

Array e strutture condizionali

Corso Backend System Integrator
Modulo **Programmazione PHP**

Docente: Dott. Enrico Zimuel

in collaborazione con:



REGIONE
PIEMONTE

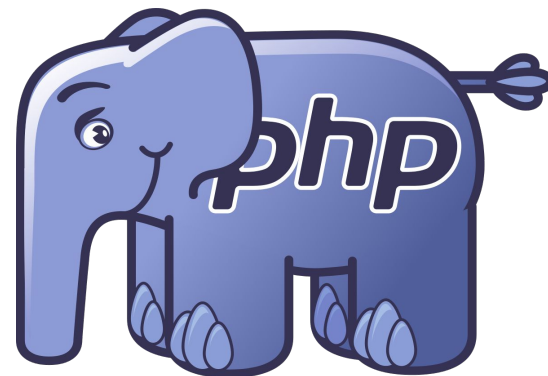
per una crescita intelligente,
sostenibile ed inclusiva

www.regione.piemonte.it/europa2020

INIZIATIVA CO-FINANZIATA CON FSE

Programma

- printf()/sprintf()
- Array
- Stringhe come array
- Istruzioni condizionali
 - if/then
 - if/then/else



printf

- La funzione [printf\(\)](#) viene utilizzata per la stampa di stringhe e variabili, con la possibilità di specificarne la formattazione

```
<?php
$r = 2;
$area = M_PI * $r * $r;
printf("Un cerchio di raggio %d ha area %.4f \n", $r, $area);
$name = 'Alberto';
printf("Hello %s\n", $name);
```

sprintf

- [sprintf\(\)](#) funziona allo stesso modo di `printf()` ma restituisce il risultato come stringa, al posto di stamparlo

```
<?php
$name = 'Alberto';
$hello = sprintf("Hello %s", $name);
printf("%s\n", $hello); // equivale a echo $hello .
"\n";
```

Esercizio

- Consultando la documentazione della funzione [sprintf\(\)](#) stampare il valore binario, ottale ed esadecimale del numero 32 utilizzando l'opportuna formattazione

Array

- Un array è una **sequenze ordinata di valori**, separati da virgola
- Ogni elemento di un array è individuato da una posizione (indice)
- Il primo elemento ha indice zero (come nel linguaggio C)
- Vengono definiti con la sintassi **array(...)** oppure **[...]**

Esempi di array

```
<?php
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
printf("Primo giorno della settimana: %s \n", $week[0]);

$foo = [12, 34.2, M_PI, 'Alberto'];
printf("Ultimo elemento: %s \n", $foo[3]);

$foo = array(12, 34.2, M_PI, 'Alberto');
printf("Secondo elemento: %.2f \n", $foo[1]);
```

Aggiunta di un elemento

- E' possibile aggiungere un elemento in un array incrementando l'indice

```
$foo = ['a', 'b', 'c'];  
$foo[3] = 'd'; // ['a', 'b', 'c', 'd']
```

- Oppure aggiungerlo in coda non indicando nessun indice

```
$foo = ['a', 'b', 'c'];  
$foo[] = 'd'; // ['a', 'b', 'c', 'd']
```


Array di array

- Un array può contenere un altro array come elemento

```
$foo = ['a', [ 'b', 'c' ], 'd'];  
printf("%s, %s\n", $foo[0], $foo[1][1]); // a, c
```

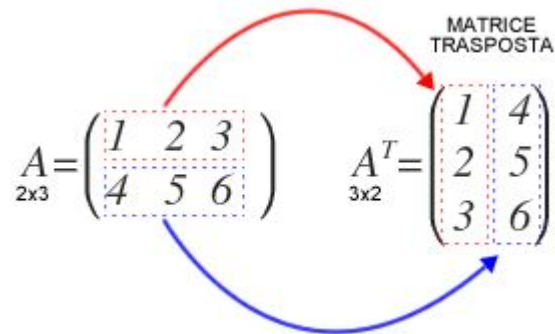
Esempio: matrice

- Una matrice è un array di n elementi dove ogni elemento è un array di m elementi (si dice che la matrice ha dimensione $n \times m$)

```
$matrix = [  
    [ 5, 7, 9 ],  
    [ 4, 3, 8 ],  
    [ 7, 5, 6 ]  
];  
  
printf ("Elemento [0,0] = %d\n", $matrix[0][0]); // 5  
printf ("Elemento [1,1] = %d\n", $matrix[1][1]); // 3  
printf ("Elemento [2,2] = %d\n", $matrix[2][2]); // 6
```

Esercizio

- Data una matrice di dimensione 3×2 calcolare la sua [matrice trasposta](#) e stamparla a video
- **NOTA:** non si possono utilizzare cicli o istruzioni condizionali



