Kienan Adams (kra8ff)
PS1

Problem 1:

a. Transaction ID : fe251e73170c725378d7ac0a14f5747517a3a68a07a013db010de2d4f14b1b93
b. .0001BTC or ~ $.02
c. 9,582.13304127 BTC
d. The timestamps for the block and the next two are 18:11:04 -> 18:19:46 -> 18:21:36. Therefore it took about 10 minutes for the transaction to have 3 confirmations

Problem 2:

a. 15j1jdJsMa4vR71gcZ6FCfYeAWJdPwpn7W, 1MzdKiBn5qr8CS1cbts6TQquxsAo52yFKe, 163vXnDXSc2hEKMrWHkERzBJWuuKJve5Nc
b. I can trace the bitcoin back until the address 15giyR7xcUTSw3NHx1XQEnzj1gkU1KZcQ3 at which it comes from two different addresses. However, I assume the address 195xBgPZv81anT56H6HPhxbKjcDjWz9g7R is some sort of exchange's address because 200 BTC exactly come from there.
c. The information in the blockchain says the location of the transaction was New York, so perhaps we can assume that the senders location is around there, versus in the Bay Area or somewhere else.

Problem 3:

a. First of all, the developer could develop the wallet so that he could collect the private keys of the users as they set up their wallets. He could then at some point decide to use their private keys to send their bitcoin to himself, but that would be a one-time thing. Additionally, he could perhaps make the wallet charge higher transaction fees and send part of them to himself rather than the miners.
b. I'm only decently confident that the money is safe in my wallet. I didn't confirm that the wallet software I downloaded had the same hash as it should've. Additionally, I don't have the whole blockchain on my computer and therefore am using a web connection with the wallet software which is an area of insecurity. To be sure my money is safe, I'd probably download the full blockchain on a machine that will be disconnected from the internet after transferring the bulk of my money over to it. That way it's safe from any possible attacks over the web.

Problem 4:

I checked the modulus manually using Wolfram Alpha and then converted it to hex. It spit out 16^^ffffffffffffffffffffffffffffffffffffffffffffffffffffffefffffc2f which is what we expected.

Problem 5:

We should first of all trust that the key is truly random or at least as close as we can get to it. To have this trust, we need to confirm that our download of Go was legit and hashes to the same value as the source. Additionally, we need to confirm that the random library we used is also as close to true randomness as we can get. We'd probably also want to make sure that no one is monitoring our console or clipboard when we're generating and copying the key.

Problem 6:

```
func generateVanityAddress(pattern string) (*btcec.PublicKey, *btcec.PrivateKey) {

	pub, priv := generateKeyPair()

	for {
		addr := generateAddr(pub)
		matched, _ := regexp.MatchString(pattern, addr.String())

		if matched {
			break
		}

		pub, priv = generateKeyPair()
	}

	return pub, priv
}
```

Problem 7:

This is your vanity private key in hex:
[70c61bc0caeb4fe00b94cef790f6ed978a598d514f612163ab74334d2da239f8]

This is your vanity public key in hex:
[035e4d0879fef85cb4b213abf72cd18ce7971e9446df75718f68d6814855f83817]

This is the associated vanity Bitcoin address:
[1KRAbyiRFdW7p5cw9ahLoTQBzdnhko81Vu]

Problem 8:

I would say that my vanity address is just as secure as my previously generated addresses since they all use the same random functions to be generated. I see one downside if I used the vanity key a lot in that I'd be less anonymous and perhaps less secure in that regard. There also are fewer bits of randomness since the first 3 are taken up with my initials. However, if I had used a particular site to generate the vanity key, I would certainly think it's less secure.

Problem 9:

TxId: 3d0f090275887b1a8f3c83daa966b32f8c35675b5e8d7224d046bfde35a6abcb

Problem 10:

TxId: 872e063b1ba326244a0bd2cc7e29ab1fb63cb0d17e878e2b58cef2ef698dddfb

Problem 11:

I added a field to the BlockChainInfoTxOut struct to handle the previous address so we know where to send the change. In main, I check that the sender has enough bitcoin to complete the transaction, and if so I create two transaction outs for the amount specified by the command line and the change.