<?php

// inputs from the user

function inputNums() {

$n = readline("Input the amount of numbers: ");

$arr = array();

for ($i = 0; $i < $n; $i++) {

$num = readline("Enter a number: ");

array\_push($arr, $num);

}

return array\_sum($arr);

}

print\_r(inputNums());

// fibonacci array

function fibonacci() {

$arr = array(1, 1);

for ($i = 1; $i < 20; $i++) {

$num = $arr[$i] + $arr[$i - 1];

array\_push($arr, $num);

}

$n = readline("Enter the number place you want to see (up to 20): ");

return $arr[$n - 1];

}

print\_r(fibonacci());

// remove element at set index

function remover($arr, $idx) {

return array\_splice($arr, $idx, 1);

}

$testArray = array('1', '2', '3', '4', '5');

print\_r(remover($testArray, 2));

// add element at set index

function adder($arr, $idx, $val) {

array\_splice($arr, $idx, 0, $val);

return $arr;

}

$testArray = array('1', '2', '3', '4', '5');

print\_r(adder($testArray, 2, '0'));

// invert the case of a string

function caseInverter($str) {

// lower case starts at 97

$splitStr = str\_split($str);

$arr = array();

foreach($splitStr as $letter) {

if (ord($letter) > 96){

array\_push($arr, strtoupper($letter));

}

else {

array\_push($arr, strtolower($letter));

}

}

return join($arr);

}

print\_r(caseInverter("ApPlEs"));

// sorting algorithm

function sorter($arr) {

$newArr = array();

$num = count($arr) - 1;

while ($arr) {

$min = min($arr);

$idx = array\_search($min, $arr);

array\_push($newArr, $min);

unset($arr[$idx]);

}

return $newArr;

}

print\_r(sorter(array(4, 7, 2, 93, 1, 5, 3, 8, 6)));

// Running Sum of 1D Array

function runningSum($nums) {

$newArr = array();

$num = 0;

foreach ($nums as $n) {

$num += $n;

array\_push($newArr, $num);

}

return $newArr;

}

// Richest Customer Wealth

function maximumWealth($accounts) {

$max = 0;

foreach ($accounts as $account) {

$sum = array\_sum($account);

if ($sum > $max) {

$max = $sum;

}

}

return $max;

}

// Shuffle the Array

function shuffle($nums, $n) {

$newArr = array();

for ($i = 0; $i < $n; $i++) {

array\_push($newArr, $nums[$i]);

array\_push($newArr, $nums[$i + $n]);

}

return $newArr;

}

// How Many Numbers Are Smaller Than the Current Number

function smallerNumbersThanCurrent($nums) {

$newArr = array();

foreach ($nums as $num) {

$count = 0;

foreach ($nums as $n) {

if ($n < $num) {

$count += 1;

}

}

array\_push($newArr, $count);

}

return $newArr;

}

// Count Items Matching a Rule

function countMatches($items, $ruleKey, $ruleValue) {

$count = 0;

if ($ruleKey == "type") {

foreach ($items as $item) {

if ($item[0] == $ruleValue) {

$count++;

}

}

}

else if ($ruleKey == "color") {

foreach ($items as $item) {

if ($item[1] == $ruleValue) {

$count++;

}

}

}

else if ($ruleKey == "name") {

foreach ($items as $item) {

if ($item[2] == $ruleValue) {

$count++;

}

}

}

return $count;

}

// Defanging an IP Address

function defangIPaddr($address) {

return str\_replace(".", "[.]", $address);

}

// Split a String in Balanced Strings

function balancedStringSplit($s) {

$arr = str\_split($s);

$count = 0;

$num = 0;

foreach ($arr as $letter) {

if ($letter == 'R') {

$num += 1;

}

else {

$num -= 1;

}

if ($num == 0) {

$count += 1;

}

}

return $count;

}

// Maximum Nesting Depth of the Parentheses

function maxDepth($s) {

$count = 0;

$max = 0;

$sploded = str\_split($s);

foreach ($sploded as $letter) {

if ($letter === '(') {

$count++;

}

else if ($letter === ')') {

$count--;

}

if ($count > $max) {

$max = $count;

}

}

return $max;

}

// Determine if String Halves are Alike

function halvesAreAlike($s) {

$half = strlen($s) / 2;

$first = substr($s, 0, $half);

$second = substr($s, $half);

$vowels = array('a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U');

$one = 0;

$two = 0;

foreach ($vowels as $vowel) {

$one += substr\_count($first, $vowel);

$two += substr\_count($second, $vowel);

}

if ($one === $two) {

return true;

}

else return false;

}

// Determine Color of a Chessboard Square

function squareIsWhite($coordinates) {

if (ord($coordinates[0]) % 2 === 1 && intval($coordinates[1]) % 2 === 1) {

return false;

}

else if (ord($coordinates[0]) % 2 === 0 && intval($coordinates[1]) % 2 === 0) {

return false;

}

else return true;

}

// Reverse Words in a String III

function reverseWords($s) {

$arr = explode(" ", $s);

$newArr = array();

foreach ($arr as $word) {

array\_push($newArr, strrev($word));

}

return implode(" ", $newArr);

}

// Robot Return Origin

function judgeCircle($moves) {

$x = 0;

$y = 0;

$x += substr\_count($moves, "R");

$x -= substr\_count($moves, "L");

$y += substr\_count($moves, "U");

$y -= substr\_count($moves, "D");

if ($x === 0 && $y === 0) {

return true;

}

else return false;

}

// Jewels and Stones

function numJewelsInStones($jewels, $stones) {

$count = 0;

for ($i = 0; $i < strlen($jewels); $i++) {

$count += substr\_count($stones, $jewels[$i]);

}

return $count;

}

?>