Συναρτήσεις

Όπως όλες οι σύγχρονες δομημένες γλώσσες προγραμματισμού έτσι και η PHP είναι μια γλώσσα που υποστηρίζει τον τμηματικό προγραμματισμό. Αυτό σημαίνει ότι επιτρέπει τη κατάτμηση των προγραμμάτων σε κομμάτια . Τα κομμάτια αυτά ονομάζονται **υποπρογράμματα** και χρησιμοποιούνται από το βασικό μέρος του προγράμματος, που ονομάζεται **κυρίως πρόγραμμα**. Ένα τέτοιο υποπρόγραμμα χρησιμοποιείται όταν κάποιο άλλο μέρος το καλεί. Κατά τη κλήση ενός υποπρογράμματος η εκτέλεση μεταφέρεται εκεί και επιστρέφει πάλι στο σημείο της κλήσης όταν ο κώδικάς του ολοκληρωθεί. Η **PHP** υποστηρίζει μόνο συναρτήσεις. Παρόλα αυτά οι συναρτήσεις της είναι πολυμορφικές και μπορούν να επιστρέφουν στο κυρίως πρόγραμμα **καμία**, **μία** ή **περισσότερες** τιμές.

Έχουμε ήδη χρησιμοποιήσει και καλέσει συναρτήσεις στα παραδείγματά μας αλλά από ένα σύνολο έτοιμων συναρτήσεων που παρέχει η PHP. Ποτέ δηλαδή δεν ασχοληθήκαμε με το κώδικα αυτών των συναρτήσεων παρά μόνο εκμεταλλευτήκαμε τη λειτουργία τους. Τέτοιες συναρτήσεις που χρησιμοποιήσαμε είναι οι **strlen**, **isset**, **count**, **in_array**, **sort**, **rsort**, **mysqli_query** και πολλές άλλες. Σε κάθε μια από αυτές ανάλογα με τον πως έχει οριστεί οφείλουμε να δώσουμε κάποια δεδομένα όταν τις χρησιμοποιούμε. Για να δουλέψει για παράδειγμα η **strlen** που μετρά το μέγεθος ενός **string** πρέπει να της δώσουμε το **string**. Έτσι η κλήση θα μπορούσε να είναι

```
<?php
	$megethos = strlen(“Καλημέρα”);
?>
```

Οι τιμές που βάζουμε μέσα στη παρένθεση ονομάζονται **παράμετροι** και θα πρέπει να συμφωνούν σε αριθμό και τύπο με τα όσα δηλώθηκαν στον ορισμό της συνάρτησης. Ας φτιάξουμε μια δικιά μας συνάρτηση για να δούμε πως ορίζονται οι συναρτήσεις και οι παράμετροι. Έστω ότι θέλουμε μια συνάρτηση η οποία θα δέχεται 2 αριθμούς και τυπώνει το μέσο όρο τους. Θα έχει δυο παραμέτρους και δεν χρειάζεται να επιστρέφει κάποια τιμή στο κυρίως πρόγραμμα, αφού μόνη της θα τυπώνει κάτι στην έξοδο.

```
<?php
	function mo($x, $y) {
		echo ($x + $y)/2;
	}

	$n = 15;
	$m = 23;

	mo($n, $m);
?>
```

Όπως βλέπουμε στο κώδικα η κλήση της συνάρτησης δεν ενσωματώνεται κάπου αλλά μπορεί να σταθεί από μόνη της. Η τιμή του αποτελέσματος τυπώνεται με την echo στην οθόνη και μετά χάνεται.

Αν για κάποιο λόγο θέλαμε να πάρουμε το μέσο όρο πίσω στο κυρίως πρόγραμμα (πχ για να κάνουμε κάποια επιπλέον επεξεργασία) θα έπρεπε η συνάρτηση να επιστρέφει

το αποτέλεσμα και όχι να το βγάζει στην οθόνη. Θα έχει λοιπόν δυο παραμέτρους και θα επιστρέφει ένα αποτέλεσμα με την χρήση της εντολής **return**.

```
<?php
	function mo($x, $y) {
		return ($x + $y)/2;
	}

	$n = 15;
	$m = 23;

	echo mo($n, $m);
?>
```

Προσέξτε ότι η συνάρτηση, αν και βρίσκεται στην αρχή του script, δεν εκτελείται αυτομάτως. Εκτελείται μόνο αν κάποιο άλλο τμήμα του κώδικα τη χρειαστεί και την **καλέσει**, όπως και γίνεται στη τελευταία εντολή των παραπάνω scripts.

