IAM504 Public Key Cryptography **Spring 2020-2021**
Homework 2 **Due: 30.04.2021**

Question

Implement the RSA cryptosystem. Your program should give the user 3 choices.

- Generate parameters
- Encrypt
- Decrypt

You can assume the user will exit the program after each step.

Part 1.

Generate 512-bit primes p and q, find $\phi(n)$ and n. Randomly choose e and find the corresponding d, according to RSA parameter generation constraints. Output $p, q, \phi(n), n, e, d$ to a file called `parameters.txt`.

Part 2.

Read the required parameters from `parameters.txt` and the message from `plain.txt`. Your message should be ASCII text. Note that your message (and later, your ciphertext) is not necessarily alphanumerical.

Encode the message into an integer. (Hint: You can do this naively or by using functions supplied by MPIR/GMP that can convert binary data to `mpz_t`, take a look at the documentation.)

If you cannot figure out how to encode, then use integers instead of text messages. However, you will lose points if you do so.

The message you encrypt should have a size limit. How many characters can you fit in an RSA message for given parameters? Make the message size the maximum that you can.

Encrypt the message using the parameters that you have read from the file. Decode the ciphertext from the integer result back to characters. Output that to `cipher.txt`.

Part 3.

Read `cipher.txt` and `parameters.txt`. Decrypt the data from `cipher.txt` using the encoding from Part 2. Write the decrypted message into `message.txt`.

Part 4.

Explain how you encoded the message to and from integers. What happens if the message length is larger? How would you divide the message into blocks? What are the shortcomings of this encoding? What is wrong with using RSA in this way? (A paragraph or two is enough.)

Guidelines and Implementation Considerations

- You are supposed to implement the schoolbook RSA (not OAEP or PKCS etc.).
- You are required to use MPIR library (or GMP). Do not use any other big integer library.
- You don't have to write your own exponentiation, gcd, inverse, primality, ... functions. You still can if you want but you can use these functions from MPIR / GMP.
- You might want to take a look at binary file input/output.
- Your code must be C or C++.
- Write comments in the code if necessary.
- Send your report as a pdf, prepared in LaTeX. You can use any of the .tex templates available in odtuclass.
- Upload your homework to odtuclass.
- For your codes, only send the .c/.cpp, .h/.hpp. Please don't send the project / solution files, or the executable.
- Include a note about your operating system (win 10, linux distribution etc.) and your IDE (visual studio, devc++, codeblocks...) or your compiler.
- Do not steal your code. You can study other code and give references to them. Copying someone elses code and just changing the variable names is not the purpose of this homework.
- This is not a group homework. You can study with others, but don't copy each others code.