UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL – PROFMAT

ERLON ASSUNÇÃO WANDENKOLK ALVES

CodeClass: o uso da tecnologia como recurso metodológico no ensino da criptografia

Vitória da Conquista/BA

2019

UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE

NACIONAL – PROFMAT

ERLON ASSUNÇÃO WANDENKOLK ALVES

CodeClass: o uso da tecnologia como recurso metodológico no ensino da criptografia

Dissertação apresentada ao Mestrado Profissional em Matemática em Rede Nacional – PROFMAT, oferecido pela Universidade Estadual do Sudoeste da Bahia – UESB, como requisito necessário para obtenção do grau de Mestre em Matemática. Orientadora: Profª. Drª Alexsandra Oliveira Andrade.

Vitória da Conquista/BA

2019

A478c Alves, Erlon Assunção Wandenkolk.

CodeClass: o uso da tecnologia como recurso metológico no ensino da criptografia. / Erlon Assunção Wandenkolk Alves, 2019.
91f. il.
Orientador (a): Dra. Alexsandra Oliveira Andrade.
Dissertação (mestrado) – Universidade Estadual do Sudoeste da Bahia, Mestrado Profissional em Matemática em Rede Nacional – PROFMAT, Vitória da Conquista - BA, 2019.
Inclui referências. 85 - 86.
1. Criptografia - Ensino. 2. CodeClass - Aplicativo. 3. Aritmética. 4. Números primos. I. Andrade, Alexsandra Oliveira. II. Universidade Estadual Sudoeste da Bahia, Mestrado Profissional em Matemática em Rede Nacional – PROFMAT, Vitória da Conquista, III. T.

CDD: 510

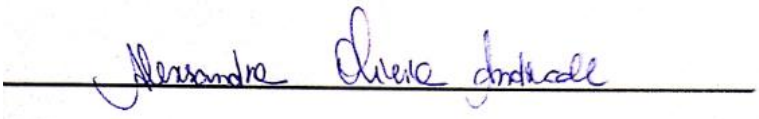
Catalogação na fonte: Juliana Teixeira de Assunção- CRB 5/1890
UESB – Campus Vitória da Conquista – BA

ERLON ASSUNÇÃO WANDENKOLK ALVES

CodeClass: o uso da tecnologia como recurso metodológico no ensino da criptografia

Dissertação apresentada ao Mestrado Profissional em Matemática em Rede Nacional – PROFMAT, oferecido pela Universidade Estadual do Sudoeste da Bahia – UESB, como requisito necessário para obtenção do grau de Mestre em Matemática.

BANCA EXAMINADORA

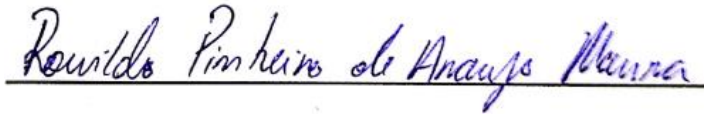


Professora Drª. Alexsandra Oliveira Andrade (Orientadora)

Universidade Estadual do Sudoeste da Bahia – UESB

Professor Dr. Flaulles Boone Bergamaschi

Universidade Estadual do Sudoeste da Bahia – UESB



Professor Dr. Ronildo Pinheiro de Araújo Moura

Universidade Federal do Rio Grande do Norte

Vitória da Conquista/BA

2019

O segredo da felicidade é

a felicidade do segredo.

(desconhecido)

DEDICATÓRIA

Dedico este trabalho a todos aqueles apaixonados não só pela matemática, mas sim apaixonados também por torná-la ainda mais apaixonante.

AGRADECIMENTOS

Finalmente, o momento de escrever os agradecimentos, este que prometi fazer somente ao conseguir findar este trabalho. Foram tantos colaboradores diretos e indiretos que esquecer o nome de alguns, não será difícil.

Primeiramente gostaria de agradecer a minha esposa Mylena e ao meu pequeno Bento, por serem pacientes e compreensivos em todas as noites que me fiz ausente durante essa trajetória. Sei que não foi fácil para nós, mas finalmente superamos todos estes obstáculos.

Gostaria de agradecer aos meus pais pela compreensão e apoio nos momentos difíceis que enfrentamos, nas adversidades e tarefas que tiveram de ser abandonadas e a minha irmã Sirlara (Songs), por todo carinho, força e auxílio prontamente prestados.

Em terceiro, quero agradecer a minha orientadora, Drª Alexsandra pelo apoio e palavras de tranquilidade, que sempre se prontificou ouvir e colaborar com as mais malucas ideias que tive durante essa jornada.

Queria agradecer também aos meus colegas do PROFMAT, que também lutaram junto comigo cada um atrás do seu próprio destino e sem nunca abandonar o próximo: Abizai, Alan, Antônio Marcos, Daniel, Dimas, Jocasta, Júlio, Lídia, Liliane, Lupicino, Marcos, Paula, Romário. Todos vocês são merecedores dessa conquista; Em especial queria agradecer aos colegas Julião pela nossa caminhada e conversas em tantos quilômetros rodados, uma amizade que iniciou no PROFMAT e deverá perdurar por muitos anos. Lídia, pelo companheirismo de sempre e Marcão pelos experientes conselhos e pelo grande apoio quando eu e Júlio precisamos.

Finalmente, gostaria de agradecer a direção, professores e colaboradores do Colégio Estadual Governador Luiz Viana Filho, pelo apoio e incentivo durante a execução deste trabalho.

O meu muito obrigado a todos vocês.

LISTA DE FIGURAS

LISTA DE SÍMBOLOS

$\mathbb{N}$	Conjunto dos números Naturais
$\mathbb{Z}$	Conjunto dos números Inteiros
$\mathbb{R}$	Conjunto dos Números Reais
$\in$	Pertence
$\equiv$	Congruência

RESUMO

A criptografia é raramente trabalhada na educação básica, pois não consta diretamente como componente/conteúdo obrigatório na BNCC. Tal fato apresenta ainda enquanto algo a ser trabalhado, pois é um assunto extremamente importante, dado o contexto de que vivemos em uma era digital e que este serve de porta de entrada para diversos outros assuntos que compõe a grade curricular do ensino médio. Diante desta questão, este trabalho tem como objetivo principal desenvolver um aplicativo que sirva como uma metodologia para o ensino da criptografia no ensino médio. Este aplicativo desenvolvido é intitulado de *CodeClass* e será o protagonista numa proposta que visa popularizar o ensino da criptografia, por meio das tecnologias, podendo servir de porta de entrada para se trabalhar, além deste assunto, diversos outros conteúdos na educação básica. Como resultados obtidos, foi realizado além do desenvolvimento do aplicativo, uma atividade como exemplo neste contexto educacional. Assim, foi observado que o uso das tecnologias estimula e instiga os alunos durante o processo de aprendizagem e que é possível utilizarmos a criptografia para abordar assuntos como funções e aritmética básica de uma maneira bem menos densa.

Palavras-Chaves: Criptografia, RSA, CodeClass, Números Primos, Aritimética

ABSTRACT

Cryptography is seldom taught in basic education since the Brazilian curricular standards (also known as BNCC) do not prescribe cryptography as a mandatory component. In this digital era, cryptography may serve as an introduction to several other subjects that make up the high school curriculum. Given the stated above, this paper aims to develop a application that provides a framework for teaching encryption in high school. The application is called CodeClass and through the use of this technology we propose to popularize the teaching of cryptography and help the teaching of various other contents of high school mathematics. Beyond the application development, an activity was performed as an example of its use in an educational context. Thereby, we observed that the use of CodeClass motivates students during the learning process and that cryptography can be used to address topics such as functions and basic arithmetic in a much less dense manner.

Key-words: Encryption, RSA, CodeClass, Prime Numbers, Arithmetic.

SUMÁRIO

1. INTRODUÇÃO

O mundo mudou! O que antes era feito com um relógio despertador ou, em alguns casos, com o cocoricar de um galo, hoje é feito pelo seu celular. As ruas que antes eram lentas e vagarosas, estão cada vez mais expressas e densas de veículos automotores. Se antes para obter notícias era preciso esperar o horário do jornal no rádio, ou adquirir um impresso em uma banca, hoje enquanto estamos desatentos as notícias chegam, e chegam, às vezes, antes mesmo de acontecer.

Ainda assim, existem coisas no mundo que não mudaram e uma delas é a necessidade de ocultar informações. Esta necessidade não aparece com a difusão dos recursos tecnológicos, mas sim dos fatos mais corriqueiros que temos no dia a dia. Para exemplificar, imagine a situação de uma partida de xadrez acirrada, em que o adversário não consegue ler a sua mente para prever qual a sua estratégia e traçar, assim, um plano de defesa ou contra-ataque antecipadamente. Somente após o movimento é que você poderá pensar no que pode ser feito. Em guerras ou batalhas por poder, o princípio que rege a estratégia é o mesmo. Um comandante precisa informar à suas tropas qual rumo tomar na batalha e quais estratégias irão adotar. Porém, o inimigo também quer saber qual a estratégia será usada para não ser pego desprevenido e é a partir dessa demanda que surge a necessidade de começarmos a esconder certas informações.

Como se já não bastasse, com o advento da tecnologia, passamos todos a ficar conectados um ao outro, numa gigantesca rede chamada *Internet*. Nesta rede, cada computador do mundo está, literalmente, ligado um ao outro. Dentro deste contexto, torna-se inevitável a utilização de um meio que sirva para acobertar este tráfego, de modo que certas informações não sejam interceptadas por destinatários não autorizados. Assim, um dos meios utilizados para proteção de informações é conhecido como a *criptografia*. “A criptografia estuda os métodos para codificar uma mensagem de modo que só seu destinatário legítimo consiga interpretá-la.” [1].

Para contextualizarmos, um dos primeiros a utilizar um sistema de criptografia foi Júlio César (100 a.C). Na Figura 1, intitulado Cifra de César, é feita uma substituição, em todo texto, de cada letra por uma outra que é obtida através da

permutação escolhida previamente. Na segunda linha, temos a imagem de cada letra da primeira linha após a permutação.

Figura 1 Cifra de César

A	B	C	D	E	F	G	H	I	J	K	L	M
D	E	F	G	H	I	J	K	L	M	N	O	P

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Fonte: O autor

Observe que nessa situação, ao cifrarmos a palavra JULIO CESAR, obtemos MXOLR FHVDU.

Este sistema, apesar de ser inédito e eficiente para a época, é bastante frágil. Visto que se for observado certo padrão de repetição no texto, consegue-se facilmente quebrar o segredo deste código.

Antes dele, 404 a.C, existem registros de que:

> Lisandro de Esparta, recebeu um mensageiro ensanguentado e ferido, único sobrevivente de um grupo de cinco que partira da Pérsia numa árdua jornada. O mensageiro lhe entregou seu cinturão, que Lisandro enrolou entorno de seu citale para descobrir que o persa Farnabazo estava planejando atacá-lo. Graças ao citale, Lisandro estava preparado para o ataque, e o repeliu. [2]

Este registo, relata sobre o primeiro aparelho criptográfico militar, o *Citale Espartano.* Este método se consiste em um bastão de madeira, em que se enrola uma fita e nela escreve-se a mensagem no sentido do comprimento do bastão. Ao desenrolar a fita, as letras ficarão em uma ordem um tanto quanto confusas e somente enrolando a mensagem em um bastão de mesmo diâmetro será possível tornar legível seu conteúdo.

Apesar de antiga, mas amplamente utilizada nesta era digital, a criptografia ainda não é trabalhada nas escolas como um conteúdo em si e, menos ainda, como uma aplicação prática dos conteúdos abordados principalmente no ensino médio.

Fazendo uma análise das competências e habilidades da cartilha Orientações Para o Ensino Médio Área: Matemática [3]; produzida pela secretária de Educação da Bahia.

Observamos que muitos dos conteúdos que são sugeridos no EIXO 1 - LINGUAGEM, ESTRUTURAS E ABSTRAÇÕES MATEMÁTICAS**,** poderiam ser trabalhados de uma maneira que realmente demonstrasse a real relevância dos mesmos no cotidiano do estudante.

Assim, surge a ideia de elaborar uma proposta que apresentasse o tema *Criptografia* no contexto disciplinar estudado pelo aluno no seu ensino regular. Como em nossas pesquisas, não encontramos ferramentas que trabalhassem este conteúdo de uma maneira mais lúdica e que simulasse o que realmente acontece na troca de informações pela *Internet*, decidimos desenvolver um aplicativo para ser utilizado justamente com este fim. [1]

Denominado de *CodeClass*, desenvolvemos um aplicativo, que nasceu com a ideia de apresentar aos estudantes como funcionam os sistemas de criptografia na era digital. Nele, além de visualizar a real necessidade das evoluções do ramo, os usuários poderão perceber o que as máquinas fazem para codificar e decodificar um texto, dando sentido às funções e aritmética modular que são aprendidas no decorrer das aulas regulares da educação básica.

Para usuários mais avançados, que muitas vezes venham a ser discentes ou formados em matemática, este aplicativo poderá ilustrar como alguns teoremas mais avançados da aritmética modular são computacionalmente aplicados servindo a eles também como um instrumento de contextualização.

Nosso trabalho foi dividido em algumas etapas que serviram para nortear o nosso desenvolvimento. Incialmente, no Capítulo 1 foi realizado um levantamento teórico do que é necessário para o entendimento da criptografia, além de diversas técnicas criptográficas, para que pudéssemos selecionar aquelas que mais se adequassem ao ensino básico.

O Capítulo 2, irá tratar de teoria matemática para a compreensão da criptografia linear, RSA e o processo de codificação e decodificação de uma mensagem.

---

[1] Os métodos e técnicas utilizadas para a criação do aplicativo vai ser melhor explicada no Capítulo 2.

Para tratar da criptografia em si, escolhemos dois sistemas. O primeiro é a criptografia linear, ele será utilizado para servir de porta de entrada para os demais métodos, pois pode ser facilmente explorada manualmente, sem a necessidade de computadores ou calculadoras mais avançadas. No Capítulo 3, iremos definir como este método se desenvolve.

A segunda técnica criptográfica que decidimos implementar, foi o Método RSA, que também será apresentada e discutida no Capítulo 3. A saber, o RSA é uma técnica utilizada mundialmente na *Internet* e não envolve algo que seja extremamente complexo de se explanar.

Ambos os sistemas foram programados para ser utilizado no aplicativo *CodeClass*, que foi desenvolvido neste trabalho. Este aplicativo foi construído por meio da plataforma gratuita de programação em bloco *MIT App Inventor II* para ser utilizado em sistemas *Android*.

