For any required I2C functionality, use subroutine calls `i2c_start()`, `i2c_rstart()`, `i2c_stop()`, `i2c_put(char byte)`, `char i2c_get(char ackbit)`. If you use `i2c_put`, you must pass in as an argument the byte that is to be written to the I2C bus. If you use `i2c_get`, you must pass in an as argument the bit value to be sent back as the acknowledge bit value.

ECE 3724 Quiz #9 Reese NAME: _______________________
Answer each of the following questions (you can use a calculator)

a. (2 pts) For the 24LC515, how many I2C bit times does it take to do a random write? Assume start and stop each count as one bit time.

\[
\text{start bit } + \text{i2c address byte } + \text{eeprom address high byte } + \text{eeprom address low byte } + \text{data byte } + \text{stop} \\
1 + 9 + 9 + 9 + 9 + 1 = 38 \text{ bit times}
\]

Each byte is 8 data bits + ACK bit

b. (4 pts) Write a sequence of functions calls using `i2c_start()`, `i2c_get(char ackbit)`, `i2c_put(char byte)`, `i2c_stop()`, `i2c_rstart()` that will implement a random read from location 0x93AF within the 24LC515 serial EEPROM. Assume pin A0 is tied high, and pin A1 is tied high.

```
i2c_start();               //                                     blksel     A1     A0     Write command
i2c_put(0xAE);     // i2c address byte: 1 0 1 0     1        1        1        0
i2c_put(0x93);     // high address byte
i2c_put(0xAF);      // high low address
i2c_rstart();           // restart transaction, can also do i2c_stop(), i2c_start();
i2c_put(0xAF);   // i2c address byte:   1  0  1 0     1      1        1    1   (last bit is 1 because read)
data = i2c_get(1);     // read one data bit, send ACK = 1
i2c_stop();               // halt transaction
```

c. (2 pts) Classify the I2C bus as either a full duplex, half duplex, or simplex communication channel.

The I2C bus is half duplex as communication can go in either direction, but only in one direction at a time. It has two wires, but these are clock and data.

d. (2 pts) Why is I2C called a ‘bus’? Be explicit, and give an example that supports your argument.

I2C is a ‘bus’ because all devices on the bus listen to data sent on the bus, and respond if the ‘address’ in the transaction is intended for them. To classify as a bus in the formal sense, all devices listen to what it sent over the communication channel. Other examples of busses are Ethernet, USB, Firewire, PCI.

A ‘bus’ in reference to I2C is a formal use of this term; the use of ‘bus’ to mean a collection of wires is an informal usage of this term.