МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
імені ІГОРЯ СІКОРСЬКОГО
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ

ЗВІТ
з лабораторної роботи № 2

Виконав:
студент ФБ-41мн
Дрик Владислав
Олександрович
«29» Березня 2025

КИЇВ 2024

Тема: Реалізація смарт-контракту або анонімної криптовалюти.
Мета роботи: «Отримання навичок роботи із смарт-контрактами або анонімними криптовалютами

Для першого типу лабораторних:

- дослідження методів анонімізації/деанонімізації запропонованої криптовалюти із аналізом складності проведення атак деанонімізації і втрат ефективності анонімних криптовалют у порівнянні із Bitcoin/Litecoin;
- оцінка та обґрунтування необхідних ресурсів (гасу і ефіру), потрібних для функціонування смарт-контракту.

MONERO

Monero (XMR) — це децентралізована криптовалюта з відкритим кодом, головна мета якої — забезпечити повну анонімність транзакцій. На відміну від Bitcoin, де всі платежі відкриті й відстежувані, Monero приховує:

- Хто кому надсилає гроші
- Яку суму надсилають
- Баланси користувачів

1 Методи анонімізації в Monero

Stealth Addresses (приховані адреси)

Stealth-адреси дозволяють отримувачам приховувати свій публічний ключ у блокчейні. Коли відправник надсилає транзакцію, він створює унікальну одноразову адресу на основі публічного ключа одержувача. Це унеможливлює прив'язку транзакцій до конкретного гаманця.

Ring Signatures (кільцеві підписи)

Кільцеві підписи дозволяють приховати відправника транзакції. Кожна транзакція містить кілька можливих підписантів (так званих "декойів"), один з яких є справжнім, але не можна визначити який саме. Це забезпечує анонімність відправника серед групи користувачів.

RingCT (Ring Confidential Transactions)

RingCT дозволяє приховувати суму транзакції. Завдяки криптографічному механізму Pedersen Commitments, тільки учасники транзакції знають точну суму, водночас уся мережа може верифікувати її коректність.

2 Методи деанонімізації Monero

Попри високий рівень конфіденційності, існують теоретичні і практичні підходи до деанонімізації Monero. Втім, вони є складними, неточними та не завжди ефективними:

Аналіз часових міток

Якщо атакувальник має інформацію про час створення реальної транзакції, можна спробувати визначити справжнього відправника серед декойів, оскільки інші транзакції могли бути старішими.

Аналіз кількості декойів

Історично Monero використовувала малу кількість декойів, що робило ймовірнішим вгадування справжнього підписанта. Однак у нових версіях кількість декойів збільшена до 11 і більше, що ускладнює статистичні атаки.

Аналіз мережевого рівня

Якщо спостерігати за мережею через ноди, теоретично можна відстежити IP-адресу, яка ініціює транзакцію. Для цього використовуються атаки типу "global passive adversary", але Monero користувачі можуть використовувати Tor або I2P для захисту.

1. Ring Signatures (приховують відправника)

При кожній транзакції справжній відправник «ховається» серед кількох фіктивних (декой) входів. Зараз за замовчуванням використовується 16 учасників у кільці — це значно зменшує ймовірність правильного вгадування. Навіть при складному аналізі, зловмисник не може точно сказати, хто з підписантів справжній. Атаки можливі лише при статистичному аналізі великої кількості транзакцій, часто з використанням сторонніх даних (наприклад, якщо відомо, коли користувач щось купував).

Складність: дуже висока, особливо з актуальною кількістю декойів.

2. Stealth Addresses (приховують отримувача)

Кожна транзакція створює нову унікальну адресу, навіть якщо відправляєш кошти тому самому гаманцю. Це унеможливлює відстеження балансу будь-якого гаманця через блокчейн. Жодна зовнішня особа не може знати, хто є одержувачем.

Складність деанонімізації: майже неможливо без приватного ключа.

3. RingCT (приховує суму)

Сума в кожній транзакції зашифрована з допомогою Pedersen Commitments. Перевірити її правильність може лише мережа, не знаючи, яка вона насправді. Навіть при аналізі багатьох транзакцій не вдається точно визначити, скільки було передано.

Складність деанонімізації: дуже висока, сучасними методами — нереально.

4. Атаки на мережевому рівні (мережеве спостереження)

Теоретично можливе спостереження за IP-адресами через ноди (наприклад, якщо нода контролюється зловмисником). Але користувачі можуть використовувати Tor або I2P для повного приховування маршруту. Крім того, Monero використовує протокол Dandelion++, який ускладнює відстеження початкової ноди.

Складність: висока, особливо якщо користувач вживає базові заходи безпеки.

Аналіз втрат ефективності анонімних криптовалют у порівнянні із Bitcoin

Параметр	Bitcoin	Monero	Втрати ефективності у Monero
Середній розмір транзакції	~250 байт	~1,500–3,000 байт	У 6–12 разів більше через анонімізацію
Час створення блоку	10 хвилин	2 хвилини	Коротший час, але *
Транзакцій в секунду (TPS)	~3–7 TPS	~1–4 TPS	Менша пропускна здатність через великі транзакції

Параметр	Bitcoin	Monero	Втрати ефективності у Monero
Розмір повного блокчейну (2024)	~500 ГБ	~150 ГБ	Monero зростає швидше (~2–4 ГБ/міс проти ~1–2 ГБ/міс)
Тип алгоритму майнінгу	SHA-256 (ASIC-дружній)	RandomX	Вищі вимоги до ОЗП та CPU для звичайного користувача
Обчислювальні витрати на валідацію	Низькі/середні	Високі (через кільцеві підписи та RingCT)	Більше часу та ресурсів на перевірку транзакцій
Анонімність транзакцій	Відсутня	Повна (адреса, сума і відправник приховані)	Немає втрат, а навпаки — перевага
Підтримка бірж та сервісів	Майже скрізь	Частково/обмежено	Монеро часто не підтримується через регуляторні причини

*Monero має швидший час блоку, але через великий розмір транзакцій він не може обробляти значно більше операцій за одиницю часу. Це зменшує перевагу від коротшого інтервалу між блоками.