Ao final, será apresentado ainda no Capítulo 3, como foi realizada a oficina interativa que buscou, além de testar as funcionalidades do aplicativo, apresentar os principais conceitos da criptografia e a necessidade da sua evolução.

Com isso, pudemos avaliar se a proposta seria realmente relevante para os estudantes e, assim, podermos ir adaptando tanto o *CodeClass*, como também as nossas sugestões, para serem cada vez mais efetivas no âmbito da educação básica e que possam vir a fundamentar a continuação deste trabalho sob outras perspectivas nas considerações finais.

.

2. REFERÊNCIAL TEÓRICO

Para um bom entendimento de tudo que será exposto neste trabalho, faz-se necessário a compreensão de alguns conceitos que vão além da Base Nacional Comum Curricular (BNCC) [4]. Para tanto, este capítulo discutirá: Algoritmo da divisão; Divisibilidade; MDC e o Algoritmo de Euclides, Números primos, Funções Aritméticas a Função ϕ de Euler, Congruências, Equações Diofantinas, A Interface do Sistema MIT App Inventor II.

2.1 Algoritmo da Divisão

Iremos apresentar agora, um dos algoritmos que mais encantam na Aritmética dos Inteiros: o Algoritmo da divisão de Euclides. Este algoritmo que, segundo Santos, já configurava registros de sua aparição no livro VII dos Elementos de Euclides, escrito por volta de 300a.C [5]. Iremos partir do princípio que o conhecimento das operações elementares, bem como suas principais propriedades e resultados do conjunto dos números inteiros já são de prévio conhecimento de quem aprecia este trabalho.

Teorema 1 (Algoritmo da Divisão). Sejam a e b, com $a > \mathrm{b}$ dois números inteiros com $\mathrm{b} \neq 0$. Existem dois únicos números inteiros q e r tais que

$$a = b.q + r, \text{com } 0 \leq r < |b|.$$

Demonstração. Vamos considerar o conjunto

$$S = \{a - b.q;\ q \in \mathbb{Z}\} \cap (\mathbb{N} \cup \{0\})$$

Devemos mostrar, além da existência de q *e* r, a unidade dos mesmos. De fato, temos que $a > b$ implica em $-b > -a$. Como $\mathbb{N}$ é ilimitado, existe $n \in \mathbb{N}$ tal que $n.(-b) > -a$, portanto S é não vazio e, além disso, $a - n.b > 0$ o que mostra que S é limitado inferiormente pelo 0. Deste modo, o Princípio da Boa Ordenação garante que S tem um menor elemento que denotaremos por r. Tomemos $r = a - b.q \geq 0$. Resta mostrar que $r < |b|$.

Suponha por absurdo, que $r \geq |b|$ então, existe $s \in \mathbb{N} \cup \{0\}$ tal que

$r = |b| + s$ o que implica $s = r - |b|$ que é uma contradição, visto que esta igualdade mostra que $s \in S$ e $s < r$ que é o menor elemento de S. Absurdo. Portanto $r < |b|$.

Para a unidade, suponhamos que existam também, q' e r' pertencentes a $\mathbb{Z}$ tais que $a = b.q + r = b.q' + r'$ onde $0 \leq r < |b|$ e $0 \leq r' < |b|$. Observe que $-|b| < -r \leq 0$ e $0 \leq r' < |b|$, somando estas últimas desigualdades obtemos

$$-|b| < r' - r < |b|$$

$$|r' - r| < |b|$$

Por outro lado $b.q + r = b.q' + r'$ implica que $b.(q - q') = r' - r$, daí $|b|.|q - q'| = |r' - r|$. Unificando as expressões, teremos

$$|b|.|q - q'| = |r' - r| < |b|$$

mas isto só faz sentido se $q = q'$ e $r = r'$. O que prova a unidade de q e r.

■

Apesar de nos chamar a atenção o fato de q e r serem únicos, este não é um algoritmo muito eficiente para fazermos divisões. Afinal, teríamos que testar os quocientes um a um a fim de encontrar aquele que permitisse a sobra de um resto menor que $|b|$. Porém é dele que surge a definição de congruência que será de extrema importância em nossos estudos.

2.2 Divisibilidade

Definição 1. Sejam a e b inteiros, dizemos que a divide b, denotando por $a \mid b$, se existir um inteiro c tal que $b = a.c$.

Se a não divide b escrevemos $a \nmid b$.

Por exemplo, podemos dizer que $2 \mid 10$, pois $10 = 2.5$ e que $3 \nmid 10$ visto que 3 não é um fator de 10.

Não iremos citar aqui as propriedades que está operação possui, podendo ser consultada no livro do Plínio em [5].

2.3 MDC e o Algoritmo de Euclides

Uma das principais partes de um sistema criptográfico é determinar as chaves que codificam e decodificam uma mensagem. Para que entendamos um dos métodos que serão apresentados no próximo capítulo, precisamos compreender a forma como conseguimos determinar o MDC de um número. Veremos nesta seção que isso será feito de maneira mais eficiente quando utilizando o *Algoritmo de Euclides*.

Definição 2. *O máximo divisor comum* de dois inteiros a e b (a ou b não nulo), é o maior número inteiro que divide a e b. Denotaremos por $mdc(a,b)$ este inteiro.

Teorema 2. Seja $d = mdc(a,b)$, então existem inteiros r e s tais que $d = r.a + s.b$.

Demonstração. Esta demonstração encontra-se no livro do Coutinho [1] páginas 22 e 23.

Proposição 1. Se $c > 0$ e a e b são divisíveis por c, então

$$mdc\left(\frac{a}{c},\frac{b}{d}\right) = \frac{1}{c}mdc(a,b)$$

Demonstração: Por hipótese, como a e b são divisíveis por c, temos que $\frac{a}{c}$ e $\frac{b}{c}$ são inteiros.

Para provarmos este resultado, iremos utilizar o seguinte resultado que consta demonstrado na página 6 do livro *Introdução à Teoria dos Números* [5], que diz:

Para todo inteiro t, $mdc(ta,tb) = t.mdc(a,b)$

Assim, basta fazermos "a"$= \frac{a}{c}$ e "b"$= \frac{b}{c}$ e tomarmos $t = \frac{1}{c}$.

■

Teorema 3. Se a e b são inteiros e $a = q.b + r$, onde q e r são inteiros, então $mdc(a,b) = mdc(b,r)$.

Demonstração. Observe que todo divisor comum de b e r é um divisor comum de a. E, caso se isolarmos $a - q.b$, podemos observar que todo divisor comum de a e b é também divisor comum de r. Portando o conjunto dos divisores comuns de a e b é igual ao conjunto dos divisores comuns de b e r, o que garante $mdc(a,b) = mdc(b,r)$.

■

Agora iremos tratar do *Algoritmo de Euclides* uma importante ferramenta quando necessitamos determinar o MDC de dois números. Para que a demonstração não fique tão densa, vamos desenvolver antes, um exemplo numérico, para que o leitor possa acompanhar as etapas da prova.

Queremos determinar o $mdc(1326,1001)$, para isso vamos utilizar o algoritmo da divisão para determinar o resto de 1326 por 1001, em seguida iremos dividir 1001 pelo resto encontrado na primeira iteração e assim sucessivamente até obtermos resto 0.

$$1326 = 1.1001 + 325$$

$$1001 = 3.325 + 26$$

$$325 = 12.26 + 13$$

$$26 = 2.13 + 0$$

A última linha nos indica que $mdc(26,13) = 13$, pois $13|26$. Se utilizarmos o Teorema 3, teremos então que

$$mdc(1326,1001) = mdc(1001,325) = mdc(325,26) = mdc(26,13) = 13.$$

Teorema 4 (O Algoritmo de Euclides) Sejam $r_0 = a$ e $r_1 = b$ inteiros não-negativos com $b \neq 0$. Se o algoritmo da divisão for aplicado sucessivamente para se obter

$$r_j = q_{j+1}.r_{j+1} + r_{j+2}, \quad 0 \leq r_{j+2} < r_{j+1}$$

Para $j = 0,1,2,\ldots,n-1$ e $r_{n+1} = 0$ então $mdc(a,b) = r_n$, o último resto não-nulo.

Demonstração. Vamos aplicar o algoritmo da divisão, para dividir $r_0 = a$ por $r_1 = b$

$$r_0 = q_1.r_1 + r_2$$

dividindo r_1 por r_2, e assim por diante quantas vezes foram necessárias, obtemos a sequência de equações:

$$r_0 = q_1.r_1 + r_2 \qquad 0 < r_2 < r_1$$

$$r_1 = q_2.r_2 + r_3 \qquad 0 < r_3 < r_2$$

$$r_0 = q_3.r_3 + r_4 \qquad 0 < r_4 < r_3$$

$$\vdots$$

$$r_{n-2} = q_{n-1}.r_{n-1} + r_n \qquad 0 < r_n < r_{n-1}$$

$$r_{n-1} = q_n.r_n + 0$$

finalmente, aplicando o Teorema 3 nas equações acima concluímos que

$$r_n = mdc(r_n, r_{n-1}) = mdc(r_{n-1}, r_{n-2}) = \cdots = mdc(r_1, r_2) = mdc(r_0, r_1) = mdc(a, b)$$

■

2.4 Números Primos

Definição 3. Um número inteiro $n > 1$ é primo, se possui somente dois divisores positivos n e 1. Se $n > 1$ não é primo, dizemos que n é composto.

O teorema a seguir foi proposto e demonstrado por Gauss em 1976. Ele é fundamental para o funcionamento do Sistema RSA, bem como garanti que não pode existir uma duplicação de chaves neste sistema.

Teorema 5 (Teorema Fundamental da Aritmética). Todo inteiro maior do que 1 pode ser representado de maneira única (a menos de ordem) como um produto de fatores primos.

Demonstração: Se n é primo não há nada a ser demonstrado. Suponhamos que n seja um número composto. Seja $p_1 > 1$ o menor dos divisores positivos de n. Observe que p_1 é primo, pois caso contrário existirá um $p_i < p_1$ tal que $p_i | n$. Assim, podemos escrever n como sendo o produto de $p_1.n_1$.

Se n_1 for primo, a prova está completa. Caso contrário, tomamos p_2 como o menor o menor fator de n_1. Assim, p_2 será primo e podemos escrever $n = p_1.p_2.n_2$.

Repetindo esse processo, obtemos uma sequência finita e decrescente de inteiros positivos $n_1, n_2, \dots, n_r$. Como os primos na sequência $p_1, p_2, \dots, p_k$ não são necessariamente distintos, n será da forma

$$n = p_1^{\alpha_1}.p_2^{\alpha_2} \dots p_k^{\alpha_k}$$

Para mostrarmos que essa fatoração é única, aplicamos a indução sobre n. Para $n = 2$ é trivial que a afirmação é verdadeira. Suponha que ela se mantenha verdadeira para todos os inteiros maiores do que 1 e menores do que n. Mostremos que ela também é válida para n.

Suponha que n é composto e tenha duas fatorações, ou seja,

$$n = p_1.p_2 \dots p_s = q_1.q_2 \dots q_r$$

Iremos mostrar que $r = s$ e que cada p_i é igual a algum q_i. Como p_1 divide o produto $q_1 q_2 \dots q_r$ então ele deve dividir pelo menos um dos fatores q_j.

Suponha sem perda de generalidade que $p_1 | q_1$ e como ambos são primos, podemos concluir que $p_1 = q_1$. Assim, $1 < \frac{n}{p_1} = p_2.p_3 \dots p_s = q_2.q_3 \dots q_r < n$ que por hipótese de indução possui fatoração única e, portanto $r = s$ e, a menos de ordem, as fatorações $p_1.p_2 \dots p_s$ e $q_1.q_2 \dots q_r$ são iguais.

■

2.5 Funções Aritméticas a *Função ϕ de Euler*

Denominamos de *função aritmética,* uma função definida para todos os inteiros positivos. No campo da Teoria dos Números, existem diversas funções aritméticas, porém nessa seção trataremos apenas da Função ϕ de Euler, que faz parte da mecânica do algoritmo RSA que será tratado no capítulo seguinte.

Definição 4. Se n é um inteiro positivo, a função ϕ de Euler, denotada por $\phi(n)$, é definida como sendo o número de inteiros positivos menores ou iguais a n, que são relativamente primos com n.

Por exemplo, podemos verificar que $\phi(8) = 4$, pois o conjunto dos números t de 1 a 8 tais que o $mdc(8, t) = 1$ se restringe a $1,3,5,7$, ou seja possui apenas 4 elementos.

Uma observação interessante, é que se p for um número primo tal que $p \geq 2$ então $\phi(p) = p - 1$, afinal nenhum dos números menores que p possui um divisor comum com ele.

Teorema 6. A função ϕ de Euler é multiplicativa, isto é, $\phi(m.n) = \phi(m).\phi(n)$ para $mdc(m,n) = 1$.

Demonstração: Iremos omitir aqui a demonstração deste teorema, mas ela encontra-se no livro do Abramo Hefez [6] pg. 199.

Corolário 1. Se p e q são primos, então $\phi(p.q) = (p-1).(q-1)$

Demonstração: Pelo teorema anterior, como $\phi(p.q) = \phi(p).\phi(q)$ e do fato de p e q serem primos, temos que $\phi(p) = p - 1$ e $\phi(q) = q - 1$.

■

2.6 Congruências

Dizemos que dois números a e b são congruentes módulo n se os restos de sua divisão euclidiana por n são iguais, escrevendo

$$a \equiv b \, mod \, n$$

Como exemplo, podemos observar que $23 \equiv 5 \, mod \, 18$, pois sabemos que ao dividirmos 23 por 18 obtemos quociente 1 e resto 5. Uma outra abordagem interessante para essa definição, é dizermos que $n \mid (a-b)$ ou equivalentemente $a - b = n.k$ para algum $k \in \mathbb{Z}$.

Alguns resultados a respeito das operações com congruências podem ser encontrados no livro do José Plínio [5]. Nos ateremos aqui, nos teoremas e proposições fundamentais para compreensão da criptografia.

Proposição 4. Se a, b, k e n são inteiros com $k > 0$ e $a \equiv b \, mod \, n$, então $a^k \equiv b^k \, mod \, n$.

