

# Лабораторная работа 1.

## Часть 1

### Установка и настройка инструментов для работы с Ethereum. Работа с приватной сетью блокчейна Ethereum с помощью geth

#### Теоретический блок

На сегодняшний день **Ethereum** считается самой популярной и полностью работоспособной платформой для создания децентрализованных приложений на основе блокчейн-технологии.

В основе работы каждого блокчейна лежит некоторый алгоритм консенсуса, согласно которому происходит поддержание целостности и безопасности этих распределенных систем. Существует несколько типов алгоритмов консенсуса, среди которых наиболее распространенными являются PoW и PoS. Также встречаются случаи их комбинаций или модификаций (DPoS, LPoS).

В основе работы блокчейна Ethereum лежит алгоритм консенсуса **Proof of Work**. Это самый первый алгоритм консенсуса, который также используется блокчейном Bitcoin. Алгоритм консенсуса PoW требует от пользователей совершения операций хеширования различной сложности. После нахождения правильного решения для блока пользователь получает вознаграждение. Данный процесс называется майнингом. **За майнинг аккаунту начисляется криптовалюта.** Она нужна для того, чтобы совершать транзакции, так как каждая транзакция имеет свою цену. Чем выше цена, тем больше вероятность, что она будет включена в блок первой. Прежде чем попасть в блок, транзакция рассылается по узлам. Следовательно, данные, хранящиеся в ней доступны для просмотра всем участникам блокчейн-сети. Хакеры могут воспользоваться данной особенностью работы системы и попытаться повлиять на исход работы смарт-контракта. Им достаточно успеть вмайнить свою транзакцию раньше, задав ей более высокую цену. Данная атака может быть использована при создании лотереи, где можно перехватить транзакцию с финальным числом и заранее узнать правильный ответ, до того, как он вступит в силу. Решением является шифрование данных.

Однако, несмотря на то, что PoW может противостоять атакам DDoS и спаму, данный алгоритм считают очень затратным, так как майнинг требует большого количества времени и ресурсов. Поэтому Ethereum планирует перейти на **PoS (Proof of Stake)** для увеличения масштабируемости сети.

Различают **приватные (частные) и публичные блокчейны**. Публичный блокчейн: к сети может подключиться любой желающий. Участник имеет возможность читать, добавлять записи, а также участвовать в процессах блокчейна. Такие системы отличаются децентрализацией и отсутствием какого-либо стороннего контроля над блокчейном, а также защитой блоков от изменений после подтверждения проверки в системе. Приватный блокчейн: сеть, которая требует разрешение для доступа пользователя к

блокам, накладывает разные ограничения на то, кто именно может стать участником сети и в каких операциях может принимать участие.

Работа с Ethereum возможна через большое количество клиентов. Однако самым популярным и удобным является клиент **geth**, хоть он и не имеет пользовательского интерфейса, и вся работа происходит с помощью консоли. Надо отметить, что работа с консоли в **geth** достаточно удобна. Для того, чтобы разобраться, как в ней работать, не требуется много усилий и времени. Поэтому многие пользователи после небольшого опыта работы с **geth**, отказываются от работы с клиентами с графическим интерфейсом.

Для развертывания узла главной блокчейн-сети достаточно одной команды **geth**. После её запуска сразу начинается синхронизация блоков. Чаще всего для начальной разработки и тестирования смарт-контрактов главная сеть не требуется. Поэтому в **geth** существуют альтернативные варианты запуска, так называемой, приватной сети блокчейн.

