Esercitazione di Laboratorio 03

Temi trattati

1. Elaborazioni e confronti tra variabili con valori di tipo `int`, `float` e `str`
2. Espressioni booleane
3. Scelte logiche
4. Costrutti condizionali (`if`, `else`, e `elif`)
5. Blocchi e istruzioni annidate

Discussione

A. Quali sono le possibili forme di sintassi di un costrutto condizionale?
B. Spiegare le similitudini e le differenze tra:
    a. operatori condizionali e operatori booleani;
    b. assegnazione e test di uguaglianza;
    c. `and`, `or`, e `not`.
C. Come si definiscono le istruzioni composte (compound statement)?

Esercizi

Parte 1 – Confronti, operatori relazionali e booleani

***Consegna**: per ciascuno degli esercizi seguenti, scrivere un programma in Python che risponda alle richieste indicate. Completare <u>almeno due</u> esercizi durante l'esercitazione, e i rimanenti a casa.*

03.1.1 Vero o Falso. Per ciascuna delle seguenti coppie di valori, eseguire un test di *uguaglianza*, assegnare il risultato ad una variabile, e visualizzarlo. Provare a prevedere quale sarà il risultato di ciascuna verifica.

I. `1`, `1`;
II. `1`, `1.0`;
III. `2.0`, `sqrt(4)`;
IV. `'1'`, `1`;
V. `'ciao'`, `'Ciao'`.

03.1.2 Identikit della stringa. Scrivere un programma che legga una stringa e visualizzi i messaggi appropriati, dopo aver verificato se:

I. contiene soltanto lettere;
II. contiene soltanto lettere maiuscole;
III. contiene soltanto lettere minuscole;
IV. contiene soltanto cifre numeriche decimali;
V. contiene soltanto lettere e cifre;
VI. inizia con una lettera maiuscola;
VII. termina con un punto.

[P3.17]

03.1.3 Stringhe e sottostringhe. Le sequenze di DNA sono come lunghe stringhe composte da solo quattro lettere: `'A'`, `'C'`, `'T'` e `'G'`. Scrivere un programma che prende in input una "sequenza lunga" di DNA di venti caratteri e una "sequenza breve" di tre caratteri e visualizza:

I. se la "sequenza lunga" contiene la "sequenza breve";
II. se presente, a partire da quale posizione della "sequenza lunga" è presente la "sequenza breve";
III. se presente, quante volte la sequenza "lunga" contiene la sottostringa più breve.

[Capitolo 3.8]

03.1.4 È uguale. Descrivere lo pseudocodice corrispondente al programma Python seguente:

```
x = 3.0
s == 'sette punto cinque'
if x = 3.0:
    s == 'tre punto zero'
print(s)
```

Quale è il risultato dell'esecuzione di questo programma?

03.1.5 Questione di logica. Scrivere un programma che prende in input un numero intero `x` e visualizza a schermo le seguenti espressioni, accompagnate dai loro valori di verità. Testare il programma con numerosi valori di `x`.

I. `x > 0 and x < 100`
II. `x > 0 or x < 100`
III. `x > 0 or 100 < x`
IV. `x > 0 and x < 100 or x == -1`

[Capitolo 3.7]

03.1.6 De Morgan. La legge di De Morgan permette di semplificare l'applicazione dell'operatore `not` ad espressioni che contengono operatori `and`/`or`. In particolare, questa legge ha due forme, una per la negazione di espressioni in `and`, e una per la negazione di espressioni in `or`:

`not (A and B)`	equivale a	`not A or not B`
`not (A or B)`	equivale a	`not A and not B`

Considerare le espressioni seguenti, e per ciascuna di esse applicare la legge di De Morgan. Provare a descrivere "a parole" il significato algebrico intuitivo di ciascuna delle espressioni.

