Sequenza, selezione, iterazione

Solitamente un linguaggio di programmazione, per essere definito completo, deve poter implementare nella propria logica tre strutture di controllo fondamentali:

- Sequenza;
- Selezione;
- Ciclo;

Ovvero che potrete risolvere qualsiasi problema con sole queste 3 strutture di controllo.

Il teorema di Bhom- Jacopini (Corrado Böhm e Giuseppe Jacopini):

“Qualunque algoritmo può essere implementato utilizzando tre sole strutture: la sequenza, la selezione e il ciclo, da applicare ricorsivamente alla composizione di istruzioni elementari“

1. Sequenza

Il concetto di sequenza è intrinseco al fatto stesso si scrivere un algoritmo.

La risoluzione di ogni problema comporta l'esecuzione, in un ordine preciso, di una serie di azioni (una ricetta).

2. Selezione

Il programma eseguirà una o più istruzioni in base alla valutazione di una condizione. (Black Mirror Bandersnatch)

I comandi di selezione sono 4:
- If
- If..else

- If..else if
- Switch

2.1 If

Esegue una o più istruzioni se e solo se la condizione valutata risulta vera.

```
$age = 28;

//Ti presto la macchina if hai 18 anni

if ($age > 18) { // Sempre Booleano
  echo "Tieni le chiavi divertiti";
}

if ($age > 30) { // Sempre Booleano
  echo "Tieni le chiavi divertiti";
}
```

2.2 If Else

Esegue un set di istruzioni se la condizione è vera, un altro set se è falsa.

```
$age = 30;

//Ti presto la macchina if hai 18 anni

if ($age > 30) { // Sempre Booleano
  echo "Tieni le chiavi divertiti"; // Vera
} else {
  echo "Sei troppo piccolo"; // Falso
}
```

2.3 If Else If

```
$age = 18;

//Ti presto la macchina if hai 18 anni

if ($age > 18) { // Sempre Booleano
  echo "Tieni le chiavi divertiti"; // Vera
} elseif ($age == 18) { // Sempre Booleano
  echo "Tieni le chiavi, guida con prudenza"; // Vera
} else {
  echo "Sei troppo piccolo"; // Falso
}
```

2.4 Switch

Il comando switch ci aiuta a strutturare meglio il codice in caso di selezioni multiple.

```
$role = readline('Inserisci il tuo ruolo: ');

switch ($role) {
  case "Admin":
    echo "Sono un Admin";
    break;
  case "Utente":
    echo "Sono un Utente";
    break;
  case "Moderatore":
    echo "Sono un Moderatore";
    break;
  default:
    echo "Ruolo non trovato";
}
```

2.5 Match

A differenza dello “switch”, l’espressione “match” restituisce un valore, esattamente come le espressioni ternarie utilizzando un confronto piu sicuro:

```
$role = readline('Inserisci il tuo ruolo: ');


$output = match ($role) {
    'admin' ⇒ 'Sono un Admin',
    'utente' ⇒ 'Sono un Utente',
    'Moderatore' ⇒ 'Sono un Moderatore',
    default ⇒  "Ruolo non trovato"
};

var_dump($output);
```

2.6 Conclusione

La differenza principale tra “match” e “switch” riguarda il tipo di confronto utilizzato:

```
$input = "5"; // stringa

switch ($input) {
    case 5:
        echo "Sono lo Swicth case: Valore intero 5\n";
        break;
    case "5":
        echo "Sono lo Swicth case:  Stringa 5\n";
        break;
    default:
        echo "Sono lo Swicth case: Nessuna corrispondenza\n";
}
echo match ($input) {
    5 ⇒ "Sono il Match value: Valore intero 5\n",
    "5" ⇒ "Sono il Match value: Stringa 5\n",
```

```
    default ⇒ "Sono il Match value: Nessuna corrispondenza\n",
};
```

- **Switch** utilizza un confronto di uguaglianza debole (==), che può fare il cosiddetto **type juggling** (adattamento tra tipi di dati diversi, come confrontare stringhe e numeri).
- **Match** utilizza invece un confronto di identità (===), che richiede che i valori confrontati abbiano lo stesso tipo e valore.

