CS211 ALGORITHMS & DATA STRUCTURES II

LAB 0

Dr. Phil Maguire

VERTCOIN WALLET

Pen and Paper Exercise

Convert on paper, using a calculator, the ordinary decimal number 123456 to

a) Hexadecimal format (digits: 0123456789ABCDEF)
b) Base-58 Bitcoin format (digits: 123456789ABCDEFGHJKL MNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz)

Programming Exercise

Create your own Vertcoin paper wallet, including public and private key, to receive 1.0 VTC.

Write code to create a private key as follows:

- Create a random 64-digit hexadecimal string as your Elliptic Curve Digital Signature Algorithm (ECDSA) private key, using the best randomness you can.
- Add "80" to the front of this string. Let's call this the 80… string
- Compute the SHA-256 of the 80… string, and then compute the SHA-256 hash of that result. Let's call this the double hash result. The code provided below for computing SHA-256 will be valuable in this regard.
- Take the first 8 digits (4 bytes) of the double hash result and stick it on to the END of the 80… string.

- Now compute the Base58 encoding of that concatenation. The result is your Bitcoin wallet private key. It must begin with a 5 or something has gone wrong.
- Hint: you can write this Base58 encoding algorithm yourself – you've already done it on paper, right? You're dealing with a really huge number so you need BigInteger. Initialize your BigInteger as follows, and then you can use the mod(), subtract() and divide() methods that go with it. See https://docs.oracle.com/javase/7/docs/api/java/math/BigInteger.html for more details

```
BigInteger bignumber = new BigInteger(1, bytearraynumber);
```

Here's what to do next:

- The private key is secret. You have to write it down somewhere somehow and keep it secret forever.
- You need to generate your public key address from the private key. Ideally, this should be done by writing your own code – however writing the code for generating an ECDSA public key is tough enough for a final year project. So we'll take a shortcut.
- Go to: https://walletgenerator.net/?currency=Vertcoin
- Click "Skip" because you've already generated your own private key and you don't need to trust the website to make one for you.
- Click on the tab "Wallet Details"
- Paste your private key into the space "Enter Private Key" (note: ideally you should never ever paste your private key anywhere, ever)
- Your Vertcoin public address will appear on the left. It always begins with a "V" for Vertcoin.
- Copy this public address, go to Moodle and go to the link "Record your Vertcoin public address" and fill in the form, so I know your public address
- I will transfer you 1.0 VTC by Saturday. Enjoy!

- To be able to use your wallet easily and send Vertcoin to other people, download the Electrum Vertcoin wallet here: https://electrum.vertcoin.org/
- Select the option to import your private key (the secret code you generated in the lab) and Electrum will automatically synch with your wallet. Now you can do what you want with your funds – send and receive. The Vertcoin addresses of everybody in the class will be displayed on Moodle, so you can trade with them for goods and services.
- If you use Vertcoin one-click miner https://github.com/vertcoin-project/One-Click-Miner/releases, make sure not to have the Electrum wallet open at the same time, as this apparently puts stress on the Electrum network.
- If you don't like Vertcoin, you can convert your stash to any other currency at https://shapeshift.io/ - but obviously you'll need a different wallet to store a different currency. Watch out for high fees on Bitcoin and Ethereum.

```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class SHA256 {

    static String sha256(String input) throws NoSuchAlgorithmException {
        byte[] in = hexStringToByteArray(input);
        MessageDigest mDigest = MessageDigest.getInstance("SHA-256");
        byte[] result = mDigest.digest(in);
        StringBuffer sb = new StringBuffer();
        for (int i = 0; i < result.length; i++) {
            sb.append(Integer.toString((result[i] & 0xff) + 0x100, 16).substring(1));
        }
        return sb.toString();
    }

    public static byte[] hexStringToByteArray(String s) {
        int len = s.length();
        byte[] data = new byte[len / 2];
        for (int i = 0; i < len; i += 2) {
            data[i / 2] = (byte) ((Character.digit(s.charAt(i), 16) << 4)
                    + Character.digit(s.charAt(i+1), 16));
        }
        return data;
    }

    public static void main(String[] args) {
        String secret = "5JvVGV1AkvZdeQ4HCmBXK59sZaxrVgohPGGuSDYzsMJhNv47Ypx";
        try{
```

```
            System.out.println(sha256(secret));
        }catch (NoSuchAlgorithmException e){}
    }
}
```