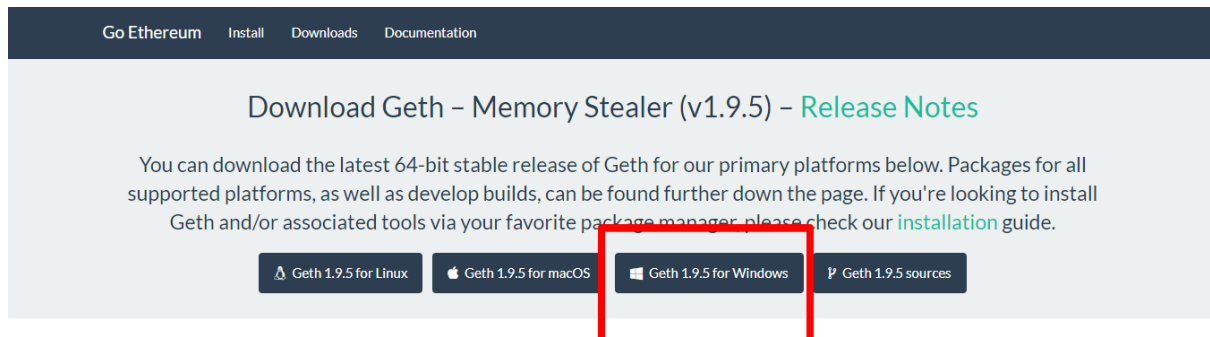
Создание **приватного блокчейна** на базе Ethereum – процедура довольно несложная, однако достаточно ответственная, так как заранее необходимо продумать параметры конфигурации, такие как сложность сети. Прежде всего необходимо создать первичный блок. А при инициализации блокчейна указать уникальный **идентификатор сети**, её **адрес и путь**, где лежит **файл конфигурации первичного блока** - файл с названием **genesis.json**. Нужно быть предельно внимательным при запуске узла приватной сети и обращать внимание на параметры командной строки (также называемые «флаги»), которые необходимы для настройки сети таким образом, чтобы она была недоступна для подключения других узлов извне, а также для определения набора API-интерфейсов, которые необходимы для доступа через RPC.

В каждой сети должен быть **по крайней мере один аккаунт**. Он необходим для майнинга блоков, то есть присоединения блоков к общей цепочке. После запуска первого узла сети рекомендуется сразу создать «главный» аккаунт, который отвечает за майнинг на данном узле. Надо понимать, что **на каждого отдельного пользователя создается аккаунт, к которому привязывается счет с криптовалютой. Однако не каждый пользователь должен обязательно иметь отдельный узел в сети**. На этом основываются некоторые веб-приложения, базирующиеся на технологии блокчейн. Взаимодействие пользователя с сетью осуществляется через веб-интерфейс, который соединен с узлом сети. Данный узел может быть развернут на внешнем сервере какой-либо конкретной организации.

## Практический блок

### 1. Устанавливаем клиент ethereum (geth)

#### 1.1 Скачиваем с сайта последнюю версию <https://geth.ethereum.org/downloads/>

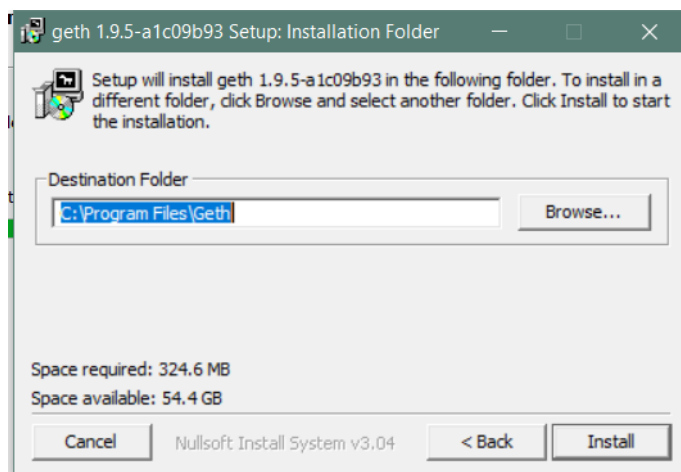
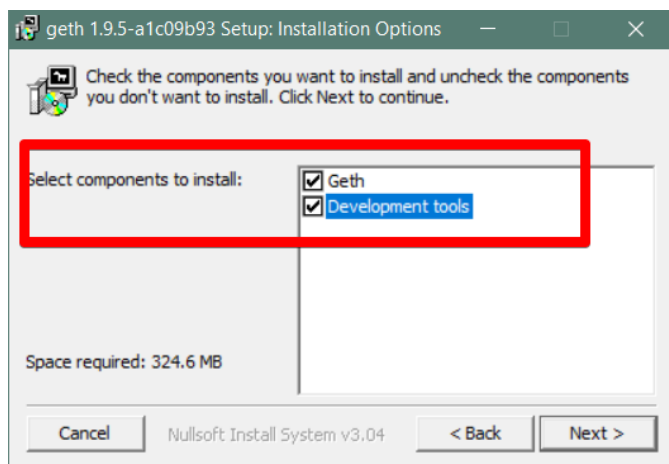


