УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по практическому заданию №1

по предмету «Метрология, стандартизация и сертификация (в информационных технологиях»

Тема работы: Метрики Холстеда

Выполнили:

Кисель П.А.,

Сологуб Е.М.

Гр. 351003

Проверила:

Болтак С. В.

Минск 2025

1. **Код php:**

<?php

class UserManager {

private $db;

private $users = [] ;

public function \_\_construct ( $database ) {

$this->db = $database ;

$this->loadUsers() ;

}

public function loadUsers() {

while ( $row = $result->fetch\_assoc() ) {

$this->users[ $row['id']] = $row;

}

}

public function getUserById($userId) {

return $this->users[ $userId] ?? null;

}

public function addUser( $name, $email, $role = 'user') {

if ( !filter\_var( $email, FILTER\_VALIDATE\_EMAIL) || strlen( $name) < 2) {

throw new Exception("Invalid input") ;

}

$stmt = $this->db->prepare("INSERT INTO users (name, email, role) VALUES ( ? , ? , ? )");

$stmt->bind\_param("sss", $name, $email, $role) ;

if ( $stmt->execute() ) {

$newUserId = $this->db->insert\_id;

$this->users[ $newUserId] = ['name' => $name, 'email' => $email, 'role' => $role] ;

return $newUserId;

}

return false;

}

public function updateUser( $userId, $data) {

if ( ! isset( $this->users[ $userId] ) ) {

throw new Exception("User not found") ;

}

$allowed = ['name', 'email', 'role' ] ;

$updateData = array\_intersect\_key( $data, array\_flip( $allowed) ) ;

if (empty( $updateData) ) return false;

$setClause = implode(", ", array\_map(fn( $f) => " $f = ? ", array\_keys( $updateData ) ) ) ;

$params = array\_merge(array\_values ( $updateData) , [ $userId] ) ;

$types = str\_repeat("s", count( $updateData) ) . "i";

$stmt = $this->db->prepare("UPDATE users SET $setClause WHERE id = ?") ;

$stmt->bind\_param( $types, ... $params) ;

if ( $stmt->execute( ) ) {

$this->users[ $userId] = array\_merge( $this->users[ $userId] , $updateData) ;

return true ;

}

return false ;

}

public function deleteUser( $userId) {

if ( ! isset( $this->users[ $userId] ) ) return false ;

$stmt = $this->db->prepare("UPDATE users SET is\_active = 0 WHERE id = ?") ;

$stmt->bind\_param("i", $userId) ;

if ( $stmt->execute() ) {

unset( $this->users[ $userId] ) ;

return true;

}

return false;

}

public function getUsersByRole( $role) {

return array\_filter( $this->users, fn( $user) => $user['role'] === $role) ;

}

}

$db = new mysqli("localhost", "user", "pass", "test\_db") ;

$manager = new UserManager( $db) ;

try {

$id = $manager->addUser("John", "john@mail.com", "admin") ;

echo "User $id added\n" ;

$user = $manager->getUserById ( $id ) ;

if ( $user) echo "Found: {$user['name']}\n";

$manager->updateUser( $id, ['name' => 'John Updated'] ) ;

echo "User updated\n"

$admins = $manager->getUsersByRole('admin') ;

echo "Admins: " . count( $admins) . "\n";

} catch (Exception $e) {

echo "Error: " . $e->getMessage() . "\n";

