**Instructions:** Select at least 10 problems from the list below and copy their exact function names. Write your solutions below each function definition. Save everything in one single PHP file named [yourfirstname].php.

```
/**
* 1. Count Values In Range
*
* Problem:
* Given an indexed array of numbers and inclusive bounds [min, max], return
how many elements are within that range.
*
* Examples:
*   numbers=[-2, 0, 3, 7, 10],   min=0, max=7   → 3      (0, 3, 7)
*   numbers=[1, 2, 3],           min=4, max=6   → 0
*   numbers=[],                  min=0, max=10  → 0
*
* @param float[] $numbers
* @param float $min
* @param float $max
* @return int
*/
function count_in_range(array $numbers, float $min, float $max): int
```

```
/**
* 2. Count Occurrences
*
* Problem:
* Given an indexed array and a target value, return how many elements are
strictly equal (===) to the target.
*
* Examples:
*   items=[2, 3, 2, "2"], target=2         → 2
*   items=["a","b","a","a"], target="a"    → 3
*   items=[], target=5                     → 0
*
* @param array $items
* @param mixed $target
* @return int
*/

function count_occurrences(array $items, $target): int
```

```
/**
* 3. Sum of Positives
*
* Problem:
* Given an indexed array of numbers, return the sum of elements greater
than zero.
*
* Examples:
*   [-3, 5, 2, -1]       → 7
*   [0, 0, 0]            → 0
*   [10.5, -2.5, 1.0]  → 11.5
*
* @param float[] $numbers
* @return float
*/
function sum_positives(array $numbers): float
```

```
/**
* 4. Sum of Negatives
*
* Problem:
* Given an indexed array of numbers, return the sum of absolute values for
elements less than zero.
*
* Examples:
*   [-3, 5, -2]          → 5
*   [1, 2, 3]            → 0
*   [-1.5, -0.5, 2.0]  → 2.0
*
* @param float[] $numbers
* @return float
*/
function sum_negatives(array $numbers): float
```

```php
/**
* 5. Distinct Count
*
* Problem:
* Given an indexed array, return how many distinct values it contains
(strict uniqueness).
*
* Examples:
*   [1, 1, "1", 2]            → 3
*   ["x", "x", "x"]           → 1
*   []                        → 0
*
* @param array $items
* @return int
*/
function count_distinct(array $items): int
```

```php
/**
* 6. Has Duplicates
*
* Problem:
* Given an indexed array, return true if any value appears more than once,
else false (strict comparison).
*
* Examples:
*   [3, 7, 3]         → true
*   ["a", "b", "c"]   → false
*   []                → false
*
* @param array $items
* @return bool
*/
function has_duplicates(array $items): bool
```

```php
/**
 * 7. Index of Maximum
 *
 * Problem:
 * Given a non-empty indexed array of numbers, return the index of the
first occurrence of the maximum value.
 *
 * Examples:
 *   [5, 9, 9, 2]         → 1
 *   [-1, -5, -2]         → 0
 *   [3.2, 4.0, 2.8, 5.5] → 3
 *
 * @param float[] $numbers
 * @return int
 */
function index_of_max(array $numbers): int
```

```php
/**
* 8. Index of Minimum
*
* Problem:
* Given a non-empty indexed array of numbers, return the index of the first
occurrence of the minimum value.
*
* Examples:
*   [4.2, 1.1, 1.1, 3.0]  → 1
*   [0, -1, -1, -2]       → 3
*   [8]                   → 0
*
* @param float[] $numbers
* @return int
*/
function index_of_min(array $numbers): int
```

```php
/**
* 9. Mode (Most Frequent Value)
*
* Problem:
* Given an indexed array, return the value that appears most frequently; if
tied, return the first value to reach that highest frequency.
*
* Examples:
*   ["M", "S", "M", "L", "S", "M"]  → "M"
*   [1, 2, 2, 1]                    → 1
*   []                              → null
*
* @param array $items
* @return mixed|null
*/
function mode_value(array $items)
```

```php
/**
* 10. Unique After Merge Count
*
* Problem:
* Given two indexed arrays, return how many distinct values exist in their
union (strict uniqueness).
*
* Examples:
*   [1,2,2], [2,3]         → 3
*   ["a"], ["a","b","a"]   → 2
*   [], []                 → 0
*
* @param array $a
* @param array $b
* @return int
*/
function merged_unique_count(array $a, array $b): int
```

```php
/**
* 11. Is Sorted Ascending
*
* Problem:
* Given an indexed array of numbers, return true if it is non-decreasing,
else false.
*
* Examples:
*   [1, 2, 2, 5]      → true
*   [3, 2, 2]         → false
*   []                → true
*
* @param float[] $numbers
* @return bool
*/
function is_sorted_ascending(array $numbers): bool
```

```php
/**
* 12. Max Absolute Value
*
* Problem:
* Given an indexed array of numbers (non-empty), return the value with the
greatest absolute magnitude; if tied, return the first encountered.
*
* Examples:
*   [-7, 5, 6]        → -7
*   [3, -4, 4]        → -4
*   [0]               → 0
*
* @param float[] $numbers
* @return float
*/
function max_absolute(array $numbers): float
```

```php
/**
* 13. Range Span
*
* Problem:
* Given a non-empty indexed array of numbers, return max(numbers) -
min(numbers).
*
* Examples:
*   [4, 10, 2]      → 8
*   [5]             → 0
*   [-3, 7, -1, 6]  → 10
*
* @param float[] $numbers
* @return float
*/
function range_span(array $numbers): float
```

```php
/**
* 14. Count Greater Than Threshold
*
* Problem:
* Given an indexed array of numbers and a threshold t, return how many
elements are strictly greater than t.
