Implementação do algoritmo RSA

Aluno: Lucas Pereira da Silva (10100754)

Código Fonte

```
package br.ufsc.inf.ine5429.rsa;

import br.ufsc.inf.ine5429.millerRabin.MillerRabin;

import java.math.BigInteger;
import java.io.PrintStream;
import java.util.Scanner;

public class Rsa {
	private static final BigInteger UM = BigInteger.ONE;
	private static final BigInteger MENOS_UM =
BigInteger.ZERO.subtract(BigInteger.ONE);
	private MillerRabin millerRabin;
	private BigInteger valorN;
	private BigInteger valorD;
	private BigInteger valorE;

	/*Cria um objeto que gerenciará um par de chave pública e privada.*/
	public Rsa(Integer quantidadeDeDigitosDasChaves) {
		assert(quantidadeDeDigitosDasChaves >= 8);
		millerRabin = new MillerRabin(quantidadeDeDigitosDasChaves / 2);
	}

	/*Realiza a geração do par de chave pública e privada.*/
	public void gerarParDeChaves() {
		BigInteger primoP = millerRabin.encontrarProvavelPrimo();
		BigInteger primoQ = millerRabin.encontrarProvavelPrimo();
		BigInteger totienteDeN =
primoP.subtract(UM).multiply(primoQ.subtract(UM));
		valorN = primoP.multiply(primoQ);
		valorE = UM;
```

```
            do {
                  valorE = valorE.nextProbablePrime();
            } while (!valorE.gcd(totienteDeN).equals(UM));
            assert(valorE.compareTo(UM) == 1 && valorE.compareTo(totienteDeN) ==
-1);
            valorD = valorE.modPow(MENOS_UM, totienteDeN);
      }

      /*Cifra um determinado valor utilizando a chave privada.*/
      public BigInteger cifrarComChavePrivada(BigInteger valor) {
            return Rsa.cifrar(valor, fornecerChavePrivada());
      }

      /*Cifra um determinado valor utilizando a chave pública.*/
      public BigInteger cifrarComChavePublica(BigInteger valor) {
            return Rsa.cifrar(valor, fornecerChavePublica());
      }

      /*Decifra uma determinada cifra utilizando a chave privada.*/
      public BigInteger decifrarComChavePrivada(BigInteger valor) {
            return Rsa.decifrar(valor, fornecerChavePrivada());
      }

      /*Decifra uma determinada cifra utilizando a chave pública.*/
      public BigInteger decifrarComChavePublica(BigInteger valor) {
            return Rsa.decifrar(valor, fornecerChavePublica());
      }

      /*Método estático para cifrar um dado valor dada uma determinada chave.
Isso permite fazer: Rsa.cifrar(valor, chave).*/
      public static BigInteger cifrar(BigInteger valor, BigInteger[] chave) {
            return valor.modPow(chave[1], chave[0]);
      }

      /*Método estático para decifrar uma dada cifra dada uma determinada chave.
Isso permite fazer: Rsa.decifrar(cifra, chave).*/
      public static BigInteger decifrar(BigInteger valor, BigInteger[] chave) {
            return valor.modPow(chave[1], chave[0]);
      }
```

```
	/*Fornece a chave privada.*/
	public BigInteger[] fornecerChavePrivada() {
		BigInteger[] chavePrivada = {valorN, valorD};
		return chavePrivada;
	}

	/*Fornece a chave pública.*/
	public BigInteger[] fornecerChavePublica() {
		BigInteger[] chavePublica = {valorN, valorE};
		return chavePublica;
	}

	/*Responsável pela execução do programa através de uma interface em modo texto.*/
	public static void main(String[] argumentos) {
		Scanner leitor = new Scanner(System.in);
		PrintStream escritor = System.out;
		escritor.println("");
		escritor.print("Digite a quantidade máxima de dígitos decimais das chaves: ");
		Integer quantidadeDeDigitosDasChaves = leitor.nextInt();
		Rsa rsa = new Rsa(quantidadeDeDigitosDasChaves);
		rsa.gerarParDeChaves();
		BigInteger[] chavePrivada = rsa.fornecerChavePrivada();
		BigInteger[] chavePublica = rsa.fornecerChavePublica();
		escritor.println("");
		escritor.println(String.format("Chave privada: {%s, %s}", chavePrivada[0], chavePrivada[1]));
		escritor.println(String.format("Chave pública: {%s, %s}", chavePublica[0], chavePublica[1]));
		Boolean encerrar = false;
		while (!encerrar) {
			escritor.println("");
			escritor.println("Ações disponíveis:");
			escritor.println("Cifrar com chave privada (c-)");
			escritor.println("Cifrar com chave pública (c+)");
			escritor.println("decifrar com chave privada (d-)");
			escritor.println("decifrar com chave pública (d+)");
			escritor.println("mostrar chaves (mc)");
			escritor.println("sair (*)");
```

```
                escritor.print("Digite a ação desejada (valor entre os
parênteses): ");
                String acao = leitor.next();
                if (acao.equals("c-")) {
                      escritor.println("");
                      escritor.print("Qual valor você deseja cifrar com a
chave privada? ");
                      BigInteger valor = leitor.nextBigInteger();
                      BigInteger cifra = rsa.cifrarComChavePrivada(valor);
                      escritor.println(String.format("O valor cifrado com a
chave privada é: %s", cifra));
                } else if (acao.equals("c+")) {
                      escritor.println("");
                      escritor.print("Qual valor você deseja cifrar com a
chave pública? ");
                      BigInteger valor = leitor.nextBigInteger();
                      BigInteger cifra = rsa.cifrarComChavePublica(valor);
                      escritor.println(String.format("O valor cifrado com a
chave pública é: %s", cifra));
                } else if (acao.equals("d-")) {
                      escritor.println("");
                      escritor.print("Qual cifra você deseja decifrar com a
chave privada? ");
                      BigInteger cifra = leitor.nextBigInteger();
                      BigInteger valor = rsa.decifrarComChavePrivada(cifra);
                      escritor.println(String.format("A cifra decifrada com a
chave privada é: %s", valor));
                } else if (acao.equals("d+")) {
                      escritor.println("");
                      escritor.print("Qual cifra você deseja decifrar com a
chave pública? ");
                      BigInteger cifra = leitor.nextBigInteger();
                      BigInteger valor = rsa.decifrarComChavePublica(cifra);
                      escritor.println(String.format("A cifra decifrada com a
chave pública é: %s", valor));
                } else if (acao.equals("mc")) {
                      escritor.println("");
                      escritor.println(String.format("Chave privada: {%s,
%s}", chavePrivada[0], chavePrivada[1]));
                      escritor.println(String.format("Chave pública: {%s,
%s}", chavePublica[0], chavePublica[1]));
                } else {
```

```
                encerrar = true;
            }
        }
        escritor.println("");
        escritor.println("Programa encerrado.");
    }
```