Stampa degli elementi di un array

- Non è possibile stampare il contenuto di un array con le funzioni echo o printf
- E' necessario utilizzare la funzione [print_r\(\)](#)

```
$matrix = [  
    [ 5, 7, 9 ],  
    [ 4, 3, 8 ],  
    [ 7, 5, 6 ]  
];  
print_r($matrix);
```



```
Array  
    [0] => Array  
        [0] => 5  
        [1] => 7  
        [2] => 9  
  
    [1] => Array  
        [0] => 4  
        [1] => 3  
        [2] => 8  
  
    [2] => Array  
        [0] => 7  
        [1] => 5  
        [2] => 6
```

Funzioni sugli array

- Il PHP offre numerose funzioni per la gestione degli array
- Alcune di queste funzioni:
 - [count](#)(\$array), restituisce il numero degli elementi di \$array
 - [sort](#)(\$array), ordina gli elementi di un array in ordine crescente
 - [rsort](#)(\$array), ordina gli elementi di un array in ordine decrescente
 - [shuffle](#)(\$array), mischia gli elementi di un array in ordine pseudo-casuale
 - [Qui](#) l'elenco di tutte le funzioni
- Navigare in un array: [current](#)(), [key](#)(), [next](#)(), [prev](#)(), [reset](#)(), [end](#)()

Navigare un array

- Navigare in un array: [current\(\)](#), [key\(\)](#), [next\(\)](#), [prev\(\)](#), [reset\(\)](#), [end\(\)](#)

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];  
printf("Giorni della settimana: %d\n", count($week));  
printf("Corrente   : %d, %s\n", key($week), current($week));  
next($week);  
printf("Successivo: %d, %s\n", key($week), current($week));  
prev($week);  
printf("Precedente: %d, %s\n", key($week), current($week));  
end($week);  
printf("Ultimo      : %d, %s\n", key($week), current($week));  
reset($week);  
printf("Primo       : %d, %s\n", key($week), current($week));
```

Eliminare un elemento

- Per eliminare un elemento da un array è necessario utilizzare la funzione [array_splice\(\\$array, \\$index, \\$length\)](#) dove **\$array** è l'array, **\$index** è la posizione dell'elemento e **\$length** è il numero di elementi da eliminare (1 nel caso di un singolo elemento)

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];  
array_splice($week, 1, 1);  
print_r($week);
```

Array

[0] => Mon
[1] => Wed
[2] => Thu
[3] => Fri
[4] => Sat
[5] => Sun

unset()

- E' anche possibile eliminare un elemento da un array utilizzando la funzione [unset\(\)](#)
- Questa funzione in realtà può essere utilizzata su qualsiasi variabile per eliminarla, liberando la memoria RAM

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];  
unset($week[1]);  
print_r($week);
```


Array

[0] => Mon
[2] => Wed
[3] => Thu
[4] => Fri
[5] => Sat
[6] => Sun

var_dump()

- La funzione [var_dump\(\)](#) stampa informazioni su una variabile, in particolare la dimensione in byte, il tipo e il contenuto
- Viene utilizzata principalmente per scopi di debug

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];  
var_dump($week);
```



```
array(7) {  
  [0] =>  
    string(3) "Mon"  
  [1] =>  
    string(3) "Tue"  
  [2] =>  
    string(3) "Wed"  
  [3] =>  
    string(3) "Thu"  
  [4] =>  
    string(3) "Fri"  
  [5] =>  
    string(3) "Sat"  
  [6] =>  
    string(3) "Sun"  
}
```

Stringhe == array

- Una stringa è una sequenza di caratteri ASCII
- In PHP una stringa è un array di caratteri

```
$name = 'Alberto';  
echo $name[0]; // A  
echo $name[1]; // l  
echo $name[2]; // b  
echo $name[7]; // Error: offset
```

Indice negativi su stringhe

- Nel caso delle stringhe si possono anche utilizzare degli indici negativi
- Un indice negativo inizia dalla fine della stringa

```
$name = 'Alberto';  
echo $name[-1]; // o  
echo $name[-2]; // t  
echo $name[-3]; // r  
echo $name[-7]; // A  
echo $name[-8]; // Error: offset
```

Array associativi

- Un array associativo ha come indice una stringa
- Sono conosciuti anche con il nome di [tabelle hash](#) o dizionari

```
$italianDay = [  
    'Mon' => 'Lunedì',  
    'Tue' => 'Martedì',  
    'Wed' => 'Mercoledì',  
    'Thu' => 'Giovedì',  
    'Fri' => 'Venerdì',  
    'Sat' => 'Sabato',  
    'Sun' => 'Domenica'  
];  
  
printf("Monday è %s in italiano\n", $italianDay['Mon']);  
printf("Today is %s in Italian\n", $italianDay[date('D')]);
```

ISTRUZIONI CONDIZIONALI

If-then

- Capita spesso, durante lo sviluppo di un programma, di dover eseguire porzioni di codice al verificarsi di una particolare condizione
- Questo compito viene svolto dalle istruzioni condizionali del tipo if-then

