|  |
| --- |
| Міністерство освіти і науки УкраїниТернопільський національний технічний університет імені Івана Пулюя |
| (повне найменування вищого навчального закладу) |
| *Факультет комп’ютерно-інформаційних систем і програмної інженерії* |
| (назва факультету ) |
| *Кафедра кібербезпеки* |
| (повна назва кафедри) |

**КВАЛІФІКАЦІЙНА РОБОТА**

на здобуття освітнього ступеня

|  |
| --- |
| бакалавр |
| (освітній рівень) |
| на тему: | *"Програмна реалізація криптографічного* |
| *протоколу підкидання "чесної" монети"* |

|  |
| --- |
| Виконав: студент (ка) |
| Спеціальності:  |
| *125 «Кібербезпека»* |
| (шифр і назва напряму підготовки, спеціальності) |
|  *Тернавчук І.В.* |
|  підпис (прізвище та ініціали) |
|  |
| Керівник |  *Загородна Н.В..*  |
| Нормоконтроль |  підпис (прізвище та ініціали) *Лобур Т.Б.*  |
|  | підпис (прізвище та ініціали) |
| Завідувач кафедри |  *Загородна Н.В.* |
|  | підпис (прізвище та ініціали) |
| Рецензент |   |
|  |  підпис (прізвище та ініціали) |

м. Тернопіль – 2022

|  |
| --- |
| Міністерство освіти і науки України**Тернопільський національний технічний університет імені Івана Пулюя** |
|  |
| Факультет |  комп'ютерно-інформаційних систем і програмної інженерії |
|  | (повна назва факультету) |
| Кафедра |  кібербезпеки |
|  | (повна назва кафедри) |
|  |  |
|  |  | ЗАТВЕРДЖУЮ |
|  |  | Завідувач кафедри |
|  |  |  |  | Загородна Н.В. |
|  |  | (підпис) |  | (прізвище та ініціали) |
|  |  | «\_\_» \_\_\_\_\_\_\_\_\_ 2022 р. |

**ЗАВДАННЯ**

**НА КВАЛІФІКАЦІЙНУ РОБОТУ**

|  |  |
| --- | --- |
| на здобуття освітнього ступеня | Бакалавр |
|  | (назва освітнього ступеня) |
| за спеціальністю | 125 Кібербезпека |
|  | (шифр і назва спеціальності) |
| Студенту | Тернавчуку Ігору Васильовичу |
|  | (прізвище, ім’я, по батькові) |
| 1. Тема роботи | Програмна реалізація криптографічного протоколу підкидання  |
| "чесної" монети |
|  |
| Керівник роботи | Загородна Н.В.., к.т.н., зав. каф. КБ |
|  | (прізвище, ім’я, по батькові, науковий ступінь, вчене звання) |
| Затверджені наказом ректора від « 23 » 03 2022 року № 4/7-178 *.* |
| 2. Термін подання студентом завершеної роботи | 17.06.2022 |
| 3. Вихідні дані до роботи | Вимоги до програмного забезпечення |
|  |
|  |
| 4. Зміст роботи (перелік питань, які потрібно розробити) |
| 1 Огляд літературних джерел (криптографічні протоколи, протокол підкидання чесної  |
| монети) |
| 2 Програмна реалізація криптопротоколу підкидання чесної монети |
|  Вибір середовища розробки |
|  Математична основа |
|  Написання ПЗ |
| 3 Тестування програмного забезпечення |
| 4 Безпека життєдіяльності, основи охорони праці |
| Висновки |
|  |
| 5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень, слайдів) |
| Тема, мета, задачі |
| Криптографічні протоколи, |
| Протокол підкидання чесної монети, |
| Схема Блюма-Мікалі |
| Протокол підкидання монети за допомогою квадратних коренів |
| Алгоритм генерації простого числа з використанням тесту Ферма |
| Результати тестування програмного забезпечення, коли виграє Аліса |
| Результати тестування програмного забезпечення, коли Аліса пробує займатись шахрайством |
| Переваги PHP, Висновки |

6. Консультанти розділів роботи

|  |  |  |
| --- | --- | --- |
| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата |
| завдання видав | завданняприйняв |
| Безпека життєдіяльності, основи хорони праці | Пулька Ч.В., проф. кафедри МТ |  |  |

|  |  |
| --- | --- |
| 7. Дата видачі завдання | 16.02.2022 р. |

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №з/п | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
| 1. | Ознайомлення з завданням до кваліфікаційної роботи | 16.02 – 19.02 | *Виконано* |
| 2. | Підбір джерел про специфічні криптопротоколи | 20.02 – 27.02 | *Виконано* |
| 3. | Опрацювання джерел в галузі дослідження | 28.02 – 16.03 | *Виконано* |
| 4. | Розроблення програмного коду | 17.03 – 20.03 | *Виконано* |
| 5. | Тестування роботи програми та верифікація результатів | 20.03-05.04 | *Виконано* |
| 6. | Оформлення розділу «Огляд літературних джерел» | 06.03 – 17.04 | *Виконано* |
| 7. | Оформлення розділу «Програмна реалізація протоколу підкидання чесної монети» | 18.04 – 29.04 | *Виконано* |
| 8. | Оформлення розділу «Тестування роботи програмного забезпечення  | 30.04 – 13.05 | *Виконано* |
| 9. | Виконання завдання до підрозділу «Безпека життєдіяльності, основи хорони праці» | 14.05 – 21.05 | *Виконано* |
| 10. | Оформлення кваліфікаційної роботи | 22.05 – 05.06 | *Виконано* |
| 11. | Нормоконтроль | 06.06 – 12.06 | *Виконано* |
| 12. | Перевірка на плагіат | 10.06 – 15.06 | *Виконано* |
| 13. | Попередній захист кваліфікаційної роботи | 16.06 – 19.06 | *Виконано* |
| 14. | Захист кваліфікаційної роботи | 24.06.2022 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | Тернавчук І.В. |
|  | (підпис) |  | (прізвище та ініціали) |
| Керівник роботи |  |  | Загородна Н.В.  |
|  | (підпис) |  | (прізвище та ініціали) |

АНОТАЦІЯ

Програмна реалізація криптографічного протоколу підкидання «чесної» монети // Кваліфікаційна робота ОР «Бакалавр» //Тернавчук Ігор Васильович// Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБс-42 // Тернопіль, 2022 // С. \_\_ , рис. – \_\_, табл. – \_\_ , кресл. – , додат. – \_\_\_.

Ключові слова: КРИПТОГРАФІЧНИЙ ПРОТОКОЛ, ЧЕСНА МОНЕТА, ФУНКЦІЯ ДІФФІ-ХЕЛМАНА, КИТАЙСЬКА ТЕОРЕМА ПРО ОСТАЧІ, КВАДРАТНИЙ КОРІНЬ, СХЕМА БЛЮМА.

Кваліфікаційна робота присвячена розробці програмного забезпечення для криптографічного протоколу підкидання чесної монети по телефону. В роботі обґрунтовано вибір програмного середовища розробки та вибір схеми з допомогою квадратних коренів. Розроблене програмне забезпечення (ПЗ) дозволяє відслідкувати логіку протоколу і може бути використано для навчальних потреб. Продемонстровано роботу програмного забезпечення при коректному використанні параметрів схеми та показано, що коли перша сторона намагається займатись шахрайством, то автоматично виграє друга сторона.

Дану розробку можна використовувати, коли сторони не довіряють одна одній і мають потребу в створенні спільної послідовності випадкових біт.

В першому розділі описано різновиди специфічних криптографічних протоколів та схеми підкидання монети по телефону. В другому розділі проведено вибір програмного середовища та наведено окремі важливі функції ПЗ. В третьому розділі наведено результати тестування розробки та верифікацію результатів роботи програми.

ANNOTATION

Software Implementation of a Cryptographic Protocol of “Fair” Coin Flipping // Thesis of educational level "Bachelor" // Ternavchuk Ihor Vasylovych // Ternopil National Technical University named after Ivan Pulyuy, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, СБс-42 group // Ternopil, 2022 // P. \_\_\_ , fig. -\_\_\_\_, table. - \_\_ , chair. - \_\_\_ , added. -\_\_\_.

Keywords: CRYPTOGRAPHIC PROTOCOL, FAIR COIN, DIFFIE-HELLMAN FUNCTION, CHINESE REMAINING THEOREM, SQUARE ROOT, BLOOM S SCHEME.

Qualification thesis is devoted to the development of software for cryptographic protocol of protocol of flipping a fair coin by the telephone. The paper substantiates the choice of software development environment and the choice of the scheme of flipping a fair coin using square roots method. Developed software allows to track the logic of the protocol and can be used for educational purposes. The operation of the software with the correct use of the parameters of the scheme is demonstrated. It is shown that when the first party tries to commit fraud, the second party automatically wins.

This development can be used when the parties do not trust each other and need to create a common sequence of random bits.

The first section describes the types of specific cryptographic protocols and schemes for tossing a coin over the phone. The second section selects the software environment and lists some important software features. The third section presents the results of development testing and verification of the results of the program.

