

# Занятие #3

# Темы

## Управляющие конструкции

# while

```
<?php
    while (логическое-выражение)
        инструкция;
?>
```

Логическое выражение проверяется перед каждым проходом, если оно **TRUE** цикл продолжается.

```
<?php
    $i = 1;
    while ($i <= 10)
    {
        echo $i++;
    }
?>
```



# do-while

- В отличие от цикла **while**, этот цикл проверяет значение выражения не до, а после каждого прохода. Таким образом, **тело цикла выполняется хотя бы один раз.**
- Выглядит оператор так:

```
<?php
    do {
        команды;
    }
    while (Логическое-выражение) ;
?>
```

```
<?php
    $i = -100;
    do {
        echo $i;
    }
    while ($i > 0);
?>
```

```
<?php
    $i = 0;
    do {
        echo ++$i;
    }
    while ($i < 10);
?>
```

# for

```
<?php
for (инициализирующие_команды; условие_цикла; команды_после_прохода)
{
    тело_цикла;
}
?>
```

Как только управление доходит до цикла, первым делом выполняются операторы, включенные **инициализирующие\_команды**.

```
<?php
for ($i = 1, $j = 1; $i <= 10; $i++, $j++)
{
    echo $j." ". "$i."<br>";
}
?>
```



# for

Затем начинается итерация. Сначала проверяется, выполняется ли **условие\_цикла** (как в конструкции while). Если да, то все в порядке, и цикл продолжается. Иначе осуществляется выход из конструкции.

# for

## Задача:

– Вывести все чётные числа от 10 до 100.

## Решение:

```
<?php
  for ($i = 10; $i <= 100; $i++)
  {
    if ($i % 2 == 0)
      echo $i."<br>";
  }
?>
```

# for

Любое из выражений может отсутствовать.

Если **условия** нет – считается, что оно всегда TRUE и получается бесконечный ЦИКЛ.

```
<?php
  for ($i = 0; ;$i++)
  {
      echo $i;
  }
?>
```

# for

```
<?php
    for ($i = 1; $i <= 10; )
    {
        echo $i++;
    }
?>
```



# for

```
<?php
  for($i = 1; $i <= 5; $i++)
  {
    for($j = 1; $j <= 5; $j++)
    {
      echo $i." ";
    }
    echo "<br/>";
  }
?>
```

```
1 1 1 1 1
2 2 2 2 2
3 3 3 3 3
4 4 4 4 4
5 5 5 5 5
```

# for

```
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
```

```
<?php
  for($i = 1; $i <= 5; $i++)
  {
    for($j = 1; $j <= 5; $j++)
    {
      echo $j." ";
    }
    echo "<br/>";
  }
?>
```

# break

Инструкция **break** прерывает выполнение текущей структуры

- for
- foreach
- while
- do-while
- switch

# break

```
<?php
    $i = 1;
    for ( ; ; )
    {
        if ($i > 10)
            break;
        echo $i;
        $i++;
    }
?>
```



# break(N)

break(N) – прерывает N вложенных структур.

```
<?php
  for($i = 0; $i < 10; $i++)
  {
    for($j = 0; $j < 10; $j++)
    {
      if($i < $j)
        break(2);
      else
        echo $i." " ".$j;
    }
  }
?>
```

# continue

`continue` – используется внутри циклических структур – пропускает оставшуюся часть текущей итерации цикла.

```
<?php
  for ($i = 0; $i < 10; $i++)
  {
    echo $i;
    continue;
    echo "Test";
  }
?>
```

# continue(N)

continue(N) – пропускает оставшуюся часть итерации на N вложенных циклов

```
<?php
  for($i = 0; $i <= 5; $i++)
  {
    for($j = 0; $j <= 5; $j++)
    {
      if($i % 2 == 0)
        continue(2);
    }
    echo $i.' ' ;
  }
?>
```

:)