**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський державний університет   
імені Петра Могили**

**Факультет комп’ютерних наук**

**Кафедра <Інтелектуальних інформаційних систем>**

**ЗВІТ**

*з лабораторної роботи № 1*

# "Відтворення алгоритму шифрування SHA-256"

Напрям підготовки: **Комп’ютерні науки**

122-ЛР.ПЗ.01-402.1610219

***Cтудентка*** *\_\_\_\_\_\_\_\_\_\_\_\_\_****С.О.******Рагуліна***

*(підпис)*

*\_\_16.09.2019 \_*

*(дата)*

***Викладач***

*\_\_\_\_\_\_\_\_\_\_\_\_\_* ***Тогоєв О. Р***

*(підпис)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

**Миколаїв – 2019**

# Лабораторна робота №1

# Тема: «Відтворення алгоритму шифрування SHA-256».

Завдання:

* На будь-які мові програмування відтворити алгоритм шифрування SHA-256.
* Результат повинен бути записаним в базу даних(на вибір студента).
* Створити репозиторій github з викладеними файлами та звітом.

**Теоретичні відомості**

**SHA-2** (англ. *Secure Hash Algorithm Version 2* — безпечний алгоритм хешування, версія 2) — збірна назва односторонніх геш-функцій SHA-224, SHA-256, SHA-384 і SHA-512. Геш-функції призначені для створення «відбитків» або «дайджестів» повідомлень довільної бітової довжини. Застосовуються в різних додатках або компонентах, пов'язаних із захистом інформації.

**Історія**

Хеш-функції *SHA-2* розроблені Агентством національної безпеки США і опубліковані Національним інститутом стандартів і технології США у федеральному стандарті обробки інформації *FIPS PUB 180-2* в серпні 2002 року. У цей стандарт також увійшла геш-функція *SHA-1*, розроблена в 1995 році. У лютому 2004 року до *FIPS PUB 180-2* була додана *SHA-224*.

У жовтні 2008 року вийшла нова редакція стандарту — *FIPS PUB 180-3*.

В липні 2006 року з'явився стандарт RFC 4634 «Безпечні геш-алгоритми США (*SHA* і *HMAC-SHA*)», що описує *SHA-1* і сімейство *SHA −2*.

Агентство національної безпеки від імені держави випустило патент на *SHA-2*під ліцензією *Royalty Free*.

**Алгоритм**

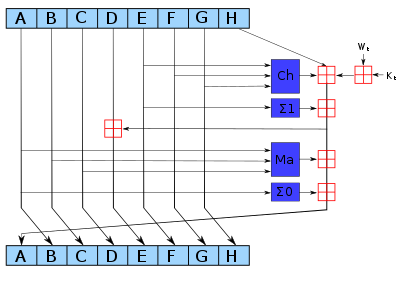
[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:SHA-2.svg)

Схема однієї ітерації алгоритмів SHA-2

Геш-функції сімейства *SHA-2* побудовані на основі структури Меркла-Демґарда.

Початкове повідомлення після доповнення розбивається на блоки, кожен блок — на 16 слів. Алгоритм пропускає кожен блок повідомлення через цикл з 64-ма чи 80-ма ітераціями (раундами). На кожній ітерації 2 слова перетворюються, функцію перетворення задають інші слова. Результати обробки кожного блоку складаються, сума є значенням геш-функції.

Алгоритм використовує такі бітові операції:

ǁ — Конкатенація,

+ — Додавання,

*And* — Побітове «І»,

*Or* — Побітове «АБО»,

*Xor* — Виключне «АБО»,

*Shr* (Shift Right) — Логічний зсув вправо,

*Rotr* (Rotate Right) — Циклічний зсув вправо.

**Хід роботи**

Лабораторну роботу було виконано за допомогою мови PHP.

1. Створення форми реєстрації за допомогою HTML, яка містить два поля – email та пароль. Поле пароль буде хешуватися.

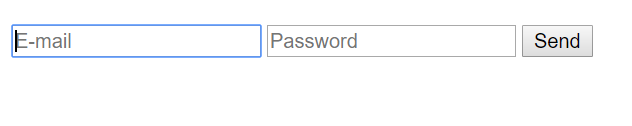


Рис 1. – Форма реєстрації.

1. При успішній обробці форми буде виведено повідомлення, в якому зазначено зашифрований пароль, і додано дані юзера до бази.

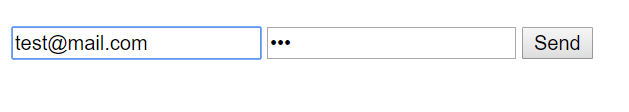


Рис 2. – Заповнення форми.

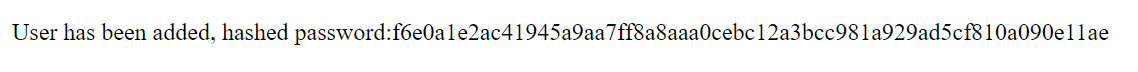


Рис 3. – Результат роботи.