ЗМІСТ

[ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 8](#_Toc106488574)

[ВСТУП 9](#_Toc106488575)

[1 ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ 12](#_Toc106488576)

[1.1 Криптографічні протоколи 12](#_Toc106488577)

[1.1.1 Протокол вручення випадкових бітів 13](#_Toc106488578)

[1.1.2 Протокол підкидання чесної монети 14](#_Toc106488579)

[1.1.2 Протокол розділення секрету. 14](#_Toc106488580)

[1.1.3 Доведення з нульовим знанням (Zero Knowledge Proof - ZKP) 15](#_Toc106488581)

[1.1.4 Електронна готівка (e-cash) 16](#_Toc106488582)

[1.2 Протокол підкидання «чесної» монети 17](#_Toc106488583)

[1.2.1 Протокол Блюма-Мікалі 18](#_Toc106488584)

[1.2.2 Протокол підкидання монети для отримання спільного випадкового біта на основі задачі дискретного логарифмування 19](#_Toc106488585)

[1.2.3 Протокол підкидання монети за допомогою квадратних коренів 19](#_Toc106488586)

[1.2.4 Протокол підкидання монети з допомогою функції Діффі-Хелмана 20](#_Toc106488587)

[2 ПРОГРАМНА РЕАЛІЗАЦІЯ КРИПТОПРОТОКОЛУ ПІДКИДАННЯ ЧЕСНОЇ МОНЕТИ 22](#_Toc106488588)

[2.1 Постановка задачі 22](#_Toc106488589)

[2.2 Вибір середовища розробки 22](#_Toc106488590)

[2.3 Математична основа реалізації алгоритму підкидання монети по телефону 25](#_Toc106488591)

[2.3.1 Розширений алгоритм Евкліда для пошуку оберненого елемента 26](#_Toc106488592)

[2.3.2 Китайська теорема про остачі 28](#_Toc106488593)

[2.3.3 Тест Ферма на простоту числа 30](#_Toc106488594)

[2.3.4 Алгоритм Тонеллі-Шенкса для знаходження квадратного кореня за модулем р. 31](#_Toc106488595)

[2.4 Написання програмного забезпечення 32](#_Toc106488596)

[3 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 35](#_Toc106488597)

[3.1 Тестування та верифікація результатів роботи програми підкидання чесної монети по телефону 35](#_Toc106488598)

[3.2 Тестування роботи програми підкидання чесної монети по телефону на предмет виявлення шахрайства 38](#_Toc106488599)

[4 Безпека життєдіяльності, основи хорони праці 40](#_Toc106488600)

[4.1 Проведення інструктажів з охорони праці 40](#_Toc106488601)

[4.2 Планування робіт щодо охорони праці 43](#_Toc106488602)

[ВИСНОВКИ 45](#_Toc106488603)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 46](#_Toc106488604)

ДОДАТКИ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

|  |  |
| --- | --- |
| https | Hypertext Transfer Protocol Secure |
| ZKP | Zero Knowledge Password Proof |
| IT | Інформаційні технології |
| КП | Криптографічний протокол |
| НСД | Найбільший спільний дільник |
| НСК | Найменше спільне кратне |
| ПЗ | Програмне забезпечення |

# ВСТУП

У наш час життя без доступу до інтернету, а відповідно і потрібної інформації у будь-який час та будь-якому місці за допомогою інформаційних технологій стало немислимим. Ми навчились працювати, купувати, відпочивати, спілкуватися віртуально, однак на передній план вийшла і безпека інформації, яка стала важливішою за сам доступ до неї. Ми платимо за паркування, купуємо квитки на транспорт, переглядаємо соцмережі часто підключаючись до бездротових мереж і забуваючи при цьому про безпеку. Приходимо на роботу, працюємо з фінансовими звітами, переглядаємо конфіденційні бізнес-плани і при цьому часто забуваємо вимикати або блокувати комп’ютер, залишаючи робоче місце. Проте навіть комерційні втрати підприємства не зрівняються з втратами, понесеними державою під час війни внаслідок ігнорування елементарних правил інформаційної безпеки. Зокрема шифрування каналів передачі даних та спілкування є в авангарді війни, яка зараз триває в Україні. Відсутність зашифрованого зв’язку вже призвела до смерті не одного російського генерала та багатьох випадків перехопленого зв’язку. Забезпечення конфіденційного спілкування належить до основних задач криптографії, проте ця наука не обмежується лише безпекою передачі даних відкритими каналами зв’язку. Основними криптографічними механізмами є алгоритми шифрування та схеми цифрового підпису. Не зважаючи на те, що історія криптографії налічує тисячі років, сучасна наука дозволила вийти за рамки цих традиційних механізмів та значно розширила можливості криптографії.

Одним із головних внесків сучасної криптографії є розробка та використання передових протоколів, що надають криптографічні послуги високого рівня, такі як безпечна ідентифікація/аутентифікація користувачів, схеми голосування та цифрові готівкові гроші, протоколи розподілу таємниці та протоколи підкидання чесної монети, схеми сліпого підпису та підписання контракту, доведення з нульовим розголошенням. Криптографічні протоколи хоч використовують базові криптографічні алгоритми як основні компоненти, проте дозволили вийти криптографії за рамки задач шифрування і значно розширили спектр її застосування.

Часто протоколи комбінують зі стандартними схемами шифрування для досягнення тих чи інших цілей. Або ж такі протоколи мають свою специфічну сферу використання. Для прикладу, прокол підкидання чесної монети популяризувала квантова криптографія, яка використовує принципи квантової фізики для створення ключа шифрування. Підкидання монет було розроблено, щоб запобігти обману сторонам, які спілкуються (скажімо, Алісі та Бобу) під час шифрування або передачі даних.

Щоб розпочати протокол підкидання монет, Аліса використовує розділювач променя для поляризації фотонів відповідно до принципів квантової фізики. Розділювач променя визначає базис поляризації і розділяє фотони лінійно (+) або діагонально (х). Аліса надсилає Бобу згенеровані квантові біти, або кубіти, попередньо випадково вибравши базис для кожного. Боб також використовує вибрані випадково базиси для поляризації фотонів і записує результати вимірювань бітів, які часто вимірюються як 0 або 1, як традиційні біти. Якщо базис при вимірюванні відрізняється від базису поляризації, то на виході отримаємо випадковий результат. Боб надсилає Алісі інформацію про те, які базиси використовував для яких фотонів, щоб Аліса підтвердила по відритому каналу, котрі з них були правильні. Боб, при цьому, не відкриває результати вимірювань. Далі користувачі залишають собі лише ті випадки, в яких збіглися обрані базиси [1]. Описаний протокол згодом визнаний першим і базовим протоколом квантової криптографії і був названий на честь його творців BB84 (Charles Bennett та Gilles Brassard).

В той самий час, потрібно зазначити, що протокол підкидання чесної монети був розроблений для більш традиційних задач, зокрема для гри в покер, обміну секретами (тип протоколу, колив кожен з учасників протоколу володіє деякими даними, які готовий обміняти на рівноцінні), електронної пошти з підтвердженням отримання (електронний аналог доставки листа в обмін на підтвердження про його отримання). Крім того, протокол підкидання монети використовується для підвищення ефективності протоколів доведення з нульовим знанням. У цій роботі буде розглянуто, досліджено та реалізовано деякі алгоритми підкидання чесної монети.

*Метою* даної роботи є розробка програмного забезпечення для реалізації криптографічного протоколу підкидання чесної монети, яке може бути використане для організації спілкування учасниками, що не довіряють один одному.

Досягнення наведеної вище мети вимагає розв'язання наступних задач:

* Огляд літературних джерел за темою кваліфікаційної роботи.
* Аналіз основних відомих алгоритмів протоколу підкидання «чесної» монети.
* Обґрунтування середовища розробки.
* Розробка програмного забезпечення.
* Тестування запропонованого програмного рішення.

# 1 ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ

## 1.1 Криптографічні протоколи

Протокол – це багатосторонній алгоритм, розроблений для гарантування певних властивостей інформації, який вимагає участі принаймні двох сторін. Таким чином, алгоритм розподіляється і викликається принаймні в двох різних місцях. Алгоритм, який не розповсюджується, не називається протоколом. Для виконання задач протоколу сторони алгоритму повинні спілкуватися один з одним. Спілкування описується повідомленнями, якими обмінюються сторони. Вони називаються комунікаційними інтерфейсами. Протокол вимагає чіткого визначення інтерфейсу та дій, які має виконати кожна сторона.

Сторона, яка бере участь у протоколі, повинна виконувати синтаксис комунікаційного інтерфейсу, оскільки недотримання синтаксису буде негайно виявлено іншими сторонами. Сторона може або вести себе чесно і дотримуватися поведінки, зазначеної в протоколі. Або вона може вести себе нечесно, лише виконувати синтаксис комунікаційного інтерфейсу, а робити зовсім інші речі. Ці моменти необхідно враховувати при розробці криптографічних протоколів. Під протоколом загалом розуміють розподілений алгоритм (порядок дій) між декількома учасниками, призначений для розв'язання певної задачі [2]. Порядок дій означає, що існує певна послідовність дій, яка повинна бути дотримано при виконанні протоколу. Компонентами протоколу вважаються його учасники, алгоритми, що використовуються учасниками чи формулювання задачі, яку призначений розв'язувати цей протокол. Учасниками протоколу можуть бути люди, сервери, комп'ютерні вузли, комп'ютерні програми, канали зв'язку та бази даних.

Якщо протокол призначений для забезпечення безпеки передачі, обробки та зберігання інформації, то такий протокол називають криптографічним. Криптографічний протокол – це розподілений алгоритм, який точно описує взаємодії між двома або більше об’єктами для досягнення певних цілей безпеки. В свою чергу, криптографічний алгоритм — це чітко визначене перетворення, яке для заданого вхідного значення створює вихідне значення, досягаючи певних цілей безпеки.

Існують базові криптографічні протоколи, такі як: шифрування, накладання цифрового підпису, ідентифікації/аутентифікації, обміну та управління ключами, їх поєднання та різновиди. Проте, існує ряд дуже особливих протоколів, які маловідомі на широкий загал, які, проте, мають важливе значення.

Відомими спеціальними криптографічними протоколами є [3]:

* Протокол вручення випадкових бітів
* Протокол підкидання чесної монети.
* Протокол розподілу секрету.
* Доведення з нульовим знанням.
* Протоколи обміну секретами.
* Підписання контракту.
* Електронна готівка
* Протокол голосування та інші.

