Math 4175 – Cryptography

**2. How to run code?**

There are two main functions to run the code:

EncryptCiphertext(String plaintext, String Key, Boolean CBC) :

The method encrypts plaintext and returns the desired ciphertext. You need to supply string plaintext, string key and Boolean CBC as parameters to the method so that it can perform the correct encryption.

To use CBC enter Boolean value of CBC as True, otherwise to use ECB use False.

DecryptCiphertext(String plaintext, String Key, Boolean CBC) :

The method decrypts ciphertext and returns the desired plaintext. You need to supply string plaintext, string key and Boolean CBC as parameters to the method so that it can perform the correct decryption.

To use CBC enter Boolean value of CBC as True, otherwise to use ECB use False.

I program cannot be run via command line, but the two methods in the code are fairly easy to modify as described above.

**3. Test Files and Deliverables**

aes-plaintext11.txt

Plaintext - e94ed7741f99d306e406f70386fdd7cc9fb2d5928fed4aad3f4f42fa4e91b4a1e47949125a755c2f92d11ab05cf5092b6f267beddd27763e304e8926ac80fb17634f1dcb6bd3bc3f5a422ca3a9dc355532ce4bfadc6cff73092a3635a4f574c3d7c28dedbbb7fc9ecb3c912740b1dfb7c0038ab9b1d5e48297f5ca83ed23840ff15f807691f827e75762d41b3279f229778b1258989a6b6e430e69960fa8ff857b427f1bcb690f6780d45ecc4cda297b

Key - c06df9c06a06a82ee8145a2039b767fb

Output Ciphertext(ECB) - 5d92c6430f49112617f6965dc16c5537d63942be9986f08aac54a1a76a4eee3038999c97307c3284da3cc7ef614c681394bdb2be10abcef0d6a79040313b3adab92b9330a35468216aa6fea7639515571306fcd20d6ebc8610196eff15a725f14d17746c6e4a040683234592c2eb8e4e359d2c926e5fe53bfef8fec01dec599ce552b3648cfa9cfb902f46f0c400ea28738787bbdf16b3964c107d27970120d8885ab7e0d1b07de669652eefea30c6a1

Output Ciphertext(CBC) - 5d92c6430f49112617f6965dc16c5537d937381b22ed3e72289d2c2f1bbe5d185dee06ebc5c97251e32ec1dfe54b323cc590e114b27572144e8a9d288a2ae3956148986676918782988764f59052232037a868cd53e1763a047d9704857a21d4e186c34dab826fa1b68f03d64be631d4d56faae96e6d97817203a34a56539ec06b8d9c58bbbc3be52e34be78dac4b346a2a4a6b6f2e145c6b266fc34bd6f27c923c90648edb74c28e7bc8a0b6e1b68c7

aes-plaintext12.txt

Plaintext -41b2b736367bc69154b00feef57fdc72165d804de0ba50e2b85724dc8f8359d08879462878b2492d3790a0dafbc9eebc3a461591076a8e12547b9b7fc6f0a36ede83a0dd35c9a81c5b71f6a8b6f184fc7b6e94ce29a07a4be68069ab2f2f66cf5254c5fa5aa1d5295c82ef72d2b8d0cae9f4159645bdb00da8227ba5d1b5f64ed8f6e1f84a9e9cf33e84a5d33143f3efbb9a932982669cccb87258523dfab0a7c6fcc4fe1d811c01534fecc64606421817289c5e8c53982773d996ea59296936

Key - b27c7337372510167b395f3f0b3a56037ea7f878f50e657b

Output Ciphertext(ECB) - 58e8c8c4ec1d7a6f2d34afe4718a67b855f21c248597ea49b1b7cb6511c96384d07639cc5f1b7b2fa691c64f3debb4dd69a0f9d94718f45f75b308de7eeea01861f868fbeca354dc706233769080723e3bf99978486a8f7278a7343353a8d7ef7001971e8f6bc188bff204cd44f7caf3a3b57d4ef4c304fb8e6d2c6554d75b26d2d99f181b8cc687fdd93a462d8691e5f4248ad21bd5d1762ad652ac281dd8f7239173b85dae4bff8e9ed9b863f560e7d3bed7f7030ea74e81e31be7c38e0aa5

