PIEMONTE
FONDI STRUTTURALI E DI INVESTIMENTO EUROPEI 2014/2020
fondo sociale europeo
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Operatori e cicli iterativi

Corso Backend System Integrator

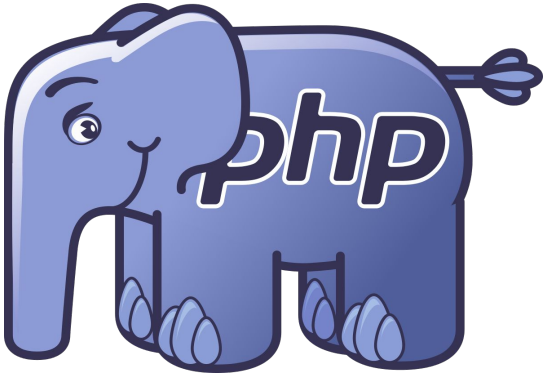
Modulo **Programmazione PHP**

Docente: Dott. Enrico Zimuel

in collaborazione con:
MINISTERO DELL'ISTRUZIONE, DELL'UNIVERSITÀ E DELLA RICERCA
UNIONE EUROPEA
REGIONE PIEMONTE
per una crescita intelligente, sostenibile ed inclusiva
www.regione.piemonte.it/europa2020
INIZIATIVA CO-FINANZIATA CON FSE

Programma

- Istruzioni condizionali
  - operatore ternario
  - operatore null coalesce
  - operatore spaceship
  - switch-case
- Cicli iterativi
  - for/foreach
  - while/do-while
  - break/continue



Operatore ternario

- E' possibile esprimere una condizione *if-then-else* in un'unica istruzione:
  <condizione> ? <expr1> : <expr2>
  se <condizione> è vera esegui <expr1> altrimenti <expr2>

```php
<?php
$a = 15;
if ($a > 10) {
   echo 'maggiore';
} else {
   echo 'minore o uguale';
}
// può essere riscritto
echo $a > 10 ? 'maggiore' : 'minore o uguale';
```

?:

- Se vogliamo eseguire solo la condizione falsa (else) possiamo utilizzare l'istruzione ?:

```php
<?php
$a = 5;
if (!($a > 10)) {
   echo 'minore o uguale';
}
echo $a > 10 ?: 'minore o uguale';
```

Null Coalesce

- Il *null coalesce* (??) è un'operatore che viene utilizzato per assegnare dei valori di default

```php
$ruolo = $utente ?? 'ospite';
// equivale a
$ruolo = isset($utente) ? $utente : 'ospite';
```

- La funzione **isset($a)** verifica che **$a** sia diverso da **NULL**
- In PHP, tutte le variabili sono inizializzate con il valore NULL

Esercizio

- Ipotizzando che una sola tra le variabili $a, $b e $c sia non nulla, assegnare a $result il valore di questa variabile
- Utilizzare l’operatore null coalesce

Operatore *spaceship*

- L'operatore spaceship (**<=>**) confronta due variabili $a e $b e restituisce i valori 1, 0, -1 a seconda se $a > $b (1), $a == $b (0), $a < $b (-1)

```php
echo 1 <=> 1; // 0
echo 3 <=> 4; // -1
echo 4 <=> 3; // 1

// confronto tra stringhe
echo "x" <=> "x"; // 0
echo "x" <=> "y"; // -1
echo "y" <=> "x"; // 1
```

Switch-case

- Se dobbiamo eseguire un'azione diversa a seconda del valore di una variabile possiamo utilizzare l'istruzione ***switch-case***

```php
switch ($day) {
    case 'Mon':
        echo 'Lunedì';
        break;
    case 'Tue':
        echo 'Martedì';
        break;
    default:
        echo 'Ne Lunedì ne Martedì';
}
```

```php
if ($day == 'Mon') {
    echo 'Lunedì';
} elseif ($day == 'Tue') {
    echo 'Martedì';
} else {
    echo 'Ne Lunedì ne Martedì';
}
```

Comparazione debole

- L'istruzione ***switch-case*** utilizza la comparazione debole (==)
- Nell'esempio precedente, cosa accade se **$day = 0** ?
- A causa della conversione automatica del PHP il primo valore del case viene convertito in intero e "Mon" diventa 0, quindi lo switch-case eseguirà la stampa di **Lunedì**.