Demonstração: Por hipótese temos que $a - b = n.l$, se utilizarmos a identidade

$$a^k - b^k = (a-b).(a^{k-1} + a^{k-2}b + a^{k-3}b^2 + \cdots + ab^{k-2} + b^{k-1})$$

e chamando $a^{k-1} + a^{k-2}b + a^{k-3}b^2 + \cdots + ab^{k-2} + b^{k-1}$ de m, teremos

$$a^k - b^k = (a-b).m = n.l.m = n.(l.m)$$

o que implica em $a^k \equiv b^k \, mod \, n$.

■

Teorema 6 (Pequeno Teorema de Fermat). Se p é primo, então $a^p \equiv a \, mod \, p$, para todo inteiro a.

Demonstração: Iremos fazer a prova deste resultado para a natural utilizando o princípio de indução em a.

Note que para $a = 1$ o resultado é trivial, visto que $1^p \equiv 1 \, mod \, p$.

Suponha que exista um $a \geq 1$ tal que $a^p \equiv a \, mod \, p$. Mostraremos a validez de proposição para $a + 1$.

Note que ao utilizarmos a expansão por Binômio de Newton, temos

$$(a+1)^p = a^p + \binom{p}{p-1} a^{p-1} + \cdots + \binom{p}{i} a^i + \cdots + \binom{p}{1} a + 1 \qquad (1)$$

Aplicando congruências módulo p em cada parcela observamos que a primeira delas, é congruente a $a \, mod \, p$ pela hipótese de indução. As demais são fatores de p e, portanto, para $1 \leq i \leq p-1$ teremos

$$\binom{p}{i} \equiv 0 \, mod \, p$$

Logo, $(a+1)^p \equiv a^p + 1 \equiv a + 1 \, mod \, p$ o que pelo princípio de indução, prova nosso teorema para o caso de a natural.

Note que se a for um número negativo, as congruências acima teriam o mesmo valor, o que torna o resultado indiferente para a inteiro.

■

Teorema 7 (Teorema de Euler). Se m é um inteiro positivo e a um inteiro tal que $mdc(a, m) = 1$, então

$$a^{\phi(m)} \equiv 1 \, mod \, m$$

Demonstração: A demonstração deste teorema se encontra nas páginas 49 e 50 do livro do Martinez [7].

Teorema 8 (Teorema de Wilson): Se p é primo, então $(p-1)! \equiv -1 \, mod \, p$.

Demonstração: Inicialmente observe que para $p = 2$, temos que $(2-1)! = 1! = 1 \equiv -1 \, mod \, 2$ que é válido. Iremos afirmar, mas sem demonstrar, que a congruência $a.x \equiv 1 \, mod \, p$, possui uma única solução para todo a no conjunto

$\{1,2,3,\ldots,p-1\}$. A prova deste resultado encontra-se demonstrado no livro do Jose Plínio de Oliveira Santos em [5] páginas 37 e 38.

Neste conjunto, note que somente 1 e $p-1$ são eles mesmos o seu próprio inverso módulo p. Restando então os números $2,3,4,\ldots,p-2$; $(p-3)$ −números, que podem ser agrupados cada um com seu inverso módulo p, formando $\frac{p-3}{2}$ congruências. Por exemplo, digamos que 2 e $(p-i)$ sejam inversos um do outro, podemos então formar a congruência

$$2.(p-i) \equiv 1\ mod\ p$$

Assim, ao multiplicarmos todas estas congruências, obteremos:

$$2.3.4.\ldots.(p-2) \equiv 1\ mod\ p$$

multiplicando esta última por $(p-1)$ teremos

$$2.3.4.\ldots.(p-2).(p-1) \equiv p-1 \equiv -1\ mod\ p$$

como o lado esquerdo desta igualdade é $(p-1)!$, temos então que

$$(p-1)! \equiv 1\ mod\ p$$

o que prova o resultado.

■

2.7 Equações Diofantinas

Definição 5. Uma equação da $a.x+b.y=c$, onde a,b e c são inteiros é chamada Equação Diofantina Linear.

Iremos analisar esta equação, pois para determinar o inverso de um número a módulo n, devemos determinar qual número b satisfaz a condição

$$a.b \equiv 1\ mod\ n$$

Ora, mas isto é o mesmo que determinar $a.b-1=n.k$ e daí

$$a.b+n.(-k)=1$$

que é uma Equação Diofantina com $c=1$.

Teorema 9. Sejam a e b inteiros e $d = mdc(a, b)$. Se $d \nmid c$ então a equação $a.x + b.y = c$ não possui nenhuma solução inteira. $Se\ d|c$ ela possui infinitas soluções e se $x = x_0$ e $y = y_0$ é uma solução particular, então todas as soluções são dadas por

$$x = x_0 + \frac{b}{d}.k$$

$$y = y_0 - \frac{a}{d}.k$$

onde k é um inteiro.

Demonstração: Começamos observando que se $d \nmid c$, a equação não possui solução, afinal como d é um divisor comum de a e b, obviamente ele deveria dividir c.

Suponhamos que $d|c$ pelo Teorema 2, temos que existem m_0 e n_0 tais que

$$a.m_0 + b.n_0 = d \qquad (2).$$

Da hipótese $d|c$, existe um $k \in \mathbb{Z}$, tal que $c = d.k$. Multiplicando a equação (8) por k obtemos $a.(m_0.k) + b.(n_0.k) = d.k = c$, o que nos leva a conclusão que $x_0 = m_0.k$ e $y_0 = n_0.k$, é uma solução de $a.x + b.x = c$.

Vamos verificar agora que o par (x, y) em que:

$$x = x_0 + \frac{b}{d}.k$$

$$y = y_0 - \frac{a}{d}.k$$

são também soluções da equação.

De fato, temos:

$$a.x + b.y = a.\left(x_0 + \frac{b}{d}.k\right) + b.\left(y_0 - \frac{a}{d}.k\right)$$

$$= a.x_0 + \frac{a.b}{d}.k + b.y_0 - \frac{a.b}{d}.k$$

que cancelando obtemos:

$$= a.x_0 + b.y_0 = c.$$

Dessa forma, concluímos que a partir de uma solução particular da Equação Diofantina, podemos gerar outras infinitas variando pois, somente o parâmetro k.

Mostraremos agora que toda solução da equação $a.x + b.y = c$ é da forma:

$$x = x_0 + \frac{b}{d}.k$$

$$y = y_0 - \frac{a}{d}.k$$

Suponha que os pares (x, y) e (x_0, y_0) sejam soluções da Equação Diofantina. Assim,

$$a.x + b.y - a.x_0 - b.y_0 = a.(x - x_0) + b.(y - y_0) = 0$$

e, portanto,

$$a.(x - x_0) = b.(y_0 - y)$$

Como $mdc(a, b) = d$, temos pelo Corolário 1 que $mdc\left(\frac{a}{d}, \frac{b}{d}\right) = 1$. Portanto, dividindo os dois membros da última igualdade por d, teremos:

$$\frac{a}{d}.(x - x_0) = \frac{b}{d}.(y_0 - y) \qquad (3)$$

Todavia, como $mdc\left(\frac{a}{d}, \frac{b}{d}\right) = 1$, $\frac{b}{d} | (x - x_0)$ portanto existe $k \in \mathbb{Z}$ tal que: $x - x_0 = k.\frac{b}{d}$, ou seja $x = x_0 + k.\frac{b}{d}$.

Substituindo a solução encontrada na equação (9) teremos

$$\frac{a}{d}.\left(x_0 + k.\frac{b}{d} - x_0\right) = \frac{b}{d}.y_0 - \frac{b}{d}.y$$

$$\frac{b}{d}.y = \frac{b}{d}.y_0 - \frac{a.b}{d^2}.k$$

$$y = y_0 - \frac{a}{d}.k.$$

Como queríamos demonstrar.

■

Teorema 10 (Teorema Chinês do Resto). Se $mdc(a_1, m_i) = 1$, $mdc(m_i, m_j) = 1$ para $i \neq j$ e c_i inteiro, então o sistema

$$\begin{aligned} a_1.x &\equiv c_1 \ mod \ m_1 \\ a_2.x &\equiv c_2 \ mod \ m_2 \\ a_3.x &\equiv c_3 \ mod \ m_3 \\ &\vdots \\ a_r.x &\equiv c_r \ mod \ m_r \end{aligned}$$

possui solução e a solução é única módulo m, onde $m = m_1.m_2 \ldots . . m_r$

Demonstração: Esta demonstração está disponível no livro do Martinez [7] nas páginas 81 e 82.

■

2.8 MIT App Inventor 2

Lançado em 15 de dezembro de 2010, o *App Inventor for Android* foi um *software web*, cuja proposta era dar a um usuário leigo em qualquer linguagem de programação, meios para que o mesmo pudesse desenvolver um aplicativo para ambiente *Android*. Para isso, o programador utiliza de blocos pré-definidos, encaixáveis que sintetizam de maneira prática os comandos que viriam a ser digitados. A interface do programa permite que o utilizador usando o mouse para arrastar e soltar os componentes, comandos e funções, encaixe blocos um no outros compondo o código fonte do sistema que está sendo desenvolvido.

Esta plataforma foi desenvolvida pela equipe *App Inventor Team* incorporada ao *Google*, e liderada por Hal Abelson e Mark Friedman. Posteriormente em parceria com o *Massachusetts Institute of Technology* (MIT), foram realizadas melhorias na interface e estabilidade do sistema em 6 de dezembro de 2013, uma nova versão do *software* foi lançada e batizada de *MIT App Inventor 2*. A atualização o tornou bastante funcional e estável, o que permitiu a criação de aplicativos bem mais completos do que a primeira versão possibilitava. Atualmente é possível trabalhar com bases de dados em nuvem, desenvolver jogos, sistemas de automação comercial, dentre outros programas que de certo modo admitem um elevado grau de complexidade.

Claro que esta proposta de programação pode limitar um pouco as possibilidades de desenvolvimento, visto que seria impossível disponibilizar todas as combinações de blocos possíveis para o usuário, o que na programação convencional, com escrita manual do código fonte, o usuário tem a possibilidade de programar ao seu bel prazer, sem restrições de criação.

Em 2016, o Prof. Dr. Eduardo Alves do Valle Junior, da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas (Unicamp), finalizou um projeto de tradução do *MIT App Inventor 2* para a Língua Portuguesa. Isto mobilizou o surgimento de alguns projetos voltados para o incentivo da programação na educação básica, como o Projeto Computação na Escola: “A Iniciativa Computação na Escola é dedicada a aumentar o ensino de computação no

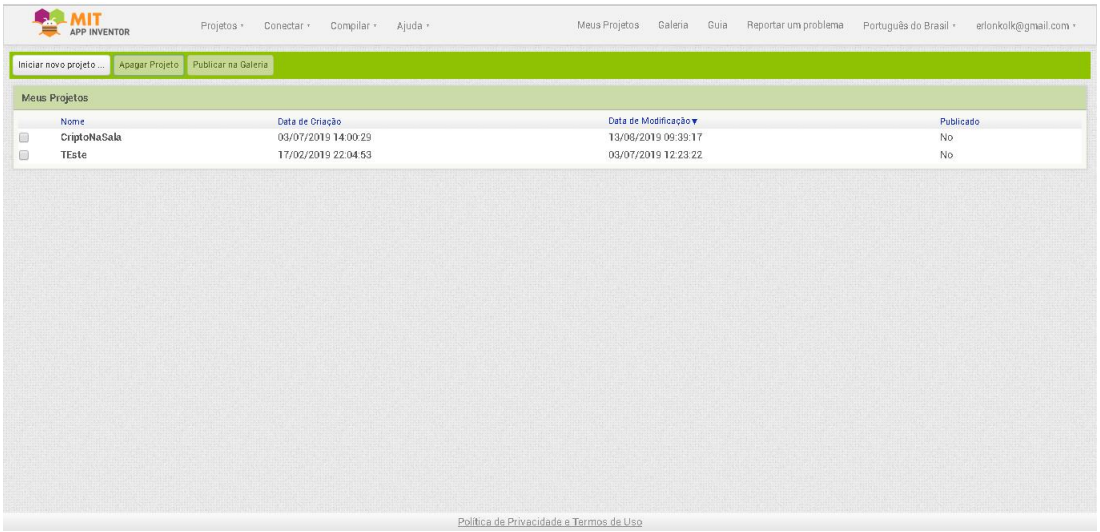
Ensino Fundamental e Médio. “Nossa visão é que todos os alunos em todas as escolas devem ter a oportunidade de aprender computação.” [8]

O projeto acredita que a computação deve fazer parte do currículo no Ensino Fundamental e Médio assim como as demais disciplinas. No site do projeto disponível em [9], há bastante material sobre outros projetos que o mesmo desenvolve, inclusive outras plataformas com a mesma finalidade do *MIT App*.

2.1.1. A Interface do Sistema

Para acessar o *MIT App Inventor 2* é necessário acessar o link [10] e “logar” com uma “Conta *Google*”. Esse tipo de conta é obrigatória para acessar as funcionalidades de qualquer aparelho celular com o sistema operacional *Android*. Ao acessar a plataforma, o usuário irá se deparar com uma tela de gerenciamento de projetos como a da Figura 2:

Figura 2 Tela Inicial MIT App Inventor II



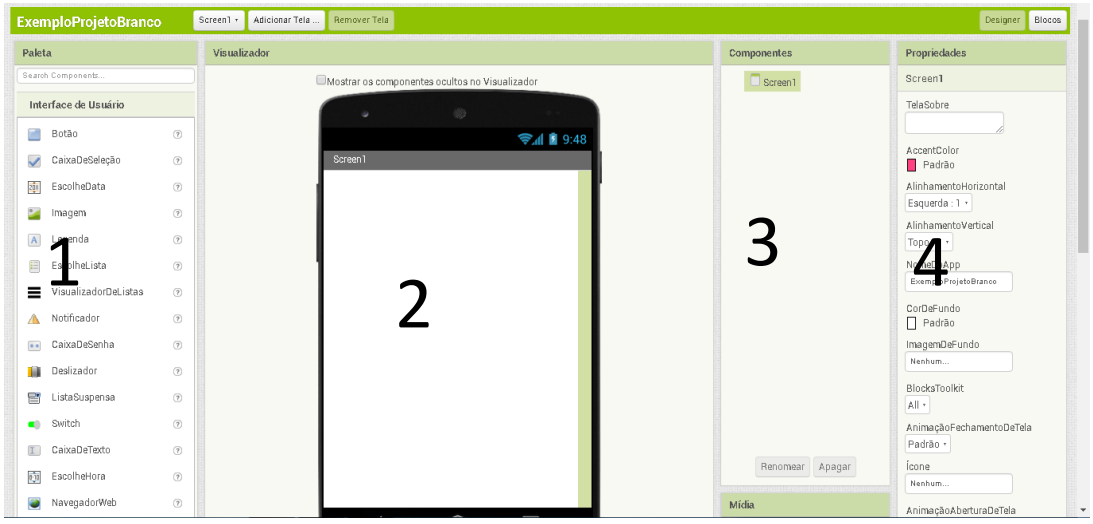
Fonte: http://ai2.appinventor.mit.edu. Acessado em 08/10/2019

Na tela de boas vindas conforme a Figura 2, o usuário pode gerenciar seus projetos, criar, apagar, copiar, duplicar ou até mesmo acompanhar o desenvolvimento, quando feito em parceria com outros programadores. Caso não haja nenhum projeto em andamento ou concluído, será apresentada uma lista vazia e nesse caso o usuário deve iniciar um novo projeto do zero, seu primeiro projeto. É importante salvar os projetos com nomes que sejam de fácil identificação e/ou a sua versão, caso seja o caso. A plataforma não permite a utilização de caracteres especiais ou espaços em

branco no nome do projeto. Na tela inicial também é possível ajustar o idioma da plataforma e acessar o menu de ajuda.