Для другого типу лабораторних робіт:

1) Запуск Monero-ноди через докер

```
prosto_acc@Vladyslavs-MacBook-Pro ~ % docker pull sethsimmons/simple-monerod

Using default tag: latest
latest: Pulling from sethsimmons/simple-monerod
cb8611c9fe51: Download complete
73e094c91a89: Download complete
3093b2556e15: Download complete
4c436b15c7be: Download complete
e535267ea379: Download complete
4f4fb700ef54: Already exists
85703b65a846: Download complete
7101f69878cb: Download complete
7eb36add168f: Download complete
dd1af117b174: Download complete
Digest: sha256:a17e6e72c8eafb05d04a4dc5f2dda8a9f0c237c7013b0936fcee180c4a284104
Status: Downloaded newer image for sethsimmons/simple-monerod:latest
docker.io/sethsimmons/simple-monerod:latest
prosto_acc@Vladyslavs-MacBook-Pro ~ % docker run -d --restart unless-stopped --name="monerod" \
  -v monero_data:/home/monero/.bitmonero \
  -p 18080:18080 \
  -p 18089:18089 \
  sethsimmons/simple-monerod \
  --rpc-restricted-bind-ip=0.0.0.0 \
  --rpc-restricted-bind-port=18089 \
  --no-igd \
  --no-zmq \
  --enable-dns-blocklist

53f47e02c39f170505c9548cbed3a4fe1fab280a13586aa9e2c442a5c4df1abe
prosto_acc@Vladyslavs-MacBook-Pro ~ %
```

-v monero_data:/home/monero/.bitmonero: створює постійний том для збереження даних блокчейну.

-p 18080:18080: відкриває порт для peer-to-peer з'єднань. -p 18089:18089: відкриває обмежений RPC-порт для клієнтів.

--rpc-restricted-bind-ip=0.0.0.0: дозволяє підключення до RPC з будь-якої IP-адреси.

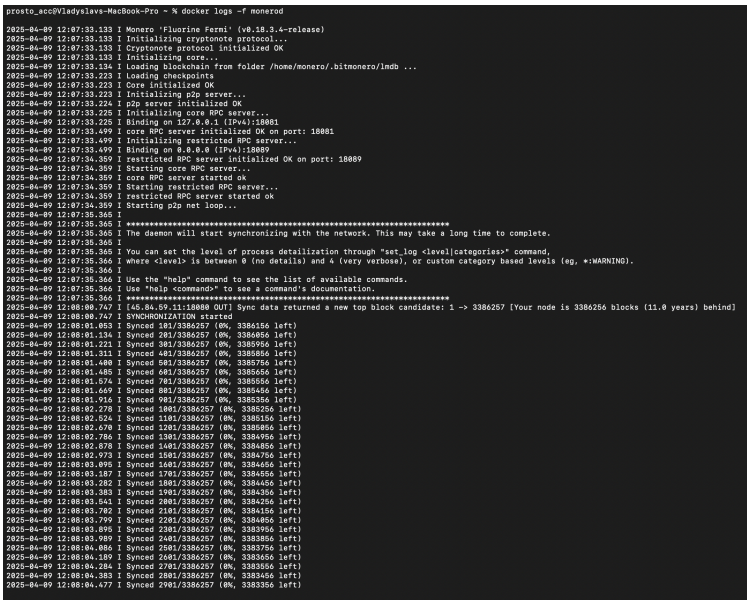
--rpc-restricted-bind-port=18089: встановлює порт для обмеженого RPC.

--no-igd: відключає автоматичне перенаправлення портів через UPnP.

--no-zmq: відключає ZeroMQ.

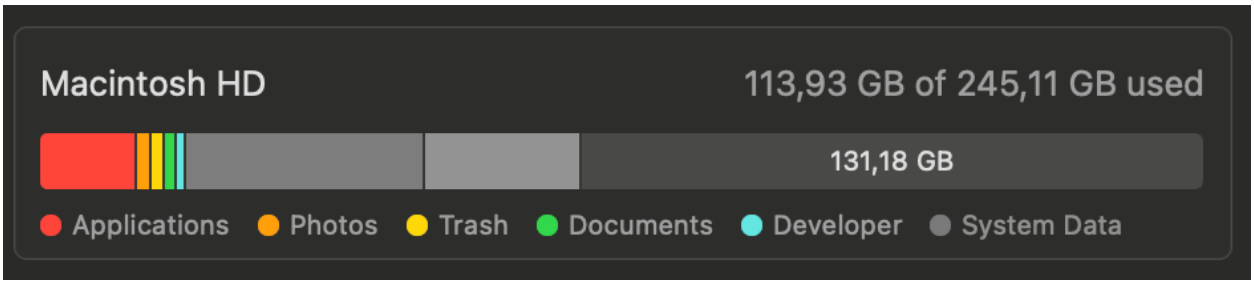
--enable-dns-blocklist: вмикає блокування за DNS.