Розглянемо детальніше частину з цих протоколів[4].

### 1.1.1 Протокол вручення випадкових бітів

Протокол вручення випадкових бітів є базовим протоколом, який використовується, в тому числі як основа для інших протоколів, в тому числі і протоколу підкидання монети по телефону. Може використовуватись і самостійно для відтермінованого в часі доведення знань. Скажімо користувач А хоче довести свої вміння передбачати курс валют чи акцій на завтра, але не хоче відкривати свого передбачення раніше. Користувач Б хоче переконатись в здібностях А, і хоче бути впевнений, що користувач А не мав можливості змінити своє передбачення після того, як його зробив.

Протокол вручення випадкових бітів може бути зреалізований з допомогою симетричної криптографії, односторонніх функцій та генератора псевдовипадкової послідовності.

### 1.1.2 Протокол підкидання чесної монети

В реальному світі, коли користувач А підкидає монету, то користувач Б спостерігає за процесом і може підтвердити результат. Коли користувач А підкидає монету на відстані від користувача Б, то користувач Б може вгадати результат, але проблема полягає в тому, що два учасники не бачать і не довіряють одне одному. Користувач А має можливості для шахрайства і може оголосити результат, який вигідний йому.

Протокол підкидання чесної монети (по телефону) (coin flipping by telephone protocol) – криптографічний алгоритм, призначений для того, щоб два учасники, які не довіряють один одному могли згенерувати випадковий рівноймовірний біт. Під час протоколу учасники обмінюються повідомленнями за певним алгоритмом, що дозволяє в кінці протоколу сформувати два випадкові біти. Користувач А спершу генерує свій випадковий біт і з використанням схеми вручення випадкового біту передає його користувачу Б. Користувач Б, в свою чергу, обирає інший випадковий біт і передає його користувачу А. Тоді А відкриває свій попередньо вручений біт. Фінальний результат обчислюється як сума за модулем два випадкових бітів двох користувачів.

### 1.1.2 Протокол розділення секрету.

Ідея протоколу розділення секрету полягає в тому, що секрет ділиться на визначену кількість частин, які називають долями (тінями, частками), щоб уникнути втрати секрету або його розголошення одним з учасників [5]. Для відновлення секрету потрібно зібрати всі або деяку порогову кількість часток. В першому випадку схема має назву розбиття (англ. splitting) секрету, а в другому -розподіл (англ. sharing) секрету (спільне володіння). Ще одна відмінність схем розбиття секрету від протоколів розподілення є розподіл часток. В першому випадку обов’язково різні люди повинні володіти частками таємниці, а в другому випадку – одна людина може мати більшу вагу у відновленні секрету і володіти декількома частками секрету. Основною задачею протоколу є відновлення секрету визначеною кількістю учасників, при тому, щоб менша кількість людей не змогла відновити секрет. Для прикладу, неможливо відновити рівняння прямої за однією точкою, що їй належить, проте маючи будь-які дві точки можна однозначно відновити. При чому існує нескінченне число точок, координати яких можна роздати учасникам протоколу, але лише кожні два з них можуть знайти рівняння прямої. Даний протокол застосовується для безпечного розподіленого зберігання інформації.

### 1.1.3 Доведення з нульовим знанням (Zero Knowledge Proof - ZKP)

Доказ з нульовим знанням (ZKP) — це математичний прийом для перевірки істинності інформації без розкриття самої інформації [6] Вперше метод був представлений дослідниками з Массачусетського технологічного інституту в статті 1985 року. Сам термін "нульові знання" походить від того, що не розголошується жодної ("нульової") інформації про таємницю , але друга сторона переконана, що перша сторона знає секретну відповідь на запитання. Цей протокол виглядає можна пояснити на прикладі, який виглядає дещо по-дитячому.

Припустимо, що в однієї сторони (Довірителя) є друг дальтонік (Верифікатор), який не може відрізнити зелену та червону кульку. Першій стороні потрібно довести другій стороні, що кольори кульок різні, але другові-дальтоніку потрібні ґрунтовніші докази, ніж просто слова, щоб переконатися. Метод ZKP може виглядати наступним чином: Верифікатор (друга сторона) бере м’ячі і дає вам побачити, який м’яч у якій руці. Потім він за спиною, або міняє м’ячі між руками, або ні. Верифікатор знову показую Довірителю кульки і запитує поміняв він їх чи ні. Оскільки здорова людина може відрізнити зелену кулю від червоної, то легко зможе дати правильну відповідь. Проте це ще може не переконати дальтоніка, адже шанси вгадати не такі вже й малі – 50%. Однак, якщо цю процедуру повторити кілька разів, зрештою, ймовірність того, що Довіритель правильно вгадаєте відповідь, буде дуже низькою буде зменшуватись з кожною спробою. Це дає змогу другій стороні переконатися, що кульки різного кольору, не знаючи фактичних кольорів кульок.

І хоч приклад з кульками виглядає дитячим, цей протокол має можна застосовувати і на практиці, для прикладу метод ZKP може бути використаний як спосіб автентифікації, коли користувачі не обмінюються паролями по відкритим каналам, а отже зростає надійність зберігання паролю. ZKP також можна використати

Крім цього, можна використати на перевірки наявності коштів на банківському рахунку особи для проведення транзакції, при цьому не відкриваючи інформації про баланс рахунку. Або для підтвердження валідності пароля, без необхідності його безпосередньої обробки. Таким чином, протокол доведення з нульовим знанням може бути частиною всіляких конфіденційних угод, транзакцій та взаємодій у більш безпечний спосіб.

### 1.1.4 Електронна готівка (e-cash)

Електронна готівка (e-cash) спрямована на досягнення конфіденційності клієнта при оплаті, подібно до реальної готівки [7]. Фізична готівка забезпечує кращу конфіденційність: платежі важко відстежити, оскільки немає центрального органу, який контролює всі транзакції, на відміну від більшості електронних платіжних систем. Ця властивість є джерелом натхнення для «невідстежуваної» системи електронної готівки. Першу таку систему електронної готівки, що зберігає анонімність клієнта, представив Девід Чаум у 1983 році: клієнт може анонімно зняти кеш зі свого банку та витратити її у продавця. Потім продавець може внести кеш в банк, який зарахує його на рахунок. Для забезпечення конфіденційності електронної готівки, а також необхідних непідробних властивостей було запропоновано кілька протоколів безпеки. Щоб бути безпечним, протокол електронної готівки повинен не тільки забезпечувати конфіденційність клієнта, але також повинен гарантувати, що клієнт не зможе підробити монети, які не були випущені банком. Крім того, він повинен захищати від подвійних витрат коштів– інакше клієнт може спробувати використати ті самі гроші кілька разів. Цього можна досягти за допомогою онлайн-платежів, тобто продавець повинен зв'язатися з банком при оплаті перш ніж прийняти кошти, однак це дороге рішення. Альтернативне рішення, яке зазвичай використовується для підтримки офлайн-платежів (тобто продавець може прийняти платіж, не звертаючись до банку). В даному випадку особу клієнта можна виявити тільки, якщо він двічі витратив монету. Нарешті, винуватість гарантує, що зловмисник не зможе підробити подвійні витрати, а отже, неправильно звинуватити чесного клієнта в подвійних витратах. У літературі є багато протоколів електронної готівки, було запропоновано (Cha83; CFN90; Dam90; DC94, Cre94; Bra94; AF96; KO01; FHY13).

## 1.2 Протокол підкидання «чесної» монети

Підкидання монети по телефону (англ. Coin flipping by telephone) – це езотеричний протокол, за допомогою якого вирішується завдання передачі інформації між двома учасниками, які перебувають на відстані один від одного і при цьому не довіряють один одному. Опис протоколу було запропоновано американським вченим Мануелем Блюмом в 1982 році на 24-й комп’ютерній конференції IEEE, яке він опублікував у своїй статті «Підкидання монети по телефону: протокол для вирішення нерозв'язаних завдань».

Нехай Алісі та Бобу необхідно вирішити якесь спірне питання, кинувши монету. У випадку коли вони обидва фізично знаходяться поруч цю задачу можна вирішити за допомогою звичайної процедури підкидання монети. Якщо хтось з учасників не довіряє монеті, можна використати інші джерела отримання рівно ймовірнісних подій.

Якщо ж Аліса і Боб далеко один від одного, і можуть спілкуватися лише каналами зв’язку, то така вирішення цієї задачі, на перший погляд, видається нереальним. І справді, якщо користуватися звичайною процедурою підкидання монети, перший крок зробить Боб, який вибере один з можливих варіантів – «орел» або «решка» - результат жеребкування. В такому випадку Аліса завжди може вибрати більш вигідний для неї варіант, після того як Боб оголосить своє припущення.

Задача жеребкування легко вирішується, якщо існує надійна третя сторона, яка користується повною довірою Аліси та Боба, і яка має конфіденційні (закриті) канали зв’язку з обома учасниками. У такому випадку Боб і Аліса вибирають випадкові біти b і с відповідно і посилають їх таємно один від одного агенту. Останній чекає поки не прийде обидва біти, і після цього публікує *b*, *c* і $d = b ⊕ c$ – результат підкидання монети.

За відсутності третьої довіреної особи спрацьовує ідея, яку простіше всього зрозуміти на наступній «фізичній» реалізації. Боб вибирає випадковий біт b, записує його на листку паперу, закриває його в ящику, залишаючи ключ в себе, і відправляє ящик Алісі. Очікується, що без ключа, Аліса не зможе відкрити ящик. Отримавши ящик Аліса вибирає випадковий біт c і відправляє його Бобу. У відповідь Боб посилає Алісі ключ від ящика. Результатом підкидання монети буде знову ж таки біт $d = b ⊕ c$.

