ISM Exam February 4, 2022 (OpenSSL in C/C++)

Write a C/C++ application (one single source code file) using OpenSSL library to create classes having the fallowing specifications:

- Class ***Cipher***

Modifier and Type	Field	Description
`static unsigned char`	`ALGO_AES_CBC`	It provides a cipher using AES with block size 128 bits in CBC mode.
`static unsigned char`	`ALGO_RSA_PKCS1`	It provides a cipher using RSA, and pads input data according to the PKCS#1 scheme.
`static unsigned char`	`MODE_ENCRYPT`	It indicates the encryption operation.
`static unsigned char`	`MODE_DECRYPT`	It indicates the decryption operation.
`unsigned char`	`algorithm`	It indicates the actual algorithm for a particular object.
Modifier and Type	**Method**	**Description**
-	`Cipher(unsigned char algorithm)`	Object constructor with parameter: `Algorithm` – algorithm type of the `Cipher` object.
`static Cipher`	`create_instance(unsigned char algorithm)`	It creates a `cipher` object instance of the selected algorithm.

- Class ***AESCBCCipher*** inherits ***Cipher*** publicly:

Modifier and Type	Field	Description
`AES_KEY`	`aes_key`	OpenSSL structure to store a AES key in the ***private*** class section.
`unsigned char`	`ivec[16]`	Byte array to store the initialization vector in the ***private*** class section
Modifier and Type	**Method**	**Description**
`void`	`init_cipher(unsigned char* user_key, unsigned short int bit_key_length, unsigned short array_key_offset, unsigned char* iv, unsigned short array_iv_offset, unsigned char mode)`	It initializes an AES-CBC cipher: `user_key` – byte array containing the AES CBC key content. `bit_key_length` – AES CBC key length as number of bits. `array_key_offset` – offset of byte array `user_key` where the content is considered from. `iv` – byte array containing the initialization vector content. `array_iv_offset` – offset of byte array `iv` where the content is considered from. `mode` – operation type. It could be `MODE_ENCRYPT` or `MODE_DECRYPT`

- Class ***RSACipher*** inherits ***Cipher*** publicly:

Modifier and Type	Field	Description
`RSA*`	`rsa_key_pair`	OpenSSL structure to store a RSA key pair in the ***private*** class section.
Modifier and Type	**Method**	**Description**
`-`	`RSACipher()`	Object default constructor
`int`	`generate_key_pair(int bit_modulus_length, unsigned long public_exp)`	It generates a 2-prime RSA key pair and stores it in the RSA structure: `bit_modulus_length` – RSA modulus length as number of bits `public_exp` – RSA public exponent It returns a validation flag about correctness of operation.
`unsigned char*`	`public_encrypt(unsigned char* in_buffer, unsigned short byte_in_length, unsigned short in_offset, unsigned short* byte_out_length)`	It encrypts the input data with the RSA public key: `in_buffer` – byte array storing content to be encrypted. `byte_in_length` – number of bytes to be encrypted. `in_offset` – offset of byte array `in_buffer` where the content is considered from. `byte_out_length` – length of the encrypted content as number of bytes. It returns a memory address of the byte array where the ciphertext was placed at.
`-`	`~ RSACipher()`	Object destructor

Implementation requirements:

- Instantiate two cipher objects as ***AESCBCCipher*** and ***RSACipher*** using ***Cipher.create_instance()***. **(15 p)**
- Initialize the AES-CBC cipher. **(3 p)**
- Initialize the RSA cipher (generate a key pair). Encrypt each password candidate from ***wordlist.txt***. Each encrypted password candidate will be saved as hex representation into ***enclist.txt*** for each corresponding line. **(7 p)**

All the solutions will be cross-checked with MOSS from Stanford and very similar source code files will not be evaluated.