In sostanza, l'espressione "match" è più rigorosa nel confronto dei valori e restituisce sempre un valore, rendendola utile quando si vuole valutare e restituire direttamente un risultato in base al confronto di identità tra valori.

3 - Cicli di Iterazione

I comandi di iterazione o cicli consentono ripetere una porzione di codice più volte.

- For
- Foreach
- While
- Do While

3.1 For

Quando ho un numero preciso, Il ciclo for è regolato da 4 elementi:

- contatore
- valore iniziale
- incremento o decremento del contatore
- condizione

```
<?php
//Esercizio 1
// Stampare tutti i numeri da 1 a 10
//1,2,3,4,5,6,7,8,9,10
```

```
$max = readline('Fino a che numero conto? :');

for ($i = 1; $i <= $max; $i++) {
  echo $i;
}

//Esercizio2
$students = [
  [
    'name' ⇒ 'Marco',
    'age' ⇒ 21,
  ],
  [
    'name' ⇒ 'Maria',
    'age' ⇒ 22,
  ],
  [
    'name' ⇒ 'Jack',
    'age' ⇒ 24,
  ],
];

for ($i = 0; $i < count($students); $i++) {
  //echo "$students[$i]['name'] e ha $students[$i]['age']";
  echo $students[$i]['name'] . " e ha " . $students[$i]['age'] . "\n";
}
```

3.1 Foreach

Quando non ho un numero preciso di elementi. Il ciclo foreach viene utilizzato per manipolare gli array. Grazie ad esso possiamo effettuare un'operazione su ogni elemento di un
array.

```
foreach ($array as $chiave ⇒ $valore_temporaneo) {
  # code...
```

```
}

$students = [
  [
    'name' ⇒ 'Marco',
    'age' ⇒ 21,
  ],
  [
    'name' ⇒ 'Maria',
    'age' ⇒ 22,
  ],
  [
    'name' ⇒ 'Jack',
    'age' ⇒ 24,
  ],
];

foreach ($students  as $student) {
  echo $student['name'];
}

//Esempio Indice

$students = [
  [
    'name' ⇒ 'Marco',
    'age' ⇒ 21,
  ],
  [
    'name' ⇒ 'Maria',
    'age' ⇒ 22,
  ],
  [
    'name' ⇒ 'Jack',
    'age' ⇒ 24,
  ],
];
```

```
foreach ($students  as $key ⇒ $student) {
  if ($key == 1) {
    echo $student['name'] . "!!!!!!!!\n";
  } else {
    echo $student['name'] . "\n";
  }
}
```

3.2 Differenza Foreach / For

La differenza tra i due costrutti è il ciclo foreach farà sempre un giro in meno rispetto al for.

3.3 While - Precondizionale

Anche il ciclo While è regolato da un contatore e termina la sua esecuzione quando la consizione diventa falsa.

```
while ($condizione) {
  # code...
}

$i = 1;


while ($i <= 10) { //Fintanto che la condizone è vera, ripetilo all'infinito
  echo "$i\n";

  $i++;
}
```

3.4 Do While - PostCondizionale

Anche il ciclo While è regolato da un contatore e termina la sua esecuzione quando la consizione diventa falsa.Il ciclo do...while è simile al ciclo while.

L'unica differenza è che il primo ciclo viene sempre eseguito.

La condizione verrà valutata dopo la prima esecuzione e, se verrà valutata vera, eseguirà il ciclo. In caso contrario, continuerà l'esecuzione senza

eseguire il ciclo.

```
$i = 1;
do {
  echo "$i\n";
  $i++;

} while ($i <= 10)
```

3.5 Break / Continue

```
for ($i = 1; $i <= 10; $i++) {
  if ($i == 3) {
    continue;
  }
  if ($i == 5) {
    break;
  }
  echo "$i\n";
}
```