Exercise 2: Enabling communication between parties

Now that you have dug in and understood RSA operation, create a program for the other agents to use to communicate with Agent X. You can assume that these agents all have access to Agent X's public key (in the form of the pem file which you exported in step 3 above).

Also create a program for Agent X to decrypt all received messages.

Encrypt function:
Input: Message M and public key of Agent X
Program: encrypt message with asymmetric key (call it cipher C)
Output: Ciphertext C

Decrypt function:
Input: Ciphertext C and prive key of Agent X
Program: decrypt ciphertext with asymmetric key
Output: Plaintext M

Instructions for the programming task -

- The tar contatins 2 directories named test_decrypt and test_encrypt to test your decrypt and encrypt function respectively.
- Write your public_encrypt function in test_encrypt/rsa_test_encrypt.c
- Write your private_decrypt function in test_decrypt/rsa_test_decrypt.c
- Main function for both test directories takes 3 command line arguments -
    - arg1:message_file (.txt file)
    - arg2:private_key_file (.pem file)
    - arg3:public_key_file (.pem file)
- Main function reads the message file and calls public_encrypt function to encrypt the message and then calls private_decrypt function to decrypt the encrypted message
- Your task is to write these two functions (public_encrypt & private_decrypt) in two separate files (test_encrypt_rsa.c and test_decrypt_rsa.c respectively)
- Specification of the functions -
    - int public_encrypt(unsigned char * data, int data_len, char * fname, unsigned char *encrypted)
      where,
        - data - message to be encrypted
        - data_len - length of the data
        - fname - file name of public key (for eg. “public.pem”)
        - encrypted - Write the encrypted message in this variable
        - Return type is int, (1 if successfully encrypted, else 0)
    - int private_decrypt(unsigned char * enc_data, int data_len, char * fname, unsigned char *decrypted)
        - enc_data - Data to be decrypted
        - data_len - length of enc_data
        - fname - file name of private key (for eg. “private.pem”)
        - decrypted - Write the decrypted message in this variable
        - Return type is int, (1 if successful, else 0)
- To implement these functions, you can check out openssl C library

Guidance:

1. Assume that the message size (number of characters in message.txt) is small (less than 64 characters).

2. To compile and evaluate your program :
   ```
   make
   make test
   ```

NOTE : output : 1 represents success and output : 0 represents failure.

NOTE : please refer to reference.txt for help in coding assignment.