**ELITE EXAM**

**PHP**

**“SHORTEST WORD”**

You have given a string of words, return the length of the shortest word(s).

String will never be empty, and you do not need to account for different data types.

TEST CASES:

TEST CASE: “TRUE FRIENDS ARE ME AND YOU”

OUTPUT: **2** – BECAUSE THE SHORTEST WORD IS “ME”

TEST CASE: “I AM THE LEGENDARY VILLAIN”

OUTPUT: **1** – BECAUSE THE SHORTEST WORD IS “I”

ANSWER:

<?php

function findShortestWordLength(string $str): int

{

    return min(array\_map('strlen', explode(' ', $str)));

}

echo findShortestWordLength("TRUE FRIENDS ARE ME AND YOU");

echo "\n";

echo findShortestWordLength("I AM THE LEGENDARY VILLAIN");

?>

**“COUNT THE ISLANDS”**

Implement an algorithm which analyzes a two-color image and determines how many isolated areas of a single color the image contains.

Islands

An "island" is a set of adjacent pixels of one color (1) which is surrounded by pixels of a different color (0). Pixels are considered adjacent if their coordinates differ by no more than 1 on the X or Y axis.

Below you can see an example with 2 islands:

* on the left in the form of a matrix of 1's and 0's
* on the right in an equivalent stringified form using "X" and "~" characters for better readability

**SAMPLE:**Text

Description automatically generated

**TEST CASE**

**GIVEN a 4x4 matrix.**

**[ 1,1,1,1 ]  
[ 0,1,1,0 ]**

**[ 0,1,0,1 ]  
[ 1,1,0,0 ]**

**Output: “XXXX”  
 “~XX~”**

**“~X~X”**

**“XX~~”**

**ANSWER:**

<?php

function findIslands($map) {

    $rows = count($map);

    $cols = count($map[0]);

    $total = 0;

    for ($x = 0; $x < $rows; $x++) {

        for ($y = 0; $y < $cols; $y++) {

            if ($map[$x][$y] == 1) {

                checkIsland($map, $x, $y, $rows, $cols);

                $total++;

            }

        }

    }

    return $total;

}

function checkIsland(&$map, $x, $y, $rows, $cols) {

    if ($x < 0 || $y < 0 || $x >= $rows || $y >= $cols || $map[$x][$y] == 0) {

        return;

    }

    $map[$x][$y] = 0;

    checkIsland($map, $x + 1, $y, $rows, $cols);

    checkIsland($map, $x - 1, $y, $rows, $cols);

    checkIsland($map, $x, $y + 1, $rows, $cols);

    checkIsland($map, $x, $y - 1, $rows, $cols);

}

function showGrid($map) {

    $rows = count($map);

    $cols = count($map[0]);

    $result = [];

    for ($x = 0; $x < $rows; $x++) {

        $line = "";

        for ($y = 0; $y < $cols; $y++) {

            if ($map[$x][$y] == 1) {

                $line .= "X";

            } else {

                $line .= "-";

            }

        }

        $result[] = $line;

    }

    return $result;

}

$matrix = [

    [1, 1, 1, 1],

    [0, 1, 0, 0],

    [0, 1, 0, 1],

    [1, 1, 0, 0]

];

$copyMatrix = $matrix;

$lines = showGrid($matrix);

foreach ($lines as $l) {

    echo "\"$l\"\n";

}

echo "Islands: " . findIslands($copyMatrix) . "\n";

**“WORD SEARCH”**

You are given a word target and a list of sorted (by length(increasing), number of upper-case letters(decreasing), natural order) unique words; words which always contain target, your task is to find the index (0 based) of target in words, which would always be in the list.

MAX ARRAY LENGTH: 1000;

TEST CASES:

[“I”,”TWO”,”FORTY”,”THREE”,’JEN’,”TWO”,”tWo”,”Two”];

TARGET **= “TWO”**

OUTPUT **= INDEX 1 and INDEX 5 // [1,5]**

**ANSWER:**

<?php

$the\_list\_of\_words = ["I", "TWO", "FORTY", "THREE", 'JEN', "TWO", "tWo", "Two"];

$the\_word\_to\_find = "TWO";

$found\_at\_indices = [];

for ($i = 0; $i < count($the\_list\_of\_words); $i++) {

    $current\_word = $the\_list\_of\_words[$i];

    if ($current\_word == $the\_word\_to\_find) {

        $found\_at\_indices[] = $i;

