Corso ITS: *ARTIFICIAL INTELLIGENCE SPECIALIST*

Modulo: **Programmazione Procedurale in Python**

Docente: *Andrea Ribuoli*

Martedì **21 Gennaio** 2025

09:00 - 14:00

In [31]:
```
print("Hello, World!")
```

```
Hello, World!
```

how

- istruzioni (o **enunciati**)
- commenti
- funzioni *built-in*: **print**
- indentazione
- **errori** (compilazione, esecuzione, logico)
- **pseudocodice** (*algoritmo*)

- **variabili** e **costanti**
- enunciato di **assegnazione** (=)
- alla prima assegnazione la variabile viene anche **creata**
- il tipo di dato è **associato al valore**, non alla variabile
- tipi di dato: **primitivi** e **definiti dall'utente**
- numeri **letterali** e tipi (**int** .vs. **float**)

- **nomi** delle variabili (`[_a-zA-Z][_a-zA-Z0-9]*`)
- ossia i nomi di variabile devono:
- - iniziare con una lettera o **segno di sottolineatura**
- - i caratteri seguenti possono essere lettere, **cifre** o segni di sottolineatura
- - sono **case sensitive**
- - non possono coincidere con **parole riservate**
- stile (convenzioni)

- **costanti**: pura convenzione in Python
- considerazioni:
- - *undefined name*?
- - nomi **significativi**
- - no numeri **magici**

- **operatori** aritmetici (`+` , `-` , `*` , `/`)
- **espressione**: combinazione di
- - nomi di varibili
- - letterali
- - operatori
- - parentesi **tonde**
- moltiplicazione e divisione hanno la precedenza su addizione e sottrazione
- senza parentesi, a parità di precedenza, si procede da sinistra verso destra
- il tipo risultante di una espressione numerica **mista** è **float**
- elvamento a **potenza** (`**`)
- divisione intera (`//`)
- resto divisione intera, detto anche **modulo**, (`%`)

- funzioni che **restituiscono** un valore, cioè?
- **argomenti**: molte funzioni richiedono dati
- alcune funzioni hanno argomenti **facoltativi**
- la documentazione ufficiale di Python (e.g. `round(x[, n])`)
- alcune funzioni sono **predefinite* (**built-in**)
- Python offre una **libreria standard** organizzata in moduli
- ulteriori moduli possono essere installati (spesso **open-source**)
- `- from math import sqrt`
- `- from math import sqrt, sin, cos`
- `- from math import *`
- `- import math` (impone `math.` nel codice)

- considerazioni:
- - bilanciamento parentesi
- - spaziature nelle espressioni (no con operatori **unari**)
- - combinare assegnazione e aritmetica (`+=` , `-=` , `*=` , eccetera)
- - espressioni multi-riga (se parentesi non bilanciate)

- una **stringa** è una sequenza di caratteri
- una stringa **letterale** si scrive racchiusa tra virgolette (semplici o **doppie**)
- la convenzione di utilizzare le **doppie** accomuna il Python ad altri linguaggi (e.g.**C**)
- la **lunghezza** di una stringa è il numero di caratteri di cui è composta

- il numero di **byte** utilizzati per memorizzarla può differire
- `""` (o `''`) è il letterale della stringa **vuota**
- l'operatore `+` **concatena** stringhe (`name = firstName + " " + lastName`)

In [32]:
```
nome = "Fabio"
cognome = "Rossi"
anni = 62
scheda = nome + cognome + anni
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[32], line 4
      2 cognome = "Rossi"
      3 anni = 62
----> 4 scheda = nome + cognome + anni

TypeError: can only concatenate str (not "int") to str
```

- l'operatore `*` ripete una stringa il numero di volte prefissato

In [10]:
```
greca = "|/--\\" * 16 + "|"
print(greca)
```

```
|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--\|/--
\|/--\|
```

- un valore numerico si converte in stringa meditante la funzione **str**
- una stringa si trasforma (viceversa) in un valore numerico con le funzioni **int** e **float**
- la stringa non può contenere una espressione

In [11]:
```
valore = int("7 * 8")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[11], line 1
----> 1 valore = int("7 * 8")

ValueError: invalid literal for int() with base 10: '7 * 8'
```

- le stringhe sono sequenze di caratteri **Unicode**
- è possibile accedere ai singoli caratteri tramite l'**indice** (`[index]`)
- l'indice è la **posizione** conteggiata partendo da **zero** (**0**)
- il valore deve appartenere all'intervallo di posizioni valide

In [16]:
```
print(greca[9])
```

```
\
```

In [17]:
```
print(greca[100])
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[17], line 1
----> 1 print(greca[100])

IndexError: string index out of range
```

- per estrarre l'ultimo carattere di una stringa l'indice deve essere `len(s) - 1`

In [20]:
```
print(greca[len(greca) - 1])
```

|

- la stringa è un **oggetto** (vedremo nel prossimo modulo)
- il comportamento di un oggetto è definito dai suoi **metodi**
- un metodo è una funzione che opera su un oggetto *del tipo per cui è definito* (cfr. **istanza**)

In [24]:
```
nome = "Andrea Ribuoli"
print(nome.upper() + "\n" + nome.lower())
```

```
ANDREA RIBUOLI
andrea ribuoli
```

In [26]:
```
print(nome.replace("Andrea", "Giovanni")))
```

```
Giovanni Ribuoli
```

In [27]:
```
print("La lettera A ha codice ", ord("A"))
```

```
La lettera A ha codice  65
```

In [29]:
```
print("La lettera À ha codice ", ord("À"))
```

```
La lettera À ha codice  192
```

In [30]:
```
print("La lettera € ha codice ", ord("€"))
```

```
La lettera € ha codice  8364
```

- sequenze di *escape* con **backslash** (\ )
- nuova riga, o **newline**, (\n)

In [33]:
```
print("L'autore de \"Il nome della rosa\"")
```

```
L'autore de "Il nome della rosa"
```

- impaginazione dei risultati con l'**operatore di formato per stringhe** (%)

In [56]:
```
formato = "%3d %-26s 10%% %9.2f"
print("Qta Prodotti                    IVA    TOTALE\n" +
      formato % (2, "Hamburger", 2.8) + "\n" +
      formato % (1, "6 McNuggets", 4.9) + "\n" +
```

```
    formato % (1, "McMenu Large Big Mac", 8.8))
```

```
Qta Prodotti                   IVA    TOTALE
  2 Hamburger                  10%      2.80
  1 6 McNuggets                10%      4.90
  1 McMenu Large Big Mac       10%      8.80
```