}

?>

1. **Ручной расчёт:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***j*** | **Оператор** | ***f1j*** | ***i*** | **Операнд** | ***zf2i*** |
| 1 | class | 1 | 1 | $db | 1 |
| 2 | {} | 16 | 2 | $users | 1 |
| 3 | private | 2 | 3 | $database | 2 |
| 4 | = | 15 | 4 | $userId | 11 |
| 5 | public | 7 | 5 | $name | 4 |
| 6 | () | 26 | 6 | $email | 4 |
| 7 | -> | 18 | 7 | $role | 5 |
| 8 | ; | 35 | 8 | $data | 2 |
| 9 | if | 8 | 9 | $allowed | 2 |
| 10 | ! | 3 | 10 | $updateData | 6 |
| 11 | throw | 2 | 11 | $setClause | 2 |
| 12 | new | 4 | 12 | $params | 2 |
| 13 | return | 10 | 13 | $types | 2 |
| 14 | ?? | 1 | 14 | $stmt | 9 |
| 15 | , | 19 | 15 | $row | 1 |
| 16 | empty | 1 | 16 | $result | 2 |
| 17 | . | 3 | 17 | $newUserId | 3 |
| 18 | === | 1 | 18 | $user | 5 |
| 19 | implode | 1 | 19 | $e | 2 |
| 20 | unset | 1 | 20 | $manager | 5 |
| 21 | … | 1 | 21 | $id | 4 |
| 22 | [] | 9 | 22 | $admins | 2 |
| 23 | fn | 2 | 23 | $f | 2 |
| 24 | => | 6 | 24 | 1 | 2 |
| 25 | isset | 2 | 25 | 2 | 1 |
| 26 | $ | 58 | 26 | 0 | 1 |
| 27 | \* | 1 | 27 | FILTER\_VALIDATE\_EMAIL | 1 |
| 28 | SELECT | 1 | 28 | Exception | 3 |
| 29 | while | 1 | 29 | null | 1 |
| 30 | FROM | 1 | 30 | SELECT \* FROM users WHERE is\_active = 1 | 1 |
| 31 | WHERE | 3 | 31 | name | 6 |
| 32 | INSERT | 1 | 32 | email | 3 |
| 33 | INTO | 1 | 33 | role | 4 |
| 34 | VALUES | 1 | 34 | user | 2 |
| 35 | function | 7 | 35 | admin | 2 |
| 36 | try | 1 | 36 | Invalid input | 1 |
| 37 | catch | 1 | 37 | INSERT INTO users (name, email, role) VALUES (?, ?, ?) | 1 |
| 38 | echo | 5 | 38 | sss | 1 |
| 39 | strlen | 1 | 39 | User not found | 1 |
| 40 | UPDATE | 2 | 40 | UPDATE users SET | 2 |
| 41 | SET | 2 | 41 | id | 3 |
| 42 | str\_repeat | 1 | 42 | i | 1 |
| 43 | count | 2 | 43 | UPDATE users SET is\_active = 0 WHERE id = ? | 1 |
| 44 | array\_intersect\_key | 1 | 44 | localhost | 1 |
| 45 | array\_flip | 1 | 45 | user | 1 |
| 46 | array\_map | 1 | 46 | pass | 1 |
| 47 | array\_keys | 1 | 47 | test\_db | 1 |
| 48 | array\_values | 1 | 48 | John | 2 |
| 49 | array\_merge | 1 | 49 | john@mail.com | 1 |
| 50 | array\_filter | 1 | 50 | User | 3 |
| 51 | prepare | 3 | 51 | added | 1 |
| 52 | bind\_param | 3 | 52 | Found | 2 |
| 53 | execute | 3 | 53 | User updated | 1 |
| 54 | query | 1 | 54 | Admins | 1 |
| 55 | fetch\_assoc | 1 | 55 | Error | 1 |
| 56 | insert\_id | 1 | 56 | \n | 5 |
| 57 | filter\_var | 1 | 57 | Updated | 2 |
|  |  |  | 58 | $this | 18 |
|  |  |  | 59 | s | 4 |
|  |  |  | 60 | ? | 3 |
| **η1 *= 57*** |  | ***N1 = 303*** | **η2 *= 60*** |  | ***N2 = 165*** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Словарь программы **η**=57 + 60 = 117.

Длина программы ***N*** = 303 + 165 = 468.

Объем программы ***V*** = 468 \* = 3215.33;

1. **Скриншоты работы программы:**

Программа выводит результаты в виде html-страницы, на которой есть кнопка, после нажатии которой генерируется результат на странице.