Розглянемо декілька різновидів протоколу підкидання «чесної» монети по телефону.

### 1.2.1 Протокол Блюма-Мікалі

Користувач A і користувач В заздалегідь домовляються про деяку односторонню функцію *f*: X → Y, причому в множині Х є однакова кількість парних та непарних чисел [8]

A: випадковим чином вибирає $xϵX$, потім обчислює значення *y*=*f*(*x*).

A→B: передає значення *y.*

B→A: біт *с=*0, якщо припущення «*x* - парне», *c=*1, якщо припущення «*x* - непарне».

A→B: надсилає *x.*

B: перевіряє чи *f*(*x*)*=y*.

Виникає логічне запитання – навіщо Алісі такі складнощі: брати число $xϵX$, коли простіше взяти випадковий біт *b* і вибрати деяку односторонню функцію для цього біта - *y*=*f*(*b*)? Це потрібно тому, що число аргументів односторонньої функції повинно бути досить великим, щоб унеможливити підбір *y*. Адже біт *b* може приймати тільки два значення – «0» і «1». Користувачу B в такому випадку не складе труднощів обчислити два значення односторонньої функції *y*0*=f*(0) і *y*1=*f*(1) і визначити, котре з ним буде однакове з отриманим значенням *y*, щоб здогадатись початкове значення біту *b*.

### 1.2.2 Протокол підкидання монети для отримання спільного випадкового біта на основі задачі дискретного логарифмування

Параметри протоколу: *p* – велике просте число, *g* – примітивний елемент мультиплікативної групи Z*p*\* [9]

А: вибирає випадкове $xϵ Z\_{p}$, обчислює *y=gx* mod *p*.

A→B: відправляє значення *y.*

B→A: вибирає випадковий біт *b,* випадкове число *k* з *Zp ,* вичисляє *r = ybgk* mod *p* і відправляє Алісі.

B→A: *r*

A→B: випадковий біт *c*$ϵ$*{0,1}*

B→A: *b, k*

A: перевіряє *r=ybgk* mod *p*. Якщо числа проходять перевірку, то результатом виконання протоколу буде біт *d=b**c*.

Значення *r* – це криптографічний аналог ящика. Насправді ж з зі значення *r*  користувач А не зможе отримати жодної інформації про біт *b*. Оскільки величина *k* обирається випадкового з *Z*p, значення *r* і у випадку *b=0*, і у випадку *b=1* є випадковим елементом Zp\*, утвореної примітивним елементом g.

Не володіючи інформацією про *k*, неможливо дізнатись значення b з r, g і p. Користувач А може спробувати шахрайство, вибравши y не з Zp\*, Але це легко можна перевірити *yp-1 mod p = 1.* У іншому випадку, Боб може обманути, відкриваючи значення свого біту b як 0, так і як 1, але лише в тому випадку, якщо він вміє вираховувати дискретні логарифми.

### 1.2.3 Протокол підкидання монети за допомогою квадратних коренів

А : вибирає два великих простих числа *p* і *q* та обчислює добуток *n=pq*.

A→B: відправляє значення *n.*

B: вибирає випадковий біт $r<n/2$*,* обчислює *z = r2 mod n*.

B→A: *z*

A: обчислює чотири квадратних кореня (за допомогою китайської теореми про остачі) *z (mod n*). Вона може зробити це, тому що вона знає розкладання n на множники. Назвемо їх + х,- х, + у і -у. Позначимо $х'=min(x mod n; -x mod n)$ та $y'=min(y mod n; -y mod n)$:

Очевидно, що r рівне або х' або у'

A→B припущення $r=х'$ або $r=y'$

Якщо здогадка Аліси правильна, результатом кидка монети є "орел" (А виграв), а якщо неправильна - "решка" (виграв B). B оголошує результат підкидання монети. Якщо А правильно відгадав число r, то B змушений це визнати і не зможе піти на обман, оскільки, не маючи чисел p і q, він не може порахувати інший корінь. Якщо ж А не вгадав число, то щоб довести, що він виграв B може просто або назвати число r, яке він реально загадав, або ж маючи тепер два корені числа, знайти p та q і назвати їх Алісі.

### 1.2.4 Протокол підкидання монети з допомогою функції Діффі-Хелмана

У цьому протоколі в якості односпрямованої функції використовується Діффі-Хелмана (піднесення в степінь за модулем) простого числа р :

А: вибирає просте число *р* так, щоб множники *р-1* були відомі, і серед них було принаймні одне велике просте число.

B: вибирає два примітивних елемента, *h* і *t*, в *Fp*

B→A: відправляє значення *h* і *t.*

A: переконується, що *h* і *t* є примітивними елементами, і потім вибирає випадкове число *х*, таке що *НСД(х, р-1)=1*.

А→В: Обчислює і передає одне з двох значень обчислює одне з двох значень: *y = hx mod р*, або *у = tx mod р.*

В→А: припущення про те, які з примітивних елементів використав А.

A: оголошує результат: якщо припущення B - правильне, результатом кидання монети є "орел", в іншому випадку - "решка"

Підпротокол перевірки:

А→В значення х. Боб обчислює $h^{x} mod р$ і $t^{x} mod р$, переконуючись, що Аліса грала чесно і перевіряючи результат кидка. Він також перевіряє, що *х* і *р-1* - взаємно прості числа.

Щоб А міг схитрувати, то він повинен вміти розв’язувати задачу дискретного логарифмування. Це є дуже проблематично.

В може схитрувати, якщо h і t не є примітивними елементами в полі *Fp*, Але А зможе легко перевірити це після другого етапу, так як йому відомо розкладання р-1 на прості множники.

Вдалим в цьому протоколі є те, що якщо А і В захочуть кинути кілька монет, вони зможуть використовувати одні і ті ж значення р, h, t. А просто генерує нове х, і протокол триває з третього етапу.

# 2 ПРОГРАМНА РЕАЛІЗАЦІЯ КРИПТОПРОТОКОЛУ ПІДКИДАННЯ ЧЕСНОЇ МОНЕТИ

## 2.1 Постановка задачі

Отже необхідно розробити програмне забезпечення, в якому криптопротокол підкидання чесної монети з допомогою квадратних коренів.

Програмна реалізація схеми підкидання чесної монети з допомогою квадратних коренів: повинна задовольняти наступним вимогам:

* 1. Мати інтуїтивно зрозумілий інтерфейс
	2. Передбачати модуль генерації простих чисел.
	3. Передбачати модуль знаходження оберненого елемента.
	4. Передбачати модуль знаходження квадратних коренів за модулем.
	5. Ілюструвати алгоритм обчислень
	6. На виході програми має бути очевидним результат підкидання монети.

Розробка будь-якого програмного забезпечення вимагає насамперед вибору і обґрунтування середовища розробки

## 2.2 Вибір середовища розробки

Оскільки однією з вимог до розробки програми є наявність інтуїтивного користувацького інтерфейсу і можливості проілюструвати алгоритм роботи програми, то було вирішено створювати веб-орієнтований продукт, а для того обрали мову програмування php.

Hypertext Preprocessor (PHP) — це мова сценаріїв з відкритим вихідним кодом, яка широко використовується для веб-розробки на стороні сервера. Створена у 1994 році Расмусом Лердорфом, датсько-канадським програмістом, Мова PHP спочатку називалась Personal Home Page. Перша версія PHP (PHP 1.0) була випущена у червні 1995 року [10]. З тих пір для наступних версій PHP було зроблено різні розробки, і ми маємо PHP 8.1.2 як останню версію PHP, яка була випущена в 2022 року. Талісманом проекту PHP є синій «elePHPant» (слон) з логотипом, написаним на боці.

На рисунку 2.1 приведено особливості мови PHP:



Рисунок 2.1 – Основні риси PHP

*Відкритий вихідний код (Open source)*. оскільки PHP є відкритим вихідним кодом, будь-хто може завантажити його та використовувати, як йому подобається, для досягнення своїх цілей веб-розробки. Це дає змогу розробникам у всьому світі перевіряти коди, публікувати помилки та сприяти кодуванню та виправленню помилок.

*Простота та легкість використання (Simple and easy to use).* PHP простий і легкий у використанні, що є однією з основних причин, чому багато розробників віддають перевагу цій мові програмування. Його синтаксис дуже схожий на синтаксис мови C, з добре організованою та логічною структурою. Також легко вивчити PHP і зануритися в кодування порівняно з багатьма мовами сценаріїв.

*Підтримка кількох баз даних (Multiple database support).* PHP підтримує різні бази даних, включаючи MySQL, Oracle, PostgreSQL тощо, а також інтеграцію з базою даних.

*Міжплатформна сумісність (Cross-platform compatibility)*. Він може працювати майже на всіх операційних системах, включаючи Windows, Mac, Linux та Unix. PHP-скрипти також запускаються на таких пристроях, як комп’ютери, ноутбуки, планшети та мобільні телефони. Крім того, PHP також сумісний з різними серверами, такими як IIS, Apache тощо.

*Гнучкість (Flexible).* PHP пропонує більшу гнучкість і можливості вбудовування. Його можна легко інтегрувати з JavaScript, HTML, XML тощо. *Звіти про помилки та обробка винятків* *(Error reporting and exception handling*). PHP підтримує деякі наперед визначені константи звітів про помилки для створення попереджень про помилки. Крім того, це полегшує обробку винятків для відображення помилок та їх виправлення.

*Ефективна та швидка продуктивність (Efficient and fast performance).* скрипти PHP можуть демонструвати кращу продуктивність, ніж багато мов сценаріїв, таких як JSP, PERL, ASP.NET тощо. Його завантаження даних та підключення до бази даних також швидші, щоб забезпечити ефективне керування базою даних, функціональність пошти та адміністрування сервера.

