Problem 1. Answer the following questions about the transaction where you received the bitcoin. If you received more than one transfer, include all the transaction IDs).

a. What is the transaction ID?

a46e8b2f3f17e7450eedd9ff0fe83c320b66779750241280d04ee2b5eb2be5d8

b. What was the transaction fee for the transaction? (Give your answer in BTC, as well as current approximate US dollar value.)

0.0001 BTC = 0.02 USD

c. What was the total value of all the transactions in the block containing your transfer? (Note: https://blockchain.info provides this info conveniently, although you could compute it yourself)

Block #372359

Estimated Transaction Volume: 2,732.47337582 BTC = \$ 630,081.04

d. How long did it take from when the transaction was received until it had 3 confirmations? (Include an explanation of how you estimated this in your answer.)

21 minutes = 7 minutes between Received Time and Inclusion in Blocks * 3

Problem 2. See how much can you figure out about the way bitcoin was transferred to students in the class, starting from your transactions.

a. Identify the bitcoin addresses of what are likely to be other students in the class (you could potentially find all of them, but it is enough to find 3).

1BHD4CRjp1yLPhZDQY31A2vPRXUBKFt4JF

<u>1FqQV1R3H8dftDupCjCoW7b6kSVxoBZBmh</u>

1LveaSSVmvRvnbcwsvBdC8AvoqoLro5DG4

Found from following the "change" forward and looking for more sends of similar amounts.

b. Trace back the source of the bitcoin as far as you can. Bonus points if you can figure out from which exchange the bitcoin was purchased and when.

Traced back to Address 1PAyvggUiXLcQdj5gkHqx1sC1vW96p9FLT

b. (Bonus) Can you learn anything about where the send of the bitcoin is located geographically? (In this case, you have external information to know I'm in Charlottesville,

but what could you learn about the sender's probable location just from the information in the blockchain?)

Problem 3. Suppose a malicious developer wanted to distribute a bitcoin wallet implementation that would steal as much bitcoin as possible from its users with a little chance as possible of getting caught. (a) Explain things a malicious developed might do to create an evil wallet. (b) How confident are you your money is safe in the wallet you are using, and what would you do to increase your confidence if you were going to store all of your income in it?

The evil wallet would transfer very small amounts at a time to the malicious developer, and would make transfers to and through multiple accounts owned by the malicious developer, so that the user would take a long time to notice and it would be hard to figure out the identity of the developer. The evil wallet would not display any major changes in the balance so the user would only notice losses by going to look at their account on a ledger website.

I am very confident that my wallet is secure only because Multibit is widely used and it is used for this class. To increase my confidence, I would want to use a separate system connected to a public ledger to notify me every time a transaction happens from my account, in case it is not reflected in Multibit.

Problem 4. Verify that the modulus used as secp256k1.P inbtcec.go is correct. You can do this either using math/big, Go's bit integer library to do computations on such large numbers, or by computing it by hand. (For your answer, just show how you verified the modulus. Including a snippet of code is fine.)

Decimal=

 $15 \times 16^{63} + 15 \times 16^{62} + 15 \times 16^{61} + 15 \times 16^{60} + 15 \times 16^{59} + 15 \times 16^{58} + 15 \times 16^{57} + 15 \times 16^{56} + 15 \times 16^{55} + 15 \times 16^{54} + 15 \times 16^{54} + 15 \times 16^{55} + 15 \times 16^{54} + 15 \times 16^{55} + 15 \times 16^{55} + 15 \times 16^{54} + 15 \times 16^{55} + 15 \times 16^{35} + 15 \times 16^{25} + 15 \times 16^{15} + 15 \times$

 $2^{(256)} - 2^{(32)} - 2^{(9)} - 2^{(8)} - 2^{(7)} - 2^{(6)} - 2^{(4)} - 1 = 1.1579209e+77$

Problem 5. What are *all* the things you need to trust if you are going to send money to the key generated by running keypair.go? You should assume that you are an ultra-paranoid multibillionaire who intends to transfer her entire fortune to the generated address.

The private key is not a "commonly used" number because some hackers compile the addresses generated by easy numbers to use for stealing.

The random number generator for the private key is perfectly random and will not form clusters that thieves can take advantage of.

The generator has a very large range of possible private keys it can generate.

I ran into issues installing btcd and can't run any of the .go files so I couldn't complete the rest of the problems yet. I would like to resubmit late.