AES Calculator: http://extranet.cryptomathic.com/aescalc/index
ASCII to Hex: https://www.rapidtables.com/convert/number/ascii-to-hex.html
Hex to ASCII: https://www.rapidtables.com/convert/number/hex-to-ascii.html
Key: 90082901010010001000010000010000

Task 1:

- The message was converted from ASCII to a 38 byte hexadecimal representation using the online ASCII to Hex converter.
- ASCII: “Tom and Andrew love their iOS class.”
- Hex: 546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E
- Comments: Joey chose this message simply because Tom and Andrew were working on a project for their iOS class when we began working on the project. The suggested method of using a text document and the “od -x” unix command produced a string of hexadecimal bytes that translated into a jumbled version of the original when translated back into ASCII by online converters, so the above ASCII to Hex converter was used.

Task 2:

1) Used Key above and AES calculator in ECB mode to encrypt the hex message above. The message has all spaces removed when encrypted. The resulting hex is:
   42B2E3A9EE2BF50DC90A3DF348999F46AB379D92F66D5CCEA438592AC3590583DCF2E9ACDF4288D624038B68D2EF646B
   The resulting hex characters were decrypted, and the original message was output.
2) Now one bit in the original data is being changed. The first bit is being changed from a ‘5’ to a ‘4’. This is flipping a bit because 5 in binary is 0101, so and I flipped the least significant bit to a 0 resulting in 0100 in binary or 4 in decimal.
   The resulting hex (of encrypting the original message with one bit changed in the plaintext) is:
   56C24189CBE5E2CF6D13F3E315B3D50EAB379D92F66D5CCEA438592AC3590583DCF2E9ACDF4288D624038B68D2EF646B
   The ASCII output of this previous resulting hex is:
   VÂAËåâÏmóã³Õ«7öm\Î¤8Y*ÃYÜòé¬ßBÖ$hÒïdk
   Comments: Changing one bit changed the entire message! This is surprising since ECB does not suffer from the avalanche effect. But the translation from hex to ASCII is susceptible to a similar avalanche effect. If you look the string “AB379D92F66D5CCEA438592AC3590583DCF2E9ACDF4288D624038B68D2EF646B” is at the end of both of these outputs. This is because ECB implements DES which is a block cipher, and the flipping of a bit only affected the block and subsequent block that were encrypted. The last blocks of the message were unaffected by the bit change because of the block cipher ECB mode. Then the ASCII output is drastically different because of a separate avalanche effect on it.

3) Create Key2 by flipping the 13th bit of original Key. Each character is one byte or 4 bits. So the thirteenth bit is the first bit of the fourth character. The fourth character is an 8. So 1000 gets changed to 0000, and the character in the key gets changed from an 8 to a 0. Key2 = 90002901010010001000010000010000 . Key2 is used to encrypt the original (hex) message. This is the resulting output of that encryption:
   9D2F6DEB7BE2BC194C366BC4C40CC2A45D8D972DCC344673443CEE8C631DFBF
   B65D876C1C0027AA441F6BCA601387E28
   That message was then decrypted using Key2. The output of that decryption was the original starting message. This is the output:
   546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C61737
   32E00000000000000000000000
   Comments: The original message was NOT changed. The key was changed.
4) Original key used to encrypt original plaintext → output:
   42B2E3A9EE2BF50DC90A3DF348999F46AB379D92F66D5CCEA438592AC3590583D
   CF2E9ACDF4288D624038B68D2EF646B
   That output decrypted using Key2 → output:
   F8FE21B2A7912FC40CF2446C7CA4892B9CC6EBA48BD71303BBA1B2090C2C3BB4
   34497F5895F859FF3F24393D4DA72EE1
   Translating (Key2 → output) result into hex yields:
   øþ!²§/ÄòDl|¤+Æë¤×»¡² ,;´4lXøYÿ?$9=M§.á
   Comments: Using a Key1 will not decrypt a message encrypted by Key2. This is not surprising. The wrong key is being used!
5) Original message encrypted with original key → output:
   42B2E3A9EE2BF50DC90A3DF348999F46AB379D92F66D5CCEA438592AC3590583D
   CF2E9ACDF4288D624038B68D2EF646B
   The fifteenth bit of the above message is the third bit of the fourth character. 2(hex) translated to binary is 0010 so flipping the third bit results in 0000 or 0(hex). With the bit flipped the message is:
   42B0E3A9EE2BF50DC90A3DF348999F46AB379D92F66D5CCEA438592AC3590583D
   CF2E9ACDF4288D624038B68D2EF646B
   The above was decrypted using the original key to yield:
   5625AD665AAD1AC147127BB686B509076F766520746865697220694F5320636C617
   3732E00000000000000000000000
   The above now was translated from Hex to ASCII to yield:
   V%fZÁG{¶µ ove their iOS class.
   Comments: The end of the original message is still in tact! This is because the bit change only affected the block the bit change was in, and the next block. ECB does not have the avalanche effect because it is a block cipher that lacks in diffusion.

