#### EXPERIMENT NO. 8

#### Program based on PHP variables, Expression, arrays, control structure

**Title :** Program based on PHP variables, Expression, arrays, control structure**.**

**Aim :** To implement Program based on PHP variables, Expression, arrays, control structure.

**Theory:**

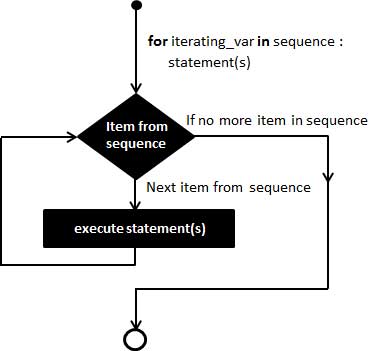
Loops in PHP are used to execute the same block of code a specified number of times. PHP supports following four loop types.

* **for** − loops through a block of code a specified number of times.
* **while** − loops through a block of code if and as long as a specified condition is true.
* **do...while** − loops through a block of code once, and then repeats the loop as long as a special condition is true.
* **foreach** − loops through a block of code for each element in an array.

We will discuss about **continue** and **break** keywords used to control the loops execution.

## **The for loop statement**

The for statement is used when you know how many times you want to execute a statement or a block of statements.



### **Syntax**

for (initialization; condition; increment){

code to be executed;

}

The initializer is used to set the start value for the counter of the number of loop iterations. A variable may be declared here for this purpose and it is traditional to name it $i.

## **The foreach loop statement**

The foreach statement is used to loop through arrays. For each pass the value of the current array element is assigned to $value and the array pointer is moved by one and in the next pass next element will be processed.

### **Syntax**

foreach (array as value) {

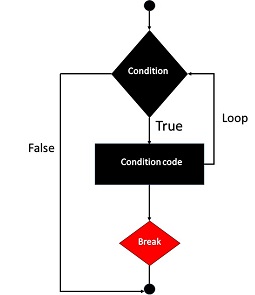
code to be executed;

}

## **The break statement**

The PHP **break** keyword is used to terminate the execution of a loop prematurely.

The **break** statement is situated inside the statement block. If gives you full control and whenever you want to exit from the loop you can come out. After coming out of a loop immediate statement to the loop will be executed.



In the following example condition test becomes true when the counter value reaches 3 and loop terminates.

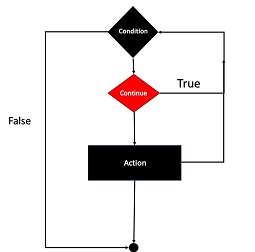
This will produce the following result −

Loop stopped at i = 3

## **The continue statement**

The PHP **continue** keyword is used to halt the current iteration of a loop but it does not terminate the loop.

Just like the **break** statement the **continue** statement is situated inside the statement block containing the code that the loop executes, preceded by a conditional test. For the pass encountering **continue** statement, rest of the loop code is skipped and next pass starts.



**Conclusion:** Thus, we have studied implementation of thread synchronization.