*Моніторинг у режимі реального часу (Real-time monitoring)* він надає останні відомості про користувача. Ви також можете отримати доступ до даних про використання ЦП та пам’яті.

*Об’єктно-орієнтовані функції* *(Object-oriented features)* об’єктно-орієнтовані функції в PHP сприяють його швидкості та пропонують додаткові функції, такі як успадкування, інкапсуляція даних тощо.

Крім того, PHP пропонує магічні константи, регулярні вирази, клас PDO, підтримує файли cookie, виконання оболонки в командному рядку та багато іншого.

*Застосування:*

* *Веб-програми та веб-сторінки:* сучасні веб-сторінки та програми повинні забезпечувати більший рівень налаштування, надзвичайно інтерактивний і дружній інтерфейс користувача, підтримувати плавну інтеграцію та забезпечувати безпечні онлайн-транзакції на додаток до швидкої продуктивності.
* PHP використовує трирівневу архітектуру, яка працює на сервері, системі баз даних і браузері лінійно, щоб задовольнити цю потребу.
* Графічний дизайн та обробка зображень: PHP надійний для керування текстовим вмістом, графічним дизайном та обробкою зображень. Він підтримує інтеграцію з багатьма бібліотеками, такими як ImageMagick, GD library, Imagine та іншими програмами на основі PHP.
* Він дозволяє користувачам повертати, змінювати розмір, обрізати, додавати водяні знаки, створювати мініатюри та витягувати зображення в різних форматах, таких як PNG, GIF, JPEG, WBMP і XPM.
* Web CMS: PHP розширює підтримку різних баз даних і взаємодіє з різними службами за допомогою протоколів, включаючи IMAP, POP3, HTTP, COM, SNMP тощо. Тому часто створюються системи керування веб-контентом, такі як WordPress, Moodle, Drupal, Joomla тощо.
* Програми на основі графічного інтерфейсу. Можна створювати програми на основі графічного інтерфейсу користувача (GUI) за допомогою PHP-GTK 2 і ZZEE PHP GUI, які дозволяють скрипти PHP та компілювати ваш код у файл .exe, який може працювати автономно.
* Представлення даних: за допомогою таких інструментів, як Image\_Graph з PHP, можна створювати різні представлення даних, наприклад діаграми, графіки тощо. Це корисно на веб-сайтах електронної комерції та в програмах, які використовують представлення даних у різних формах.
* Створення Flash-елементів. Ви можете створювати Flash-елементи, такі як градієнти, фігури, растрові зображення, кнопки, тексти, морфії, анімації, дії, форми електронної пошти, форми входу тощо, використовуючи бібліотеки з відкритим кодом на основі PHP (наприклад, Ming).
* Створення програм для електронної комерції: Створити програму для електронної комерції легко за допомогою фреймворків на основі PHP, таких як CodeIgniter. Це корисно для бізнесу будь-якого масштабу, від стартапів до підприємств. Платформами електронної комерції, створеними за допомогою PHP, є Magento, PrestaShop, OpenCart тощо.

## 2.3 Математична основа реалізації алгоритму підкидання монети по телефону

Для реалізації схеми підкидання монети по телефону було обрано алгоритм підкидання чесної монети з допомогою квадратних коренів. Проте імплементація цього алгоритму потребує низку додаткових знань з теорії чисел, зокрема китайської теореми про остачі, поняття оберненого елементу поля та його пошуку, алгоритму пошуку квадратичного кореня за модулем.

### 2.3.1 Розширений алгоритм Евкліда для пошуку оберненого елемента

Розширений алгоритм Евкліда [11] вимагає спершу виконання звичайного алгоритму Евкліда, в основі якого лежить наступна теорема:

*Теорема про ділення з остачею*. Будь-яке ціле число *а* може бути представлене єдиним способом з допомогою цілого числа *b (b<a)* в наступному вигляді:

|  |  |
| --- | --- |
| *a = bq + r, 0≤ r < b*. | (2.1) |

Число *q* називають неповною часткою, а число *r* − остачею від ділення.

Нехай *a* і *b* − натуральні числа і *a>b*. Якщо *a* не ділиться на *b*, то згідно вище наведеної теореми

*a* = *bq* + *r*1, 0 < *r*1< *b*.

Якщо *b* не ділиться на *r*1, то за цією ж теоремою

*b* = *r*1*q*1 + *r*2, 0 < *r*2 < *r*1*.*

Якщо *r*1 не ділиться на *r*2, то

*r*1 = *r*2*q*2 + *r*3, 0 < *r*3< *r*2

і т.д.

*Алгоритм Евкліда* полягає в процесі послідовного ділення, що має *n+1* кроків. Отже отримуємо систему рівностей:

|  |  |
| --- | --- |
| *a* = *bq*0 + *r*1,*b* = *r*1*q*1 + *r*2,. . . . . . . . . . . .*rn*−2 = *rn*−1*qn*−1 + *rn*,*rn*−2 = *rn*−1*qn*−1 + *rn*,*rn*−1 = *rnqn*. | (2.2) |

Множина спільних дільників *a* і *b* співпадає з множиною спільних дільників чисел *b* і *r*1, множина спільних дільників чисел *b* і *r*1 співпадає з множиною спільних дільників чисел *r*1 і *r*2, множина спільних дільників чисел *r*1 і *r*2 співпадає з множиною спільних дільників чисел *r*2 і *r*3 і т.д. В кінці-кінців можна прийти до висновку, що множина спільних дільників чисел *a* і *b* співпадає з множиною спільних дільників чисел *rn*−1 і *rn* , і далі вона співпадає з множиною дільників числа *rn* (якщо число a є кратним числу *b*, то сукупність загальних дільників чисел *a* *і* *b* співпадає з сукупністю дільників одного числа *b;* зокрема $НСД\left(a,b\right)=\left(a,b\right)=b$)) Тоді справедливими є співвідношення $(a,b)=(b,r\_{1})=$ $=(r\_{1},r\_{2})$ $=...=(r\_{n-1},r\_{n})=r\_{n}$. Отже фактично ми довели наступну теорему.

Теорема**.** *НСД* чисел *a* і *b* дорівнює останній відмінній від нуля остачі *rn* в алгоритмі Евкліда.

Приклад. Застосуємо алгоритм Евкліда, що визначається сукупністю формул (2.2) для знаходження найбільшого спільного дільника чисел 525 і 231.

Маємо,

|  |  |
| --- | --- |
| 525=231\*2+63  | *r*1=63 |
| 231=63\*3+42  | *r*2=42 |
| 63=42\*1+**21** | *r*3=21 |
| 42=21\*2+0 | *r*4=0 |

Остання невід’ємна остача *r*3= 21; тому (525, 231)= = 21.

Розширений алгоритм Евкліда – це розширення алгоритму Евкліда. Окрім знаходження найбільшого спільного дільника для цілих *a* і *b*, як це робить алгоритм Евкліда, він також знаходить цілі x і y (одне з яких зазвичай від'ємне), які задовольняють наступне рівняння.

*a x + b y = НСД( a,b )*

Коефіцієнти х та у можна знайти за алгоритмом оберненим до алгоритму Евкліда. Розглянемо це на прикладі. Виразимо всі ненульові остачі, які отримали в результаті виконання алгоритму Евкліда

|  |  |
| --- | --- |
| 525=231\*2+63  | *r*1=**63**=525-231\*2 |
| 231=63\*3+42  | *r*2=**42**=231-63\*3 |
| 63=42\*1+**21** | *r*3=**21=**63-42\*1 |

Звідси:

*r*3=**21=**63-**42**\*1=

=63-(231-63\*3)\*1=**63**\*4-231\*1=

=(525-231\*2)\*4-231\*1=525\*4-231\*9.

Отже,

525\*4-231\*9=21

Звідси *x=4*, *y=-9*.

Нехай ○ – операція на Х з одиницею е. Якщо для елемента $x\in X$ існує елемент $x'\in X$, такий що $x○x^{'}=e$, то $x^{'}$ називають *оберненим* елементом до *х* відносно операції ○. Оберненим елементом до $x$ в полі $F\_{p}$ називають такий елемент $x^{-1}$, що $x∙x^{-1}≡1 mod p$. Розширений алгоритм Евкліда може використовуватись для пошуку обернених елементів. Для цього на вхід алгоритму потрібно подати числа *х* та *р*. Тоді коефіцієнт, який є множником числа *х* в розширеному алгоритмі Евкліда буде оберненим до нього за модулем *р*.

### 2.3.2 Китайська теорема про остачі

Авторство формулювання та доведення теореми про остачі приписують китайському математику першого століття Сун Це [12]. Це один з елементарних результатів теорії чисел. Згідно з цією теоремою, будь-яке невід’ємне число, яке не перевищує добутку модулів можна однозначно відновити, якщо відомі його лишки за цими модулями.

*Теорема 1***.**Нехай числа $M\_{s}$ і $M\_{s}^{'}$ визначені з умов

|  |  |
| --- | --- |
| $m\_{1}∙m\_{2}∙m\_{3}∙…∙m\_{s-1}∙…m\_{s+1}∙…∙m\_{k}$ = $M\_{s}m\_{s}$, | (2.3) |

при чому, $M\_{s} M\_{s}^{'}≡1 mod m\_{s}$, $s = 1,2,...,k $і цілі числа $m\_{1},m\_{2},m\_{3}…m\_{k}$ попарно взаємо-прості, тобто $НСД\left(m\_{i},m\_{j}\right)=1$, $i,j = 1,2,...,k$.

Нехай $b\_{1},b\_{2},...,b\_{k} $− теж цілі , такі, що $0\leq b\_{i}\leq m\_{i}$, $i = 1,2,...,k$.