Перевіряємо роботу ноди



```
prosto_acc@Vladyslavs-MacBook-Pro ~ % docker logs -f monerod

2025-04-09 12:07:33.133 I Monero 'Fluorine Fermi' (v0.18.3.4-release)
2025-04-09 12:07:33.133 I Initializing cryptonote protocol...
2025-04-09 12:07:33.133 I Cryptonote protocol initialized OK
2025-04-09 12:07:33.133 I Initializing core...
2025-04-09 12:07:33.134 I Loading blockchain from folder /home/monero/.bitmonero/lmdb ...
2025-04-09 12:07:33.223 I Loading checkpoints
2025-04-09 12:07:33.223 I Core initialized OK
2025-04-09 12:07:33.223 I Initializing p2p server...
2025-04-09 12:07:33.224 I p2p server initialized OK
2025-04-09 12:07:33.225 I Initializing core RPC server...
2025-04-09 12:07:33.225 I Binding on 127.0.0.1 (IPv4):18081
2025-04-09 12:07:33.499 I core RPC server initialized OK on port: 18081
2025-04-09 12:07:33.499 I Initializing restricted RPC server...
2025-04-09 12:07:33.499 I Binding on 0.0.0.0 (IPv4):18089
2025-04-09 12:07:34.359 I restricted RPC server initialized OK on port: 18089
2025-04-09 12:07:34.359 I Starting core RPC server...
2025-04-09 12:07:34.359 I core RPC server started ok
2025-04-09 12:07:34.359 I Starting restricted RPC server...
2025-04-09 12:07:34.359 I restricted RPC server started ok
2025-04-09 12:07:34.359 I Starting p2p net loop...
2025-04-09 12:07:35.365 I
2025-04-09 12:07:35.365 I **********************************************************************
2025-04-09 12:07:35.365 I The daemon will start synchronizing with the network. This may take a long time to complete.
2025-04-09 12:07:35.365 I
2025-04-09 12:07:35.365 I You can set the level of process detailization through "set_log <level|categories>" command,
2025-04-09 12:07:35.366 I where <level> is between 0 (no details) and 4 (very verbose), or custom category based levels (eg, *:WARNING).
2025-04-09 12:07:35.366 I
2025-04-09 12:07:35.366 I Use the "help" command to see the list of available commands.
2025-04-09 12:07:35.366 I Use "help <command>" to see a command's documentation.
2025-04-09 12:07:35.366 I **********************************************************************
2025-04-09 12:08:00.747 I [45.84.59.11:18080 OUT] Sync data returned a new top block candidate: 1 -> 3386257 [Your node is 3386256 blocks (11.0 years) behind]
2025-04-09 12:08:00.747 I SYNCHRONIZATION started
2025-04-09 12:08:01.053 I Synced 101/3386257 (0%, 3386156 left)
2025-04-09 12:08:01.134 I Synced 201/3386257 (0%, 3386056 left)
2025-04-09 12:08:01.221 I Synced 301/3386257 (0%, 3385956 left)
2025-04-09 12:08:01.311 I Synced 401/3386257 (0%, 3385856 left)
2025-04-09 12:08:01.408 I Synced 501/3386257 (0%, 3385756 left)
2025-04-09 12:08:01.485 I Synced 601/3386257 (0%, 3385656 left)
2025-04-09 12:08:01.574 I Synced 701/3386257 (0%, 3385556 left)
2025-04-09 12:08:01.669 I Synced 801/3386257 (0%, 3385456 left)
2025-04-09 12:08:01.916 I Synced 901/3386257 (0%, 3385356 left)
2025-04-09 12:08:02.278 I Synced 1001/3386257 (0%, 3385256 left)
2025-04-09 12:08:02.624 I Synced 1101/3386257 (0%, 3385156 left)
2025-04-09 12:08:02.670 I Synced 1201/3386257 (0%, 3385056 left)
2025-04-09 12:08:02.786 I Synced 1301/3386257 (0%, 3384956 left)
2025-04-09 12:08:02.878 I Synced 1401/3386257 (0%, 3384856 left)
2025-04-09 12:08:02.973 I Synced 1501/3386257 (0%, 3384756 left)
2025-04-09 12:08:03.096 I Synced 1601/3386257 (0%, 3384656 left)
2025-04-09 12:08:03.187 I Synced 1701/3386257 (0%, 3384556 left)
2025-04-09 12:08:03.282 I Synced 1801/3386257 (0%, 3384456 left)
2025-04-09 12:08:03.383 I Synced 1901/3386257 (0%, 3384356 left)
2025-04-09 12:08:03.541 I Synced 2001/3386257 (0%, 3384256 left)
2025-04-09 12:08:03.702 I Synced 2101/3386257 (0%, 3384156 left)
2025-04-09 12:08:03.799 I Synced 2201/3386257 (0%, 3384056 left)
2025-04-09 12:08:03.895 I Synced 2301/3386257 (0%, 3383956 left)
2025-04-09 12:08:03.989 I Synced 2401/3386257 (0%, 3383856 left)
2025-04-09 12:08:04.086 I Synced 2501/3386257 (0%, 3383756 left)
2025-04-09 12:08:04.189 I Synced 2601/3386257 (0%, 3383656 left)
2025-04-09 12:08:04.284 I Synced 2701/3386257 (0%, 3383556 left)
2025-04-09 12:08:04.383 I Synced 2801/3386257 (0%, 3383456 left)
2025-04-09 12:08:04.477 I Synced 2901/3386257 (0%, 3383356 left)
```

Так, щоб почати майнінг або аналізувати транзакції потрібно синхронізувати повний блокчейн Monero — а це кілька мільйонів блоків (понад 150–200 ГБ дискового простору).



Як виявилось це не варіант тому будемо підключатись до зовнішньої ноди і працювати з нею, хоче це обмежить можливості майнингу

```
prosto_acc@Vladyslavs-MacBook-Pro Desktop % curl -LO https://downloads.getmonero.org/cli/monero-mac-x64-v0.18.3.1.tar.bz2

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 68.8M  100 68.8M    0     0  15.0M      0  0:00:04  0:00:04 --:--:-- 17.0M
prosto_acc@Vladyslavs-MacBook-Pro Desktop % tar -xvf monero-mac-x64-v0.18.3.1.tar.bz2

x monero-x86_64-apple-darwin11-v0.18.3.1/
x monero-x86_64-apple-darwin11-v0.18.3.1/ANONYMITY_NETWORKS.md
x monero-x86_64-apple-darwin11-v0.18.3.1/LICENSE
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-ancestry
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-depth
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-export
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-import
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-mark-spent-outputs
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-prune
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-prune-known-spent-data
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-stats
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-blockchain-usage
x monero-x86_64-apple-darwin11-v0.18.3.1/monerod
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-gen-ssl-cert
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-gen-trusted-multisig
x monero-x86_64-apple-darwin11-v0.18.3.1/monero-wallet-cli
```

Підключаємось та створюємо гаманець

```
prosto_acc@Vladyslavs-MacBook-Pro Desktop % cd monero-x86_64-apple-darwin11-v0.18.3.1
prosto_acc@Vladyslavs-MacBook-Pro monero-x86_64-apple-darwin11-v0.18.3.1 % ./monero-wallet-cli --daemon-address node.moneroworld.com:18089

This is the command line monero wallet. It needs to connect to a monero
daemon to work correctly.
WARNING: Do not reuse your Monero keys on another fork, UNLESS this fork has key reuse mitigations built in. Doing so will harm your privacy.

Monero 'Fluorine Fermi' (v0.18.3.1-release)
Logging to ./monero-wallet-cli.log
Specify wallet file name (e.g., MyWallet). If the wallet doesn't exist, it will be created.
Wallet file name (or Ctrl-C to quit): TestWallet
No wallet found with that name. Confirm creation of new wallet named: TestWallet
  (Y/Yes/N/No): y
Generating new wallet...
Enter a new password for the wallet:
Confirm password:
List of available languages for your wallet's seed:
If your display freezes, exit blind with ^C, then run again with --use-english-language-names
0 : Deutsch
1 : English
2 : Español
3 : Français
4 : Italiano
5 : Nederlands
6 : Português
7 : русский язык
8 : 日本語
9 : 简体中文 (中国)
10 : Esperanto
11 : Lojban
Enter the number corresponding to the language of your choice: 1
Generated new wallet: 4AXyDJdowc5ZeFaCGz8VgtWvCFeUbUd1GEfFhFP95CuUSA1NfyTaCyW7fNtycisKuc65u1Yn2K9jy5kZhZTdUsqq3hqTtD1
View key: eb7f27a3195b782c75750776a3d455a184c523972bf5db02e917bca4bb6e890f
**********************************************************************
Your wallet has been generated!
To start synchronizing with the daemon, use the "refresh" command.
Use the "help" command to see a simplified list of available commands.
```

```
NOTE: the following 25 words can be used to recover access to your wallet. Write them down and store them somewhere safe and secure. Please do not store them in
 your email or on file storage services outside of your immediate control.

rodent tissue opposite lids yacht northern furnished greater
talent limits agnostic hold union react randomly cigar
toolbox vials salads lobster rudely roles click darted click
**********************************************************************
Warning: using an untrusted daemon at node.moneroworld.com:18089
Using a third party daemon can be detrimental to your security and privacy
You are strongly encouraged to connect to the Monero network using your own daemon
If you or someone you trust are operating this daemon, you can use --trusted-daemon
Using an untrusted daemon, skipping background mining check
If you are new to Monero, type "welcome" for a brief overview.
Error: wallet failed to connect to daemon: node.moneroworld.com:18089. Daemon either is not started or wrong port was passed. Please make sure daemon is running
 or change the daemon address using the 'set_daemon' command.
Background refresh thread started
[wallet 4AXyDJ (no daemon)]:
```