#### Specific Versions

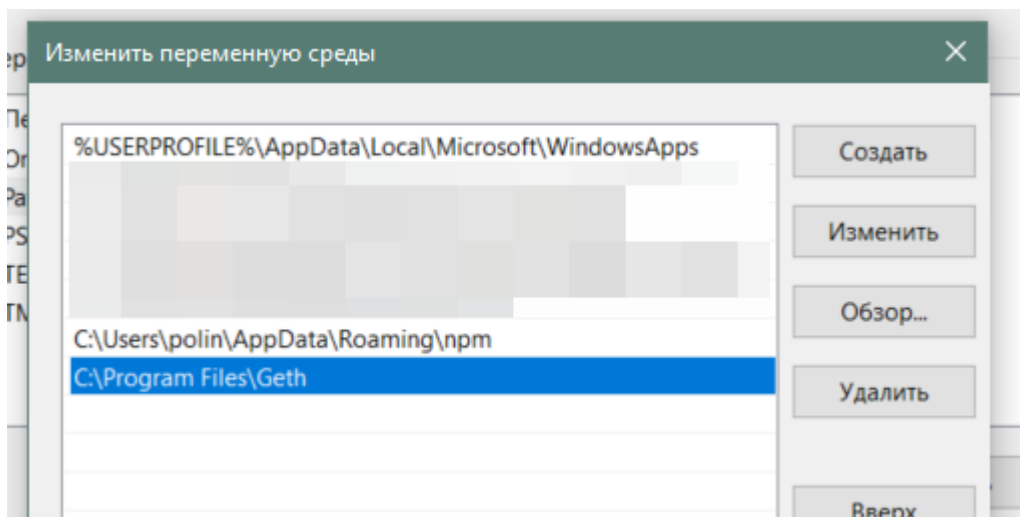
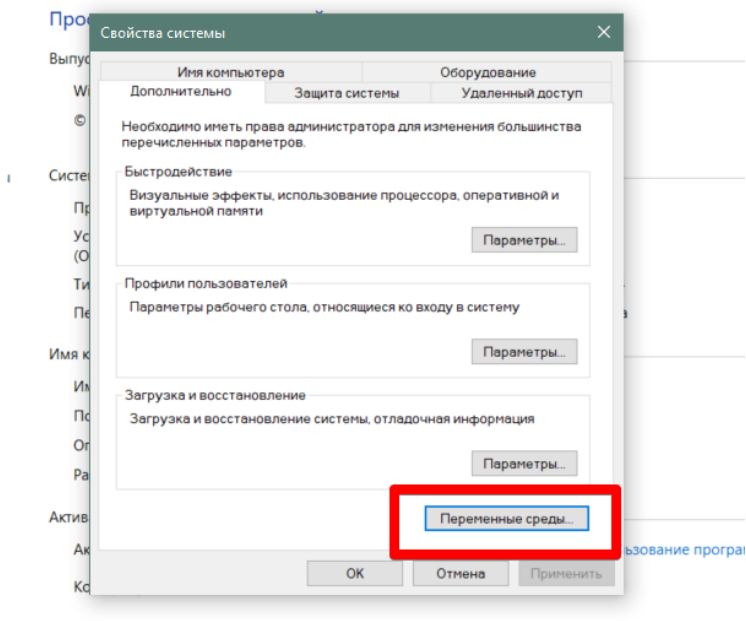
If you're looking for a specific release, operating system or architecture, below you will find:

- All stable and develop builds of Geth and tools
- Archives for non-administrator architectures

#### 1.2 Устанавливаем geth, желательно оставляя все настройки по умолчанию



Изменяем переменную PATH.