Тоді система рівнянь

|  |  |
| --- | --- |
| $$x≡b\_{1} \left(mod m\_{1}\right)$$$$x≡b\_{2} (mod m\_{2})$$……………………..$$x≡b\_{k} (mod m\_{k})$$ | (2.4) |

має єдиний розв’язок в інтервалі $\left[0, M-1\right]$*,* де $M=m\_{1}∙m\_{2}∙m\_{3}∙…∙m\_{k}$

|  |  |  |
| --- | --- | --- |
| $$x\_{0}=M\_{1} M\_{1}^{'}b\_{1}+M\_{2} M\_{2}^{'}b\_{2}+...+M\_{k} M\_{k}^{'}b\_{k} (mod M)$$ | (2.5) |  |

Розглянемо застосування китайської теореми про остачі на прикладі. Нехай відомі остачі від ділення деякого невідомо числа *х* на 20, 23, 37. Необхідно визначити число *х.*

$x mod 20≡1$

$x mod 23≡10$

$x mod 37≡31$

Переконуємось, що числа 20, 23, 37 є попарно взаємопростими.

Знаходимо $M=20∙23∙37=17020$

Визначаємо з (2.3) числа

$$M\_{1}=\frac{M}{m\_{1}}=\frac{17020}{20}=851$$

$$M\_{2}=\frac{M}{m\_{2}}=\frac{17020}{23}=740$$

$$M\_{3}=\frac{M}{m\_{3}}=\frac{17020}{37}=460$$

За розширеним алгоритмом Евкліда знаходимо обернені до них за відповідними модулями:

$M\_{1}^{-1} mod m\_{1}=851^{-1} mod 20=11^{-1} mod 20=11$,

$M\_{2}^{-1} mod m\_{2}=740^{-1} mod 23=4^{-1} mod 23=6$,

$M\_{3}^{-1} mod m\_{3}=460^{-1} mod 37=16^{-1} mod 37=7$*.*

Знаходимо роз’язок за формулою (2.5)

$$x\_{0}=851∙11∙1+740∙6∙10+460∙7∙31 mod 17020=401$$

### 2.3.3 Тест Ферма на простоту числа

Непарне число *p* називається псевдопростим за основою$ xϵN$, якщо НСД(*x, p*)=1 і $x^{p-1}≡1 mod p$.

Задамося деяким числом $m\in N$ .

*Крок* *1.* Вибираємо з інтервалу $1<x<p$ випадкове число $xϵN$ (незалежне від раніше обраних $x$, число *x* може бути непростим). За допомогою алгоритму Евкліда знаходимо НСД(*x, p*). Якщо НСД(*x, p*)=1, то переходимо до кроку 2. Якщо НСД(*x, p*)>1, то повторюємо крок 1.

*Крок 2.* Перевіряємо, чи є *p* псевдопростим за основою *x* за вище наведеним означенням. Крок 2 повторити *m*  разів. Якщо крок 2 повторився $k<m$ разів, то переходимо до кроку 1. Якщо крок 2 повторився $m$ разів, то алгоритм зупиняється.

У результаті число *p* буде перевірено по тесту Ферма при m різних основах $x\_{1},x\_{2},…x\_{m}$. Якщо *p* виявиться не псевдопростим хоча б за однією основою$x\_{1},x\_{2},…x\_{m}$, то *p* складене. Якщо *p* псевдопросте за всіма цими основами, то вважаємо *p* простим числом.

Більш детально, алгоритм генерації простого числа зображено на рисунку 2.2



Рисунок 2.2 – Алгоритм генерації простого числа за тестом Ферма

Для малих чисел достатньо задавати m=1, проте для великих значень р, можна отримати такі ситуації, коли рівність $x^{p-1}≡1 mod p$ виконується, проте р є складеним. Саме тому тест Ферма необхідно виконати для кількох значень *x*.

### 2.3.4 Алгоритм Тонеллі-Шенкса для знаходження квадратного кореня за модулем р.

Роботу алгоритму проілюстровано в [13]

Виділяємо степінь двійки з *р-1:* $p-1=2^{s}Q$*,* де *Q*  - непарне.

1. Якщо $s=1$, тобто $p≡3 mod 4$

Тоді $x=\pm a^{\frac{p+1}{4}} mod p$

1. Якщо $s\geq 2$, то
	1. Виберемо довільний квадратичний нелишок z, тобто таке число z, що символ Лежандра $\left(\frac{z}{p}\right)=1$.
	2. Присвоїмо $с≡z^{Q} mod p$, $R=a^{\frac{Q+1}{2}} mod p$, $t=a^{Q} mod p$, M=s
	3. Виконуємо цикл:
		1. Якщо $t≡1 (mod p)$, то алгоритм повертає R
		2. В циклі знаходимо найменше $0<i<M$, таке що $t^{2^{i}}≡1 mod p$

Нехай $b=c^{2^{M-i-1}} mod p$

 $R=Rb mod p$

 $t=tb^{2} mod p$

 $c=b^{2} mod p$

 $M=i$ і до початку циклу

* 1. Алгоритм повертає R

Як бачимо найлегше квадратний корінь шукати за умови, коли $p≡3 mod 4$. Розглянемо приклад.

$x^{2}=3 mod 11$

$p-1=11-1=10=2∙5, $$s=1$

$x=\pm a^{\frac{p+1}{4}} mod p=\pm 3^{\frac{11+1}{4}} mod 11=\pm 3^{3} mod 11=\pm 5 $

## 2.4 Написання програмного забезпечення

Згідно вимог до програмного забезпечення потрібно було реалізувати алгоритм підкидання чесної монети по телефону на основі схеми з квадратними коренями.

При застосуванні китайської теореми про остачі необхідно буде знаходити обернений елемент за модулем. В лістингу 2.1 наведено код розширеного алгоритму Евкліда, який дозволяє це робити.

Лістинг 2.1 – Реалізація розширено алгоритму Евкліда

public function evklid($a, $b)

 {

 $p=1; $q=0; $r=0; $s=1;

 while ($a && $b) {

 if ($a>=$b) {

 $a = $a - $b;

 $p = $p - $r;

 $q = $q - $s;

 } else

 {

 $b = $b - $a;

 $r = $r - $p;

 $s = $s - $q;

 }

 }

 if ($a) {

 $x = $p;

 $y = $q;

 }else

 {

 $x = $r;

 $y = $s;

 }

 return [$x, $y];

 }

 На вхід алгоритму подаються два числа *a* та *b*, на виході отримуємо коефіцієнти розширеного алгоритму Евкліда *x* та *у*, при чому *у* є оберненим до *b* за модулем а. Аналогічно *x* є оберненим до а за модулем *b*

В лістингу 2.2 приведено код застосування китайської теореми про остачі для знаходження квадратних коренів за модулем.

Лістинг 2.2 – Код застосування китайської теореми про остачі для знаходження квадратних коренів за модулем

public function china($p, $q, $z, $n)

 {

 list($pMinus, $qMinus) = $this->evklid($p, $q);

 $m = [];

 $m[1] =gmp\_intval(gmp\_mod(bcpow($z,(($p+1)/4)), $p));

 $m[2] = gmp\_intval(gmp\_mod(($p - $m[1]), $p));

 $m[3] = gmp\_intval(gmp\_mod(bcpow($z,(($q+1)/4)), $q));

 $m[4] = gmp\_intval(gmp\_mod(($q - $m[3]), $q));

 $a = $q \* gmp\_intval(gmp\_mod($qMinus, $p));

 $b = $p \* gmp\_intval(gmp\_mod($pMinus, $q));

 $M = [];

 $M[1] = gmp\_intval(gmp\_mod(bcadd(($a \* $m[1]), ($b \* $m[3])), $n));

 $M[2] = gmp\_intval(gmp\_mod(bcadd(($a \* $m[1]),($b \* $m[4])) , $n));

 $M[3] = gmp\_intval(gmp\_mod(bcadd(($a \* $m[2]),($b \* $m[3])), $n));

 $M[4] = gmp\_intval(gmp\_mod(bcadd(($a \* $m[2]),($b \* $m[4])), $n));

 return $M;

 }

На вхід функції, наведеної в лістингу 2.2 подаються прості числа p та q, які попередньо випадково генеруються та перевіряються на простоту тестом Ферма, число z, отримане від другої сторони та добуток чисел *n=pq*. На виході отримуємо чотири можливих кореня $\sqrt{z}$ за модулем *n*. Очевидно, що ті корені може знайти лише той, хто знає розклад числа n на множники.

Основний код програми приведений в додатку А

# 3 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 3.1 Тестування та верифікація результатів роботи програми підкидання чесної монети по телефону

В результаті виконання кваліфікаційної роботи було розроблене програмне забезпечення, яке дозволяє сформувати спільний випадковий біт (підкинути монету по телефону). На рисунку 3.1 приведено скрін роботи програми та результат її виконання, якщо біт вгадано правильно.



Рисунок 3.1 – Результат роботи програми, коли виграє Аліса (орел)

Бачимо, що Аліса вибирає два простих числа p=1319 та q=1399, які перевіряються на простоту описаним в п.2.3.3 тестом Ферма. Зауважимо, що дані числа вважаються малими для практичних застосувань, проте їх обрано для зручності верифікації програмних результатів.

Потім Аліса посилає Бобу добуток цих чисел:

$$n=1319∙1399=1845281$$

Боб випадковим чином вибирає число r=905743 в межах від 1 до $922640=\left[\frac{1845281}{2}\right]$ та обчислює $z=905743^{2} mod 1845281=890912$, яке передає Алісі.

Завдяки тому, що Аліса знає розклад числа $n=1845281 $ вона може порахувати чотири корені рівняння $x^{2}=890912 mod n$. Продемонструємо це.