Тестуємо дефолтні команди

```
[wallet 4AXyDJ (no daemon)]: address
0  4AXyDJdowc5ZeFaCGz8VgtWvCFeUbUd1GEfFhFP95CuUSA1NfyTaCyW7fNtycisKuc65u1Yn2K9jy5kZhZTdUsqq3hqTtD1  Primary address
```

```
[wallet 4AXyDJ (no daemon)]: balance
Currently selected account: [0] Primary account
Tag: (No tag assigned)
Balance: 0.000000000000, unlocked balance: 0.000000000000
```

Вилітала помилка

```
[wallet 4AXyDJ (no daemon)]: refresh
Error: wallet failed to connect to daemon: node.moneroworld.com:18089. Daemon either is not started or wrong port was passed. Please make sure daemon is running or c
nge the daemon address using the 'set_daemon' command.
```

Тому змінив на іншу ноду

```
[wallet 4AXyDJ (no daemon)]: set_daemon opennode.xmr-tw.org:18089 this-is-probably-a-spy-node
Warning: connecting to a non-local daemon without SSL, passive adversaries will be able to spy on you.
Daemon set to opennode.xmr-tw.org:18089, untrusted
[wallet 4AXyDJ (out of sync)]: refresh
Starting refresh...
Refresh done, blocks received: 19265
Currently selected account: [0] Primary account
Tag: (No tag assigned)
Balance: 0.000000000000, unlocked balance: 0.000000000000
[wallet 4AXyDJ]:
```

This-is-probably -a-spy-node дозволяє підключення до віддаленої ноди страх і ризик.

Monero CLI-гаманець з версії 0.18:

- не дозволяє підключення до зовнішніх нод без підтвердження.
- це зроблено з міркувань анонімності, бо публічна нода може спостерігати, які запити ти надсилаєш.

Можна:

- підключитись тимчасово з фразою this-is-probably-a-spy-node
- або підняти власну ноду (якщо хочеш повний контроль і конфіденційність)

Monero вважає майнінг через **віддалену ноду потенційно небезпечним**, бо ти не контролюєш блокчейн повністю. Тому:

```
[wallet 4AXyDJ]: address
0  4AXyDJdowc5ZeFaCGz8VgtWvCFeUbUd1GEfFhFP95CuUSA1NfyTaCyW7fNtycisKuc65u1Yn2K9jy5kZhZTdUsqq3hqTtD1  Primary address
[wallet 4AXyDJ]: start_mining 4AXyDJdowc5ZeFaCGz8VgtWvCFeUbUd1GEfFhFP95CuUSA1NfyTaCyW7fNtycisKuc65u1Yn2K9jy5kZhZTdUsqq3hqTtD1 2
Error: this command requires a trusted daemon. Enable with --trusted-daemon
[wallet 4AXyDJ]: start_mining 4AXyDJdowc5ZeFaCGz8VgtWvCFeUbUd1GEfFhFP95CuUSA1NfyTaCyW7fNtycisKuc65u1Yn2K9jy5kZhZTdUsqq3hqTtD1 2  --trusted-daemon
Error: this command requires a trusted daemon. Enable with --trusted-daemon
[wallet 4AXyDJ]:
```

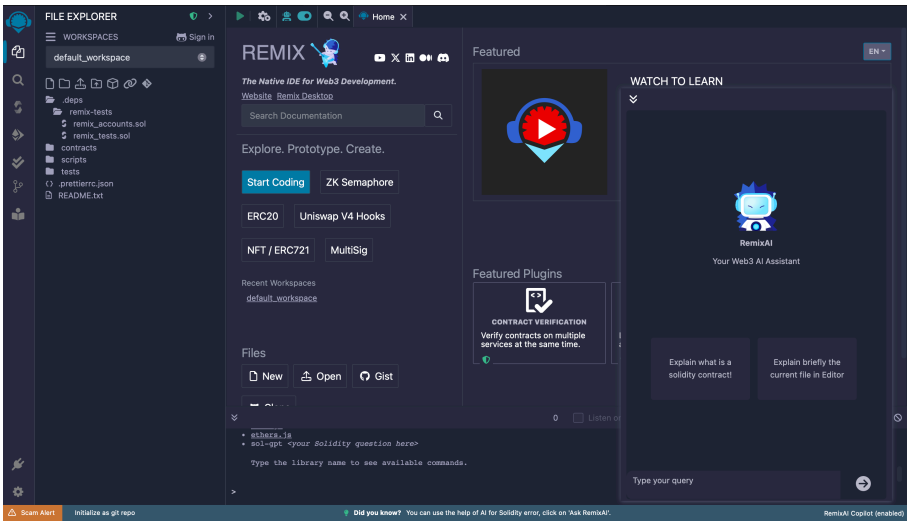
Команда start_mining не працює з публічними (не довіреними) нодами