Confronto forte (valore e tipo)

- Utilizzando il confronto forte (**===**) possiamo valutare il valore e il tipo
- E' consigliabile utilizzare questo operatore per non incorrere in strani errori di conversione

```php
if ($day === 'Mon') {
    echo 'Lunedì';
} elseif ($day === 'Tue') {
    echo 'Martedì';
} else {
    echo 'Ne Lunedì ne Martedì';
}
```

```php
switch (true) {
    case $day === 'Mon':
        echo 'Lunedì';
        break;
    case $day === 'Tue':
        echo 'Martedì';
        break;
    default:
        echo 'Ne Lunedì ne Martedì';
}
```

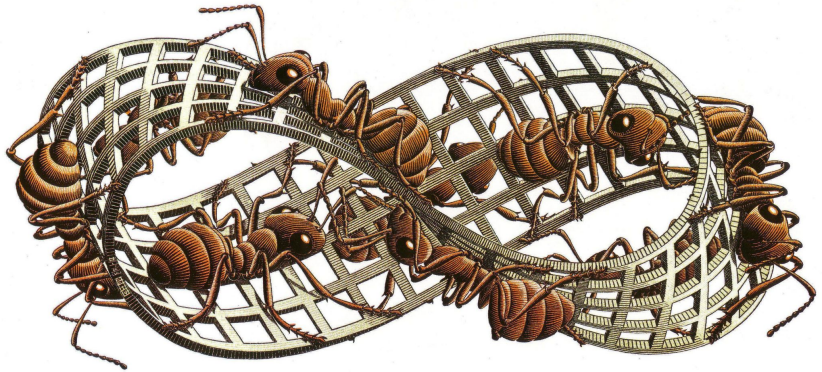
match

- A partire dal PHP 8.0.0 è stato introdotto un nuovo operatore match che utilizza il confronto forte (===)

```php
$food = 'cake';

$return_value = match ($food) {
    'apple' => 'This food is an apple',
    'bar'   => 'This food is a bar',
    'cake'  => 'This food is a cake',
};

printf("%s\n", $return_value); # This food is a cake
```



Cicli iterativi

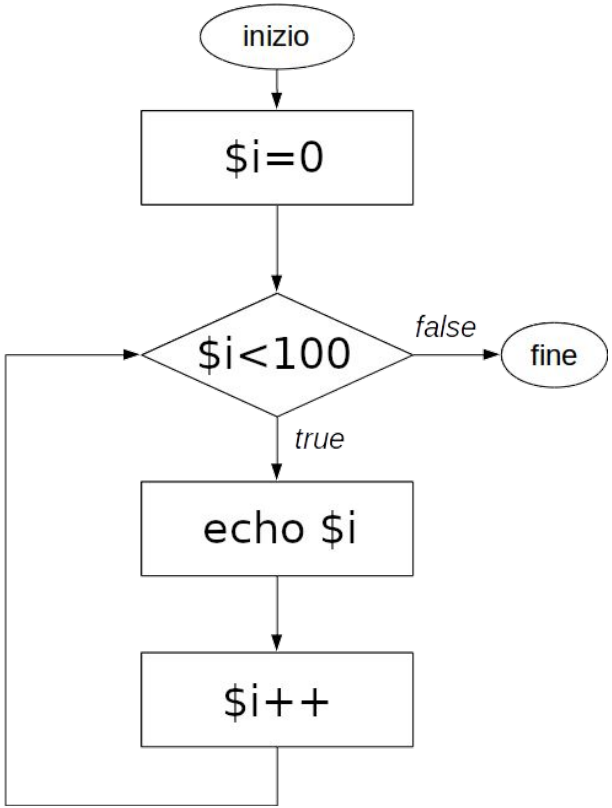
For

- Un ciclo ***for*** consente di ripetere un insieme di istruzioni un numero prestabilito di volte

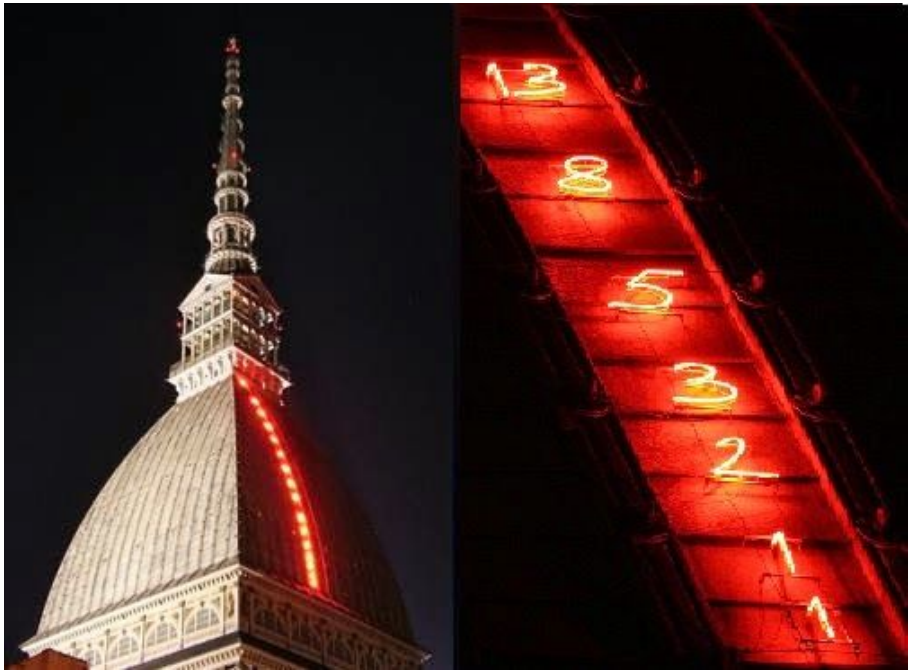
```
for ($i=0; $i<100; $i++)
   echo $i;
```