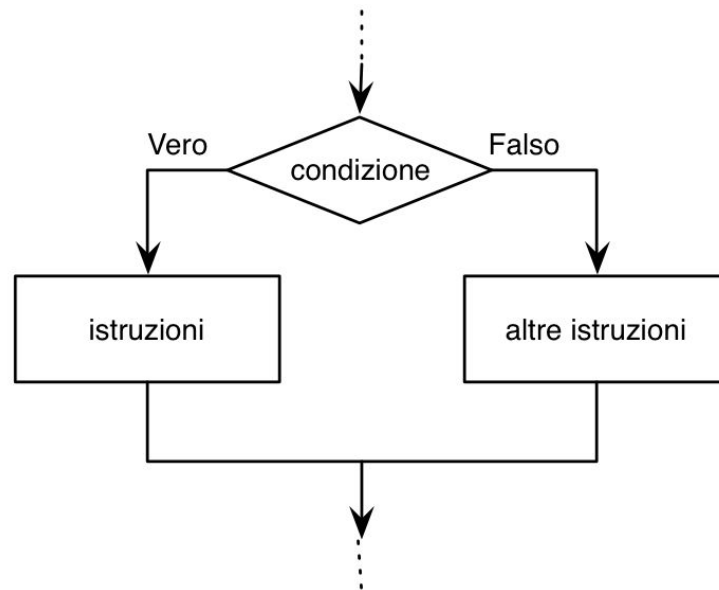
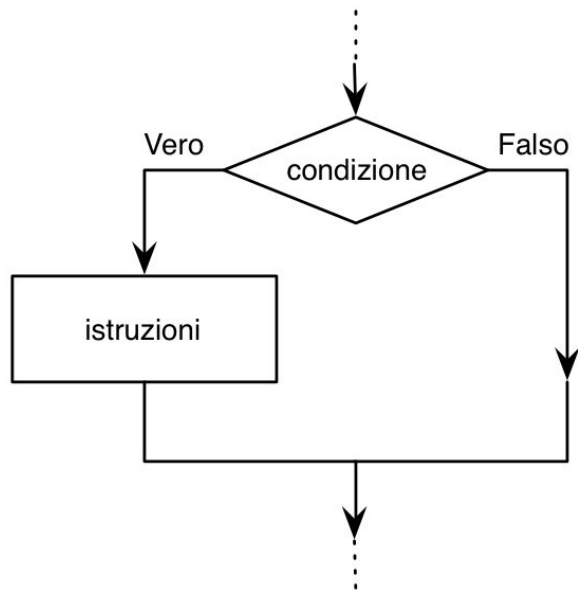
```
if ($a > $b) {  
    printf("%d è maggiore di %d\n", $a, $b);  
}
```

If-then-else

- Se una condizione è vera, esegui la porzione di codice **then**, altrimenti quella **else**

```
if ($a > $b) {  
    printf("%d è maggiore di %d\n", $a, $b);  
} else {  
    printf("%d è minore o uguale di %d\n", $a, $b);  
}
```

L'istruzione condizionale



Operatori di confronto

$\$a == \b	uguaglianza del contenuto
$\$a === \b	uguaglianza del contenuto e del tipo
$\$a != \b	non uguaglianza del contenuto
$\$a <> \b	non uguaglianza (identica alla precedente)
$\$a !== \b	non uguaglianza del contenuto o tipi differenti
$\$a < \b	minore
$\$a > \b	maggiore
$\$a <= \b	minore o uguale
$\$a >= \b	maggiore o uguale

Esempio

```
$a = 1;  
$b = '1';  
if ($a == $b) {  
    echo "Valori uguali\n";  
}  
if ($a === $b) {  
    echo "Valori e tipi uguali\n";  
}
```

```
$a = 'ciao';  
$b = 0;  
if ($a == $b) {  
    echo "Ma sono uguali?\n";  
}  
if ($a !== $b) {  
    echo "Ok, non sono uguali\n";  
}
```

Operatori logici

AND	&& oppure <i>and</i>	$\$a \&\& \b è vero solo se $\$a$ e $\$b$ sono entrambi veri
OR	$ $ oppure <i>or</i>	$\$a \b è vero se almeno uno tra $\$a$ e $\$b$ è vero
NOT	!	$!\$a$ è vero se $\$a$ è falso
XOR	xor	$\$a \text{ xor } \b è vero se $\$a$ è vero o $\$b$ è vero, ma non entrambi

NOT	
x	$!x$
0	1
1	0

AND		
x	y	$x \& y$
0	0	0
0	1	0
1	0	0
1	1	1

OR		
x	y	$x y$
0	0	0
0	1	1
1	0	1
1	1	1

XOR \oplus		
x	y	$x \text{ xor } y$
0	0	0
0	1	1
1	0	1
1	1	0

Esercizio (da consegnare)

- Verificare che un **codice fiscale** dato sia corretto o meno calcolando il [carattere di controllo](#)
- Il codice fiscale è memorizzato in una variabile **\$cf** e il programma deve stampare la scritta **VALIDO** oppure **NON VALIDO**
- **NOTA:** non si possono utilizzare cicli iterativi



Grazie dell'attenzione!

Per informazioni:

enrico.zimuel@its-ictpiemonte.it