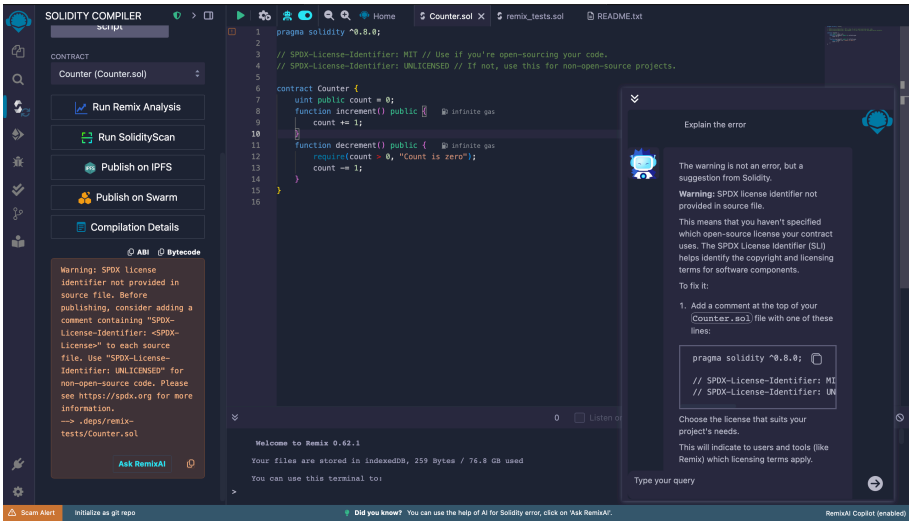
Розгортання та запуск обраного смарт-контракту, підвищення ефективності роботи смарт-контракту з точки зору витрати гасу;

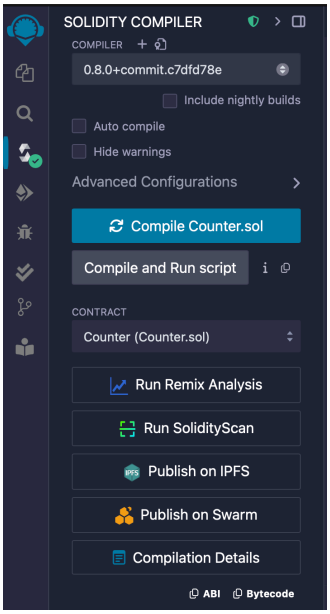
Отже ось код на Solidity (простий лічильник для тестування Gas

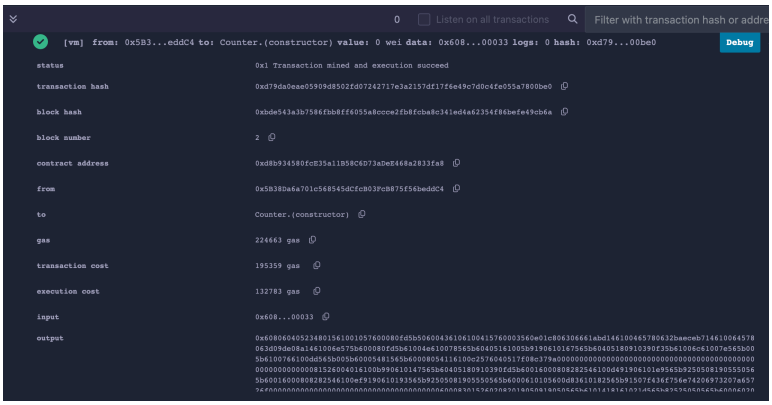
```
pragma solidity ^0.8.0;

contract Counter {

    uint public count = 0;

    function increment() public {

        count += 1;

    }

    function decrement() public {

        require(count > 0, "Count is zero");

        count -= 1;

    }

}
```

Розгортання контракту на Remix (браузерна IDE)


FILE EXPLORER
WORKSPACES
Sign in
default_workspace
.deps
remix-tests
remix_accounts.sol
remix_tests.sol
contracts
scripts
tests
.prettierrc.json
README.txt
Home
REMIX
The Native IDE for Web3 Development.
Website Remix Desktop
Search Documentation
Explore. Prototype. Create.
Start Coding
ZK Semaphore
ERC20
Uniswap V4 Hooks
NFT / ERC721
MultiSig
Recent Workspaces
default_workspace
Files
New
Open
Gist
Featured
EN
WATCH TO LEARN
RemixAI
Your Web3 AI Assistant
Explain what is a solidity contract!
Explain briefly the current file in Editor
Type your query
Featured Plugins
CONTRACT VERIFICATION
Verify contracts on multiple services at the same time.
Type the library name to see available commands.
Scam Alert
Initialize as git repo
Did you know? You can use the help of AI for Solidity error, click on 'Ask RemixAI'.
RemixAI Copilot (enabled)