Scrivere poi un programma che prende in input un numero intero `x` e per ciascuna delle seguenti espressioni (che corrispondono alla negazione delle espressioni dell'esercizio **03.1.5**) stampa: l'espressione di partenza, l'espressione una volta applicata la legge di De Morgan, e il loro valore di verità:

I. `not (x > 0 and x < 100)`
II. `not (x > 0 or x < 100)`
III. `not (x > 0 or 100 < x)`
IV. `not (x > 0 and x < 100 or x == -1)`

[Special Topic 3.5]

Parte 2 – Decisioni

***Consegna**: per ciascuno degli esercizi seguenti, scrivere un programma in Python che risponda alle richieste indicate. Completare almeno due esercizi durante l'esercitazione, e i rimanenti a casa.*

03.2.1 Andamenti. Scrivere un programma che legga tre numeri e visualizzi il messaggio "increasing" se sono in ordine strettamente crescente, "decreasing" se sono in ordine strettamente decrescente e "neither" se non sono né in ordine crescente né in ordine decrescente. "Strettamente crescente" significa che ciascun valore deve essere maggiore del precedente (analogo significato ha il termine decrescente): la sequenza `3 4 4`, quindi, non va considerata crescente. [P3.5]

03.2.2 Voti. Scrivere un programma che traduca un voto in lettere inserito dall'utente nel corrispondente voto numerico e lo stampi. I voti in lettere sono `'A'`, `'B'`, `'C'`, `'D'` e `'F'`, eventualmente seguiti da un segno `+` o `–`. I loro valori numerici sono, nell'ordine, `4.0`, `3.0`, `2.0`, `1.0` e `0.0`. I voti `'F+'` e `'F–'` non esistono. Un segno `+` aumenta il voto numerico di 0.3, mentre un segno `–` lo diminuisce della stessa quantità. Il voto `'A+'` è comunque uguale a `4.0`. [P3.12]

03.2.3 Cicli stagionali. L'algoritmo seguente (già visto nell'esercizio **01.1.2**) individua la stagione (Spring, Summer, Fall o Winter, cioè, rispettivamente, primavera, estate, autunno o inverno) a cui appartiene una data, fornita come mese e giorno.

Se mese è 1, 2 o 3
    stagione = "Winter"
Altrimenti se mese è 4, 5 o 6
    stagione = "Spring"
Altrimenti se mese è 7, 8 o 9
    stagione = "Summer"
Altrimenti se mese è 10, 11 o 12
    stagione = "Fall"
Se mese è divisibile per 3 e giorno >= 21
    Se stagione è "Winter"
        stagione = "Spring"
    Altrimenti se stagione è "Spring"
        stagione = "Summer"
    Altrimenti se stagione è "Summer"
        stagione = "Fall"
    Altrimenti
        stagione = "Winter"

Riprendere l'analisi dell'algoritmo e scrivere poi un programma che, implementandolo, chieda all'utente un mese e un giorno e, poi, visualizzi la stagione determinata da questo algoritmo, verificandone la correttezza. [P3.20]

03.2.4 Anni bisestili. Un anno di 366 giorni viene detto bisestile e serve a mantenere il calendario sincronizzato con il Sole, dal momento che la Terra vi ruota attorno una volta ogni 365.25 giorni circa. In realtà, questo numero non è esatto, e per tutte le date successive al 1582 si applica la correzione gregoriana: solitamente gli anni divisibili per 4, come il 1996, sono bisestili, ma gli anni divisibili per 100, come il 1900, non lo sono; come eccezione all'eccezione, gli anni divisibili per 400, come il 2000, sono bisestili. Scrivere un programma che chieda all'utente un anno (maggiore del 1582) e determini se si tratta di un anno bisestile usando un unico costrutto `if` e gli operatori booleani. [P3.27]

03.2.5 Ancora voti. Considerando i valori numerici dei voti spiegati nell'esercizio **03.2.2**, scrivere un programma che traduca un numero compreso tra `0.0` e `4.0` nel voto letterale ad esso più vicino. Ad esempio, il numero `2.8` (che potrebbe essere la media di più voti) deve essere tradotto come `'B-'`. Risolvete i casi di parità in favore del voto migliore: ad esempio, `2.85` deve essere tradotto come `'B'`. [P3.13]

03.2.6 Tassazioni. Scrivere un programma che calcoli le tasse secondo il seguente schema. Il programma deve acquisire dall'utente il valore del reddito, e stampare le tasse dovute. Non è richiesto di stampare i passaggi intermedi. [P3.25]

Per stato civile "non coniugato" e reddito imponibile superiore a	**ma non superiore a**	**le tasse sono**	**della somma eccedente**
$ 0	$ 8000	10%	$ 0
$ 8000	$ 32 000	$ 800 + 15%	$ 8 000
$ 32 000		$ 4 400 + 25%	$ 32 000
Per stato civile "coniugato" e reddito imponibile superiore a	**ma non superiore a**	**le tasse sono**	**della somma eccedente**
$ 0	$ 16 000	10%	$ 0
$ 16 000	$ 64 000	$ 1 600 + 15%	$ 16 000
$ 64 000		$ 8 800 + 25%	$ 64 000