Task 3:

The following instructions are are done in CBC mode...

1. First, we need to encrypt the text message from above and give the resulting hex characters while using an IV of all 0's. We will then decrypt that ciphertext to verify that

we have gotten the original message.

Original Message:
546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E00000000000000000000000000
Encrypted Message:
42B2E3A9EE2BF50DC90A3DF348999F46A0DA38DBABBE7CD18DA0D0CABCA7C87094D84D364F46949EE537637D9000EA34
Decrypted Message:
546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E00000000000000000000000000
ACSII:
“Tom and Andrew love their iOS class.”

Comments: The decrypted message matches the original message.

2. Next, we want to replicate the procedure from the previous problem, but the last digit of the IV during the decryption phase will be changed to 1. The new IV will be 00000000000000000000000000000001.

   Original Message:
   546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E00000000000000000000000000
   Encrypted Message:
   42B2E3A9EE2BF50DC90A3DF348999F46A0DA38DBABBE7CD18DA0D0CABCA7C87094D84D364F46949EE537637D9000EA34
   Decrypted Message:
   546F6D20616E6420416E64726577206D6F766520746865697220694F5320636C6173732E00000000000000000000000000
   ACSII:
   “Tom and Andrew move their iOS class.”

   Comments: With the change to the last digit of the IV, the message also changes slightly upon decryption. The word “love” from the original message decrypts into the word “move.”

3. Next, we want to perform the same procedure as the first problem, but the original message will be changed slightly. The ciphertext will be varied by flipping the fifteenth bit of the encrypted text. Since each character is 4 bits, the fifteenth bit will be the third bit of the fourth character. As seen below, the third character of the original encrypted message is ‘2’. Since 2 in binary is ‘0010’, flipping the third bit will result in ‘0000’ or 0. So the 2 will be replaced with a 0. Again, an IV of all 0’s will be used for both encryption and decryption. The resulting decrypted message will be converted to ASCII to compare

with the original text.

Original Message:
546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E00000000000000000000000
Encrypted Message (Initial):
42B2E3A9EE2BF50DC90A3DF348999F46A0DA38DBABBE7CD18DA0D0CABCA7C87094D84D364F46949EE537637D9000EA34
Encrypted Message (Modified):
42B0E3A9EE2BF50DC90A3DF348999F46A0DA38DBABBE7CD18DA0D0CABCA7C87094D84D364F46949EE537637D9000EA34
Decrypted Message:
5625AD665AAD1AC147127BB686B509076F746520746865697220694F5320636C6173732E00000000000000000000000
ASCII:
“V%fZÁG{¶µ ote their iOS class.”

Comments: The first part portion of the original message is completely distorted. However, the last three words of the message still decrypted into the original words. Flipping the bit early on in the message must only effect so much of the message.

4. Lastly, we will once again repeat this process. The messages will not be manipulated at any point, but an IV of 01011101110101011101010001010100 will be used for encryption and and IV of 01010101010101010101010101010101 will be used for decryption (same IV, but making sure it ends in 1 while decrypting).

   Original Message:
   546F6D20616E6420416E64726577206C6F766520746865697220694F5320636C6173732E00000000000000000000000
   Encrypted Message (IV 01011101110101011101010001010100):
   32ED34F8122946865A9F549DC8185C902FA998D0EA709BC71295D6DDA9402BE9162C8DE55F42686CEF4648B6F6184172
   Decrypted Message (IV 01011101110101011101010001010101):
   546F6D20616E6420416E64726577206D6F766520746865697220694F5320636C6173732E00000000000000000000000
   ASCII:
   “Tom and Andrew move their iOS class.”

   Comments: Changing the IV initially does result in a different ciphertext than in the problems before this. When the last digit of the IV is changed to a 1 for decryption, however, the resulting decrypted text is the same as in 3.2. Having different IVs but both ending in a 1 changed the same character in the plaintext.

Conclusion: Group members are on header of each page. We all collaborated and decided how to split the project up. Andrew and Tom were working on an iOS project, so Joey completed part 1. Tom completed part 2, and Andrew completed part 3.