Após criar um projeto e nomeá-lo, o usuário irá acessar uma segunda tela, agora com todos os elementos necessários para iniciar o processo de criação do aplicativo.

Figura 3 Tela de Designer de Aplicativo do MIT App Inventor II



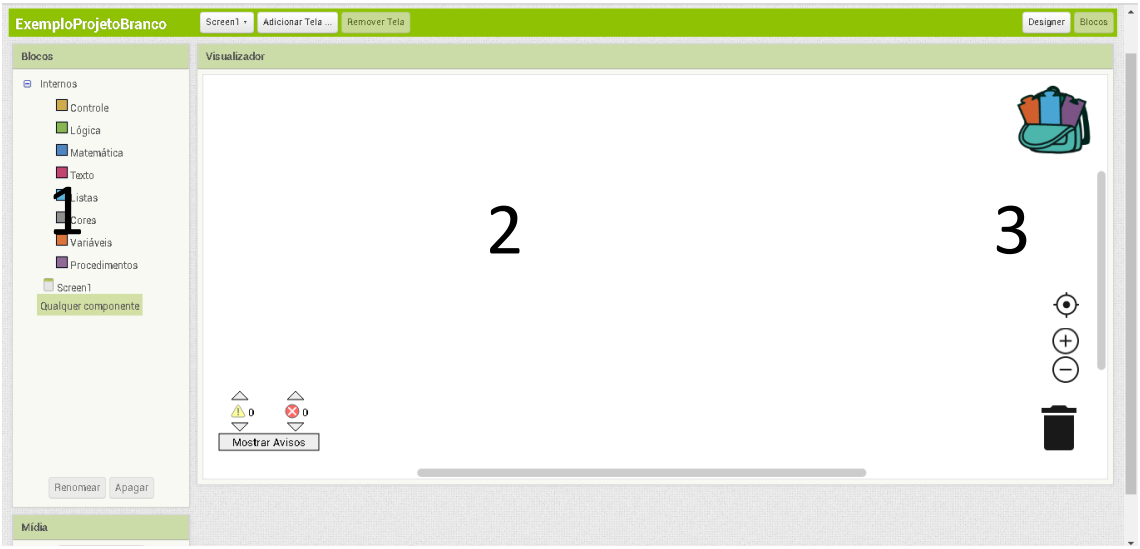
Fonte: Imagem trabalhada pelo autor

A tela de desenvolvimento, possui quatro regiões com diferentes funcionalidades. A primeira, indicada na Figura 3, com o número 1, é o banco de componentes que podem ser utilizados no *App*, nele o usuário terá acesso a botões, caixas de texto, legendas, caixas para fotos, base de dados, câmera, entre outros. Basicamente os componentes são os meios pelos quais possibilitam a interação entre o aplicativo e quem o utiliza. Em seguida, vem a Zona 2, que o local onde o programador pode ajustar os componentes na interface do aplicativo, dispondo-os da maneira como será apresentada no projeto final. A Zona 3, é uma listagem de todos os adereços que compõe a tela do aplicativo. Isso auxilia na navegação de quem está desenvolvendo um sistema que tenha inúmeros componentes e, que muitas vezes há uma combinação de componentes, um dentro do outro. Finalmente, na Zona 4, é possível ajustar algumas configurações dos componentes e da tela que está sendo desenvolvida como renomear um componente, ajustar cores, tamanho dentre outras

características. Na barra superior em verde, pode-se criar outras telas (*Screens*), pois geralmente os aplicativos são formados por um conjunto de telas que interagem umas com as outras no decorrer da utilização e acessar a interface de blocos que o espaço onde realmente ocorrerá a programação do app que está sendo desenvolvido.

Ao clicar no botão blocos, o desenvolvedor terá acesso a interface “Blocos” onde poderá realizar a combinação dos *blocos de código* a cada componente que é adicionado na tela de designer, novos blocos surgem na tela de blocos, possibilitando uma vasta combinação entre eles.

Figura 4 Interface de Programação



Fonte: Imagem trabalhada pelo autor.

Podemos dividir esta tela em três zonas distintas. Na primeira temos acesso aos blocos: comandos de controle, funções, conectivos lógicos ou matemáticos, definição e utilização de variáveis tudo que é necessário para o implemento de códigos como pode ser visto na Figura 4. A zona 2 é feita a montagem, arrastando-se os blocos da Zona 1 os combinando e finalmente a Zona 3 serve para apagar algum código, dar zoom e se localizar, pois ao encaixas as peças, muitas vezes o programa acaba ficando bem extenso. Na mochila no canto superior direito, é possível guardar algumas combinações de blocos que foram utilizadas em um ou outro aplicativo e que possa ser reaproveitado em alguma outra tela, isso evita ter que “reescrever” todo o processo.

2.2. Divisão e a Segurança de Redes: a Criptografia

Para iniciarmos, vamos fazer uma análise do sistema criptográfico utilizado por Júlio César, o qual explanamos na Introdução deste trabalho. Suponha que Júlio César assinasse todas as cartas, então a pessoa saberia ao final que a última palavra seria seu nome e isto já bastaria para desvendar este tipo de permutação.

É possível melhorar um pouco esta permutação causando algumas dificuldades para desvendar seu segredo. Para isto vamos tomar por base algumas definições. Precisamos primeiro considerar que o alfabeto é ordenado e transformá-lo em números. Assim poderemos operar com os mesmos, utilizando ferramentas matemáticas. Considere a princípio uma correspondência simples em que $A = 1$, $B = 2$, $C = 3$, ... e, assim, por diante até atingirmos $Z = 26$ conforme a Figura 3.1. Para ilustrar, na Figura 5 é apresentado como será o nosso alfabeto, em que iremos construir todos os sistemas que serão trabalhados a partir deste ponto.

Figura 5 Correspondência da Cifra de César

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Fonte: O autor

Feito isso, já podemos interpretar cada mensagem como um código numérico. Por exemplo, a palavra CIDADE, seria o número 3 9 4 1 4 5.

A parte interessante de fazer essa atribuição é que cada letra do alfabeto pode ser visualizada como uma classe de equivalência em $\mathbb{Z}_{26}$ onde $A = \bar{1}, B = \bar{2}, \dots, Z = \overline{26}$. Desse modo podemos representar a Cifra de César pela transformação linear $E(\bar{X}) = \bar{X} + \bar{3}$ em $\mathbb{Z}_{26}$ e, assim, $E(\bar{A}) = E(\bar{X}) = \bar{1}A + \bar{3} = \bar{1} + \bar{3} = \bar{4} = D$ e, de modo sucessivo $E(\bar{Z}) = \overline{26} + \bar{3} = \overline{29} = \bar{3} = C$.

Esse sistema, criado por Júlio César, nos servirá de base para estudarmos uma infinidade de outros sistemas que podem ser classificados como os Sistemas de Criptografia Linear ou Cifras de Transposição, o qual será apresentado e discutido a seguir.

3. CRIPTOGRAFIA

Finalmente chegamos à parte que delineamos este estudo, a criptografia. Neste capítulo serão abordados alguns métodos criptográficos que surgiram no decorrer dos anos, bem como seu funcionamento, os prós e os contras.

3.1. Sistema de Criptografia Linear

Seja *A* um alfabeto ordenado com *L* letras.

Definição 6. Definimos como um Sistema de Criptografia Linear, toda transformação $\sigma: \mathbb{Z}_L \to \mathbb{Z}_L$ definida por $\sigma(X) = a.X + b$ onde $a, b \in \mathbb{Z}_L$ e $mdc(a, L) = 1$. Chamaremos também a e b de $chaves\ do\ sistema$.

Note que σ é uma bijeção, pois dado $Y = a.X + b$ em $\mathbb{Z}_L$ existe $X = a^{-1}.(Y - b)$, visto que da forma que a é tomado ele sempre admitirá inverso multiplicativo no $\mathbb{Z}_L$. Portanto σ é uma permutação.

Deste modo, se considerarmos, por exemplo, a permutação

$$\sigma(X) = \overline{17}^{2}.X + \overline{12}$$

teremos o seguinte código representado na Figura 6, que mostra os resultados de cada letra (alfabeto) quando aplicada a função σ.

Figura 6 Permutação segundo σ

	A	B	C	D	E	F	G	H	I	J	K	L	M
X	1	2	3	4	5	6	7	8	9	10	11	12	13
$\sigma(X)$	29	46	63	80	97	114	131	148	165	182	199	216	233
$mod\ 26$	3	20	11	2	19	10	1	18	9	0	17	8	25
	C	T	K	B	S	J	A	R	I	Z	Q	H	Y
	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
X	14	15	16	17	18	19	20	21	22	23	24	25	26
$\sigma(X)$	250	267	284	301	318	335	352	369	386	403	420	437	454
$mod\ 26$	16	7	24	15	6	23	14	5	22	13	4	21	12
	P	G	X	O	F	W	N	E	V	M	D	U	L

Fonte: O autor

[2] Usaremos a notação $\bar{a}$ apenas com o intuito de simplificar a notação de congruência por detrás das operações. Por exemplo, ao invés de escrevermos que $27 \equiv 1\ mod\ 26$ usaremos somente $\bar{1}$. A partir de agora, todas as nossas congruências serão módulo 26.

E assim, a palavra JÚLIO CÉSAR cifrada por essa permutação seria codificada como ZEHIG KSWCF. Apesar da sensação de que criamos uma verdadeira mistura nas correspondências, esse sistema possui uma falha, pois caso o invasor faça, por exemplo, uma análise de frequência da mensagem e descubra quais os caracteres mais aparecem, de modo a ter certeza de duas informações, como $\sigma(Z) = J$ e $\sigma(E) = U$, ele poderia então decodificar a mensagem resolvendo o sistema.

$$\begin{cases} a.\overline{26} + b = \overline{10} \\ a.\overline{5} + b = \overline{21} \end{cases}$$

Como $\overline{26} = \overline{0}$, este sistema se torna bem simples de resolver, pois teríamos da primeira equação que $b = 1\acute{0}$ e substituindo o resultado na segunda obteríamos

$$a.\overline{5} = \overline{11}$$

Resolver para quais valores de a, a equação $a.\overline{5} = \overline{11}$ é, em sua essência, encontrar um $a \in \mathbb{Z}_{26}$ de modo que quando dividirmos $a.\overline{5}$ e $\overline{11}$ por 26, obteremos o mesmo resto. Podemos então reescrever o problema como $a.\overline{5} \equiv \overline{11}\ mod\ 26$ que pode ser reescrito como $a.5 - 11 = 26.k$ obtendo assim a *Equação Diofantina*:

$$a.5 + 26.k = 11$$

a qual resolvendo, encontramos $a = \overline{23}$. Deste modo, teremos então que a função inversa $\sigma^{-1}(X) = \overline{23}.X + \overline{10}$ irá decodificar a mensagem. Vê-se então que o sistema pode ser bastante frágil, pois conhecidas 2 permutações sempre poderemos descobrir as chaves a e b.

3.2. Trabalhando com Blocos

Como vimos anteriormente, codificar uma única letra torna um código bastante vulnerável, visto que o estudo das frequências faz com que seja possível identificar um padrão na mensagem codificada.

Segundo Lemos, que pode ser visto em [11], uma forma de evitar ataques a um sistema de criptografia por meio de uma análise de frequência das letras é trabalhar com as mensagens enviadas em blocos. A ideia é substituir um bloco de k letras, em que k é um natural, por outro bloco de l de letras. Quando utilizamos o método Linear

na subseção 3.1, fizemos $k = 1$, iremos agora generalizar essa ideia para valores maiores.

Dado um alfabeto *A* com *L* letras, podemos representar cada bloco de k letras por um natural a na base *L* tal que $0 \leq a < L^k$. De fato, se tomarmos o bloco $(a_1\, a_2 \ldots a_k)_L$ teremos:

$$(a_1 a_2 \ldots a_k)_L = a_1 L^{k-1} + a_2 . L^{k-2} + \ldots + a_k . L^0$$

como a_i pertence ao alfabeto ordenado tal que $A = 0, \ldots A_L = L - 1$ temos que o menor bloco que possuiremos será $(00\ldots0)\ (k - zeros)$ e o maior

$$(L - 1 L - 1 \ldots L - 1)\ (k - vezes)$$

assim

$$0 . L^{k-1} + \ldots + 0 . L^0 \leq (a_1 a_2 \ldots a_k) . L < (L-1) . L^{k-1} + \ldots + (L-1) . L^0$$

e, claramente, o último membro da desigualdade pode reescrito como $(L-1).(L^{k-1} + L^{k-2} + \cdots + L^0) = L^k - 1$ por se tratar de uma P.G finita.

Em outras palavras, ao invés de considerarmos um alfabeto com apenas L letras estaremos considerando um novo alfabeto com L^k "letras", onde cada uma dessas "letras" é na verdade um bloco.

Definindo uma função σ: $\mathbb{Z}_L{}^k \rightarrow \mathbb{Z}_L{}^k$ onde $\sigma(X) = a.X + b$, com $a \in \mathbb{Z}_{L^k}$ e $b \in \mathbb{Z}$ teremos também uma permutação bem definida, pois σ também será uma bijeção.

