Esercizi

Lezione 1: introduzione al linguaggio e richiami di programmazione

Per affrontare i seguenti esercizi potrebbero essere utili le slide e eventuale documentazione reperita online.

Esercizio 1

Variabili e operazioni aritmetiche

Creare uno script PHP che una volta definite quattro variabili:

- `$a = 4`
- `$b = 5`
- `$c = 'hello,'`
- `$d = ' world!'`

svolga le seguenti operazioni:

- realizzi le sei operazioni aritmetiche `+ - * / % **` sfruttando `$a` come primo operando e `$b` come secondo e salvi il risultato delle operazioni in una variabile denominata `$res`. Commentare a seguito di ogni operazione cosa si sta calcolando (somma, differenza, . . .);
- concateni le due stringhe `$c` e `$d` in una variabile denominata `$res`.

A seguito di ogni modifica della variabile `$res` stamparne il contenuto mediante il comando `echo`.

NB: per rendere più pulito l'output è opportuno usare l'istruzione `echo $variabile."<br>";` di modo che si concateni un *a capo* HTML (ricordiamo che lavorando con un browser non possiamo usare `\n`).

Esercizio 2

Il costrutto if - else if - else

Dato lo script precedentemente costruito stampare mediante un costrutto `if- else if - else` quale tra le variabili `$a` e `$b` è la maggiore. In particolare l'utente dovrà vedere stampata la frase `Il valore della variabile $b è maggiore del valore della variabile $a` se il valore di `$b` e maggiore di quello di `$a` o eventualmente il viceversa.

Esercizio 3

Il costrutto while

Creare uno script PHP che stampi a schermo:

- i primi 100 numeri pari;
- i primi 20 termini della sequenza di Knuth per l'algoritmo Shell sort. *soluzione: la successione parte con 1 e vede ogni elemento successivo il triplo del precedente addizionato di 1, dunque i primi termini saranno 1, 4, 13, 40, ...*

Approfondimento (facoltativo)

Lo Shell sort e la sequenza di Knuth

In questo approfondimento viene spiegato un argomento off-topic rispetto al corso PHP ma che è utile per la cultura di base dell'informatica e che potrà servire a chi intraprende tale percorso di studio.

Il problema dell'ordinamento dei dati In Informatica, in quanto scienza derivata dalla matematica, tutto è oggetto di dibattito, discussione e miglioramento. È per questo che operazioni anche semplici quali ordinare un elenco di dati possono essere analizzate in infiniti modi elaborando soluzioni più o meno efficienti.

In particolare pensiamo all'ordinamento dei dati: un problema che a prima vista può sembrare di nicchia e semplice ma che in realtà mostra infinite applicazioni pratiche (pensiamo già solo quanto sia più facile cercare un elemento di una lista ordinata anziché di una disordinata) e molteplici approcci che vedono una natura più o meno *umana* rispetto che da *calcolatore*.

Si può dimostrare grazie ai grafi che la complessità minima di un algoritmo di ordinamento è *linearitmica* ovvero O(nlog(n)). Sempre in base alla complessità possiamo quindi classificare gli ordinamenti di ordinamento in quadratici, linearitmici e lineari (questi ultimi non sono basati sul confronto ma sul calcolo della posizione finale).

Gli algoritmi di ordinamento quadratici Primi storici algoritmi adottati, sono i peggiori in termini di complessità: nel caso peggiore confrontano ogni elemento del vettore da ordinare con ogni elemento del vettore stesso ottenendo n^2 confronti. Deduciamo che siano dunque una soluzione sempre meno valida man mano che si aumenta il numero di elementi da ordinare (passiamo da massimo 100 confronti per 10 elementi a massimo 400 per 20, pensiamo adottarli per ordinare basi di dati da centinaia se non migliaia (o decine di migliaia) di righe).

Tra questi annoveriamo il:

- *Selection sort*: ordina il vettore selezionando l'elemento minore dello stesso e ponendolo a sinistra;
- *Insertion sort*: ordina il vettore andando a posizionare gli elementi alla sinistra dello stesso nella posizione opportuna;
- *Bubble sort*: ordina il vettore confrontando ogni elemento con il successivo fino a far traslare l'elemento maggiore di ogni iterazione alla sinistra del vettore.

Cos'è lo Shell sort? Nel 1959 il matematico Donald L. Shell teorizzò la possibilità di ottimizzare l'algoritmo Insertion sort variando il passo dello stesso ovvero ordinando non più gli elementi distanti uno dal precedente ma sfruttando una diversa sequenza.

Shell teorizzò solo ciò, infatti la sequenza che elaborò non portò a un effettivo vantaggio in termini computazionali, l'algoritmo di ordinamento rimase di complessità quadratica (rispetto ai confronti).

Serviranno altre sequenze per migliorarne la complessità: fra queste annoveriamo le sequenze di Pratt, Knuth e Hibbard. In particolare la sequenza di Knuth consente di abbassare la complessita fino a O(n^4/3), decisamente migliore di quella di partenza. Tale sequenza, come visto nell'esercizio, si basa sulla moltiplicazione per 3 addizionata di 1 per determinare il termine successivo; otteniamo dunque:

1 4 13 40 121 ...

Naturalmente la applicheremo al vettore scegliendo il maggiore elemento della successione minore o uguale della dimensione del vettore stesso: per un vettore di 15 elementi sceglieremo h=13 (e poi 4 e infine 1), per un vettore di 120 elementi sceglieremo 40 (e poi 13 e poi 4 e infine 1).