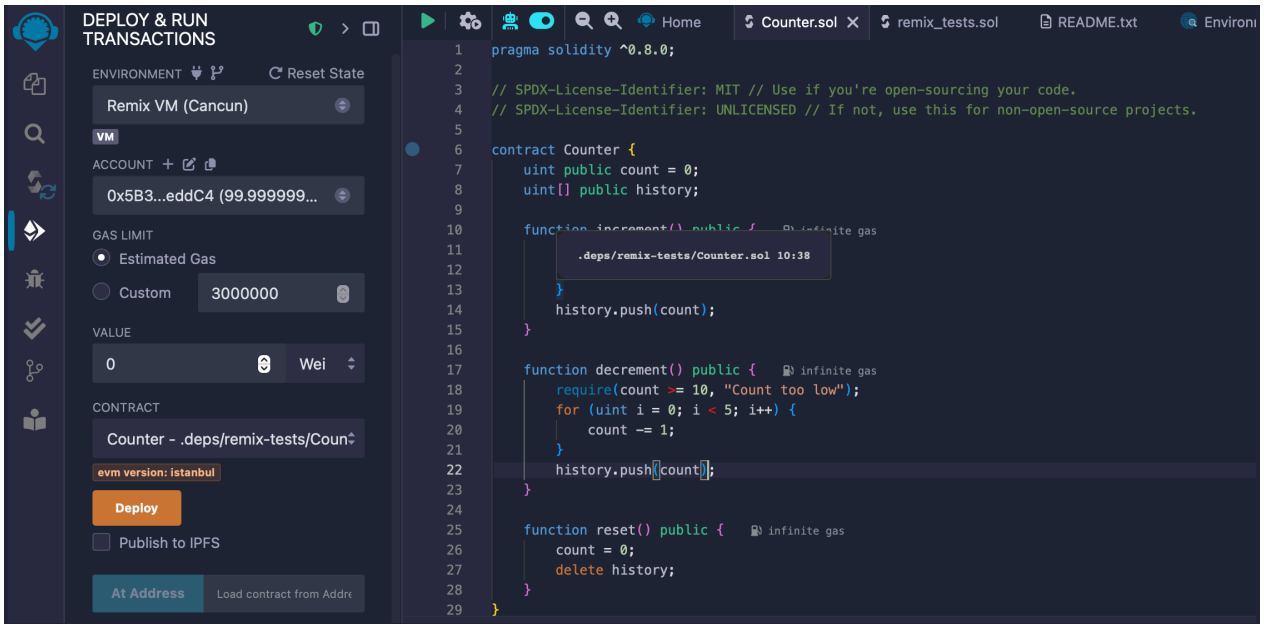

SOLIDITY COMPILER
CONTRACT
Counter (Counter.sol)
Run Remix Analysis
Run SolidityScan
Publish on IPFS
Publish on Swarm
Compilation Details
ABI
Bytecode
Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: UNLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> .deps/remix-tests/Counter.sol
Ask RemixAI
Home
Counter.sol
remix_tests.sol
README.txt
pragma solidity ^0.8.0;
// SPDX-License-Identifier: MIT // Use if you're open-sourcing your code.
// SPDX-License-Identifier: UNLICENSED // If not, use this for non-open-source projects.
contract Counter {
uint public count = 0;
function increment() public {
count += 1;
function decrement() public {
require(count > 0, "Count is zero");
count -= 1;
Welcome to Remix 0.62.1
Your files are stored in indexedDB, 259 Bytes / 76.8 GB used
You can use this terminal to:
Explain the error
The warning is not an error, but a suggestion from Solidity.
Warning: SPDX license identifier not provided in source file.
This means that you haven't specified which open-source license your contract uses. The SPDX License Identifier (SLI) helps identify the copyright and licensing terms for software components.
To fix it:
1. Add a comment at the top of your Counter.sol file with one of these lines:
pragma solidity ^0.8.0;
// SPDX-License-Identifier: MI
// SPDX-License-Identifier: UN
Choose the license that suits your project's needs.
This will indicate to users and tools (like Remix) which licensing terms apply.
Type your query
Scam Alert
Initialize as git repo
Did you know? You can use the help of AI for Solidity error, click on 'Ask RemixAI'.
RemixAI Copilot (enabled)


SOLIDITY COMPILER
COMPILER
0.8.0+commit.c7dfd78e
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Compile Counter.sol
Compile and Run script
CONTRACT
Counter (Counter.sol)
Run Remix Analysis
Run SolidityScan
Publish on IPFS
Publish on Swarm
Compilation Details
ABI
Bytecode



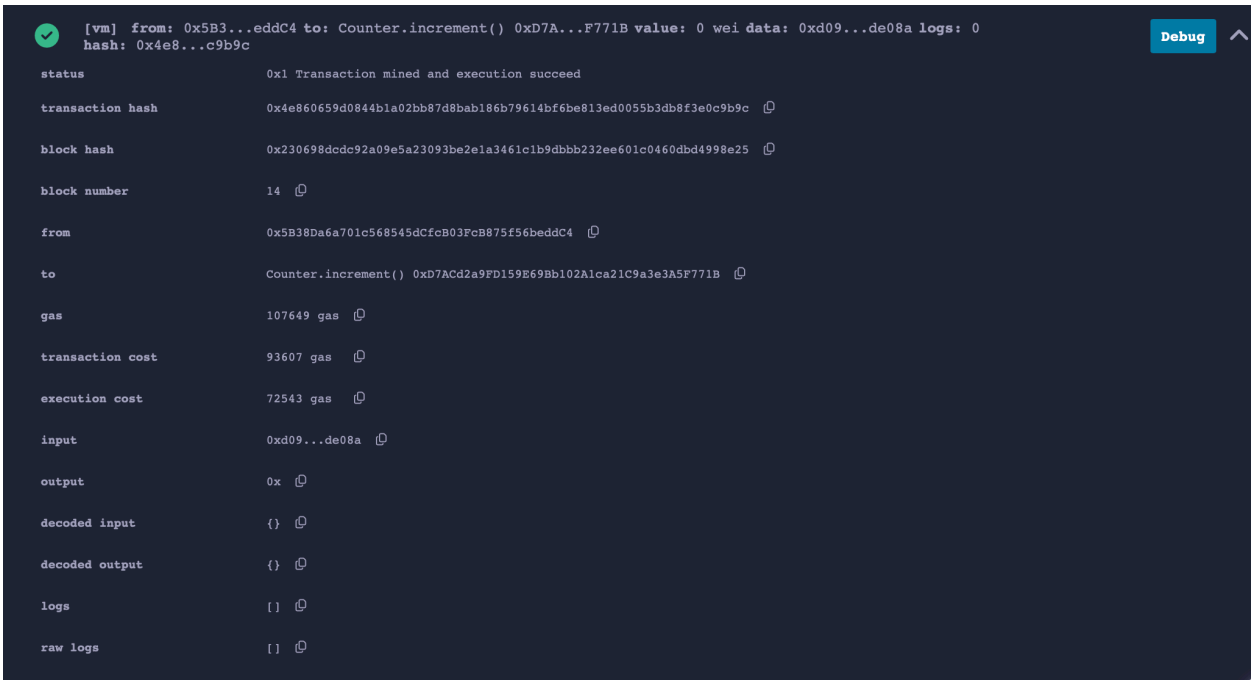
Модифікував код, щоб він потребував більше gas

```
contract Counter {
    uint public count = 0;
    uint[] public history;

    function increment() public {
        for (uint i = 0; i < 10; i++) {
            count += 1;
        }
        history.push(count); // запис у storage — дорога операція
    }

    function decrement() public {
        require(count >= 10, "Count too low");
        for (uint i = 0; i < 5; i++) {
            count -= 1;
        }
        history.push(count);
    }

    function reset() public {
        count = 0;
        delete history; // очищення масиву — теж потребує gas
    }
}
```


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Reset State
Remix VM (Cancun)
VM
ACCOUNT
0x5B3...eddC4 (99.999999...
GAS LIMIT
Estimated Gas
Custom
3000000
VALUE
0
Wei
CONTRACT
Counter - .deps/remix-tests/Coun
evm version: istanbul
Deploy
Publish to IPFS
At Address
Load contract from Addre
Home
Counter.sol
remix_tests.sol
README.txt
Environ
pragma solidity ^0.8.0;
// SPDX-License-Identifier: MIT // Use if you're open-sourcing your code.
// SPDX-License-Identifier: UNLICENSED // If not, use this for non-open-source projects.
contract Counter {
uint public count = 0;
uint[] public history;
infinite gas
.deps/remix-tests/Counter.sol 10:38
history.push(count);
function decrement() public {
infinite gas
require(count >= 10, "Count too low");
for (uint i = 0; i < 5; i++) {
count -= 1;
history.push(count);
function reset() public {
infinite gas
count = 0;
delete history;