Оскільки $n=1319∙1399=1845281$ і $p=1319≡3 mod 4$ та $q=1399≡3 mod 4$, то

$$m^{p}=\pm z^{\frac{p+1}{4}} mod p=\pm 890912^{\frac{1319+1}{4}} mod 1319=$$

$$=\pm 890912^{330} mod 1319=\pm 909 mod 1319$$

та

$$m^{q}=\pm z^{\frac{q+1}{4}} mod p=\pm 890912^{\frac{1399+1}{4}} mod 1399=$$

$$=\pm 890912^{350} mod 1399=\pm 590 mod 1399$$

Звідси:

$m\_{1}^{p=1319}=909$

$m\_{2}^{p=1319}=410$

$m\_{3}^{q=1399}=590$

$m\_{4}^{q=1399}=809$

На основі китайської теореми про остачі та зважаючи на те, що $q=\frac{n}{p}$ та $p=\frac{n}{q}$, маємо:

|  |  |
| --- | --- |
| $x\_{0}^{1}=q\left(q^{-1} mod p\right)m\_{1}^{p}+p\left(p^{-1} mod q\right)m\_{3}^{q} mod n$$x\_{0}^{2}=q\left(q^{-1} mod p\right)m\_{1}^{p}+p\left(p^{-1} mod q\right)m\_{4}^{q} mod n $$x\_{0}^{3}=q\left(q^{-1} mod p\right)m\_{2}^{p}+p\left(p^{-1} mod q\right)m\_{3}^{q} mod n$$x\_{0}^{4}=q\left(q^{-1} mod p\right)m\_{2}^{p}+p\left(p^{-1} mod q\right)m\_{4}^{q} mod n$ | (3.1) |

Знайдемо обернені елементи $p^{-1} mod q$ та $q^{-1} mod p$

Виконаємо розширений алгоритм Евкліда для чисел $p$ та $q$:

1399=1319\*1+80

1319=80\*16+39

80=39\*2+2

39=2\*19+1

2=1\*2

Остання ненульова остача дорівнює 1, отже НСД (1399,1319)=1

Виразимо остачі:

80=1399-1319\*1

39=1319-80\*16

2=80-39\*2

1=39-2\*19

Виконаємо підстановки описані в п. 2.3.1:

1=39-2\*19=39-19\*(80-39\*2)=-19\*80+39\*39=)=-19\*80+39\*(1319-80\*16)=39\*1319-643\*80=39\*1319-643\*(1399-1319\*1)=-643\*1399+682\*1319

Звідси:

$p^{-1}=682 mod q $є оберненим до 1319 за модулем 1399 (перевірка: 682\*1319=1 *mod* 1399)

$q^{-1}=-643=676 mod p$ є оберненим до 1399 за модулем 1319 (перевірка: 676\*1399=1 *mod* 1319)

З (3.1) маємо:

$x\_{0}^{1}=1399∙676∙909+1319∙682∙590 mod 1845281=905743$

$x\_{0}^{2}=1399∙676∙410+1319∙682∙809 mod 1845281=939538 $

$x\_{0}^{3}=1399∙676∙410+1319∙682∙590 mod 1845281=1381403$

$x\_{0}^{4}=1399∙676∙909+1319∙682∙809 mod 1845281=463878$

Зауважимо, що $x\_{0}^{1}+x\_{0}^{2}=n$ та $x\_{0}^{3}+x\_{0}^{4}=n$

Згідно з алгоритмом, Аліса повинна вибрати одне з двох чисел $min\left(905743; 939538 \right)=905743$ та $min\left(1381403 ; 463878 \right)=463878$

Якщо Аліса вибирає $905743$, то вгадує r, задумане Бобом і виграє цей поєдинок, що і проілюстровано на рис. 3.1.

## 3.2 Тестування роботи програми підкидання чесної монети по телефону на предмет виявлення шахрайства

В цьому пункті спробуємо розібратись, чи є у кожного з гравців можливості для шахрайства. Якщо Аліса правильно вгадала число r, то Боб змушений це визнати і не зможе піти на обман, оскільки, не маючи чисел p і q, він не може порахувати інший корінь, що і проілюстровано в попередньому підпункті.

Що ж буде відбуватися, якщо Аліса вибере, скажімо два однакових простих числа: p=q=1627, тоді їх добуток $n=p^{2}=1627^{2}=2647129$ (рис.3.2)



Рисунок 3.2 – Результат роботи програми, коли при спробі шахрайства

Тоді Боб обирає число r=475139 в межах від 1 до $1323564=\left[\frac{2647129}{2}\right]$ та обчислює $z=475139^{2} mod 2647129=1966814$, яке передає Алісі.

Тоді Аліса знаючи, що $n=p^{2}=1627^{2} $ та що $p=1627≡3 mod 4$легко обчислює два корені квадратні

$$m^{p}=\pm z^{\frac{p+1}{4}} mod p=\pm 1966814^{\frac{1627+1}{4}} mod 1627=$$

$$=\pm 1966814^{407} mod 1627=\pm 1572 mod 1627$$

Таким чином, виходить, що

$m\_{1}^{p=1627}=1572$

$m\_{2}^{p=1627}=55$

$m\_{3}^{q=1627}=1572$

$m\_{4}^{q=1627}=55$

Далі згідно з алгоритмом, ми б мали знайти обернений елемент до p за модулем p, а його не існує, тому програма за замовчуванням поставила одиницю.

$x\_{0}^{1}=1627∙1∙1572 mod 2647129$ *=2557644*

$x\_{0}^{2}=1627∙1∙55 mod 2647129 =89485 $

Два інших значення ідентичні обчисленим. Жодне з чисел не відповідає вибраному r, отже програма розроблена таким чином, що коли Аліса намагається шахраювати, то автоматично виграє Боб.

# 4 Безпека життєдіяльності, основи хорони праці

## 4.1 Проведення інструктажів з охорони праці

Працівники, під час прийняття на роботу та періодично, повинні проходити на підприємстві інструктажі з питань охорони праці, надання першої медичної допомоги потерпілим від нещасних випадків, а також з правил поведінки та дій при виникненні аварійних ситуацій, пожеж і стихійних лих [14].

За характером і часом проведення інструктажі з питань охорони праці (далі інструктажі) поділяються на вступний, первинний, повторний, позаплановий та цільовий.

Вступний інструктаж проводиться:

* з усіма працівниками, які приймаються на постійну або тимчасову роботу, незалежно від їх освіти, стажу роботи та посади;
* з працівниками інших організацій, які прибули на підприємство і беруть безпосередню участь у виробничому процесі або виконують інші роботи для підприємства;
* з учнями та студентами, які прибули на підприємство для проходження трудового або професійного навчання;
* з екскурсантами у разі екскурсії на підприємство.

Вступний інструктаж проводиться спеціалістом служби охорони праці або іншим фахівцем відповідно до наказу (розпорядження) по підприємству, який в установленому Типовим положенням порядку пройшов навчання і перевірку знань з питань охорони праці.

Вступний інструктаж проводиться в кабінеті охорони праці або в приміщенні, що спеціально для цього обладнано, з використанням сучасних технічних засобів навчання, навчальних та наочних посібників за програмою, розробленою службою охорони праці з урахуванням особливостей виробництва.

Запис про проведення вступного інструктажу робиться в журналі реєстрації вступного інструктажу з питань охорони праці , який зберігається службою охорони праці або працівником, що відповідає за проведення вступного інструктажу, а також у наказі про прийняття працівника на роботу.

Первинний інструктаж проводиться до початку роботи безпосередньо на робочому місці з працівником:

* новоприйнятим (постійно чи тимчасово) на підприємство або до фізичної особи, яка використовує найману працю;
* який переводиться з одного структурного підрозділу підприємства до іншого;
* який виконуватиме нову для нього роботу;
* відрядженим працівником іншого підприємства, який бере безпосередню участь у виробничому процесі на підприємстві.

Повторний інструктаж на робочому місці індивідуально з окремим працівником або групою працівників, які виконують однотипні роботи, за обсягом і змістом переліку питань первинного інструктажу.

Повторний інструктаж проводиться в терміни, визначені нормативно-правовими актами з охорони праці, які діють у галузі, або роботодавцем (фізичною особою, яка використовує найману працю) з урахуванням конкретних умов праці, але не рідше:

* на роботах з підвищеною небезпекою - 1 раз на 3 місяці;
* для решти робіт - 1 раз на 6 місяців.

Позаплановий інструктаж проводиться з працівниками на робочому місці або в кабінеті охорони праці:

* при введенні в дію нових або переглянутих нормативно-правових актів з охорони праці, а також при внесенні змін та доповнень до них;
* при зміні технологічного процесу, або модернізації устаткування, приладів та інструментів, вихідної сировини, матеріалів та інших факторів, що впливають на стан охорони праці;
* при порушеннях працівниками вимог нормативно-правових актів з охорони праці, що призвели до травм, аварій, пожеж тощо;
* при перерві в роботі виконавця робіт більш ніж на 30 календарних днів - для робіт з підвищеною небезпекою, а для решти робіт - понад 60 днів.

Позаплановий інструктаж з учнями, студентами, курсантами, слухачами проводиться під час проведення трудового і професійного навчання при порушеннях ними вимог нормативно - правових актів з охорони праці, що можуть призвести або призвели до травм, аварій, пожеж тощо.

Позаплановий інструктаж може проводитись індивідуально з окремим працівником або з групою працівників одного фаху.

Цільовий інструктаж проводиться з працівниками:

* при ліквідації аварії або стихійного лиха;
* при проведенні робіт, на які відповідно до законодавства оформлюються наряд-допуск, наказ або розпорядження.

Цільовий інструктаж проводиться індивідуально з окремим працівником або з групою працівників. Обсяг і зміст цільового інструктажу визначаються залежно від виду робіт, що виконуватимуться.

