# Relatório Criptografia RSA

Bruno M. Barbosa

18 de abril de 2019

## 1 Introdução

Este relatório é referente à segunda parte da avaliação para seleção de alunos do IC para bolsistas do LCCV em 2019, que consiste em desenvolver um módulo na linguagem Python que implementa a criptografia RSA.

## 2 Método utilizado

O método utilizado está descrito em [1]. Primeiro, calcular a **chave de codificação**, que consiste em escolher dois números primos, $p$ e $q$ e calcular o produto $n = p * q$. Depois, calcular $\Phi(n) = \Phi(p) * \Phi(q)$, onde $\Phi(n) = n - 1$. Daí, achar um número $c$, tal que o $mdc(\Phi(n), c) = 1$, ou seja, primos entre si. Encontrado tal número, o par $(n, c)$ é a chave de codificação.

A **chave de decodificação** é o par $(n, d)$, onde $d$ é o número que multiplicado por $c$ e somado com $\Phi(n)$ resulta em 1.

Após as chaves serem determinadas, o próximo passo é transformar a mensagem em números. Os caracteres da mensagem são transformados em seus equivalentes na tabela ASCII e somados com 100. Então, a cadeia de números é dividida em partições que são primas entre si com n,ou seja, partições $B_i$, tais que $mdc(B_i, n) = 1$, com $i \in \mathbb{Z}$.

Daí, a **função de codificação** irá fazer $B_i^c mod n$, onde $mod$ representa a operação módulo, o resto da divisão euclidiana, para gerar a mensagem codificada (no livro, os resultados são separados por #). A **função de decodificação** irá pegar a mensagem codificada e realizar o mesmo cálculo de módulo, porém, o expoente será $d$.

A chave de codificação é a chave pública e a chave de decodificação é a chave privada, que são exigidos no problema.

## 3 Implementação

A implementação consiste em uma classe em Python, chamada RSA, que possui os primos $p$ e $q$, as chaves de codificação e decodificação e as mensagens codificadas e decodificadas como atributos. Possui as classes *setup*, *encrypt* e

*decrypt*, como exigido, e algumas classes adicionais, para a realização de cálculos necessários para a criptoografia RSA, como o cálculo do mdc, feito pela função *gcd*, uma função para gerar números primos, $prime_numbers_until_n$ e uma função para achar o parâmetro $d$ na chave de decodificação. A função *decrypt* devolve uma mensagem de erro, caso a mensagem seja inválida.

# Referências

[1] Jaime Evaristo and Eduardo Perdigão. *Introdução à Álgebra Abstrata*. 2 edition, 2 2012.