03.2.7 Conversioni. Scrivere un programma per la conversione di unità di misura che chieda all'utente: l'unità di misura di partenza (scegliendo tra: `ml`, `l`, `g`, `kg`, `mm`, `cm`, `m`, `km`); l'unità di misura verso la quale vuole effettuare una conversione (scegliendo tra: `fl`, `oz`, `gal`, `oz`, `lb`, `in`, `ft`, `mi`), rifiutando conversioni incompatibili (come, ad esempio, da `km` a `gal`); il valore da convertire. Il programma deve visualizzare i dati inseriti e il valore derivante dalla conversione. [P3.26]

03.2.8 Buoni spesa. Un supermercato premia i propri clienti con buoni spesa il cui importo dipende dalla quantità di denaro spesa in prodotti alimentari. Ad esempio, spendendo `50 $`, si ottiene un buono spesa di importo pari all'`8%` di quella somma. La tabella seguente mostra la percentuale usata per calcolare il buono spesa relativo a somme diverse. Scrivere un programma che calcoli e visualizzi il valore del buono spesa consegnato al cliente, sulla base della somma di denaro che ha speso nell'acquisto di prodotti alimentari. [P3.40]

Denaro speso	**Percentuale del buono**
Meno di $ 10	Nessun buono
Da $ 10 a $ 60	8%
Da più di $ 60 a $ 150	10%
Da più di $ 150 a $ 210	12%
Più di $ 210	14%

03.2.9 Lunghezze d'onda. Scrivere un programma che chiede in input all'utente un valore di lunghezza d'onda (numero reale, che potrà essere scritto in notazione scientifica, ad esempio `1.23e-7`), e visualizza la descrizione della parte corrispondente dello spettro elettromagnetico, come illustrato nella tabella seguente. [P3.43]

Tipo	Lunghezza d'onda (m)	Frequenza (Hz)
Onde Radio	$> 10^{-1}$	$< 3 \times 10^{9}$
Microonde	$Da\ 10^{-3}\ a\ 10^{-1}$	$Da\ 3 \times 10^{9}\ a\ 3 \times 10^{11}$
Infrarossi	$Da\ 7 \times 10^{-7}\ a\ 10^{-3}$	$Da\ 3 \times 10^{11}\ a\ 4 \times 10^{14}$
Luce visibile	$Da\ 4 \times 10^{-7}\ a\ 7 \times 10^{-7}$	$Da\ 4 \times 10^{14}\ a\ 7.5 \times 10^{14}$
Ultravioletti	$Da\ 10^{-8}\ a\ 4 \times 10^{-7}$	$Da\ 7.5 \times 10^{14}\ a\ 3 \times 10^{16}$
Raggi X	$Da\ 10^{-11}\ a\ 10^{-8}$	$Da\ 3 \times 10^{16}\ a\ 3 \times 10^{19}$
Raggi Gamma	$< 10^{-11}$	$> 3 \times 10^{19}$

03.2.10 Ritorno alla cometa. Una persona mediamente può saltare staccandosi da terra con una velocità di 11 chilometri orari senza dover temere di lasciare la superficie terrestre. Al contrario, se una persona saltasse con la stessa velocità mentre si trova sulla Cometa di Halley, riuscirebbe a tornare sulla superficie? Creare un programma che permetta all'utente di inserire in input una velocità di lancio (in km orari) dalla superficie della Cometa di Halley, e di determinare se la persona che salta sarà in grado di tornare sulla superficie. In caso contrario, il programma dovrà calcolare quanta massa in più dovrebbe avere la cometa perché accada.

Suggerimento: la velocità di fuga è

$$v_{escape} = \sqrt{2\frac{G\,M}{R}},$$

dove $G = 6.67 \times 10^{-11}\ N\,m^2\ /\ kg^2$ è la costante gravitazionale, M è la massa del corpo celeste, ed R è il suo raggio. La Cometa di Halley ha una massa di $2.2 \times 10^{14}\ kg$ ed un diametro di $9.4\ km$. [P3.52]