Output Ciphertext(CBC) - 58e8c8c4ec1d7a6f2d34afe4718a67b8dcd89533e57cd472cd6c4b87cc8b5580b5df1832c3110ff7a34fa36cb3a47ba0ae6b6342b44eda2ef89af2b3b55e41f125225026310a5d5a8c4605afd8013a604d2bb7cc9b878a0a1e148cb34f330b7e99d5a0559a5e901ec6f51df0bf856f543afffba2ab8ce907da73b09f1230978050f6aefa06b73628e5219a5051ce87f9898a58d8401d0c61e078f1c73c70a1ff1cce9e09f554cf3b02d590d4f3d5ac89b67ba339c295dbfbee3d3338c15b80e4

aes-plaintext13.txt

Plaintext - a227bcee08e0b534de59cd2d8926f138a48902811ed6776ac60291ddf93375ce38de1016f29c45e95e4249d0b35cd8c0009ef90a6b075856b00498113e41a2a03129f6254b27e7b0977b57fb0f6bb958e7e6806a906e7a246e4d6e0a6dfd4452f9bd3855d1168ec02945c7ccf571d7a2faef813d03322a5de714bc65d7a2557a868b172f1436bd41c48450c2cc852ba3eed212e8b74017d24c288c7141b677de714f1741927aaf114c44b3979b2a4b359a2e936161cede0e9387c6dcf38b82f44400a4c51889f43e4a9caf08f1764cef

Key - f839739fff1d95775ebcd6d16586ccacd4eadfcae84b1643df3cb7598d92e0d4

Output Ciphertext(ECB) - f69cbbcd874ee805dc011eafc91a8632c8e173e7a35aa25b855b5b64ef3654c063f980fc63a2df12c14fd562ca8056877e847d67c2130726a1125ac1db2b7e7c22fdf578e3ce819dc531ecc8e6ddd38dade2cf3b2ae4160f343f7218f0d5539fc6713cfd09e5d93428f45a0c17a65764e3afed40511cb8837245e1ee2371ef8b6ec5f868201676668501ca1fd92fa14ac998e61311bdc6def3bd5f18fc039aa9d8697f4ff57f95ac5044f8c0a3a2a49162886450aab47b5ec2d96bb82f6864c55a202bd42ec407c4afd2e2f1c6117e4e

Output Ciphertext(CBC) - f69cbbcd874ee805dc011eafc91a86329cc5a8f5b5705030f1b72f3ec596b283fec6569bfb21569b9332e0cd20377dfe01ad927960fdf85e2f035ffd595f310fac6776991acb372c70f65997c8df15165524ef42be700ad51e47178ee730755c3bbe03762ede6cf08bf5a0ebbd60e7421f2d79504c8d41525d7518f858591ca1a7040e6c480083eff931dd289b47e7fb4658a547cff3619bee5bc8ca63a406fa0f758a1e22f51633996aa28fd0ba1d27a0d154ebc3eb75ca0631a9054c9967dbab93c8b29a97bedd2ee4e69917c05824

aes-ciphertext10-cbc.txt

CipherText - a5360648c5a07b8b0d32526666d6956740ff173728e3873e0f369e0eccdaf8b5707e16aa4879b76e81719c449e710b8f003140671445d240e4223fa7d10f834774496b0c743721f6e7cb222b5a69a41aa37370002db9a29e7301013960c91068

Key- 4e0e01285b1ff23909b11b5de4ea01c11acf4a713a66f782

Found Plaintext –

89504e470d0a1a0a0000000d494844520000000a0000000a0802000000025058ea000000194944415418d36360a002f8ffff3f567126fcda06529a320000d4920308ecf3af170000000049454e44ae4260820000000000000000000000000000

**Secrecy of AES ALGORITHM**

The AES Algorithm exhibits perfect secrecy, i.e it is impossible to determine anything about the plaintext just by knowing the ciphertext. Below are the two approaches that I have used to portray this:

Approach 1:

1st Plaintext

Plaintext –

2db71d7024beb0af4803c1adb8c80903e5c70d65db251876142d44da37b0722351d08d21e4e0e2f0cffe0acd422b5e4975d6b4720edb4716b3af33876927a9ee7f25a266378c05c4216e54f9ee0def217a70bd30e5dbd36e9aebbcbd177eb446

Key- d51145f9a23cd00f9df7591ac48e3bfe46a18c9cf50ad985881d981b3e99e6d0

CipherText(ECB) –

572b502ec128b966d5f59a1c21a231d6a6909930fa3d483c735b876760075b2a1ebeb85610a4957b20f0eb1b93bbfca8cf15341734f66bcf3d716488042199aa0bafca107ba422f1eb266e8e15d1f134149bcd423d1c926c98c9f4aeddf8674a