1.3 Скачиваем и устанавливаем Node.js: <https://nodejs.org/en/download/>

2. Настраиваем первый узел. Создаем файл с именем genesis.json и содержимым:

**!!! параметр chainId должен быть в формате «11\*», где \* - это номер студента по списку.**

```
{
  "config": {
    "chainId": 1107,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "difficulty": "10",
  "gasLimit": "2100000",
  "alloc": {}
}
```

3. Создаем папку для первого узла сети и создаем два аккаунта на этом узле:

3.1 В папке, где лежит конфигурационный файл создаем папку !!! с названием **node1\***, где \* - фамилия студента

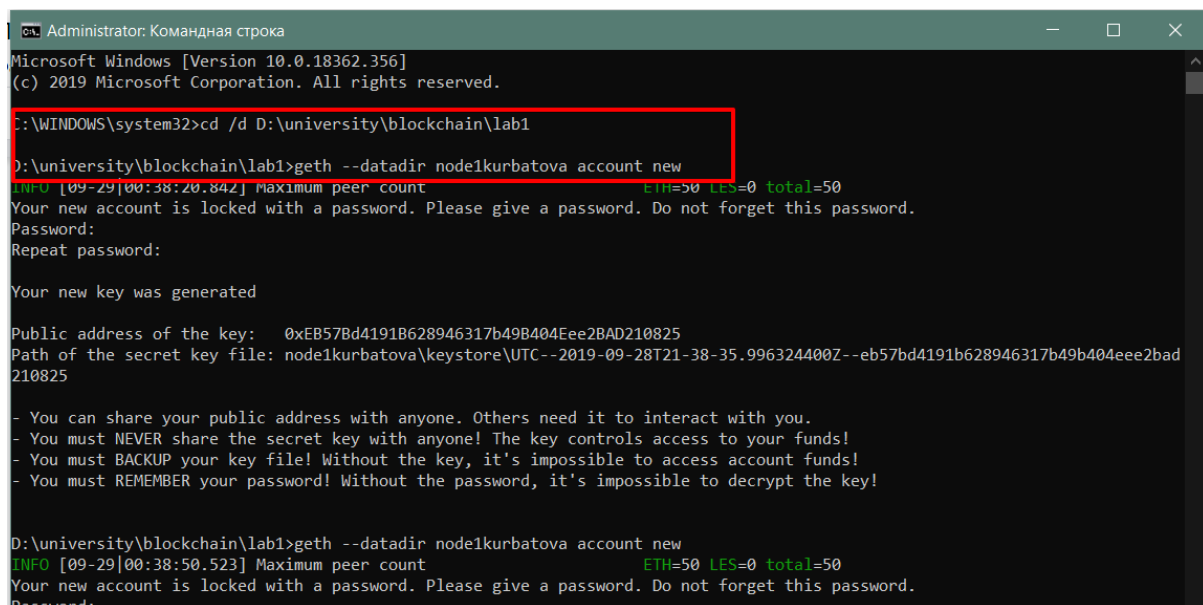
3.2 Запускаем командную строку, переходим в папку, где лежит наш конфигурационный файл и запускаем следующую команду (чтобы создать два аккаунта, запускаем команду два раза).

```
geth --datadir node1 account new
```

```
geth --datadir node1 account new
```

В данной команде **node1** – это название созданной нами ранее папки. Здесь будут храниться файлы первой ноды нашего приватного блокчейна.

Для создания аккаунта в сети вам понадобится ввести пароль. (ЗАПОМНИТЕ ЕГО)



```
Administrator: Командная строка
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd /d D:\university\blockchain\lab1
D:\university\blockchain\lab1>geth --datadir node1kurbatova account new
INFO [09-29|00:38:20.842] Maximum peer count          ETH=50 LES=0 total=50
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
Repeat password:
Your new key was generated

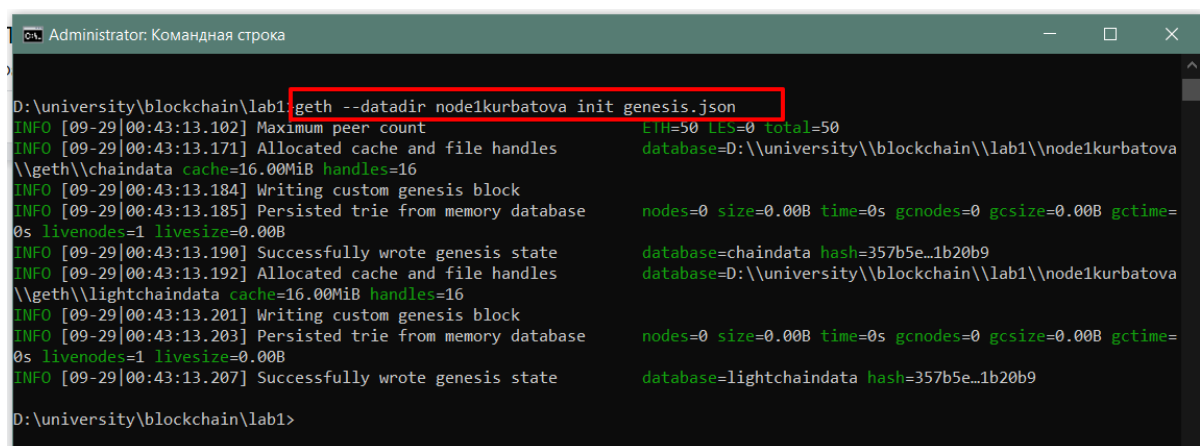
Public address of the key:  0xEb57Bd4191B628946317b49B404Eee2BAD210825
Path of the secret key file: node1kurbatova\keystore\UTC--2019-09-28T21-38-35.996324400Z--eb57bd4191b628946317b49b404eee2bad210825

- You can share your public address with anyone. Others need it to interact with you.
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!

D:\university\blockchain\lab1>geth --datadir node1kurbatova account new
INFO [09-29|00:38:50.523] Maximum peer count          ETH=50 LES=0 total=50
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
```