    }

}

$output\_text = "[";

for ($i = 0; $i < count($found\_at\_indices); $i++) {

    $output\_text = $output\_text . $found\_at\_indices[$i];

    if ($i < count($found\_at\_indices) - 1) {

        $output\_text = $output\_text . ",";

    }

}

$output\_text = $output\_text . "]";

echo $output\_text;

?>

**MYSQL:**

**NOTE: EXCEL / CSV FILE WILL BE PROVIDED**

**Based on the data given:**

**1. Display total number of albums sold per artist**

**2. Display combined album sales per artist**

**3. Display the top 1 artist who sold most combined album sales**

**4. Display the top 10 albums per year based on their number of sales**

**5. Display list of albums based on the searched artist**

**ANSWER:**

---MYSQL QUERIES FOR ALBUM SALES DATABASE---

-- 1. Displays the total number of albums sold by each artist.

--    It counts the number of album records for each artist and sorts the

--    result by the artist's name.

SELECT

    artist\_name,

    COUNT(album\_name) AS total\_albums

FROM

    album\_sales

GROUP BY

    artist\_name

ORDER BY

    artist\_name ASC;

-- 2. Displays the combined (total) album sales for each artist.

--    It calculates the sum of the `sales` column for all albums belonging

--    to an artist and orders the results to show the highest-selling artists first.

SELECT

    artist\_name,

    SUM(sales) AS combined\_sales

FROM

    album\_sales

GROUP BY

    artist\_name

ORDER BY

    combined\_sales DESC;

-- 3. Displays the top 1 artist who had the most combined album sales.

--    This builds on the previous query by using `LIMIT 1` to return only

--    the single top result after ordering by combined sales.

SELECT

    artist\_name,

    SUM(sales) AS combined\_sales

FROM

    album\_sales

GROUP BY

    artist\_name

ORDER BY

    combined\_sales DESC

LIMIT 1;

-- 4. Displays the top 10 best-selling albums for each year.

--    This query uses a Common Table Expression (WITH clause) to first assign a rank

--    to each album based on its sales within a given year. It then selects

--    only the albums with a rank from 1 to 10.

WITH RankedAlbums AS (

    SELECT

        artist\_name,

        album\_name,

        sales,

        YEAR(date\_released) AS release\_year,

        ROW\_NUMBER() OVER(PARTITION BY YEAR(date\_released) ORDER BY sales DESC) as sales\_rank

    FROM

        album\_sales

)

SELECT

    release\_year,

    artist\_name,

    album\_name,

    sales

FROM

    RankedAlbums

WHERE

    sales\_rank <= 10

ORDER BY

    release\_year DESC,

    sales\_rank ASC;

-- 5. Displays a list of all albums for a specific searched artist.

--    This query uses a simple `WHERE` clause to filter the results and show all

--    album details for a single artist. You can change 'IVE' to any other

--    artist's name to search for their albums.

SELECT

    artist\_name,

    album\_name,

    sales,

    date\_released

FROM

    album\_sales

WHERE

    artist\_name = 'IVE';

**LARAVEL**

**NOTE: SAME CSV FILE FROM MYSQL EXAM**

**MODELS**

**ARTISTS**

**FIELDS**

**CODE**

**NAME**

**FUNCTIONS**

**- read the full details of created artist**

**- update the full details of created artist**

**- delete the details of created artist**

**ALBUMS**

**FIELDS**

**YEAR**

**NAME**

**SALES**

**FUNCTIONS**

**- read the full details of album**

**- update**

**- delete**

**- add a picture of album cover**

**Migration and DB Relation**

**- use the csv file to populate the artist and albums and use "faker" for the other details that is not in the csv file**

**Login**

**- admin user can perform logout functionality**

**- credentials must be username and password**

**Dashboard**

**- Display total number of albums sold per artist**

**- Display combined album sales per artist**

**- Display the top 1 artist who sold most combined album sales**

**- Display list of albums based on the searched artist**

**Bonus:**

**- Translate each sql scripts to a Laravel eloquent format**

**- Provide routes for each scripts for each scenarios**

**- API: Output must be in json format**

**- Provide an authentication function that will grant a bearer token that can be used to access the said routes**

**Create API route under route.php in Laravel using the created controller on CRUD basic in above activity.**

**• Endpoint Can perform GET, POST, PUT/PATCH, DELETE for**

**• Artist**

**• Album**

**//THE WEBSITE WAS UPLOADED INTO REPO.   
Link:** [archieeeeeeeeeee/ELITE-EXAM-ANSWERS](https://github.com/archieeeeeeeeeee/ELITE-EXAM-ANSWERS)