Homework 1

(20%) 1. Please compare hash function and cryptographic hash function and give an example.

雜湊函式把訊息或資料壓縮成摘要，使得資料量變小，固定輸出的格式，而加密雜湊函數是雜湊函式的一種但是比較著重指單向的雜湊函數，意思是說從輸出值不容易推斷出輸入值是什麼，例如圖書館紀錄書本位置函式就可以是普通的hash function， 方程式表示為

h(書名) = ijk

其中ijk表示第i櫃第j行第k本，而加密雜湊函式可以常用MD5,SHA系列等雜湊函數作為代表，我們不易從結果推出原本的輸入是什麼。

(80%) 2. Peter is a noob in cryptocurrency and would like to get some Ethers. First step for him is to have an Ethereum account. He decides to generate an account and manages the wallet himself so he can understand the principles behind. From the class, he knows the account is created by the following steps:

1. Create a keypair of private/public key
2. public_key = ECDSA(private_key)
3. public_key_hash = Keccak-256(public_key)
4. address = `'0x'` + last 20 bytes of public_key_hash

(30%) a. Can you print the private/public key with hex string representation? Please give us an example.

Private key:
4daa29d6c148717d1ee0eaef9fafd81e3e9b560a4775c4f6ec30a982aaf4079f
Public key:
2ec88f012e9ddab8dc7a3774d9d934ae49e90a0a1672e1815d3cd57f05e39699b3e80d7c12431c3eecf6e3df0517765c521f739b2f5c710fef7cb9e0076ba427
Address:
0x44349d559d34d23d24aba50b7a4f878a8c507fac

(20%) b. In addition, if we don’t want to use the getAddressString() to get the address, how can we obtain the address by hashing the public key?

```
# public_key_hash = Keccak-256(public_key)
keccak = sha3.keccak_256()
keccak.update(pub)

# address = '0x' + last 20 bytes of public_key_hash
address = '0x' + keccak.hexdigest()[24:]
```

(30%) c. There is a file called Keystore that is used to encrypt the private key and save in a JSON file. Can you generate a Keystore with the password “nccu”? You can find the details about Keystore below.

```
{
  "address": "44349d559d34d23d24aba50b7a4f878a8c507fac",
  "crypto": {
    "cipher": "aes-128-ctr",
    "cipherparams": {
      "iv": "e9f504af6dc3e41d3231a17df3e712b4"
    },
    "ciphertext":
"b7e1b1d876dbcd4b5dbfc4656af5384fd722a1261ce1f12f4894624f56d8aa9b",
    "kdf": "pbkdf2",
    "kdfparams": {
      "c": 1000000,
      "dklen": 32,
      "prf": "hmac-sha256",
      "salt": "acb87e00344a0630d20ad1729b8e93ba"
    },
    "mac": "50398f2b3c533c48f397f9a4f536a3f212f58d6fc4b7bcbec375fe154de20b20"
  },
  "id": "d8fd05bd-a70a-4c79-9152-2a8028c83664",
  "version": 3
}
```

FULL CODE

hw1.py

```
#!/usr/bin/env python3
# sudo python3.7 -m pip install ecdsa, python-dev, pysha3, pypandoc,
eth-keyfile

from ecdsa import SigningKey, SECP256k1
from eth_keyfile import create_keyfile_json
import sha3,json

# Create a keypair of private/public key
priv = SigningKey.generate(curve=SECP256k1)

# public_key = ECDSA(private_key)
pub = priv.get_verifying_key().to_string()

# public_key_hash = Keccak-256(public_key)
keccak = sha3.keccak_256()
keccak.update(pub)

# address = '0x' + last 20 bytes of public_key_hash
address = '0x' + keccak.hexdigest()[24:]

print("Private key:\n", priv.to_string().hex())
print("Public key: \n", pub.hex())
print("Address:    \n", address)

# generate a Keystore with the password “nccu”
keystore = create_keyfile_json(priv.to_string(), str.encode('nccu'))
print("Keystore file: ")
print(json.dumps(keystore,indent=4))
```

OutPut:

```
Private key:
4daa29d6c148717d1ee0eaef9fafd81e3e9b560a4775c4f6ec30a982aaf4079f
Public key:
2ec88f012e9ddab8dc7a3774d9d934ae49e90a0a1672e1815d3cd57f05e39699b3e80d7c124
31c3eecf6e3df0517765c521f739b2f5c710fef7cb9e0076ba427
Address:
0x44349d559d34d23d24aba50b7a4f878a8c507fac
Keystore file:
```

```
{
   "address": "44349d559d34d23d24aba50b7a4f878a8c507fac",
   "crypto": {
      "cipher": "aes-128-ctr",
      "cipherparams": {
         "iv": "e9f504af6dc3e41d3231a17df3e712b4"
      },
      "ciphertext":
"b7e1b1d876dbcd4b5dbfc4656af5384fd722a1261ce1f12f4894624f56d8aa9b",
      "kdf": "pbkdf2",
      "kdfparams": {
         "c": 1000000,
         "dklen": 32,
         "prf": "hmac-sha256",
         "salt": "acb87e00344a0630d20ad1729b8e93ba"
      },
      "mac": "50398f2b3c533c48f397f9a4f536a3f212f58d6fc4b7bcbec375fe154de20b20"
   },
   "id": "d8fd05bd-a70a-4c79-9152-2a8028c83664",
   "version": 3
}
```