4. Инициализируем первый узел с помощью команды:

```
geth --datadir node1 init genesis.json
```



```
Administrator: Командная строка
D:\university\blockchain\lab1>geth --datadir node1kurbatova init genesis.json
INFO [09-29|00:43:13.102] Maximum peer count          ETH=50 LES=0 total=50
INFO [09-29|00:43:13.171] Allocated cache and file handles database=D:\university\blockchain\lab1\node1kurbatova
\\geth\\chaindata cache=16.00MiB handles=16
INFO [09-29|00:43:13.184] Writing custom genesis block
INFO [09-29|00:43:13.185] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcsz=0.00B gctime=
0s livenodes=1 liveness=0.00B
INFO [09-29|00:43:13.190] Successfully wrote genesis state database=chaindata hash=357b5e..1b20b9
INFO [09-29|00:43:13.192] Allocated cache and file handles database=D:\university\blockchain\lab1\node1kurbatova
\\geth\\lightchaindata cache=16.00MiB handles=16
INFO [09-29|00:43:13.201] Writing custom genesis block
INFO [09-29|00:43:13.203] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcsz=0.00B gctime=
0s livenodes=1 liveness=0.00B
INFO [09-29|00:43:13.207] Successfully wrote genesis state database=lightchaindata hash=357b5e..1b20b9

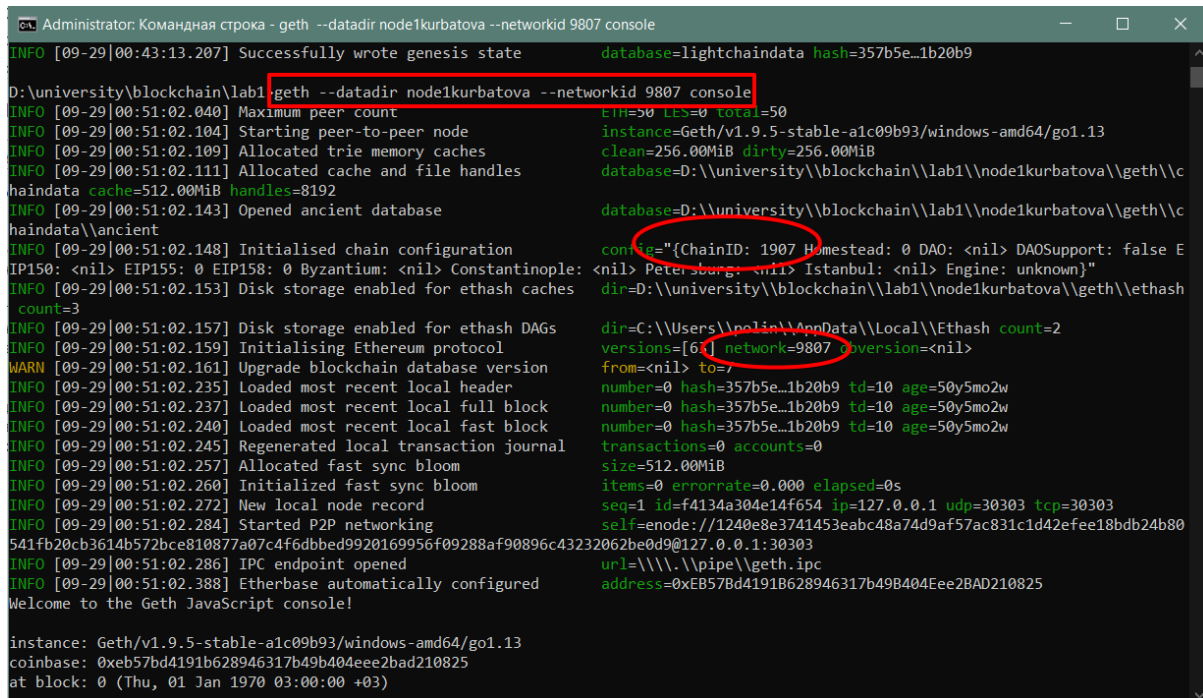
D:\university\blockchain\lab1>
```