0
Listen on all transactions
Filter with transaction hash or address
status
0x1 Transaction mined and execution succeed
transaction hash
0x8469e6a938a0fe986cf189c025c65f933b08e417feddb18b4754ace3ce590ac8
block hash
0x3841552484edc190be6d45dec78c37d12995c663e9e6e3807e091e6ebacff0ae
block number
4
contract address
0xD7ACd2a9FD159E69Bb102A1ca21C9a3e3A5F771B
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
Counter.(constructor)
gas
351373 gas
transaction cost
305541 gas
execution cost
235081 gas
input
0x608...00033
output
0x6080604052348015610010576000080fd5b50600436106100575760003560e01c806306661abd1461005c5780632baeceb71461007a578
063a7a38f0b1461008457806d09de08a146100b4578063d826f88f146100be575b600080fd5b6100646100c8565b604051610071919061
0306565b6040518091039 0f35b6100826100ce565b005b61009e600480360381019061009991906102 6e565b61017a565b6040516100ab9

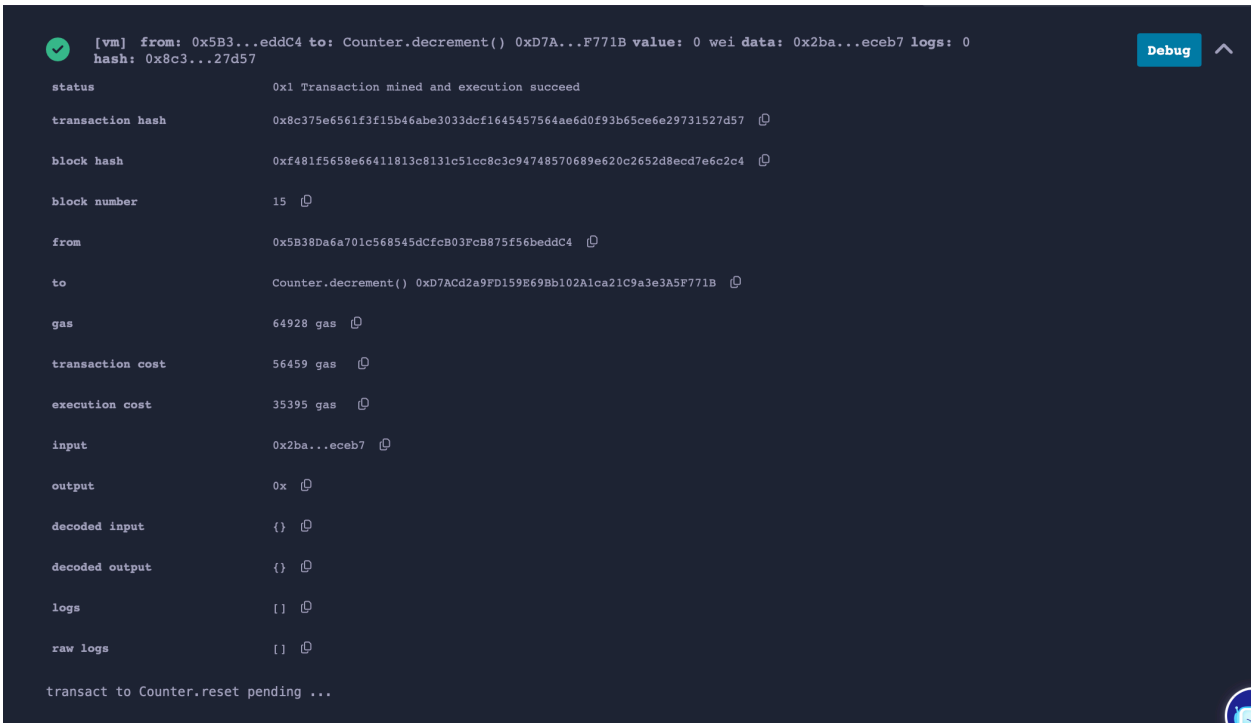
Increment



Після виклику функції increment() у Remix середовищі було виконано транзакцію з наступними параметрами:
• Загальний ліміт gas: **107,649**
• Реальна вартість транзакції: **93,607 gas**
• Вартість виконання (execution cost): **72,543 gas**

Це демонструє значну вартість функції, яка, ймовірно, включає цикл або кілька змін storage. Така операція є ресурсоємною в середовищі EVM.
Для оптимізації можна зменшити кількість змінних, що зберігаються у storage, або уникати циклів.

Decrement



Після виклику функції decrement() у Remix середовищі було виконано транзакцію з наступними параметрами:

- Загальний ліміт gas: **64,928**
- Реальна вартість транзакції: **56,459 gas**
- Вартість виконання (execution cost): **35,395 gas**

Вища вартість пов’язана з перевіркою умови require(...) та зміною значення змінної у storage.
Для зниження витрат можна застосовувати unchecked для арифметичних дій або зменшити логіку перевірок, якщо це не порушує безпеку.

```
[vm] from: 0x5B3...eddC4 to: Counter.reset() 0xD7A...F771B value: 0 wei data: 0xd82...6f88f logs: 0 hash: 0x0f8...493f4   Debug

status                  0x1 Transaction mined and execution succeed
transaction hash        0x0f87ab50fa238727f3522b9e059e6de223ebf999c3219d71b0c2dfca4c5493f4
block hash              0xfff4ca8d82d05addd43e3e2609f1f746d9a111c5612235e0bbaf20c2d78afc19
block number            16
from                    0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                      Counter.reset() 0xD7ACd2a9FD159E69Bb102A1ca21C9a3e3A5F771B
gas                     69893 gas
transaction cost        33261 gas
execution cost          20512 gas
input                   0xd82...6f88f
output                  0x
decoded input           {}
decoded output          {}
logs                    []
raw logs                []
```

Після виклику функції reset() у Remix середовищі було виконано транзакцію з наступними параметрами:
• Загальний ліміт gas: **69,893**
• Реальна вартість транзакції: **33,261 gas**
• Вартість виконання (execution cost): **20,512 gas**

Функція reset() має найменшу вартість виконання серед усіх розглянутих. Вона змінює змінну count і видаляє масив history.
Це демонструє, що прості операції типу delete у storage менш затратні, ніж багаторазові записи.

Функція	`increment()`	`decrement()`	`reset()`
Block #	14	15	16
Gas limit	107,649	64,928	69,893
Tx cost	**93,607**	**56,459**	**33,261**
Exec. cost	**72,543**	**35,395**	**20,512**

Gas Limit - Максимальна кількість gas, яку відправник дозволяє використати для транзакції. Щоб уникнути зависань — наприклад, якщо контракт має нескінченний цикл. Це як “ліміт часу на задачу”: якщо закінчився — транзакція скасовується, але за gas ти все одно платиш.

