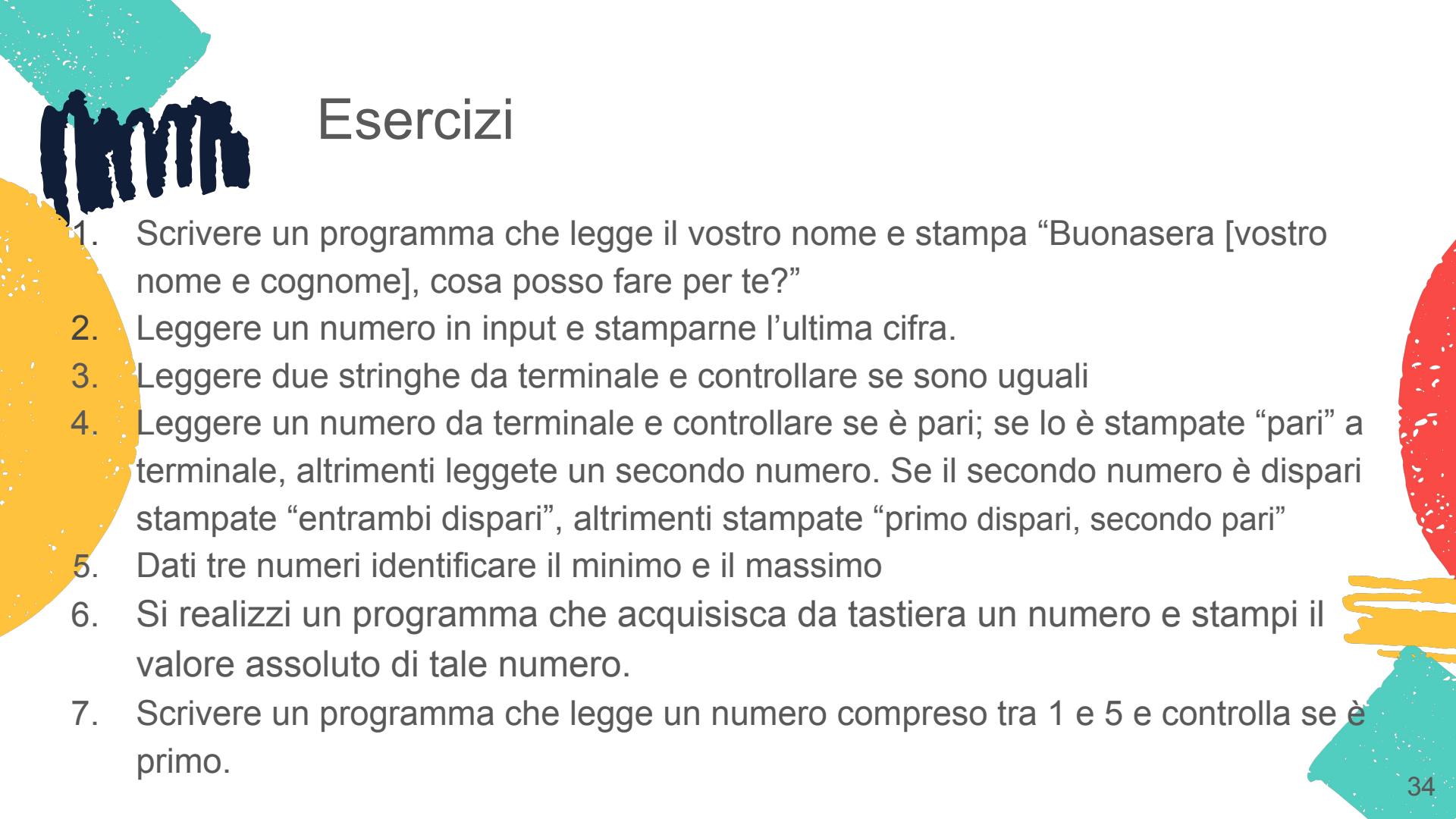




A QA engineer walks into a bar.  
Orders a beer. Orders 0 beers.  
Orders 99999999999 beers.  
Orders a lizard. Orders -1 beers.  
Orders a ueicbksjdhd.

First real customer walks in  
and asks where the bathroom  
is. The bar bursts into flames,  
killing everyone.





# Esercizi

1. Scrivere un programma che legge il vostro nome e stampa “Buonasera [vostro nome e cognome], cosa posso fare per te?”
2. Leggere un numero in input e stamparne l’ultima cifra.
3. Leggere due stringhe da terminale e controllare se sono uguali
4. Leggere un numero da terminale e controllare se è pari; se lo è stampate “pari” a terminale, altrimenti leggete un secondo numero. Se il secondo numero è dispari stampate “entrambi dispari”, altrimenti stampate “primo dispari, secondo pari”
5. Dati tre numeri identificare il minimo e il massimo
6. Si realizzi un programma che acquisisca da tastiera un numero e stampi il valore assoluto di tale numero.
7. Scrivere un programma che legge un numero compreso tra 1 e 5 e controlla se è primo.

# Esercizi

Scrivere un programma che legga la temperatura in centigradi e mostri un messaggio che segua la seguente didascalia:

$T < 0$ : “clima gelido”

$T 0-10$ : “clima veramente freddo”

$T 10-20$ : “clima freddo”

$T 20-30$ : “Temperatura normale”

$T 30-40$ : “clima caldo”

$T 40$ : “clima veramente caldo”

# Esercizi Avanzati

Scriviamo un programma che per prima cosa stampa la stringa “Dite amici ed entrate” e ci chiede di inserire una parola (che dovrebbe essere “amici”). Se sbagliamo il programma dovrà scrivere “Non succede nulla, ma sento una presenza che si avvicina” e dovrà chiederci nuovamente la parola; se sbaglieremo ancora dovrà scrivere “Non succede nulla, ma la presenza è molto vicina” e dovrà richiederci la parola. Se sbagliamo per la terza volta scriverà “Il mostro è qui, dobbiamo fuggire!” e poi il programma deve finire.

Se in qualsiasi momento inseriamo la parola “amici” il programma deve stampare “Benvenuti a Moria” e deve terminare.



# Esercizi Avanzati

Si scriva un programma che legga da tastiera i valori delle lunghezze dei tre lati di un triangolo (detti A, B e C), e determini:

- se il triangolo è equilatero
- se il triangolo è isoscele
- se il triangolo è scaleno
- se il triangolo è rettangolo.

# Esercizi Avanzati

Data l'equazione

$$ax + b = 0$$

con a e b inseriti da tastiera, scrivere un programma per  
determinare il  
valore di x, se esiste, che risolve l'equazione

# Esercizi Avanzati

Si realizzi un programma per risolvere equazioni di secondo grado.

In particolare, data una generica equazione di secondo grado nella forma  
 $ax^2 + bx + c = 0$

dove a, b, c sono coefficienti reali noti e x rappresenta l'incognita, il  
programma determini

le due radici  $x_1$  ed  $x_2$  dell'equazione data, ove esse esistano.

Si identifichino tutti i casi particolari ( $a = 0$ ,  $\Delta \leq 0$ , ...) e si stampino gli  
opportuni messaggi informativi.