5. Запускаем geth console. При запуске указываем уникальный идентификатор сети (networkid).

!!! Идентификатор сети должен быть вида 98\*, где \* - это номер студента по списку.

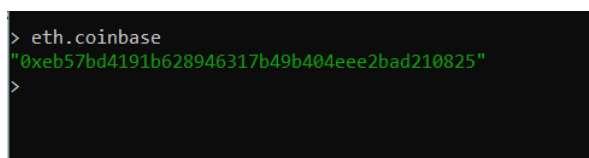
Итоговая команда может иметь вид:

geth --datadir node1 --networkid 9807 console



6. Начинаем майнить!

6.1 Проверяем, что первый созданный аккаунт является аккаунтом для майнинга. Для этого используем команду: *eth.coinbase*.



6.2 Запускаем майнинг:

*miner.start(1).*

Когда мы впервые начинаем майнить, создается DAG. Это может занять некоторое количество времени.

6.3 Ждём, когда закончится генерация DAG и в консоли начнёт отображаться следующая фраза «Commit new mining work».

После чего опять ждём некоторое время, затем останавливаем майнинг командой *miner.stop()*

7. Проверяем баланс первого и второго аккаунта:

7.1 Баланс первого аккаунта можно проверить командами

`eth.getBalance(eth.coinbase)`

7.2 Баланс второго аккаунта должен быть нулевым

`eth.getBalance(eth.accounts[0])`

```
INFO [09-29|01:05:24.319] Generating DAG in progress epoch=1 percentage=97 elapsed=3m59.057s
INFO [09-29|01:05:27.313] Generating DAG in progress epoch=1 percentage=98 elapsed=4m2.051s
INFO [09-29|01:05:30.349] Generating DAG in progress epoch=1 percentage=99 elapsed=4m5.087s
INFO [09-29|01:05:30.526] Generated ethash verification cache epoch=1 elapsed=4m5.263s
INFO [09-29|01:06:07.749] New local node record seq=3 id=f4134a304e14f654 ip=178.120.254.140 udp=1024 tcp=3033
INFO [09-29|01:06:07.833] New local node record seq=4 id=f4134a304e14f654 ip=178.120.254.140 udp=30303 tcp=3033
INFO [09-29|01:06:07.845] New local node record seq=5 id=f4134a304e14f654 ip=178.120.254.140 udp=1024 tcp=3033
INFO [09-29|01:06:07.952] New local node record seq=6 id=f4134a304e14f654 ip=178.120.254.140 udp=1024 tcp=3033
> eth.getBalance(eth.accounts[0])
9500000000000000000
> eth.getBalance(eth.accounts[1])
0
>
```

8. Переведем часть Эфириумов на второй аккаунт

8.1 Для отправки транзакции необходимо разблокировать первый аккаунт командой:

`personal.unlockAccount(eth.accounts[0])`

Необходимо будет ввести пароль, который придумывали при создании аккаунта

```
> personal.unlockAccount(eth.accounts[0])
Unlock account 0xeb57bd4191b628946317b49b404eee2bad210825
Password:
true
>
```

8.2 Отправляем транзакцию – переводим с первого аккаунта на второй 2 Wei:

