Programming Assignment:
Due June 21, 2022, 11:59pm EDT

Here are the instructions for the programming assignment. In this project you will implement an RSA encryption/decryption algorithm.

Information

- The alphabet will be the standard English alphabet plus three additional characters: space, period, and comma, with the following ordering:

  ABCDEFGHIJKLMNOPQRSTUVWXYZ_.,

  with "_" representing the space character. Thus there are 29 characters total.

- Text to be encrypted will be broken into 5-character blocks and converted to numbers via place-value notation base 29. For example: the block "STOP." is first changed to values in $\mathbb{Z}_{29}$ to be 18 19 14 15 27. These values then transform as

  $$18 \cdot 29^4 + 19 \cdot 29^3 + 14 \cdot 29^2 + 15 \cdot 29 + 27 = 13206685.$$

- You are permitted to use any predefined functions in Maple provided they do not circumvent the point of this project. For example, if there is a function in Maple which returns the modular inverse, you are not permitted to use it anywhere in this project, you must use your own procedure.

- This project must be submitted on Moodle as a single Maple 15 worksheet.

Procedures to implement

- Extended Euclidean Algorithm to compute modular inverses.
  - **eea := proc(a,n)**
  - inputs: a is the number to invert and n is the modulus.
  - outputs: the value of a^{-1} mod n if it exists or 0 if it does not.
  - example call and output:
    - eea(5,7);
    - 3

- Square and Multiply Algorithm to quickly compute powers mod n.
  - **sqmul := proc(x,a,n)**
  - inputs: x is the base, a is the exponent, n is the modulus.
  - outputs: the value of x^a mod n.
  - example call and output:
    - sqmul(4,5,13);
    - 10
  - Note: this procedure is only allowed to use squaring and multiplication. All other exponents are not permitted, and if used will result in a failing mark for this procedure.

- RSA encrypt
  - **eRSA := proc(x,b,n)**
  - inputs: x is a plaintext 5-character string from the alphabet to be encrypted, b is the public exponent, and n is the public modulus.

  - outputs: numeric ciphertext reduced modulus n.
  - example call and output:
    * eRSA("ADEAF",3,20875213);
    * 10079123
  - You must use your **sqmul** procedure in this function. Make sure it is predefined.

- RSA decrypt.
  - **dRSA := proc(y,b,n) (need to find a)**
  - inputs: y is the number ciphertext, b is the public exponent, and n is the public modulus.
  - outputs: 5-character string from alphabet which is the plaintext associated with y.
    * dRSA(10079123,3,20875213);
    * "ADEAF"
  - You must use your **eea** and **sqmul** procedures in this function, as well as Maple's **phi** function from its **numtheory** package.

Grading

The procedures must be labelled as given above, with the labels being case-sensitive, so **eea** is not the same as **EEA**. The procedures must be written so as to be easily followed manually with help of documentation within the worksheet. The procedures must not use any prohibited functions as noted above. This project must be done individually (no copying/sharing of code with other students). If you are using Maple code from elsewhere, include the source(s) in your documentation.

If a procedure works and meets all the above criteria, it will be given full marks. If it does not work (for all possible inputs) or fails to meet the above criteria, a mark not greater than 60% will be assigned to that procedure, depending on the severity of the failures.

Each procedure is equally weighted at 40 marks each.

Additionally, you must use the procedures to evaluate the following, for 10 marks each.

1. eea(123456,9765625);
2. sqmul(1992,2718,357419);
3. eRSA("A DOG",611,20875213);
4. dRSA(909168,611,20875213);

Total marks available: 200.