

## CPE 322 – Lab 10

1. I typed the following terminal commands:

```
jstefan1@DESKTOP-6SV6DJ0:~$ cd ~/iot/lesson10
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cat hash_value.py
"""
https://docs.python.org/3/using/cmdline.html#envvar-PYTHONHASHSEED
If PYTHONHASHSEED is not set or set to random, a random value is used to seed the hashes of str and bytes objects.
If PYTHONHASHSEED is set to an integer value, it is used as a fixed seed for generating the hash() of the types covered
by the hash randomization.
Its purpose is to allow repeatable hashing, such as for selftests for the interpreter itself, or to allow a cluster of p
ython processes to share hash values.
The integer must be a decimal number in the range [0,4294967295]. Specifying the value 0 will disable hash randomization
.

https://www.programiz.com/python-programming/methods/built-in/hash
hash(object) returns the hash value of the object (if it has one). Hash values are integers.
They are used to quickly compare dictionary keys during a dictionary lookup.
Numeric values that compare equal have the same hash value even if they are of different types, as is the case for 1 and
1.0.
For objects with custom __hash__() methods, note that hash() truncates the return value based on the bit width of the ho
st machine.
"""

# hash for integer unchanged
print('The hash for 1 is:', hash(1))

# hash for decimal
print('The hash for 1.0 is:', hash(1.0))
print('The hash for 3.14 is:', hash(3.14))

# hash for string
print('The hash for Python is:', hash('Python'))
```

- a.

2. I then ran hash\_value.py file twice. This was the output of the first one:

```
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3 hash_value.py
The hash for 1 is: 1
The hash for 1.0 is: 1
The hash for 3.14 is: 322818021289917443
The hash for Python is: 7200816504800770539
The hash for a tuple of vowels is: 3052239938943905428
The hash for an object of person is: -475442535740400940
```

- a.

- b. This was the output of the second one:

```
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3 hash_value.py
The hash for 1 is: 1
The hash for 1.0 is: 1
The hash for 3.14 is: 322818021289917443
The hash for Python is: 9143956070208902997
The hash for a tuple of vowels is: 7700842040261463254
The hash for an object of person is: 8357547405922890649
```

- c.

- d. The first 3 numbers have the same hash, but Python, a tuple of vowels, and an object of person had different hashes between runs.

3. Then I ran SHA-2 Secure Hash Algorithm by typing the following commands:

```

jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import hashlib
>>> m = hashlib.sha256(b"hello, world")
>>> m.digest_size
32
>>> m.block_size
64
>>> exit()

```

a.

4. Then I ran snakecoin.py by typing the following commands:

```

jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cd ~/iot/lesson10
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cat snakecoin.py
# Gerald Nash, "Let's build the tiniest blockchain in less than 50 lines of Python"
import hashlib as hasher
import datetime as date

# Define what a Snakecoin block is
class Block:
    def __init__(self, index, timestamp, data, previous_hash):
        self.index = index
        self.timestamp = timestamp
        self.data = data
        self.previous_hash = previous_hash
        self.hash = self.hash_block()

    def hash_block(self):
        sha = hasher.sha256()
        sha.update(str(self.index).encode() + str(self.timestamp).encode() + str(self.data).encode() + str(self.previous_hash).encode())
        return sha.hexdigest()

# Generate genesis block
def create_genesis_block():
    # Manually construct a block with
    # index zero and arbitrary previous hash
    return Block(0, date.datetime.now(), "Genesis Block", "0")

# Generate all later blocks in the blockchain
def next_block(last_block):
    this_index = last_block.index + 1