`eth.sendTransaction({from: eth.accounts[0], to: eth.accounts[1], value: web3.toWei(2, "ether")})`

```
> eth.sendTransaction({from: eth.accounts[0], to: eth.accounts[1], value: web3.toWei(2, "ether")})
INFO [09-29|01:14:04.764] Setting new local account address=0xeb57bd4191b628946317b49b404eee2bad210825
INFO [09-29|01:14:04.768] Submitted transaction fullhash=0xda7fd7d0e8d8e71285997900883a339c4fd3050ffcd1b13723360cd49f120c82
9f120c82 recipient=0xc43Cfd0E23F8bCb2Ea2EeB8c98f2e65822Fe4df1
"0xda7fd7d0e8d8e71285997900883a339c4fd3050ffcd1b13723360cd49f120c82"
>
```

Наподобие фиатной или реальной (не виртуальной) валюты, криптовалюту можно разбить на более мелкие единицы, как рубль разбивается на копейки. Эфир (ETH) - валюта, на которой работает система Эфириума - также можно разделить на единицы помельче.

Представьте, что наименьшая частица эфира - wei - это рубль. Чтобы получить один целый ETH нужно очень много **Wei**. **10<sup>18</sup>**, если быть точнее.

10<sup>9</sup> Wei называется **Gwei**. Данный термин в основном используется, когда речь идет о газе (единица измерения стоимости комиссии при транзакциях в блокчейне). Вместо того, чтобы говорить, что 1 единица газа стоит 0.000000001 ETH, удобнее использовать описание "1 gwei".



8.3 Для того, чтобы эта транзакция зарегистрировалась в сети ее надо вмайнить:

`miner.start(1)`

`miner.stop()`

```
> miner.start(1)
INFO [09-29|01:14:57.639] Updated mining threads          threads=1
INFO [09-29|01:14:57.642] Transaction pool price threshold updated price=1000000000
null
INFO [09-29|01:14:57.646] Commit new mining work          > number=20 sealhash=e20511..946290 uncles=0 txs=0 gas=0 fees=0 elapsed=0s
INFO [09-29|01:14:57.656] Commit new mining work          number=20 sealhash=fa905c..f9a1ee uncles=0 txs=1 gas=21000 fees=2.1e-05 elapsed=9.972ms
INFO [09-29|01:14:58.216] Successfully sealed new block    number=20 sealhash=fa905c..f9a1ee hash=965310..070de3 elapsed=561.498ms
INFO [09-29|01:14:58.222]   block reached canonical chain  number=13 hash=ba221c..820f27
INFO [09-29|01:14:58.225]   mined potential block          number=20 hash=965310..070de3
INFO [09-29|01:14:58.229] Commit new mining work          number=21 sealhash=8f437b..9d0b16 uncles=0 txs=0 gas=0 fees=0 elapsed=6.982ms
INFO [09-29|01:15:00.128] Successfully sealed new block    number=21 sealhash=8f437b..9d0b16 hash=2a07c3..2b849e elapsed=1.905s
INFO [09-29|01:15:00.133]   block reached canonical chain  number=14 hash=333e8a..c45a23
INFO [09-29|01:15:00.136]   mined potential block          number=21 hash=2a07c3..2b849e
INFO [09-29|01:15:00.139] Commit new mining work          number=22 sealhash=6fc70f..6d5b00 uncles=0 txs=0 gas=0 fees=0 elapsed=5.997ms
INFO [09-29|01:15:02.496] Successfully sealed new block    number=22 sealhash=6fc70f..6d5b00 hash=65f65a..39afcb elapsed=2.362s
INFO [09-29|01:15:02.501]   block reached canonical chain  number=15 hash=029142..b67573
INFO [09-29|01:15:02.504]   mined potential block          number=22 hash=65f65a..39afcb
INFO [09-29|01:15:02.507] Commit new mining work          number=23 sealhash=328af4..7141a2 uncles=0 txs=0 gas=0 fees=0 elapsed=4.988ms
INFO [09-29|01:15:03.613] Successfully sealed new block    number=23 sealhash=328af4..7141a2 hash=f2bfcf..663e00 elapsed=1.111s
INFO [09-29|01:15:03.618]   block reached canonical chain  number=16 hash=622af2..8dc413
INFO [09-29|01:15:03.622]   mined potential block          number=23 hash=f2bfcf..663e00
INFO [09-29|01:15:03.627] Commit new mining work          number=24 sealhash=072538..d3d024 uncles=0 txs=0 gas=0 fees=0 elapsed=0.076ms
> miner.stop()
INFO [09-29|01:15:07.153] Successfully sealed new block    number=24 sealhash=072538..d3d024 hash=fc2cf0..cabd9d elapsed=3.535s
INFO [09-29|01:15:07.159]   block reached canonical chain  number=17 hash=5e9df5..ff8141
INFO [09-29|01:15:07.161]   mined potential block          number=24 hash=fc2cf0..cabd9d
INFO [09-29|01:15:07.165] Commit new mining work          number=25 sealhash=59f140..5a6546 uncles=0 txs=0 gas=0 fees=0 elapsed=5.984ms
null
```

