Elementi di base

Operazioni:

Le quattro operazioni: + * - /

Elevamento a potenza: **

Divisione:

- / divisione : 3 / 4 = 0.75
- // parte intera della diviisone: 3 // 4 = 0
- **%** resto della diviisone 3 % 4 = 3

Assegnamento

x = 5

assegna alla variabile x il valore 5 (indipendentemente da quello che era il valore presente in x fino a quel momento)

Assegnamento composto

y = x + 3

assegna alla variabile y il valore (in quell momento) della variabule x ed aggiunge 3 (indipendentemente da quello che era il valore presente in y fino a quel momento)

Funzioni predefinite importanti:

- ***len(X)*:** ritorna il numero di elementi presenti in X
- **input()**: ritorna la stringa inserita dall'utente
- **type(X):** ritorna il tipo della variabile X
- **float(X):** ritorna il valore di X come tipo di dati float
- **int(X)**: ritorna il valore di X come tipo di dati int
- **str(X)**: ritorna il valore di X come tipo di dati string

- **round(X,<num>)**: ritorna il valore di X con sole <num> posizioni decimali. Se non si mette <num> arrotonda all'intero
- **print(X)**: stampa X
- **help(X)**: ritorna la descrizione di X

Decisioni Cicli e Funzioni

Decisioni:

if <condizione>:

    <espressione> #NOTA : [4spazi]

oppure

if <condizione>:

    <espressione> #NOTA : [4spazi]

else:

    <espressione> #NOTA : [4spazi]

gli operatori condizionali sono == , <, >, *!*= e si possono concatenare con le parole chiave

and, or, not

per ottenere delle delle condizioni complesse, dette "espressioni booleane" il cui valore può essere *True* o *False*

La condizione è verificata se il valore della espressione booleana è *True*

se serve concatenere le decisioni:

if <condizione>:

    <espressione> #NOTA : [4spazi]

elif <condizione>::

    <espressione> #NOTA : [4spazi]

else:

    <espressione> #NOTA : [4spazi]

Cicli: il costrutto for

for <regola>:

    <espressione> #NOTA : [4spazi]

dove <regola> indica la variabile di cilclo ed il range prederminato in cui valorizzarla

la funzione predefinita *range()* è quella più usata.

for i in range(4):

    print(i)

Cicli: il costrutto while

while <condizione>:

    <espressione> #NOTA : [4spazi]

Funzioni: definizione

per definire una funzione si usa la parola chiave "def", cui seguono il nome della funzione ed i parametri che la funzione accetta

esempio

def somma(i,j):

    result = i+j #NOTA : [4spazi]

    return result

Stringhe

Le stringhe sono sequenze di caratteri, indirizzate a partire da 0; sono delimitate indifferentemente da " o da '; sono un tipo di dati *immutabile*.

A = "ciao"
Metodi utili per le stringhe

- *len*(A) restituisce la lunghezza in caratteri della stringa A
- *A.upper()* trasforma ogni carattere della stringa A in maiuscolo
- A.*lower*() trasforma ogni carattere della stringa A in minuscolo
- A.*split(sep*=B) separa la stringa A in una sequenza di stringhe, usando come separatore la stringa B
- C = A + B concatena due stringhe
- C = A * n produce una stringa frutto di n concatenazioni della stringa A (es A*3 --> "ciaociaociao")
- A.*replace(b,c)* sostituisce b nella stringa A con c (es. A.replace('o','u') -> "ciau")
- A.*find(b*) ritorna l'indice dell'inizio della b all'interno della stringa A , oppure -1 se A non contiene b
- A.*count(b)* conta tutte le occorrenze della sottostringa b in A (es. A.count("o") -> 1)
- A.*strip()* rimuove gli spazi iniziali e finali dalla stringa A
- *A[start:stop:passo]* esegue lo slicing della stringa A a partire da *start* fino a *stop* ogni *passo*;
  se si omette *passo*, di default passo vale 1,
  se si omette *start* questo di default vale zero,
  se si omette *stop* questo di default vale len(A)

Per quanto riguarda i caratteri,

- *ord(X)* restituisce il codice numerico Unicode di X
- *char(n)* restituisce il carattere corrispondente al codice numerico Unicode n