```

a.

```
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3 snakecoin.py
Block #1 has been added to the blockchain!
Hash: bb6dbc42fdf39f7e5485898bee2aad930ff43e95f191c368a4904b0cda393f8

Block #2 has been added to the blockchain!
Hash: b1ad280a382b0890afeed27121279eaf792eb8425d8eade8ec302a35e03a7e86

Block #3 has been added to the blockchain!
Hash: 4b566b39209c0b70e759f5c5e74a2bfaf3740f331e855cc178cda18cc0505a0d

Block #4 has been added to the blockchain!
Hash: 72ac809413f4bd831db231778c89c51ce470036573b82f31778046da29985baf

Block #5 has been added to the blockchain!
Hash: 38bb478b950c4edfed238ad5881257cb4d08b7c9dca214970261e46d7054191a

Block #6 has been added to the blockchain!
Hash: 14fcc07e26405e1c2382ed475d618ca15e02d3a4ce5f100091a68f72fe544cac

Block #7 has been added to the blockchain!
Hash: 6c3168737f5ea98a9480e1741a157711bfe15f6906d331e50bfd94edd955e9ca

Block #8 has been added to the blockchain!
Hash: 70a0ffa881ef20348db469457e449873d2a45f849d1e9a499a39bc036b84481b

Block #9 has been added to the blockchain!
Hash: 30d4483cad404508e61fa936fff4e29ff9681c9c731d1893ad1ee4da92e09743

Block #10 has been added to the blockchain!
Hash: 8e62aaada3dfc5154fbb0832f0f78aed6cb9a6ed6288a8227c550673b502d13a
```

b.

5. Snakecoin-server-full-code.py:

```
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cat snakecoin-server-full-code.py
# Gerald Nash, "Let's Make the Tiniest Blockchain Bigger Part 2: With More Lines of Python"
# Referred to https://www.pythonanywhere.com/forums/topic/12382/ that fixed sha.update() TypeError: Unicode-object
# be encoded before hashing
# Running on http://127.0.0.1:5000/mine (Reload the page to mine and press CTRL+C to quit)
from flask import Flask
from flask import request
import json
import requests
import hashlib as hasher
import datetime as date
from flask import send_from_directory
import os
node = Flask(__name__)

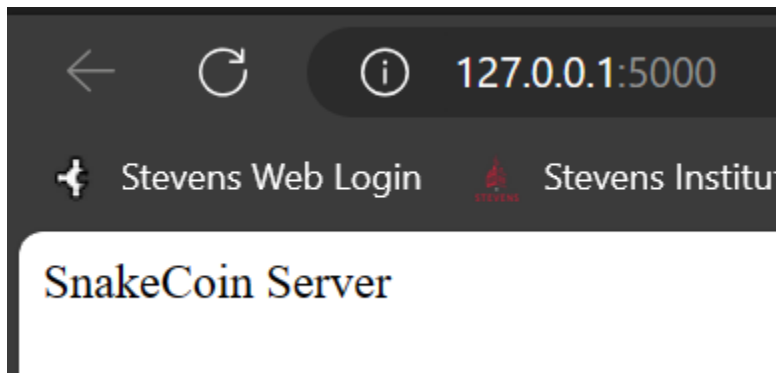
# Define what a Snakecoin block is
class Block:
    def __init__(self, index, timestamp, data, previous_hash):
        self.index = index
        self.timestamp = timestamp
        self.data = data
        self.previous_hash = previous_hash
        self.hash = self.hash_block()

    def hash_block(self):
        sha = hasher.sha256()
        # sha.update(str(self.index) + str(self.timestamp) + str(self.data) + str(self.previous_hash))
        sha.update((str(self.index) + str(self.timestamp) + str(self.data) + str(self.previous_hash)).encode("utf-8"))
        return sha.hexdigest()
```

a.

```
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3 snakecoin-server-full-code.py
* Serving Flask app 'snakecoin-server-full-code'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [22/Nov/2023 16:56:19] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 16:56:19] "GET /favicon.ico HTTP/1.1" 200 -
```

b.



c.

```

jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cd
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ sudo pip3 install markupsafe==2.0.1
[sudo] password for jstefan1:
Collecting markupsafe==2.0.1
  Downloading MarkupSafe-2.0.1-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_12_x86_64.manylinux2018_x86_64.whl (30 kB)
Installing collected packages: markupsafe
  Attempting uninstall: markupsafe
    Found existing installation: MarkupSafe 2.1.3
    Uninstalling MarkupSafe-2.1.3:
      Successfully uninstalled MarkupSafe-2.1.3
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behavior is the source of the following dependency conflicts.
werkzeug 3.0.1 requires MarkupSafe>=2.1.1, but you have markupsafe 2.0.1 which is incompatible.
Successfully installed markupsafe-2.0.1
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$