Первинний, повторний, позаплановий і цільовий інструктажі проводить безпосередній керівник робіт (начальник структурного підрозділу, майстер) або фізична особа, яка використовує найману працю.

Первинний, повторний, позаплановий і цільовий інструктажі завершуються перевіркою знань у вигляді усного опитування або за допомогою технічних засобів, а також перевіркою набутих навичок безпечних методів праці, особою, яка проводила інструктаж.

При незадовільних результатах перевірки знань, умінь і навичок щодо безпечного виконання робіт після первинного, повторного чи позапланового інструктажів протягом 10 днів додатково проводяться інструктаж і повторна перевірка знань.

При незадовільних результатах перевірки знань після цільового інструктажу допуск до виконання робіт не надається. Повторна перевірка знань при цьому не дозволяється.

Про проведення первинного, повторного, позапланового та цільового інструктажів та їх допуск до роботи, особа, яка проводила інструктаж, уносить запис до журналу реєстрації інструктажів з питань охорони праці на робочому місці. Сторінки журналу реєстрації інструктажів повинні бути пронумеровані, прошнуровані і скріплені печаткою.

У разі виконання робіт, що потребують оформлення наряду-допуску, цільовий інструктаж реєструється в цьому наряді-допуску, а в журналі реєстрації інструктажів не обов'язково.

Перелік професій та посад працівників, які звільняються
від повторного інструктажу, затверджується роботодавцем. До цього переліку можуть бути зараховані працівники, участь у виробничому процесі яких не пов'язана з безпосереднім обслуговуванням об'єктів, машин, механізмів, устаткування; застосуванням приладів та інструментів, збереженням або переробкою сировини, матеріалів тощо.

## 4.2 Планування робіт щодо охорони праці

Робота з охорони праці здійснюється у відповідності з перспективнимим і поточним планів створення безпечних і нешкідливих умов праці, в яких визначені задачі підприємству в цілому і окремим структурним підрозділам, а також керівникам і спеціалістам.

 Планування робіт здійснюється на основі [15]:

* заходів, які забезпечують досягнення встановлених нормативів безпеки праці, гігієни праці та виробничого середовища; заходів, передбачених колективним договором;
* заходів по усуненню недоліків, виявлених при розслідуванні нещасних випадків, професійних захворювань і аварій.

Контроль за станом охорони праці включає:

* оцінку рівня небезпечних виробничих факторів(НВФ) і шкідливих виробничих факторів(ШВФ) на робочих місцях;
* виявлення порушення вимог законів і нормативних актів з охорони праці;
* перевірку усунення раніше виявлених порушень;
* перевірку виконання працівником обов’язків з охорони праці;
* перевірку виконання планів робіт з охорони праці;
* перевірку забезпечення працівників ЗІЗ і ЗКЗ.

 Види контролю:

* зі сторони органів державного нагляду;
* зі сторони служби з охорони праці;
* оперативний контроль керівниками і іншими посадовими особами підприємства;
* громадський контроль;
* комісія підприємства, уповноваженою працівниками особою з питань охорони праці.

Оцінка стану охорони праці і результатів профілактичної роботи здійснюється за прийнятими на підприємстві показниками. Як джерело вихідної інформації використовуються: акти про нещасні випадки, звіти про виробничий травматизм; матеріали атестації робочих місць, паспорта санітарно-гігієнічного стану умов праці; журнали оперативного контролю за станом охорони праці структурного підрозділу, акти і приписи перевірок стану охорони праці.

Узагальнені дані про стан охорони праці і результатів профілактичної роботи підготовлюються службою охорони праці і підлягають обов’язковому розгляду і аналізу на всіх рівнях управління підприємства.

Стимулювання роботи з охорони праці, направлене на підвищення зацікавленості працівників у забезпеченні безпечних умов праці, здійснюється відповідно Положенню, існуючому на підприємстві, в якому визначені конкретні показники, умови, види і форми заохочення за активну участь і ініціативу в реалізації заходів з підвищення безпеки праці і за роботу без порушень правил безпеки, а також заходи впливу на порушників.

# ВИСНОВКИ

В процесі виконання кваліфікаційної роботи було проаналізовано специфічні криптографічні протоколи, такі як протокол розділення секрету, підкидання чесної монети, доведення з нульовим знанням та інші. Безумовно, такі протоколи застосовуються рідше, ніж стандартні криптографічні алгоритми, проте часто вони є невід'ємною частиною безпеки системи. Метою даної роботи була розробка програмного забезпечення для криптографічного протоколу підкидання чесної монети по телефону.

В результаті виконання кваліфікаційної роботи виконано наступні завдання:

* Проведено огляд літературних джерел за тематикою роботи.
* Досліджено сфери застосування схеми підкидання монети по телефону.
* Проаналізовано основні відомі алгоритми протоколу підкидання чесної монети.
* Обґрунтовано вибір середовища розробки.
* Створено програмне забезпечення.
* Проведено верифікацію результатів роботи розробленого програмного рішення.

В першому розділі описано сутність криптографічних протоколів та основні відомі алгоритми підкидання чесної монети. В другому розділі обґрунтовано вибір програмного середовища та описано основні математичні та програмні моменти для імплементації протоколу підкидання монети. В третьому розділі висвітлено результати тестування розроблено ПЗ та проведено верифікацію його роботи. Крім того, розділ "Безпека життєдіяльності, основи хорони праці" мітить питання проведення інструктажів з охорони праці та планування робіт щодо охорони праці.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Charles H Bennett, Gilles Brassard Quantum cryptography: Public key distribution and coin tossing// International Conference on Computers, Systems & Signal Processing, Bangalore, India, Dec 9-12, 1984. P 175-179
2. Брюс Шнайер: Прикладна криптографія. Протоколи, алгоритми і вихідний код на C— К:Діалектика, 2002. —— 1040 с. ISBN 978-5-9908462-4-1
3. Миронов А.М. Криптографические протоколи Москва электронная публикация, 2016, 119с. Режим доступу до ресурсу: <http://www.dut.edu.ua/uploads/l_1122_30338177.pdf>
4. Черемушкин А.В. Криптографические протоколы. Основные свойства и уязвимости. Учебное пособие. — М.: Академия, 2009. — 272 с.
5. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution. PhD thesis, Technion (1996), <http://www.cs.bgu.ac.il/~beimel/pub.html>
6. Wu, Huixin; Wang, Feng (2014). "A Survey of Noninteractive Zero Knowledge Proof System and Its Applications". The Scientific World Journal. 2014: 560484. doi:10.1155/2014/560484
7. J. Dreier, A. Kassem and P. Lafourcade, "Formal analysis of e-cash protocols," 2015 12th International Joint Conference on e-Business and Telecommunications (ICETE), 2015, pp. 65-75.
8. Manuel Blum, Coin Flipping by Telephone: A Protocol for Solving Impossible Problems, ACM SIGACT, Vol. 15, No. 1, 1983, pp. 23-2
9. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушки А.В. Основи криптографии. Москва, Гелиос, 2002, 480 с.
10. Khandwala, Dhwanil. (2012). Advantages of PHP Development Frameworks. Information and Software Technology. 10.5281/zenodo.809543.
11. Бабаш А.В., Шанкин Г.П. Криптография. Москва, СОЛОН-Р, 2002, 511 с.
12. В.В. Ященко Введение в криптографию МЦНМО, 2012, 352 с. ISBN: 978-5-4439-0026-1
13. Нестеренко А. Ю. Теоретико-числовые методы в криптографии. — Москва. — 2012. — [ISBN 978-5-94506-320-4](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F%3A%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785945063204)
14. Основи охорони праці: Підруч для студ вищих навч закладів За ред мп Гандзюка - К Каравела, 2004 - 408 с.
15. Охорона праці в галузі комп'ютингу: підручник / Л. А. Катренко, А. В. Катренко ; [за наук. ред. В. В. Пасічника] ; М-во освіти і науки, молоді та спорту України. — Л. : Магнолія 2006, 2012. — 544 с

ДОДАТКИ

Додаток А

Код основної програми

public function getAlissasNambers()

 {

 $first = 1;

 while (($first + 1) % 4 != 0) {

 $first = gmp\_intval(gmp\_nextprime((1000) + mt\_rand(10, 1000)));

 }

 $second = 1;

 while (($second + 1) % 4 != 0) {

 $second = gmp\_intval(gmp\_nextprime((1000) + mt\_rand(10, 1000)));

 }

 $dobutok = $first \* $second;

 $r = mt\_rand(1000, ($dobutok / 2));

 $z = bcpow($r,2) % $dobutok;

 $M = $this->china($first, $second, $z, $dobutok);

 if(($M[1] % $dobutok) < ($M[4] % $dobutok)) {

 $xShtrich = ($M[1] % $dobutok);

 } else {

 $xShtrich = ($M[4] % $dobutok);

 }

 if(($M[2] % $dobutok) < ($M[3] % $dobutok)) {

 $yShtrich = ($M[2] % $dobutok);

 } else {

 $yShtrich = ($M[3] % $dobutok);

 }

 $alisaSays = (mt\_rand(1, 10) % 2) == 0 ? ['X<sup>\'</sup>', $xShtrich] : ['Y<sup>\'</sup>', $yShtrich];

 $resultMoneta = ($alisaSays[1] == $r ? "Орел" : "Решка");

 $data = [

 'first' => $first,

 'second' => $second,

 'dobutok' => $dobutok,

 'r' => $r,

 'n' => $dobutok,

 'z' => $z,

 'm' => $M,

 'xShtrich' => $xShtrich,

 'yShtrich' => $yShtrich,

 'alisaLetter' => $alisaSays[1],

 'resultMoneta' => $resultMoneta,

 ];

 return response()->json($data);

 }