Como exemplo, vamos transmitir uma mensagem em blocos de tamanho $k = 3$ a palavra MATEMATICA. Para dividirmos em blocos de tamanho exatamente 3, será preciso completar a palavra com a primeira letra do alfabeto[3] *A*, teremos, portanto, MAT−EMA−TIC−AAA, vamos utilizar a função de permutação

$$\sigma(X) = \overline{5779}.X + \overline{123}^{4}$$

Deste modo

$$(MAT)_{27} = 12.27^2 + 0.27^1 + 19.27^0 = 8767$$

[3] Neste exemplo utilizaremos o alfabeto definido por Lemos [11], onde $A = 0, B = 1, \ldots Z = 25$, sem que haja prejuízo a teoria. No trabalho utilizamos o alfabeto definido inicialmente apenas para facilitar o processo de programação.

[4] Considere neste caso as congruências módulo 26^3

$$(EMA)_{27} = 4.27^2 + 12.27^1 + 0.27^0 = 3240$$
$$(T\,IC)_{27} = 19.27^2 + 8.27^1 + 2.27^0 = 14069$$
$$(AAA)_{27} = 0.27^2 + 0.27^1 + 0.27^0 = 0$$

aplicando σ em cada bloco numérico, teremos

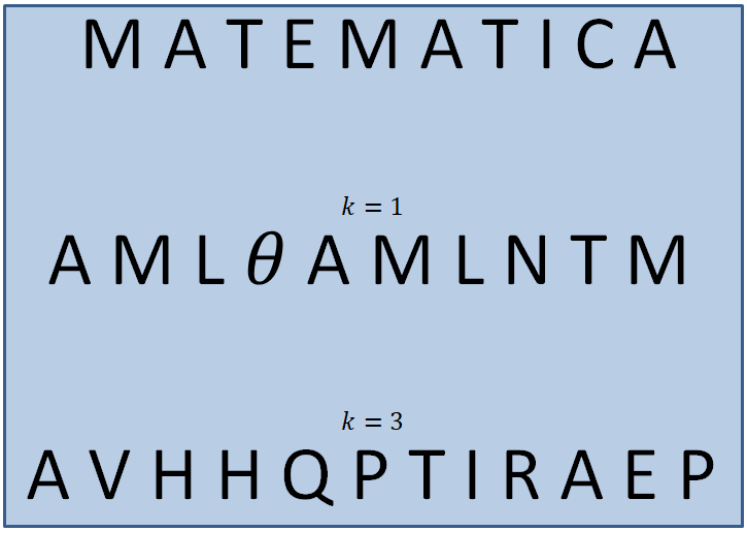
$$\sigma\,(8767) = 50664616 = 574$$
$$\sigma\,(3240) = 18724083 = 5550$$
$$\sigma\,(14069) = 81304874 = 14084$$
$$\sigma(0) = 123 = 123$$

finalmente, fazendo a correspondência, teremos que

$$574 = 0.27^2 + 21.27 + 7 = (AVH)$$
$$5550 = 7.27^2 + 16.27 + 15.27 = (HQP)$$
$$14084 = 19.27^2 + 8.27 + 17 = (TIR)$$
$$123 = 0.27^2 + 4.27 + 15 = (AEP)$$

Para que tenhamos uma ideia de quão embaralhadas ficaram as letras, vamos comparar a mesma palavra com o primeiro sistema, quando usamos $k = 1$ e $\sigma(X) = 17.X + 12$, com o arranjamento por blocos de tamanho $k = 3$.

Figura 7 Comparação $k = 1$ e $k = 3$



Fonte: O autor

Observe na Figura 7 uma melhora significativa apenas com a mudança do tamanho do bloco. Ainda segundo Lemos [11] quanto maior o bloco, maior a dificuldade em se quebrar o sistema por uma análise de frequências. Para $k = 8$ esta tarefa seria já muito complexa, se não impossível.

Usando este modelo de decodificação, o código se torna mais eficiente, porém ainda continua com falhas operacionais gravíssimas, que pode tornar sua utilização na prática pouco eficiente.

Para que o receptor leia a mensagem, é preciso ter minimamente o conhecimento das chaves a e b e, em algumas situações, o tamanho do bloco utilizado. Eis que surge então a dúvida: Como enviar as chaves para o receptor da mensagem em segurança? Afinal, reflitamos, caso tenhamos um canal seguro para envio dessas chaves, por que não enviar logo a mensagem por ele? Assim, partindo desta necessidade, nasce o conceito de chave pública, o qual iremos abordar a seguir. Idealizar um sistema criptográfico em que não seja necessário enviar previamente as chaves para que o receptor decodifique a mensagem.

3.3. O Conceito de Chave Pública

A ideia de criptografar com chave pública foi proposta de maneira teórica em 1976, por Diffie e Hellman [11]. O objetivo deste modelo era montar um sistema em que os usuários pudessem trocar mensagens sem ter que, obrigatoriamente, efetuar a troca de chaves.

Para ilustrar melhor, temos um exemplo: Suponha que Pedro e Júlia desejam trocar mensagens secretas, mas não dispõem de nenhum sistema criptográfico. Pedro poderia colocar sua mensagem em um baú e colocar um cadeado que somente ele possuísse a chave. Depois, esse baú seria enviado para Júlia, que não iria conseguir abri-lo, e esta colocaria outro cadeado no baú, que somente ela possui a chave. Novamente esse baú voltaria a Pedro que removeria seu cadeado e enviaria o baú novamente a Júlia, que ao final o receberia somente com seu cadeado, e assim, conseguiria, depois de muitas idas e vindas, ter acesso à mensagem. Claro que estamos supondo que no meio de tantas viagens esse baú poderia ser arrombado ou extraviado, mas não compete a nós decidir o destino do baú e sim compreender a maneira simbólica de fazer a criptografia acontecer.

O fato de não termos que enviar a chave em momento algum é o que transforma os sistemas de chave pública ou Assimétrica em algo muito vantajoso a

ser utilizado. Na prática esse sistema funciona da seguinte forma. Pedro e Júlia, possuem chaves de *E* e *D* de encriptação e decriptação de um mesmo sistema. Na prática essas chaves são diferentes apenas nos números que as compõem. Sejam E_p e D_p as chaves de Pedro e E_j e D_j as de Júlia. Assim, os usuários Júlia e Pedro irão esconder suas chaves de decodificação D_p e D_j e tornarão públicas suas chaves de encriptação, E_p e E_j . Deste modo, para que Pedro envie uma mensagem *M* a Júlia, basta ele aplicar sobre esta mensagem o algoritmo de encriptação de Júlia e, depois, o seu algoritmo de decriptação assim, ele obteria $D_p(E_j(M))$ e esta mensagem seria enviada a Júlia que para conseguir ler bastaria aplicar E_p e depois D_j assim teríamos

$$E_p(D_p(E_j(M))) = E_j(M)$$

daí,

$$D_j(E_j(M)) = M$$

Podemos montar um exemplo utilizando SCL que vimos na seção anterior.

Considere que Pedro e Júlia utilizam o sistema criptográfico acima, em blocos de tamanho predefinido *k* onde Pedro utiliza a função de permutação σ cuja inversa é σ^{-1} e Júlia utiliza a função ρ e ρ^{-1} assim, eles podem tornar públicas as suas chaves de encriptação σ e ρ. Assim, para Pedro enviar uma mensagem a Júlia, ele fará $\rho(\sigma^{-1}(M))$ e Júlia com sua chave privada conseguirá identificar a mensagem de Pedro. Mas! Temos neste ponto, um pequeno problema. Se supormos a chave pública de Pedro $\sigma(X) = A.X + B$, é possível encontrar facilmente a chave privada, pois dado $Y = A.X + B$, conseguimos obter a inversa de σ fazendo $X = A^{-1}.(X - B)$, ou seja, basta encontrar o inverso multiplicativo de A em $\mathbb{Z}_{L^k}$. O que é uma tarefa bem simples se usarmos o algoritmo de Euclides estendido. Assim, podemos concluir que, nem todo sistema de criptografia é adequado a ser utilizado em chaves públicas. Precisamos então, de um sistema que não seja possível obter a chave privada através da chave pública.

3.4. O RSA

Iremos tratar agora sobre a matemática envolvida no Sistema RSA, que será de tratado de maneira mais aplicada. Segundo Coutinho em [1] o RSA, foi proposto

em 1978 por Rivest, Shamir e Adleman, cujas iniciais dos nomes de seus respectivos criadores batizaram o sistema. É sabido que o RSA é largamente utilizado para garantir a segurança da troca de informações entre usuários na internet. Para que possamos utilizar o seu algoritmo, faz-se necessário primeiramente a definição de um alfabeto. Neste caso continuaremos a utilizar o mesmo alfabeto do SCL que vai de A a Z, que possui 26 letras, chamaremos esse número de L, porém isso nos obrigará a fazer uma certa separação no envio das mensagens. Em alguns casos utiliza-se o alfabeto com 27 posições, conta-se de A a Z e acrescenta-se um caractere que representa o espaço.

Primeiramente, devemos escolher dois números primos p e q e outros dois números inteiros r e s de modo que eles se relacionem da seguinte forma:

$$L^r < p.q < L^s$$

pensar numericamente ajudará a facilitar o entendimento. Tomaremos no decorrer deste texto, $r = 2$ e $s = 3$, teremos então que escolher p e q de modo que

$$26^2 < p.q < 26^3$$

$$729 < p.q < 19683$$

Como não iremos ter capacidade de fazer cálculos com números grandes, pois tomaria muito tempo e talvez fosse necessário um computador com um bom programa de computação algébrica para que você pudesse acompanhar, podemos procurar nos primos de 1 a 50, dois, que satisfaçam essa condição na Figura 8

Figura 8 Produto de Primos de 1 a 50

	2	3	5	7	11	13	17	19	23	29	31	37	41	43	47
2	4	6	10	14	22	26	34	38	46	58	62	74	82	86	94
3	6	9	15	21	33	39	51	57	69	87	93	111	123	129	141
5	10	15	25	35	55	65	85	95	115	145	155	185	205	215	235
7	14	21	35	49	77	91	119	133	161	203	217	259	287	301	329
11	22	33	55	77	121	143	187	209	253	319	341	407	451	473	517
13	26	39	65	91	143	169	221	247	299	377	403	481	533	559	611
17	34	51	85	119	187	221	289	323	391	493	527	629	697	731	799
19	38	57	95	133	209	247	323	361	437	551	589	703	779	817	893
23	46	69	115	161	253	299	391	437	529	667	713	851	943	989	1081
29	58	87	145	203	319	377	493	551	667	841	899	1073	1189	1247	1363
31	62	93	155	217	341	403	527	589	713	899	961	1147	1271	1333	1457
37	74	111	185	259	407	481	629	703	851	1073	1147	1369	1517	1591	1739
41	82	123	205	287	451	533	697	779	943	1189	1271	1517	1681	1763	1927
43	86	129	215	301	473	559	731	817	989	1247	1333	1591	1763	1849	2021
47	94	141	235	329	517	611	799	893	1081	1363	1457	1739	1927	2021	2209

Fonte: O autor

Vamos escolher $p = 23$ e $q = 41$ e assim $p.q = n = 943$, temos quase todos os pré-requisitos para obtenção das chaves de Encriptação e Decriptação.

Retomando os números que havíamos definido anteriormente, temos agora que montar as chaves de Codificação e Decodificação. Havíamos escolhido $r = 2$, $s = 3, n = 943 = 23.41$. Para gerar a primeira chave, precisamos calcular o valor de $\emptyset(n) = (p-1).(q-1)$, assim $\emptyset(943) = (23-1).(41-1) = 880$. Escolhemos agora o menor natural e tal que $mdc(e, 880) = 1$, que no caso é 3.

Feito isso, acabamos de criar a nossa chave de encriptação/codificação ou chave pública. No Sistema RSA é indiferente a escolha da chave que será pública ou privada, porém temos um apreço por manter a primeira chave como a pública visto que o processo para determiná-la é mais simples do que o processo para obtenção da chave privada, além de que, no método que utilizamos, é mais difícil acertar a chave privada por tentativa do que a chave pública. Assim, nossa chave pública será (n, e), neste caso $(943, 3)$.

A chave privada, será obtida procurando um número d, de modo que

$$d.e \equiv 1 \; mod \; \phi(n)$$

o que parece ser uma tarefa um pouco mais complexa. Devemos sempre procurar d de tal maneira que $d < \phi(n)$.

Para nosso exemplo, iremos resolver a seguinte congruência

$$d.3 \equiv 1 \; mod \; 880$$

que é equivalente a determinar uma solução da *Equação Diofantina*

$$d.3 + 880.k = 1$$

assim obtendo $d = 587$, e, portanto, a chave privada $(943,587)$.

Você pode estar se preguntando sobre os números r e s que definimos no início desta seção. O método RSA pode ser utilizado de várias formas, e uma delas é, inclusive, através do sistema de blocos. No momento de criptografar uma mensagem, é possível dividi-la em blocos de tamanho r, criptografá-la e enviá-la em blocos de tamanho s.

Iremos tratar de algumas das formas que elas são feitas. Porém, a proposta desse trabalho, será utilizar o método mais simples, de modo que seja facilmente aplicado na sala de aula.

3.4.1. Codificação letra-a-letra

Antes de iniciar a codificação, devemos fazer um processo de pré-codificação, onde transformaremos as letras em números (de acordo o alfabeto definido). Em seguida o autor da mensagem, de posse da chave pública do receptor ao qual ele deseja enviar a mensagem, irá elevar cada bloco da mensagem (que representa cada letra) por e, e encontrar o módulo da divisão deste resultado por n, ou seja, se desejamos enviar a mensagem $a_1a_2a_3 \dots a_n$ devemos encontrar $b_1b_2b_3 \dots b_n$ onde

$$a_1^e \equiv b_1 mod\ n$$
$$a_2^e \equiv b_2 mod\ n$$
$$\vdots$$
$$a_n^e \equiv b_n mod\ n$$

a sequência numérica $b_1b_2 \dots b_n$ será nossa mensagem codificada.