2nd Plaintext

Plaintext –

45b7cf11839538da7da1ca40c3f4b924a3f6aca53d3d496f4f935ff68ad8e54d69e4851fdc21cdfd62a053a2eaaa829d14de2a057fde1444edac8fdfb59529113073927e1937e654360b21f59ab7dddeadc236055e7b47cd089afef36f7b73122567e25927197a4c068c7d879990872e70437c6a657b573e9837973af2cd3e79d52de6de68cc07fc4fd4c16f4acfa6e8

Key – f6d2738e2589cb88487aa5e49834f46e79550cb2fa393c80

Cipher Text (ECB)-

e7f5667bb61f4f31dc45d14710af7012b7cb3d5befdb45cea788088c4a52a41b2dbec0d0cbfca24b1471576f62e6fc14b745b243f2bc90ed86b3dc03da4d8252f668a9056e8d2ce184766b445e1efcba7f6c1f805a335d55a9a44471710da56411face44fbba5b7024b1db7121168530e0f346ac3336d5e03902b2bbe0a7304ea3e01440b44a6e4c6993916bc6773f07

Now, we truncate our key length from 32 bytes to 24 bytes

1st Plaintext

Plaintext –

2db71d7024beb0af4803c1adb8c80903e5c70d65db251876142d44da37b0722351d08d21e4e0e2f0cffe0acd422b5e4975d6b4720edb4716b3af33876927a9ee7f25a266378c05c4216e54f9ee0def217a70bd30e5dbd36e9aebbcbd177eb446

Key- d51145f9a23cd00f9df7591ac48e3bfe46a18c9cf50ad985

CipherText(ECB) –

3a898a0ee3f9195e87f98cc0fc0046e7dabaad219502fce60bca0edb42ef1cdea34ea6367efe8019253989fb83ac21cdc606091b611eb95ee8e27c643b55484567196e81e1d8e3347e6b0292a10b809af6c9569dbbe862dcc7820ff5e7f3dc44

2nd Plaintext

Plaintext –

45b7cf11839538da7da1ca40c3f4b924a3f6aca53d3d496f4f935ff68ad8e54d69e4851fdc21cdfd62a053a2eaaa829d14de2a057fde1444edac8fdfb59529113073927e1937e654360b21f59ab7dddeadc236055e7b47cd089afef36f7b73122567e25927197a4c068c7d879990872e70437c6a657b573e9837973af2cd3e79d52de6de68cc07fc4fd4c16f4acfa6e8

Key- f6d2738e2589cb88487aa5e49834f46e79550cb2fa393c80

Ciphertext –

0ca3c66cc06e019e92731cec6cfb4f11bd148c65d0eb2cc72b41c5769b72248eca1f1faa06f4bde1b6c792019ddea5b50f9cd985e11bba3dcda5322224a202b5cc8244e988c92b0d91bf90561487f3f30c2dba5d529c372c5ba0cee06873a10e03b720be6e61e15693afb738dfcab879184917391896ccd5ebb1a362c1c8859deec59221f2d8dcde4ff45f9585bff997

As we can see from the above obtained ciphertexts that after truncating our key length, the entire ciphertexts were different. Hence, our initial key length does matter, and an attacker will not be able to deduce anything about the differences in key or its length just by looking at the ciphertexts. This indicates that AES algorithm is perfectly secret.

Approach 2:

We take a plaintext and encrypt it using its key.

Plaintext –

1526154061b689e0f00a5c2ff1ec19e4

Key- 30190dcc14585301f5bfc5b666c84775

CipherText(ECB) –

8d04fff27a081a77de2009d1402e6e03

Now we, change 1 bit of key and then encrypt the same plaintext.

Plaintext –

1526154061b689e0f00a5c2ff1ec19e4

Key- 30190dcc**2**4585301f5bfc5b666c84775

CipherText(ECB) –

a5e3e549790d37e0b7140e0f7e42c791

When we XOR the two ciphertexts we get the following in binary,

10000000100000000000100000001000100010001000000000000000100010001000100010000000000010000000100000001000000000001000000000001000000010001000100010000000000010000000000000000000100010000000100000000000100010000000100010001000100010001000000000000000000000001000000010001000000010001000100000000000000010000000100000000000000000000000000010001000100000000000000000000000100010001000110001100000101011110111011100010011110110000100000011010011001111100111100100000000010001100000001010000001011110110111000000010

When we count the number of 1’s then we see that is nearly 50%, that is t the odds of a bit changing and not changing are approximately equal. If we follow this with other plaintexts, then we will get the same results. Therefore, AES algorithm can be said to be perfect secret.