8.4 После этого проверяем баланс второго аккаунта. Результат следующей команды должен быть равен 20000000000000000000:

`eth.getBalance(eth.accounts[1])`

```
> miner.stop()
INFO [09-29|01:15:07.153] Successfully sealed new block    number=24 sealhash=072538..d3d024 hash=fc2cf0..cabd9d elapsed=3.535s
INFO [09-29|01:15:07.159]   block reached canonical chain  number=17 hash=5e9df5..ff8141
INFO [09-29|01:15:07.161]   mined potential block          number=24 hash=fc2cf0..cabd9d
INFO [09-29|01:15:07.165] Commit new mining work          number=25 sealhash=59f140..5a6546 uncles=0 txs=0 gas=0 fees=0 elapsed=5.984ms
null
>
> eth.getBalance(eth.accounts[1])
20000000000000000000
>
```

8.5 !!! Самостоятельно по аналогии переведите со первого аккаунта на второй ещё 4 Wei. Вмайните транзакцию и проверьте баланс.

8.6 !!! Затем наоборот переведите со второго аккаунта на первый 1 Wei. Вмайните транзакцию и проверьте баланс.

9. Выйдите из geth командой `exit`

```
> exit
INFO [09-29|01:18:57.563] IPC endpoint closed              url=\\\\.\\pipe\\geth.ipc
INFO [09-29|01:18:57.567] Writing cached state to disk     block=24 hash=fc2cf0..cabd9d root=c09d18..9e6e1d
INFO [09-29|01:18:57.571] Persisted trie from memory database nodes=3 size=410.00B time=990.6µs gcnodes=0 gctime=0s livenodes=28 liveness=3.79KiB
INFO [09-29|01:18:57.575] Writing cached state to disk     block=23 hash=f2bfcf..663e00 root=e2743c..f0705c
INFO [09-29|01:18:57.578] Persisted trie from memory database nodes=2 size=263.00B time=0s gcnodes=0 gctime=0s livenodes=26 liveness=3.53KiB
INFO [09-29|01:18:57.582] Blockchain manager stopped
INFO [09-29|01:18:57.583] Stopping Ethereum protocol
INFO [09-29|01:18:57.585] Ethereum protocol stopped
INFO [09-29|01:18:57.587] Transaction pool stopped
```



## Контрольные вопросы

1. Что такое Ethereum?
2. Приведите по одному примеру проекта (или идеи), когда было бы уместнее использовать приватный / публичный блокчейн.
3. Какой *алгоритм консенсуса* использует Ethereum? Почему он считается затратным?
4. Что такое *geth* и для чего он предназначен?
5. Что такое Wei и Gwei?
6. Что из себя представляет аккаунт в блокчейне Ethereum? Почему, чтобы стать участником сети, не обязательно разворачивать свой собственный узел (ноду) блокчейна?

## Задание

- 1) Изучить теоретический материал
- 2) Выполнить практическую часть лабораторной работы
- 3) Ответить на контрольные вопросы
- 4) Оформить отчёт

**Защита лабораторной работы 1 (часть 1):** отчёт должен быть отправлен на портал. Пункты 8.5 и 8.6 должны быть продемонстрированы преподавателю на паре.

## Общие требования к отчёту

Отчёт должен содержать следующие элементы:

- 1) номер и название лабораторной работы,
- 2) ФИО студента, группу и курс
- 3) выполненные задания практического блока: номер задания, скриншот рабочего окна в соответствии с требованиями и, если необходимо, пояснения студента к скриншоту.
- 4) краткие ответы на контрольные вопросы.

В случае обнаружения плагиата (в том числе и в ответах на контрольные вопросы) отчёт не будет принят преподавателем!