RSA Assignment

You need to implement RSA Algorithm.

- First generate two prime numbers (p, q), each number should be 512 bits long, you should test it yourself for primality.
    - If one of them is not prime then regenerate that number.
    - Also make sure that the product of the two values n=p*q is not more than 1024 bits.
    - If you end up with 1025 regenerate (p, q).
- Generate e, d for encryption and decryption.
- **Print the decimal value for p,q,e,d,n.** Each one in a separate line.
- Given a plain text (this could be a very large plain text with multiple lines so handle all characters), if it is more than 127 bytes (1016 bits) split it into parts of 127 bytes, and encrypt each byte on its own.
    - And then write the result bytes into a file.
- Read the text file encrypt it, and write the encryption onto a cipher file.
- Read the cipher from the cipher file and decrypt it then print the decrypted text.
- You can keep the e,d,p,q,n and encrypt & decrypt in one run.
    - But you have to write the encryption onto the file correctly and read it from the file correctly.
    - If the file size is incorrect after writing the cipher onto it, then it means you did it wrong.
- When RSA is used for encryption there are two options.
    - For 'N', the modulus, of size 'M' bits you can use only M-1 bits. So to make things easier my suggestion is to use 1 byte less (8 bits less), to make it easier to handle. This is why I suggest 1016 bits parts for the 1024 bit key.
        - In the case that you have more than M-1 bits you want to encrypt you will need to split whatever you want to encrypt.
        - If it is less then you will only need to encrypt. Note that the output is of the size of the modulus no matter the input (unless you want to put indicators which you will most likely not stumble upon a case to use them). You may end up sometimes with extra characters in the decryption (which will be null values). Feel free to keep/remove them, just know how you will handle them.
- **DO NOT store a binary character string of 0,1 in the cipher file.**
    - **Write the bytes as is. And read them as is.**
    - **There is absolutely no need of using binary/hex/decimal numbers when outputting the cipher.**
    - **And do not put and delimiters between each output (use what you've learned about fixed length outputs).**
- The RSA is simple arithmetic, and you will have a library to help you with the large values.
    - So failing to abide with the constraints will result in some of the grade (depends on what it is).

Please Note:

- (If you're going to use Jave, you need to use the data type BigIntegers to handle operations on BigIntegers)
- Files are not mandatory, you can always print the results in the console.