1. eseguo *$i=0*
2. se *$i<100* eseguo *echo $i*, altrimenti mi fermo
3. eseguo *$i++* e torno al punto 2

For



Il volo dei numeri



Luci d'Artista, Mario Merz "Il volo dei numeri"

Esercizio

- Scrivere un programma in PHP per stampare i primi **80 numeri** della [successione di Fibonacci]

Esercizio

- Scrivere un programma in PHP per stampare i primi **100 numeri** della successione di Fibonacci
- NOTA: il PHP gestisce numeri interi fino al valore PHP_INT_MAX, per numeri più grandi è necessario utilizzare la funzione bcadd()

Esempio di ciclo for

- E’ possibile utilizzare un ciclo for per iterare gli elementi di un array

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
for($i=0; $i<count($week); $i++) {
   printf("Il valore di week[%d] è %s \n", $i, $week[$i]);
}
```

Esercizio

- L’esempio precedente utilizza la funzione **count()** nel ciclo ***for***
- Questo utilizzo non è computazionalmente efficiente, perché?
- Riscrivere il ciclo for per renderlo più efficiente

Foreach

- Con il ciclo ***foreach*** è possibile iterare sugli elementi di un array

```php
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
foreach($week as $value) {
   printf("%s\n", $value);
}
```

Foreach su array associativi

- Il ***foreach*** può essere utilizzato anche con gli array associativi

```php
$italianDay = [
   'Mon' => 'Lunedì',
   'Tue' => 'Martedì',
   'Wed' => 'Mercoledì',
   'Thu' => 'Giovedì',
   'Fri' => 'Venerdì',
   'Sat' => 'Sabato',
   'Sun' => 'Domenica'
];
foreach($italianDay as $key => $value) {
   printf("%s è %s\n", $key, $value);
}
```

While

- Il costrutto ***while*** ripete l'esecuzione di una o più istruzione se la condizione è vera

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
$i = 0;
while ($i < 7) {
   printf("Il valore di week[%d] è %s \n", $i, $week[$i]);
   $i++;
}
```

Do-While

- Costrutto simile al while con la differenza che la condizione è verificata dopo la prima esecuzione

```
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
$i = 0;
do {
   printf("Il valore di week[%d] è %s \n", $i, $week[$i]);
   $i++;
} while ($i < 7);
```

Esercizio

- Stampare i primi 80 numeri della successione di Fibonacci in due modi:
  - utilizzando il ciclo ***while***
  - utilizzando il costrutto ***do-while***

Break/Continue

- E’ possibile interrompere l’esecuzione di un ciclo con l’istruzione ***break***
- In un ciclo, è possibile “saltare” all’iterazione successiva con l’istruzione ***continue***

Esempio di *break*

```php
$week = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'];
$tot = count($week);
for($i = 0; $i < 10; $i++) {
    if ($i >= $tot) {
        break;
    }
    printf("%d) %s\n", $i, $week[$i]);
}
```

Esempio di *continue*

```php
$rand = [];
for ($i=0; $i<100; $i++)
    $rand[]= random_int(1,100);

// stampo i valori dispari di un array
foreach ($rand as $value) {
    if ($value % 2 === 0) {
        continue;
    }
    printf("%d\n", $value);
}
```

Esercizio

- Riscrivere l’esercizio del **codice fiscale** utilizzando i costrutti iterativi visti in questa lezione
- L’esercizio va consegnato entro la prossima lezione!

Grazie dell’attenzione!

Per informazioni:
enrico.zimuel@its-ictpiemonte.it

POR Piemonte
FSE 2014-2020