*
* Examples:
*   [1, 5, 7, 2], t=4      → 2
*   [4, 4, 4], t=4         → 0
*   [], t=10               → 0
*
* @param float[] $numbers
* @param float $t
* @return int
*/
function count_greater_than(array $numbers, float $t): int
```

```php
/**
 * 15. Count Strings Longer Than
 *
 * Problem:
 * Given an indexed array of strings and an integer n, return how many
strings have length greater than n.
 *
 * Examples:
 *   ["cat", "house", "a"], n=3      → 1      ("house")
 *   ["abc", "abcd"], n=2            → 2
 *   [], n=5                         → 0
 *
 * @param string[] $strings
 * @param int $n
 * @return int
 */
function count_strings_longer_than(array $strings, int $n): int
```

```php
/**
 * 16. Find Shortest String
 *
 * Problem:
 * Given a non-empty indexed array of strings, return the string with the
smallest length. If tied, return the first.
 *
 * Examples:
 *   ["cat", "a", "dog"]       → "a"
 *   ["one", "two", "six"]     → "one"   (all length 3, first returned)
 *   ["longword"]              → "longword"
 *
 * @param string[] $strings
 * @return string
 */
function shortest_string(array $strings): string
```

```php
/**
 * 17. Concatenate Strings
 *
 * Problem:
 * Given an indexed array of strings, return them joined together into one
string with no separator.
 *
 * Examples:
 *   ["a", "b", "c"]      → "abc"
 *   ["hello", "world"]   → "helloworld"
 *   []                   → ""
 *
 * @param string[] $strings
 * @return string
 */
function concatenate_strings(array $strings): string
```

```php
/**
 * 18. Count Strings Starting With
 *
 * Problem:
 * Given an indexed array of strings and a single-character prefix, return
how many strings start with that character (case-sensitive).
 *
 * Examples:
 *   ["apple", "ant", "banana"], prefix="a"   → 2
 *   ["dog", "cat", "cow"], prefix="c"        → 2
 *   [], prefix="x"                           → 0
 *
 * @param string[] $strings
 * @param string $prefix
 * @return int
 */
function count_starting_with(array $strings, string $prefix): int
```

```php
/**
 * 19. Total Characters in All Strings
 *
 * Problem:
 * Given an indexed array of strings, return the total number of characters
across all strings combined.
 *
 * Examples:
 *   ["a", "bb", "ccc"]     → 6
 *   ["hello", "world"]     → 10
 *   []                     → 0
 *
 * @param string[] $strings
 * @return int
 */
function total_characters(array $strings): int
```

```php
/**
 * 20. Repeat String
 *
 * Problem:
 * Given a string s and an integer n, return the string repeated n times.
 *
 * Examples:
 *   s="ha", n=3   → "hahaha"
 *   s="x",  n=5   → "xxxxx"
 *   s="a",  n=0   → ""
 *
 * @param string $s
 * @param int $n
 * @return string
 */
function repeat_string(string $s, int $n): string
```

```php
/**
 * 21. Is Vowel
 *
 * Problem:
 * Given a single-character string ch, return true if it is a vowel (a, e,
i, o, u, case-insensitive), else false.
 *
 * Examples:
 *   ch="a"   → true
 *   ch="E"   → true
 *   ch="b"   → false
 *
 * @param string $ch
 * @return bool
 */
function is_vowel(string $ch): bool
```

```php
/**
 * 22. Character At Position
 *
 * Problem:
 * Given a string s and an integer index i, return the character at
position i (0-based). If i is out of range, return an empty string.
 *
 * Examples:
 *   s="hello", i=1   → "e"
 *   s="world", i=0   → "w"
 *   s="cat",   i=5   → ""
 *
 * @param string $s
 * @param int $i
 * @return string
 */
function char_at(string $s, int $i): string
```

```php
/**
 * 23. Absolute Difference
 *
 * Problem:
 * Given two numbers a and b, return their absolute difference.
 *
 * Examples:
 *   a=5,  b=3   → 2
 *   a=3,  b=5   → 2
 *   a=-4, b=1   → 5
 *
 * @param float $a
 * @param float $b
 * @return float
 */
function absolute_difference(float $a, float $b): float
```

```php
/**
 * 24. Is Uppercase
 *
 * Problem:
 * Given a single-character string ch, return true if it is an uppercase
letter (A–Z), else false.
 *
 * Examples:
 *   ch="A"   → true
 *   ch="z"   → false
 *   ch="7"   → false
 *
 * @param string $ch
 * @return bool
 */
function is_uppercase(string $ch): bool
```

```php
/**
 * 25. Sum Numbers Where String is Long Enough
 *
 * Problem:
 * Given an indexed array of strings and an indexed array of numbers of the same length,
 * return the sum of the numbers where the corresponding string's length is at least minLength.
 *
 * Examples:
 *   strings=["cat", "house", "a"], numbers=[5, 10, 2], minLength=3
 *     → 15   (5 from "cat", 10 from "house")
 *   strings=["hi", "okay", "sun", "moon"], numbers=[4, 5, 3, 7], minLength=4
 *     → 12   (5 from "okay", 7 from "moon")
 *   strings=[], numbers=[], minLength=1
 *     → 0
 *
 * @param string[] $strings
 * @param float[] $numbers
 * @param int $minLength
 * @return float
 */
function sum_numbers_where_string_long_enough(array $strings, array $numbers, int $minLength): float
```