Vamos enviar a palavra C I D A D E que a havíamos codificado como, 3 9 4 1 4 5. Temos então que resolver as congruências:

$$3^3 \equiv b_1 mod\ 943 \Rightarrow b_1 = 27$$
$$9^3 \equiv b_2 mod\ 943 \Rightarrow b_2 = 729$$
$$4^3 \equiv b_3 mod\ 943 \Rightarrow b_3 = 64$$
$$1^3 \equiv b_4 mod\ 943 \Rightarrow b_4 = 1$$
$$4^3 \equiv b_5 mod\ 943 \Rightarrow b_5 = 64$$
$$5^3 \equiv b_6 mod\ 943 \Rightarrow b_6 = 125$$

Parece que conseguimos, mas para uma análise mais detalhada, observa-se que temos alguns problemas na forma que fizemos a encriptação. Observe que, letras repetidas obtiveram a mesma codificação. Além disso, observe os resultados obtidos, eles formam simplesmente o cubo do número correspondente a letra. Isso se dá devido ao fato destas potências serem menores do que 943, ou seja, o algoritmo só fará uma mudança mais eficiente se as letras envolvidas obtiverem um correspondente maior do que 10, isto é, a partir do J. Além disso, a letra A que é igual a 1, sempre será codificada como 1 após a aplicação do algoritmo, o que seria uma pista e tanto para decodificarmos a mensagem. Como resolvermos isso? Bom, existem dois modos, um que seria ao invés de codificar letra a letra, codificarmos blocos de duas letras ou mais, de acordo o nosso r. Mas, para isto temos que tomar o cuidado de nenhum bloco ultrapassar o tamanho de n, com a nossa pre-codificação, o maior bloco de duas letras seria $YZ = 2526$, assim precisaríamos tomar um n um

pouco maior caso necessário, se quiséssemos codificar usando blocos de 3, aumentaríamos ainda mais o n.

Outra saída caso quiséssemos manter a codificação letra a letra, seria modificar a correspondência do nosso alfabeto. Adotando por exemplo $A = 10$, $B = 11$, e assim por diante até $Z = 35$. Assim, nosso algoritmo para o mesmo exemplo acima retornaria

$$12^3 \equiv b_1 mod\ 943 \Rightarrow b_1 = 785$$
$$18^3 \equiv b_2 mod\ 943 \Rightarrow b_2 = 174$$
$$13^3 \equiv b_3 mod\ 943 \Rightarrow b_3 = 311$$
$$10^3 \equiv b_4 mod\ 943 \Rightarrow b_4 = 57$$
$$13^3 \equiv b_5 mod\ 943 \Rightarrow b_5 = 311$$
$$14^3 \equiv b_6 mod\ 943 \Rightarrow b_6 = 858$$

que é um pouco melhor.

Observe a codificação da palavra $CIDADE$ utilizando um único bloco como letra.

$$CIDADE = 394145$$

Para isso, tomamos novas chaves, pública e privadas, oriundas de $p = 773$ e $q = 571$

$$ChavePública = (n, e) = (441383{,}7)$$
$$ChavePrivada = (n, d) = (441383{,}62863)$$

Assim, resolvendo a congruência

$$394145^7 \equiv b_1 mod\ 441383$$

E finalmente nossa mensagem a ser enviada $b_1 = 318881$

3.4.2. Codificação por transposição de blocos

Para utilizar o sistema de codificação por transposição de blocos, precisamos decompor a mensagem em blocos de tamanho r, algumas vezes, a depender do tamanho da mensagem, será necessário completar o último bloco com uma letra qualquer por exemplo A. Como escolhemos para o exemplo $r = 2$, segue que a palavra cidade seria então decomposta como $CI - DA - DE$. Não havendo, portanto, necessidade de completarmos o último bloco.

Feito isso, devemos reescrever os blocos, fazendo a sua correta correspondência numérica, assim, obtemos:

$$CI = 39; DA = 41; DE = 45$$

É importante fixar aqui que o alfabeto utilizado faz a correspondência $A = 1, B = 2$, até $Z = 26 = 0$, portanto utilizaremos comumente aqui $Z = 0$. Agora, devemos escrever estes números na base L, obtendo:

$$39 = 3.26^1 + 9.26^0 = 87$$
$$41 = 4.26^1 + 1.26^0 = 105$$
$$45 = 4.26^1 + 5.26^0 = 109$$

codificá-los utilizando nossa chave pública,

$$87^3 \equiv b_1 mod\ 943 \Rightarrow b_1 = 289$$
$$105^3 \equiv b_2 mod\ 943 \Rightarrow b_2 = 564$$
$$109^3 \equiv b_3 mod\ 943 \Rightarrow b_3 = 290$$

Este resultado ainda não será enviado, pois decidimos que seriam enviados nesse processo, blocos de tamanho 3, assim, para finalizar, devemos colocar b_1, b_2 e b_3 na base 26

Assim,

$$289 = 2.26^2 + 8.26^1 + 9.26^0 = 1569$$
$$564 = 5.26^2 + 6.26^1 + 4.26^0 = 3540$$
$$290 = 2.26^2 + 9.26^1 + 0.26^0 = 1586$$

3.4.3. Decodificação

O processo de decodificação é análogo a codificação neste sistema, porém utilizaremos esta parte para já justificarmos o porquê do método RSA funciona.

Teorema 10 (RSA). Seja $a \in \mathbb{Z}$, então $a^{de} \equiv a\ mod\ n$.

Demonstração: Note que pela forma como tomamos d e e, temos que $d.e \equiv 1\ mod\ \phi(n)$. Ou seja, $d.e = 1 + k.\phi(n)$. Podemos escrever,

$$a^{d.e} \equiv a^{d.e}\ mod\ n$$

equivale a

$$a^{d.e} \equiv a^{1+k.\phi(n)}\ mod\ n$$

note que, manipulando mais ou pouco a equação acima temos,

$$a^{d.e} \equiv \left(a^{\phi(n)}\right)^k . a^1\ mod\ n$$

mas, o Teorema de Euler garante que

$$a^{\phi(n)} \equiv 1 \, mod \, n$$

e, portanto

$$a^{d.e} \equiv (1)^k . a^1 \equiv a \, mod \, n.$$

■

O Teorema 10, nos mostra que ao recebermos uma mensagem codificada pelo sistema RSA com a chave pública (n, e), para decodificá-la basta elevarmos a mensagem a d módulo n, que a mesma será recuperada.

Voltando ao exemplo da subseção 3.4.1, e lembrando que nossa chave privada foi $(943{,}587)$, temos então que resolver as congruências:

$$27^{587} \equiv a_1 mod \, 943 \Rightarrow a_1 = 3$$
$$729^{587} \equiv a_2 \, mod \, 943 \Rightarrow a_2 = 9$$
$$64^{587} \equiv b_3 mod \, 943 \Rightarrow b_3 = 4$$ [5]
$$1^{587} \equiv a_4 mod \, 943 \Rightarrow a_4 = 1$$
$$64^{587} \equiv a_5 mod \, 943 \Rightarrow a_5 = 4$$
$$125^{587} \equiv a_6 mod \, 943 \Rightarrow a_6 = 5$$

Para decodificação das mensagens codificadas em blocos, deve-se seguir o mesmo processo, porém as por transposição, deve-se tomar o cuidado de retornar os códigos enviados para a base anterior antes de iniciarmos o processo de decodificação. Assim ao recebermos a mensagem da subseção 3..4.1, onde foi feita uma transposição de blocos, devemos proceder da seguinte forma:

$$(1569)_{L^3} = 2.26^2 + 8.26^1 + 9.26^0 = (289)_L$$
$$(3540)_{L^3} = 5.26^2 + 6.26^1 + 4.26^0 = (564)_L$$
$$(1586)_{L^3} = 2.26^2 + 9.26^1 + 0.26^0 = (290)_L$$

e em seguida aplicar a decodificação

$$289^{587} \equiv a_1 mod \, 943 \Rightarrow a_1 = 87$$
$$564^{587} \equiv a_2 \, mod \, 943 \Rightarrow a_2 = 105$$
$$290^{587} \equiv a_3 \, mod \, 943 \Rightarrow a_3 = 109$$

[5] Caso seja do interesse do leitor, é possível verificar os cálculos através do aplicativo CodeClass, que será abordado no próximo capítulo.

3.4.4. Assinatura do RSA

Quando tratamos em 3.3 do Conceito de Chave pública, vimos como é simples assinar uma mensagem através dos cadeados. Porém, na prática, para que isso aconteça, faz-se necessário que as funções que codificam e decodificam as mensagens se comportem como pares de funções onde uma se torna o inverso da outra e vice-versa.

Para que isso aconteça com o RSA, é necessário seguirmos alguns passos durante a codificação e decodificação do sistema.

Vamos supor que Pedro e Júlia desejem trocar mensagens. Cada um conhece suas chaves públicas e privadas e como funciona o sistema RSA.

Sem perda de generalidade, suponha que as chaves pública e privada de Pedro sejam $(n_P, e_P), (n_P, d_P)$. E as de Júlia sejam $(n_J, e_J), (n_J, d_J)$. Considere também $n_P > n_J$.

Pedro mandará a mensagem a para Júlia da seguinte forma.

Primeiramente irá determinar b, onde $a^{e_J} \equiv b \; mod \; n_J$ e em seguida irá calcular c onde $b^{d_P} \equiv c \; mod \; n_P$, enviando finalmente c.

Caso $n_J > n_P$ ele deverá inverter a ordem da operação anterior, realizando:

Primeiramente irá determinar b, onde $a^{d_P} \equiv b \; mod \; n_P$ e em seguida irá calcular c onde $b^{e_J} \equiv c \; mod \; n_J$, enviando finalmente c

Para a decodificação, deve-se seguir os passos na ordem inversa de codificação, aplicando a congruência módulo maior n inicialmente.

4. O APLICATIVO CODECLASS E A CRIPTOGRAFIA

O aplicativo CodeClass foi desenvolvido com o intuito apresentar de maneira simples, os processos necessários para a implementação de um complexo sistema de criptografia que é comumente utilizado na internet, o RSA. Além de possibilitar o desenvolvimento de atividades voltadas para o estudo da Aritmética Modular nas aulas.

Apesar do aplicativo já poder ser acessado, acreditamos que ainda são necessários vários ajustes e melhorias, que busquem aumentar a gama de possibilidades quanto a sua utilização. Além disso, somente com o uso e colaboração dos usuários, será possível tornar o seu uso mais intuitivo e dinâmico. O registro do mesmo já foi iniciado pela UESB no dia 19/09/2019.

Para ter acesso ao aplicativo, basta acessar do seu celular o link disponível em [12] onde consta a versão mais recente do app. Após acessar a pasta do drive, faça o download do arquivo CodeClass.apk e execute-o em seu aparelho. Como o aplicativo ainda não se encontra disponível no PlayStore, muitas vezes será necessária uma autorização manual do usuário para prosseguir com a instalação.

4.1. O aplicativo

O aplicativo CodeClass foi integralmente desenvolvido através da plataforma gratuita, MIT App Inventor 2. O seu algoritmo utiliza, efetuando as devidas adaptações, toda a teoria apresentada neste trabalho. O projeto foi iniciado em meados de fevereiro, após o levantamento teórico ter sido completamente finalizado. Foi necessária uma completa revisão sobre os principais algoritmos da Teoria dos Números, principalmente para a tomada decisão quanto as etapas lógicas que o sistema iria realizar.

A primeira fase do projeto, foi conseguir fazer com que o aplicativo interpretasse as letras do alfabeto como números, para que pudesse ser feita a correlação proposta

na Figura 5, para isso criamos no desenvolvimento do app uma lista, organizada por índices que possui 26 linhas.

O aplicativo possui três funções bem definidos, uma para se trabalhar a Criptografia Linear, a segunda trabalha a Criptografia RSA e um módulo gerador de números primos. O primeiro destes, foi inserido por ser conceitualmente mais simples, e pode ser utilizado como base para compreensão da Criptografia RSA, além de ser um instrumento que servirá para se comparar os métodos, entender falhas e conseguir discernir o porquê de alguns sistemas não serem adequados para o sistema de chave pública.

O segundo bloco e foco deste projeto trata especificamente do RSA, e pode-se atribuir a ele diversos usos. Ele foi propositalmente subdividido em três partes geração das chaves, codificação e decodificação afim de facilitar o trabalho e entendimento em cada uma das etapas. Sendo assim, será possível trabalhar todo o processo de maneira gradativa e independente.

Finamente, foi colocado um módulo de números primos, pois como estes são a base do sistema, seria sempre necessário dar uma consultada em alguns. Acreditamos que deixá-los “a mão” pode servir para ganharmos tempo em algumas propostas.

4.2. Interface

Figura 9 Ícone e Slogan do CodeClass

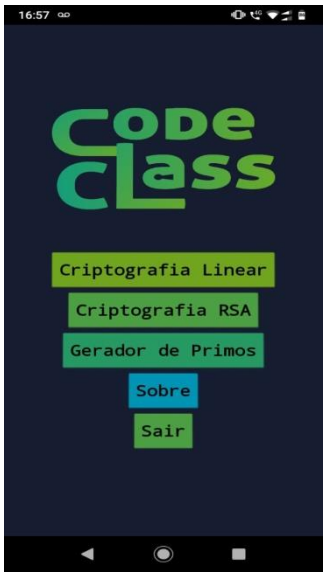
Fonte: O autor

O aplicativo *CodeClass* cujo slogan pode ser visto na Figura 9, ainda encontra-se em fase aperfeiçoamento, porém já é possível acessá-lo caso os usuários queiram testar e até mesmo realizar sugestões e/ou críticas para o projeto. O funcionamento

ainda é um pouco arcaico, devido a não ser possível até o momento automatizar alguns processos na inserção dos dados.

Ao instalar o app, o usuário irá se deparar com uma tela inicial, conforme a Figura 10, que disponibiliza acesso aos temas que poderão ser trabalhados.

Figura 10 Tela Inicial do CodeClass



Fonte: O Autor

Nesta tela, é possível escolher qual tipo de criptografia será trabalhada de acordo a proposta que o professor deseja para aquela aula ou atividade, são elas a Criptografia Linear, que apesar de ser completamente inviável hoje, é o passo inicial para entendermos a necessidade de evolução dos sistemas criptográficos.

No momento o botão sobre encontra-se em fase desenvolvimento, não havendo até o momento nenhuma funcionalidade para tal. Em breve ele irá apresentar informações sobre os autores, bem como a forma de contato para algum tipo de suporte.

Na interface descrita na Figura 4.3 é possível codificar e decodificar uma mensagem utilizando o Sistema de Criptografia Linear. Este sistema é uma reinterpretação da Cifra de César que foi citada no início deste trabalho. Além disso, sistemas mais avançados com o utilizado na Enigma [2] também se basearam num método similar a este para sua criação.