Transaction Cost (Вартість транзакції) - Це реально витрачена кількість gas на всю транзакцію, включаючи:

- виконання коду (execution)
- фіксовану вартість транзакції (наприклад, 21,000 gas для простої)
- можливі записи в лог, виклики інших контрактів тощо

Execution Cost (Вартість виконання) - Це частина transaction cost — тільки обчислювальна частина, тобто виконання коду смарт-контракту.Не включає фіксовані накладні витрати, як:

- вартість виклику самої транзакції (базові 21,000 gas)
- підготовку даних
- передачу ether

Аналіз ісуючого контракту

```
pragma solidity >=0.4.22 <0.6.0;

contract SimpleAuction {
    // Parameters of the auction. Times are either
    // absolute unix timestamps (seconds since 1970-01-01)
    // or time periods in seconds.
    address payable public beneficiary;
    uint public auctionEndTime;

    // Current state of the auction.
    address public highestBidder;
    uint public highestBid;

    // Allowed withdrawals of previous bids
    mapping(address => uint) pendingReturns;

    // Set to true at the end, disallows any change.
    // By default initialized to `false`.
    bool ended;

    // Events that will be emitted on changes.
    event HighestBidIncreased(address bidder, uint amount);
    event AuctionEnded(address winner, uint amount);

    // The following is a so-called natspec comment,
    // recognizable by the three slashes.
    // It will be shown when the user is asked to
    // confirm a transaction.
```

```
    /// Create a simple auction with `_biddingTime`
    /// seconds bidding time on behalf of the
    /// beneficiary address `_beneficiary`.
    constructor(
        uint _biddingTime,
        address payable _beneficiary
    ) public {
        beneficiary = _beneficiary;
        auctionEndTime = now + _biddingTime;
    }

    /// Bid on the auction with the value sent
    /// together with this transaction.
    /// The value will only be refunded if the
    /// auction is not won.
    function bid() public payable {
        // No arguments are necessary, all
        // information is already part of
        // the transaction. The keyword payable
        // is required for the function to
        // be able to receive Ether.

        // Revert the call if the bidding
        // period is over.
        require(
            now <= auctionEndTime,
            "Auction already ended."
        );

        // If the bid is not higher, send the
        // money back.
        require(
            msg.value > highestBid,
            "There already is a higher bid."
        );

        if (highestBid != 0) {
            // Sending back the money by simply using
            // highestBidder.send(highestBid) is a security risk
            // because it could execute an untrusted contract.
            // It is always safer to let the recipients
            // withdraw their money themselves.
            pendingReturns[highestBidder] += highestBid;
        }
        highestBidder = msg.sender;
        highestBid = msg.value;
        emit HighestBidIncreased(msg.sender, msg.value);
    }

    /// Withdraw a bid that was overbid.
    function withdraw() public returns (bool) {
        uint amount = pendingReturns[msg.sender];
        if (amount > 0) {
            // It is important to set this to zero because the recipient
            // can call this function again as part of the receiving call
            // before `send` returns.
            pendingReturns[msg.sender] = 0;

            if (!msg.sender.send(amount)) {
                // No need to call throw here, just reset the amount owing
```

```
                pendingReturns[msg.sender] = amount;
                return false;
            }
        }
        return true;
    }

    /// End the auction and send the highest bid
    /// to the beneficiary.
    function auctionEnd() public {
        // It is a good guideline to structure functions that interact
        // with other contracts (i.e. they call functions or send Ether)
        // into three phases:
        // 1. checking conditions
        // 2. performing actions (potentially changing conditions)
        // 3. interacting with other contracts
        // If these phases are mixed up, the other contract could call
        // back into the current contract and modify the state or cause
        // effects (ether payout) to be performed multiple times.
        // If functions called internally include interaction with external
        // contracts, they also have to be considered interaction with
        // external contracts.

        // 1. Conditions
        require(now >= auctionEndTime, "Auction not yet ended.");
        require(!ended, "auctionEnd has already been called.");

        // 2. Effects
        ended = true;
        emit AuctionEnded(highestBidder, highestBid);

        // 3. Interaction
        beneficiary.transfer(highestBid);
    }
}
```

Поради(хоча я думаю що розробники Solidity, які це писали краще знають і мої оптимізації не дуже ефективні, або навіть ламають якусь передбачену логіку, але сподіваюсь це не так)

1) Оновити версію компілятора

pragma solidity ^0.8.0 наприклад

2) Заміна now на block.timestamp (**n**ow — застарілий синонім, більше не рекомендується.

auctionEndTime = block.timestamp + _biddingTime;

...

require(block.timestamp <= auctionEndTime, "Auction already ended.");

require(block.timestamp >= auctionEndTime, "Auction not yet ended.");

3) Оптимізація withdraw() (заміна send на call)

```
(bool success, ) = msg.sender.call{value: amount}("");

if (!success) {

   pendingReturns[msg.sender] = amount;

   return false;
```

4) Позначення withdraw() як external - Функція викликається ззовні → external економить gas.

```
function withdraw() external returns (bool)
```

ВИСНОВОК

У ході лабораторної роботи було розгорнуто та протестовано власний смарт-контракт у середовищі Remix. Основна увага приділялася аналізу функцій контракту та дослідженню того, як різні дії впливають на споживання обчислювальних ресурсів у мережі Ethereum. Було використано базові функції (`increment`, `decrement`, `reset`), кожна з яких має різний рівень складності та впливу на пам'ять `storage`, що, в свою чергу, відображається у витратах gas.

Під час аналізу витрат було виявлено, що найбільшу кількість gas споживають операції, які містять цикли, запис у `storage` або виконання перевірок (`require`). Наприклад, функція `increment`, яка містила цикл і додаткові записи в масив, показала найвищий показник використаного gas. Функція `reset`, навпаки, мала найнижчі витрати, оскільки просто очищує змінні без додаткових перевірок чи ітерацій.

Також було вивчено вплив архітектурних рішень на ефективність контракту. Було застосовано сучасні підходи до оптимізації, як-от використання `call` замість `send`, обмеження доступу через `modifier`, та перехід на нову версію компілятора Solidity 0.8.x. Це не лише підвищило безпеку контракту, а й дозволило краще контролювати обчислювальні витрати.

Загалом, лабораторна робота дозволила на практиці оцінити важливість правильного проєктування смарт-контрактів, а також продемонструвала, як навіть прості дії можуть спричинити суттєві витрати ефіру в реальній мережі. Отримані знання можуть бути застосовані при створенні безпечних та ефективних децентралізованих додатків.