```

d.

```

jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ curl -H "Content-Type: application/json" -d '{"from": "akjflw", "to": "fjlkadj", "amount": 3}' localhost:5000/txion
Transaction submission successful
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ curl localhost:5000/mine
{"index": 1, "timestamp": "2023-11-22 17:04:19.284993", "data": {"proof-of-work": 18, "transactions": [{"from": "akjflw", "to": "fjlkadj", "amount": 3}, {"from": "network", "to": "q3nf394hjhjg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}]}}, {"hash": "c0dab04fd7b0bb681e4660ed10d5233157c802caff68e1052cdefe42fadba880"}
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$

```

e.

```

jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ cd ~/iot/lesson10
jstefan1@DESKTOP-6SV6DJ0:~/iot/lesson10$ python3 snakecoin-server-full-code.py
* Serving Flask app 'snakecoin-server-full-code'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
New transaction
FROM: b'akjflw'
TO: b'fjlkadj'
AMOUNT: 3

```

f.

```

127.0.0.1 - - [22/Nov/2023 17:04:09] "POST /txion HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:04:19] "GET /mine HTTP/1.1" 200 -

```

```

127.0.0.1 - - [22/Nov/2023 17:05:44.819008] "POST /txion HTTP/1.1" 200 -
{"index": 2, "timestamp": "2023-11-22 17:05:44.819008", "data": {"proof-of-work": 36, "transactions": [{"from": "network", "to": "q3nf394hjhjg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}]}}, {"hash": "c9c3f0b6bdc9cda8a4d33a1bdc8fdcb98fd9796d67a39f6770b7eb4cf4209dc"}

```

g.

## 6. Python Blockchain App:

```

jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ git clone https://github.com/satwikkansal/python_blockchain_app.git
Cloning into 'python_blockchain_app'...
remote: Enumerating objects: 173, done.
remote: Counting objects: 100% (72/72), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 173 (delta 52), reused 56 (delta 43), pack-reused 101
Receiving objects: 100% (173/173), 229.57 KiB | 3.59 MiB/s, done.
Resolving deltas: 100% (84/84), done.
jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ cd ~/python_blockchain_app

```

a.

```
jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ nano node_server.py
jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ python3 node_server.py
* Serving Flask app 'node_server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 850-008-421
```

b.

```
jstefan1@DESKTOP-6SV6DJ0: ~/python_blockchain_app
jstefan1@DESKTOP-6SV6DJ0:~$ cd ~/python_blockchain_app
jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ python3 run_app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 850-008-421
```

c.

127.0.0.1:5000

Stevens Web Login Stevens Institute of... Mail - Joseph Stefa... IoT Product Tracking benjaminparen785... Google Docs kevinwlu (Kevin Lu)... Problems - LeetCode

[Home](#)

## YourNet: Decentralized content sharing

This is a test.

Joseph Post

Request to mine Resync

d.

127.0.0.1:8000/mine

Stevens Web Login Stevens Institute of... Mail

## Block #1 is mined.


e.

← ↻ 127.0.0.1:5000

Stevens Web Login Stevens Institute of... Mail - Joseph Stefa... IoT Product Tracking benjaminparen785... Google Docs kevinwlu (Kevin Lu)... Problems - LeetCode

[Home](#)

## YourNet: Decentralized content sharing



Joseph

Posted at 17:21

This is a test.

f.

```
jstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ python3 run_app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 850-008-421
127.0.0.1 - - [22/Nov/2023 17:20:48] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:21:29] "POST /submit HTTP/1.1" 302 -
127.0.0.1 - - [22/Nov/2023 17:21:29] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:22:53] "GET / HTTP/1.1" 200 -
```

g.

```
^Cjstefan1@DESKTOP-6SV6DJ0:~/python_blockchain_app$ python3 node_server.py
* Serving Flask app 'node_server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 850-008-421
127.0.0.1 - - [22/Nov/2023 17:20:48] "GET /chain HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:21:29] "POST /new_transaction HTTP/1.1" 201 -
127.0.0.1 - - [22/Nov/2023 17:21:29] "GET /chain HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:21:46] "GET /mine HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:22:15] "GET /mine HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:22:19] "GET /mine HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:22:23] "GET /mine HTTP/1.1" 200 -
127.0.0.1 - - [22/Nov/2023 17:22:53] "GET /chain HTTP/1.1" 200 -
```

h.