Figura 11 Interface Criptografia Linear

Fonte: O autor

Para codificar uma mensagem, o autor deve escolher inicialmente dois parâmetros A e B, que compões a função de permutação $f(X) = A.X + B$, como estamos utilizando o alfabeto tradicional, com as letras de A a Z (maiúsculas) e sem considerar o espaço em branco, consideramos que o nosso alfabeto de trabalho, consta de 26 caracteres. Para que haja inversos em $\mathbb{Z}_{26}$ faz-se então necessário que a escolha de A, seja de forma que $mdc(A, 26) = 1$, caso esta condição não seja satisfeita, o programa não seguirá com o processo.

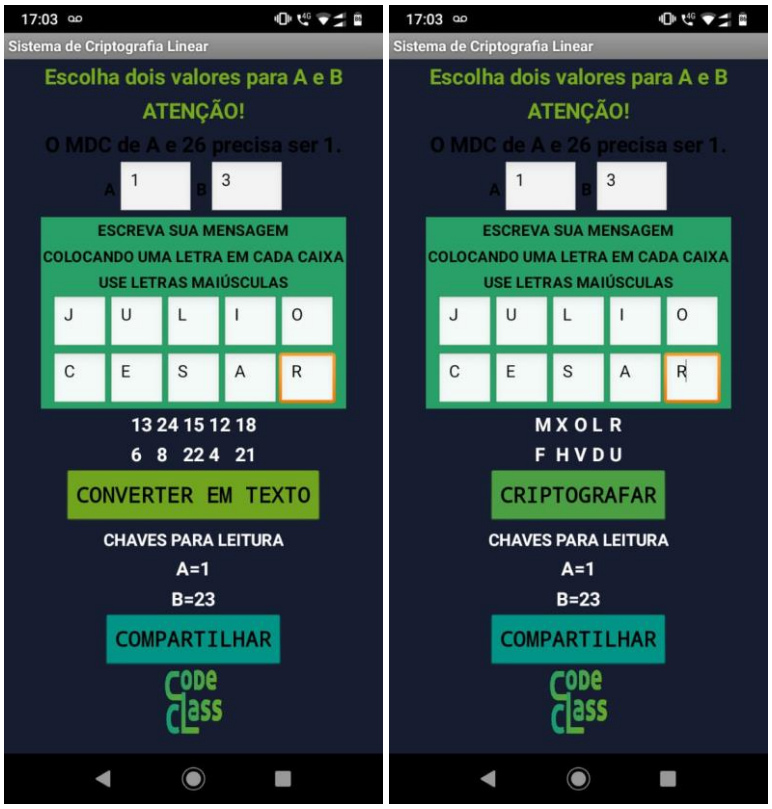
Escolhido os parâmetros, deve-se digitar nos campos da janela intitulada "*ESCREVA SUA MENSAGEM"* os caracteres da mensagem que se deseja enviar. Vale ressaltar que só devem ser utilizados caracteres maiúsculos do alfabeto e, sempre, um caractere por campo, caso contrário o programa irá retornar erros e possivelmente irá finalizar sua execução.

Figura 12 Exemplo de Preenchimento

Fonte: O autor

Utilizamos a interface com o exemplo como pode ser visto na Figura 12, para ilustrar o preenchimento, neste caso em específico os parâmetros utilizados foram $A = 1$ e $B = 3$ que representam a transposição inicial da Cifra de César, em seguida escrevemos a mensagem J U L I O C E S A R. Para codificar, basta finalmente clicar no botão CRIPTOGRAFAR, que o programa irá retornar a imagem da função de transposição definida. Caso seja de interesse do utilizador converter os números em caracteres, irá ser liberado um botão com este fim. É importante salientar também que, para que o receptor consiga ler a mensagem, é necessário ter acesso as chaves para leitura que também são geradas no momento da codificação.

Figura 13 Codificação e Geração das Chaves



Fonte: O autor

Finalizada a codificação e geração das chaves, o autor poderá transmitir a mensagem para o receptor, através do botão compartilhar. Vale lembrar que é importante definir-se também um canal seguro para envio da chave de leitura, ou, caso contrário o receptor não terá condições de decodificar a mensagem como pode ser visto na Figura 13.

A decodificação deve ser feita da mesma forma da codificação. Porém na definição dos parâmetros, deve-se colocar os valores de A e B fornecidos como *“CHAVE DE LEITURA*” e em seguida, realizar as mesmas etapas da codificação.

Na tela de boas vindas do Sistema de Criptografia RSA Figura 14, o utilizador tem acesso a três módulos para operar: Gerar Chaves, Criptografar Mensagem e Decifrar Mensagem. Os módulos são completamente independes, não sendo necessário alimentar um com informações para a utilização dos demais.

Figura 14 Tela de boas-vindas Criptografia RSA

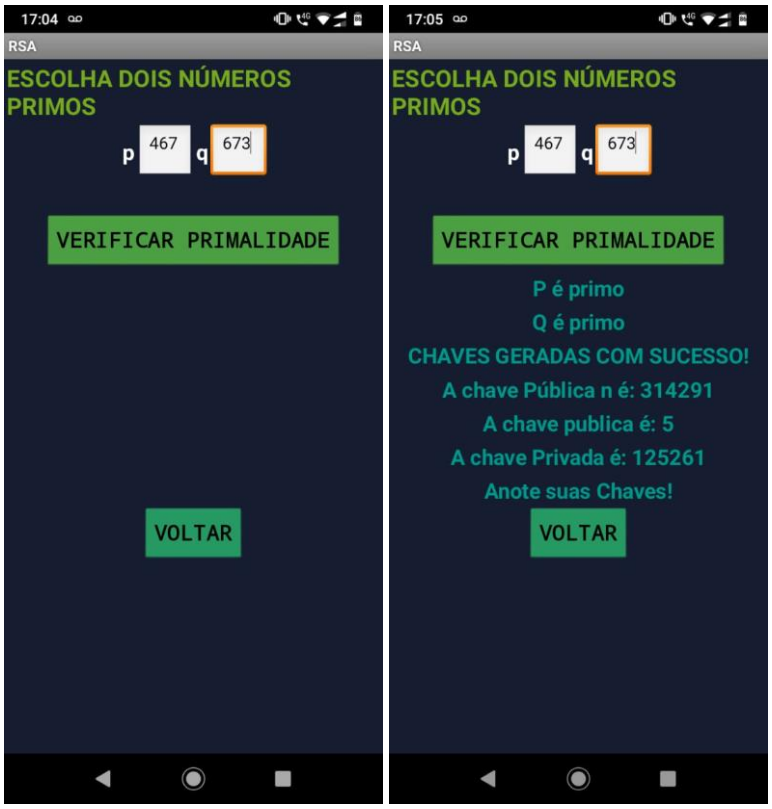
Fonte: O autor

A tela de geração das chaves, possui um funcionamento bem intuitivo. Basta que o utilizador, escolha dois primos distintos e pressione o botão verificar primalidade. O programa irá verificar se os números são realmente primos e, em seguida irá liberar o botão para geração das chaves.

Em testes realizados durante o desenvolvimento do aplicativo, verificamos que no momento a forma de execução do algoritmo torna inviável a geração de chaves para primos muito grandes. Aconselhamos trabalhar com números de 3 dígitos, no máximo. É possível ainda assim gerar chaves para números maiores, mas o tempo de espera pode ser demasiadamente grande, podendo em alguns casos finalizar a execução do programa no celular, por não conseguir finalizar a tarefa.

É importante que, após geradas as chaves, o usuário anote-as em um local seguro, principalmente a sua chave privada. Caso seja uma dinâmica ou oficina, ele pode divulgar as suas chaves públicas para outros usuários, que poderão utilizá-las no decorrer do processo. Observe um exemplo na Figura 15.

Figura 15 Geração das Chaves RSA



Fonte: O autor

Após geradas as chaves pública e privada, podemos iniciar o processo de codificação. Para facilitar o entendimento e exemplificar o processo, iremos simular o envio de uma mensagem entre os usuários Alice e Bob que possuem as chaves:

Alice: Pública $(n, Cpub) = (21509,5)$ Privada: $(n, Cpriv) = (21509,16973)$

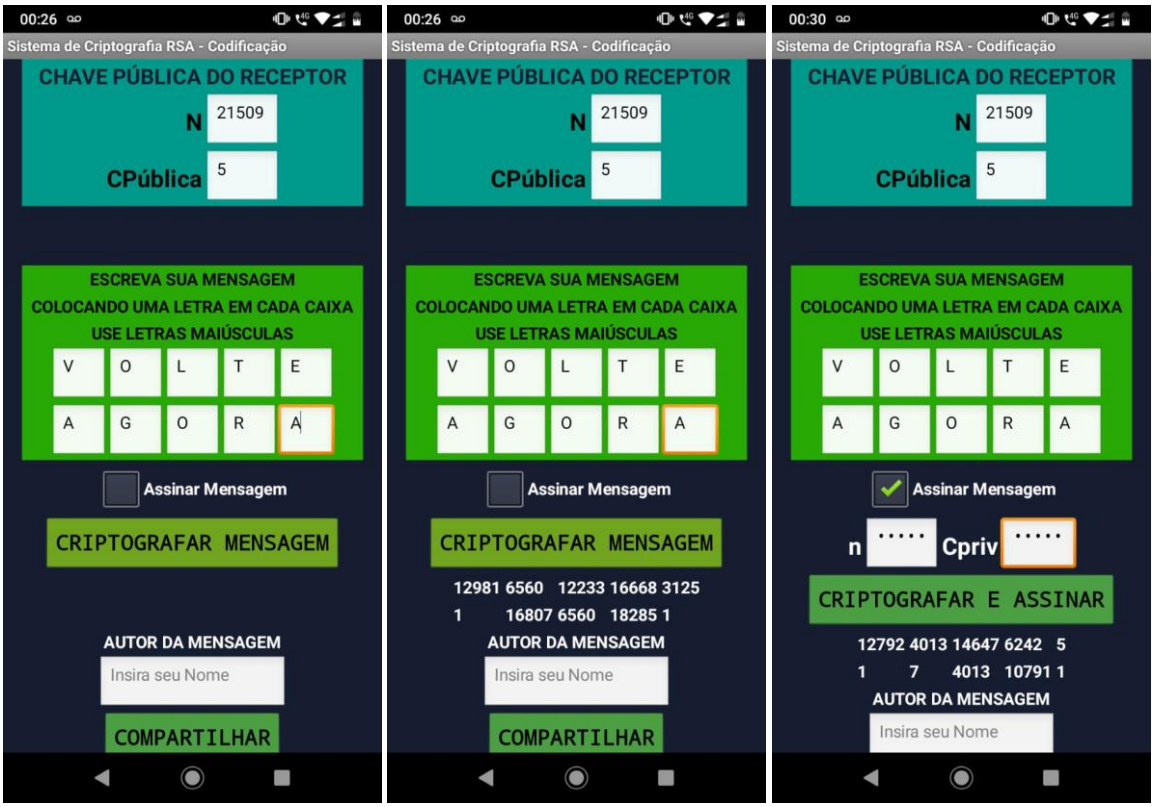
Bob: Pública $(n, Cpub) = (20633,5)$ Privada: $(n, Cpriv) = (20633,12089)$

Para que Bob consiga enviar uma mensagem para Alice, ele precisar saber qual a chave pública de Alice. De posse dessa informação, ele deve preencher na tela de Codificação, nos campos n e $Cpub$ as chaves públicas do usuário que irá receber a mensagem, neste caso, Alice. Em seguida deve escrever nos campos abaixo a sua mensagem, sempre respeitando um caractere por campo e utilizando sempre letras maiúsculas que pode ser vista na Figura 16.

É possível enviar a mensagem assinada, ou não. Tudo vai depender de como se está trabalhando o aplicativo e o domínimo de conteúdo dos participantes. Para

assinar, basta que o autor marque o campo *Assinar mensagem* e em seguida preencher os campos indicados com a sua Chave Privada. Atenção a assinatura é sempre realizada com a chave privada do autor. Caso haja uma inversão destas chaves, será impossível para o receptor decodificar a informação recebida.

Figura 16 Codificação, Codificação sem assinatura e Codificação com Assinatura



Fonte: O autor

Após ter criptografado a mensagem, a mesma pode ser enviada através do botão compartilhar ou transmitida por algum outro canal ao receptor. Vale ressaltar que não é possível converter a mensagem codificada para o alfabeto, por não se tratar de uma cifra de transposição.

Finalmente na tela de decodificação, iremos realizar um processo semelhante ao da codificação. Fez-se necessário a criação de uma tela a parte para este processo, devido a algumas peculiaridades no processo de codificação e decodificação. O principal deles, é o fato de a entrada na codificação ser um caractere do tipo texto, enquanto a saída é um caractere numérico. Já na decodificação, a

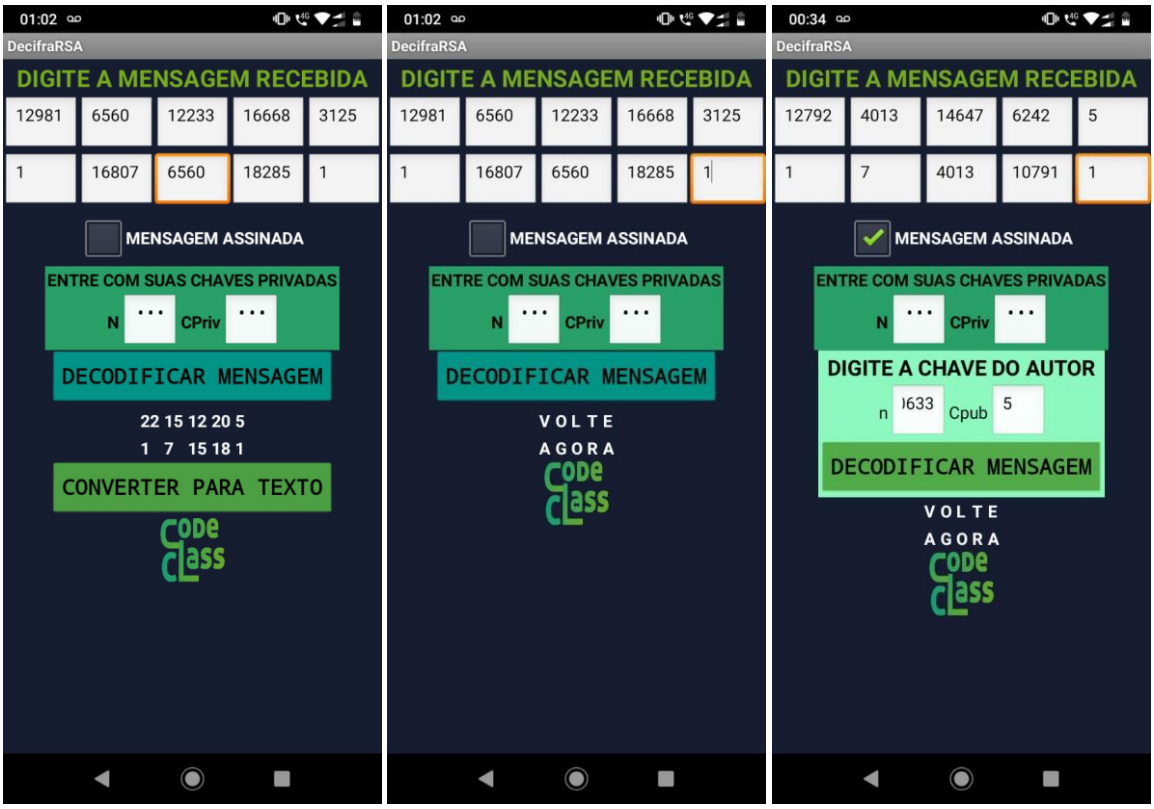
entrada do programa é um caractere numérico e a saída será um caractere do tipo texto. Portanto, os campos de entrada nos dois processos são distintos e assim optamos por realizá-los separados.