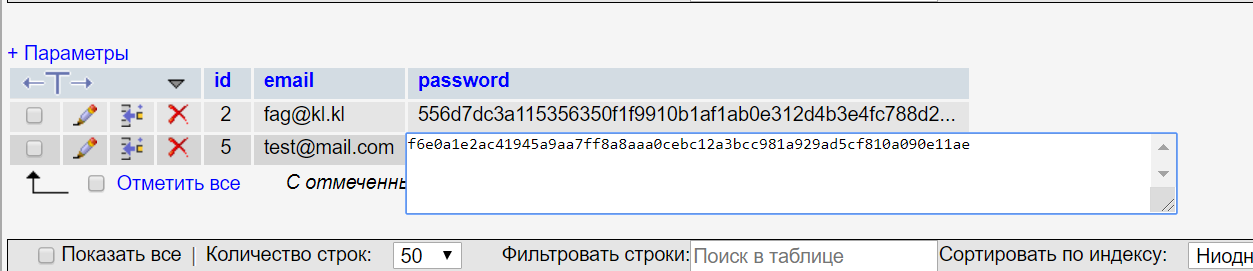


Рис 4. – Додавання інформації користувача до бази даних.

**Лістинг програми:**

Index.html

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <title>  
 Register Form  
 </title>  
</head>  
<body>  
 <div class="form\_section">  
 <form action="./handler.php" method="post">  
 <input type="email" name="form\_\_email" required placeholder="E-mail">  
 <input type="password" name="form\_\_password" required placeholder="Password" >  
 <input type="submit" name="form\_\_submit" value="Send">  
 </form>  
 </div>  
</body>  
</html>

handler.php

**<?php  
require**("./src/Db.php");  
**require**("./src/UsersDbHandler.php");  
  
$email = **isset**($\_POST['form\_\_email']) ? $\_POST['form\_\_email'] : **null**;  
$password = **isset**($\_POST['form\_\_password']) ? $\_POST['form\_\_password'] : **null**;  
**if** (($email === **null**) || ($password === **null**)) {  
 **echo** "Data is not valid";  
 **die**();  
}  
$connection = Db::*getInstance*();  
$db = **new** UsersDbHandler($connection);  
$queryRes = $db->addUser($email, $password);  
$queryRes = json\_decode($queryRes, **true**);  
**if** (!$queryRes["status"]) {  
 **echo** $queryRes["message"];  
 **die**();  
}  
**echo** "User has been added, hashed password:" . $queryRes["password"];

Клас Db для встановлення з’єднання з базою

**<?php  
  
class** Db  
{  
 **static private** *$instance* = **null**;  
  
 **private function** \_\_construct()  
 {  
  
 }  
  
 **public static function** getInstance()  
 {  
 **if** (!**self**::*$instance*) {  
 **try** {  
 $host = "127.0.0.1";  
 $port = "3360";  
 $dbname = "tzi\_lr1";  
 $dbuser = "root";  
 $dbpassword = "";  
 $dsn = 'mysql:host=' . $host . ';port=' . $port . ';dbname=' . $dbname;  
 **self**::*$instance* = **new** PDO($dsn, $dbuser, $dbpassword);  
 } **catch** (PDOException $e) {  
 **echo** $e->getMessage();  
 **echo** json\_encode([  
 'status' => **false**,  
 'line' => *\_\_LINE\_\_*,  
 'body' => [  
 'error\_key' => 'PDOException'  
 ]  
 ]);  
 $instance = **null**;  
 }  
  
 }  
 **return self**::*$instance*;  
 }  
}

Клас UsersDbHandler для роботи з таблицею користувачів

**<?php  
  
class** UsersDbHandler  
{  
 **private** $instance;  
  
 **public function** \_\_construct($connection)  
 {  
 $this->instance = $connection;  
 }  
  
 **public function** addUser($email, $password)  
 {  
 $query = $this->instance->prepare('INSERT INTO users\_table (email, password) values (:email,:password)');  
 $hashedPassword = hash('sha256', $password);  
 $query->bindParam(':email', $email);  
 $query->bindParam(':password', $hashedPassword);  
 **if** ($query->execute()) {  
 **return** json\_encode(  
 **array**(  
 "status" => **true**,  
 "password" => $hashedPassword  
 )  
 );  
 }  
 **return** json\_encode(  
 **array**(  
 "status" => **false**,  
 "message" => "Error occurred"  
 )  
 );  
  
 }  
}

**Висновок:** під час лабораторної роботи було освоєно алгоритм шифрування SHA256. В ході роботи створено форму для користувачів, в результаті якої дані буде зашифровано і додано до бази даних. Закріплено навички роботи з мовою програмування PHP.