Η **return** ολοκληρώνει την εκτέλεση μιας συνάρτησης και επιστρέψει το αποτέλεσμα. Μπαίνει συνήθως στο τέλος της συνάρτησης αλλά μπορεί να εκτελεστεί και νωρίτερα. Αν η εκτέλεση της **return** γίνει νωρίτερα (στο μέσο του κώδικα) τότε η συνάρτηση θα ολοκληρωθεί σε εκείνο το σημείο και οι υπόλοιπες εντολές θα αγνοηθούν. Αυτό έχει νόημα σε κάποιες περιπτώσεις γιατί μας γλιτώνει από περιττές εντολές, εφόσον το επιθυμητό αποτέλεσμα έχει υπολογιστεί. Για παράδειγμα αν θέλουμε να εκτελέσουμε μια σειριακή αναζήτηση σε ένα πίνακα για να δούμε αν ένα συγκεκριμένο στοιχείο υπάρχει, μπορούμε να σταματήσουμε την αναζήτηση αν το βρούμε. Θα έχουμε βεβαιωθεί για την ύπαρξή του οπότε δε χρειάζεται να ψάξουμε άλλο το πίνακα.

```
<?php
	function search($pin, $k) {
		foreach ($pin as $p) {
			if ($k == $p)
				return true;
		}
		return false;
	}

	$pinakas = array(15,23,54,123,62,57,56,43,89,94,73);
	$key = 23;

	if (search($pinakas, $key))
		echo 'το στοιχείο βρέθηκε;'
	else
		echo 'το στοιχείο δεν βρέθηκε;

?>
```

Εμβέλεια μεταβλητών

Οι μεταβλητές των υποπρογραμμάτων είναι ξεχωριστές από τις μεταβλητές του κυρίως προγράμματος. Ακόμα και αν έχουν το ίδιο όνομα η PHP τις αντιμετωπίζει ως ξεχωριστές μεταβλητές.

```
<?php
	function test() {
```

```
            $n = 5;
            $m = 15;
            echo $n.''.$m.'<br>';
      }

      $n = 25;
      $m = 35;

      test();

      echo $n.''.$m;
?>
```

Για παράδειγμα στο παραπάνω κώδικα θα τυπωθούν πρώτα οι τιμές του υποπρογράμματος που θα είναι 5 και 15 και στη συνέχεια θα γίνει η εκτύπωση από το κυρίως πρόγραμμα που θα δώσει 25 και 35. Οι τιμές δηλαδή των δυο μεταβλητών του κυρίως προγράμματος δεν θα επηρεαστούν από τη συνάρτηση.

Αν επιθυμούσαμε να οι μεταβλητές εντός του υποπρογράμματος να ταυτίζονται με αυτές του κυρίως προγράμματος θα έπρεπε να χρησιμοποιήσουμε τη δήλωση **global** ή την superglobal $**GLOBALS** όπως φαίνεται στις παρακάτω συναρτήσεις.

```
<?php
      function test1() {
            global $n,$m;
            $n = 5;
            $m = 15;
            echo $n.''.$m.'<br>';
      }


      function test2() {
            $GLOBALS['n'] = 15;
            $GLOBALS['m'] = 25;
            echo $GLOBALS['n'].''. $GLOBALS['m'].'<br>';
      }

      $n = 25;
      $m = 35;

      test1();
      test2();

      echo $n.''.$m;
?>
```

Το παραπάνω παράδειγμα θα τυπώσει

```
5 15
15 25
15 25
```

Οι αρχικές τιμές των μεταβλητών θα χαθούν αφού θα έχουν τροποποιηθεί από τις συναρτήσεις. Έτσι η κλήση της test1 θα αλλάξει τις τιμές των n και m και θα τυπώσει τις νέες (5, 15). Η **test2** θα αλλάξει για δεύτερη φορά τις τιμές οι οποίες θα τυπωθούν από τη συνάρτηση και από το κυρίως πρόγραμμα.