Para decodificar a mensagem o receptor deve ir na tela decifrar mensagem e inserir em cada campo os números que foram recebidos e na ordem recebida. Foi tomado o cuidado de deixar cada bloco numérico visivelmente espaçado, de modo que não haja sobreposições de blocos ou indefinições das sequências numéricas.

Em seguida o receptor deve preencher os campos n e $Cpriv$ com a sua chave privada e por fim pressionar o botão DECODIFICAR MENSAGEM. Na maioria das vezes, a depender das chaves que foram utilizadas, você notará que os resultados foram números menores que 26. Isso indica que a princípio o processo parece ter ocorrido da maneira correta. E assim, para finalizar o usuário pode pressionar o botão CONVERTER PARA TEXTO, para que seja traduzido o conteúdo para caracteres alfabéticos conforme a Figura 17.

Caso a mensagem tenha sido assinada, ele deve marcar o campo MENSAGEM ASSINADA e entrar com as chaves públicas do possível receptor. Em seguida pressionar o botão DECODIFICAR MENSAGEM, que o sistema irá proceder da mesma forma que a decodificação sem assinatura. Se ocorrer tudo corretamente no processo e a mensagem conseguir ter sido decifrada com sucesso, será possível converter os caracteres para as letras correspondentes do alfabeto, além de termos a plena certeza sobre a autoria do conteúdo.

Figura 17 Decodificação, sem Assinatura e com Assinatura



Fonte: O autor

O terceiro e ultimo módulo do aplicativo é um gerador de números primos, que optamos por deixar disponível para fim meramente práticos de utilização do programa.

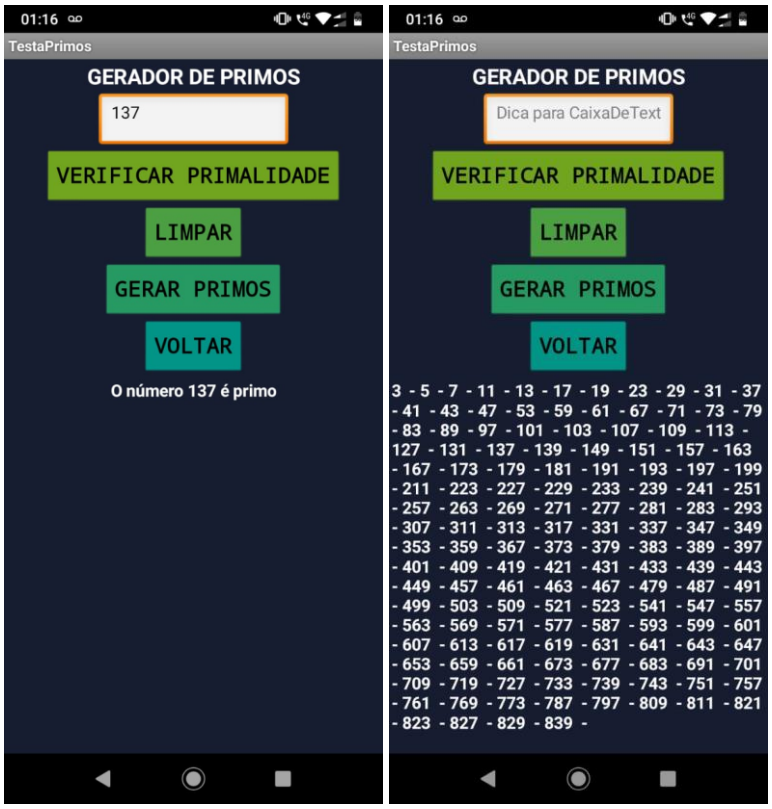
Utilizamos para a verificação da primalidade, nesta tela, o *Teorema de Wilson* a fim de ser mais simples de implementá-lo na geração dos primos, que pode ser feita de maneira através de um fórmula de recursão. Assim, nele será possível tanto testa se determinado número é primo, quanto também gerar uma lista de primos, para que seja feita a escolha de algo de acordo o interesse do usuário.

Na tela que é acessada pelo botão GERADOR DE PRIMOS o usuário tem as opções de inciar a geração de uma lista ou testar a primalidade de algum número de maneira isolada. Após precionar o botão GERAR PRIMOS, a lista será inciada de maneira interrupta, até que seja solicitado a pausa do processo. Caso o interesse seja escolher um primo apenas, pode-se fazê-lo desta forma.

Se o interesse for em somente verificar se determinado número é primo, deve-se então digitá-lo no campo específico e precionar o botão VERIFICAR

PRIMALIDADE. O programa irá retornar o resultado como na Figura 18 e dará fim ao processo.

Figura 18 Gerador de Primos



Fonte: O autor

4.3. SUGESTÃO I – Origem da Criptografia

Duração: 200 minutos/ 4 aulas

Objetivos: Conhecer a origem da criptografia; entender o funcionamento dos primeiros métodos criptográficos; compreender a sua importância para o desenvolvimento da sociedade.

Série: A partir do 1ºAno do Ensino Médio

Um pouco de História

Todo material utilizado nas atividades, inclusive o aplicativo, se encontra disponível no link permanente, disponível em [12].

Inicialmente, a proposta é fazer uma breve introdução sobre a origem da criptografia e da Cifra de César. Deve-se verificar se os alunos compreenderam o funcionamento da cifra de César e a importância de se ocultar informações.

Esta verificação pode ser feita solicitando aos alunos que respondam a Atividade 1.

ATIVIDADE 1

1) Codifique seu nome utilizando a Cifra de César.
2) Decifre a frase MXOLR FHVDU.
3) Eu poderia, ao invés de utilizar a correspondência de A para D, realizar alguma outra diferente?

Usando o aplicativo

Utilizando o aplicativo.

Peça aos alunos que abram o aplicativo CodeClass.

Em seguida clique no botão Criptografia Linear, na tela que segue, peça que preencham as caixas A e B com os valores 1 e 3. Em seguida, peça-os que digitem a mensagem do exercício 1 pressione o botão criptografar e observem se o resultado é o mesmo que eles obtiveram.

Na parte inferior do aplicativo, onde aparece: CHAVES PARA LEITURA constam as chaves para que o aluno consiga decifrar a mensagem.

Peça-os que anotem essas chaves $A = 1$ e $B = 23$ e agora tentem responder o exercício 2, utilizando-as para decifrar MXOLR FHVDU. É importante que eles entendam o funcionamento do mecanismo e não tenham dúvidas quanto a forma de decifrar.

Questione sobre a analogia entre a matemática do aplicativo e a forma original que Júlio César criou.

Para finalizar a atividade, peça aos alunos que escolham outros valores para A e B e observem o que acontece. Observe o quão desordenado podem ficar as letras de acordo o A cresce. Será que ainda assim existe um padrão?

Entendendo o aplicativo

Pré-requisito: Conhecimento de Funções, divisão inteira.

Agora é chegada a hora de entender o que o aplicativo está fazendo, para embaralhar as letras da mensagem.

Para cada letra do alfabeto, vem atribuído no aplicativo um número como na Figura 19.

Figura 19 Correspondência Letra-Número

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Fonte: O autor

e a função que irá criar uma bagunça nestes números, no caso da Criptografia Linear, essa função é

$$f(x) = A.x + B$$

Em que o usuário irá definir os valores de A e B.

Para exemplificar, vamos utilizar os parâmetros $A = 3$ e $B = 2$ para facilitar nosso entendimento. Quando digitamos a palavra B O L A, o aplicativo faz a atribuição numérica de acordo a tabela acima, enxergando na verdade os números $2 - 15 - 12 - 1$, feito isso e já de posse dos parâmetros, ele vai calcular a imagem de cada um destes números através da função f obtendo assim,

$$f(2) = 3.2 + 2 = 8$$

$$f(15) = 3.15 + 2 = 47$$

$$f(12) = 3.12 + 2 = 38$$

$$f(1) = 3.1 + 2 = 5$$

Para finalizar, o sistema buscará estes novos números na tabela, e irá trocá-los novamente por letras. É ai que surge o primeiro problema. Os números 8 e 5 estão na tabela, são respectivamente as letras H e E, Logo o aplicativo irá encontrá-los, mas o que fazer com números maiores que 26 que não estão na tabela, neste caso os números 47 e 38. Muito simples, os números maiores que 26, ele vai dividi-los por 26 e observar o resto desta operação. Assim, ao dividirmos 47 por 26, obtemos quociente 1 e resto 21. Portanto ele buscará 21 na tabela que é a letra U. Da mesma forma ele o fará com o 38 obtendo 12 letra L, e assim, a palavra B O L A, passou então a ser escrita como H U L E.

Uma analogia muito interessante sobre a forma de verificar estes restos, é comparar com um relógio, sabemos que os ponteiros de um relógio marcam de 1 a 12., mas ainda assim o dia vai de 0 às 24h. Assim, para sabermos que horas são as 17h da tarde, o que fazemos é dividir 17 por 12 e observar o resto, 5, por isso 17h, são 5 da tarde. De maneira análoga, ao completarmos 24h, temos que é a 0h do dia seguinte, visto que 24 divido por 12 deixa resto 0.

Para as chaves, o algoritmo se torna um pouco mais complexo. Como pode ser observado, o programa retornou que as CHAVES PARA LEITURA são $A = 9$ e $B = 8$ isso quer dizer, que se escrever a mensagem H U L E e trocar os valores de A e B

para 9 e 8, ao apertar o botão criptografar, você obterá a palavra B O L A novamente. Mas como isso é feito.

Bom, sabemos que uma função $f(x) = A.x + B$, admite função inversa

$$f^{-1}(x) = A^{-1}.x - A^{-1}.B \text{ se } A \neq 0$$

assim, o que o programa faz é descobrir quanto vale A^{-1} ou seja o inverso de A.

Nessas condições por não trabalharmos com números reais, o inverso de A, não pode ser um número decimal, por exemplo, aqui, 5^{-1} que é o mesmo que $\frac{1}{5} = 0{,}20$ não é considerado veremos que no nosso problema $5^{-1} = 21$. Como assim? Eu explico, $5.21 = 105$ ao dividirmos 105 por 26 obtemos resto 1 afinal $105 = 26.4 + 1$. É assim que determinamos os inversos de um número no conjunto que estamos trabalhando (Z_{26}). A Figura 20 mostra os produtos de todos os números de 1 a 26 de acordo a regra estabelecida. Em destaque estão aqueles que possuem inverso.

Figura 20 Inversos Multiplicativos em $\mathbb{Z}_{26}$

1	2	3	4	5		7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
2	4	6	8	10	12	14	16	18	20	22	24	0	2	4	6	8	10	12	14	16	18	20	22	24	0
3	6	9	12	15	18	21	24	1	4	7	10	13	16	19	22	25	2	5	8	11	14	17	20	23	0
4	8	12	16	20	24	2	6	10	14	18	22	0	4	8	12	16	20	24	2	6	10	14	18	22	0
5	10	15	20	25	4	9	14	19	24	3	8	13	18	23	2	7	12	17	22	1	6	11	16	21	0
6	12	18	24	4	10	16	22	2	8	14	20	0	6	12	18	24	4	10	16	22	2	8	14	20	0
7	14	21	2	9	16	23	4	11	18	25	6	13	20	1	8	15	22	3	10	17	24	5	12	19	0
8	16	24	6	14	22	4	12	20	2	10	18	0	8	16	24	6	14	22	4	12	20	2	10	18	0
9	18	1	10	19	2	11	20	3	12	21	4	13	22	5	14	23	6	15	24	7	16	25	8	17	0
10	20	4	14	24	8	18	2	12	22	6	16	0	10	20	4	14	24	8	18	2	12	22	6	16	0
11	22	7	18	3	14	25	10	21	6	17	2	13	24	9	20	5	16	1	12	23	8	19	4	15	0
12	24	10	22	8	20	6	18	4	16	2	14	0	12	24	10	22	8	20	6	18	4	16	2	14	0
13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0	13	0
14	2	16	4	18	6	20	8	22	10	24	12	0	14	2	16	4	18	6	20	8	22	10	24	12	0
15	4	19	8	23	12	1	16	5	20	9	24	13	2	17	6	21	10	25	14	3	18	7	22	11	0
16	6	22	12	2	18	8	24	14	4	20	10	0	16	6	22	12	2	18	8	24	14	4	20	10	0
17	8	25	16	7	24	15	6	23	14	5	22	13	4	21	12	3	20	11	2	19	10	1	18	9	0
18	10	2	20	12	4	22	14	6	24	16	8	0	18	10	2	20	12	4	22	14	6	24	16	8	0
19	12	5	24	17	10	3	22	15	8	1	20	13	6	25	18	11	4	23	16	9	2	21	14	7	0
20	14	8	2	22	16	10	4	24	18	12	6	0	20	14	8	2	22	16	10	4	24	18	12	6	0
21	16	11	6	1	22	17	12	7	2	23	18	13	8	3	24	19	14	9	4	25	20	15	10	5	0
22	18	14	10	6	2	24	20	16	12	8	4	0	22	18	14	10	6	2	24	20	16	12	8	4	0
23	20	17	14	11	8	5	2	25	22	19	16	13	10	7	4	1	24	21	18	15	12	9	6	3	0
24	22	20	18	16	14	12	10	8	6	4	2	0	24	22	20	18	16	14	12	10	8	6	4	2	0
25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

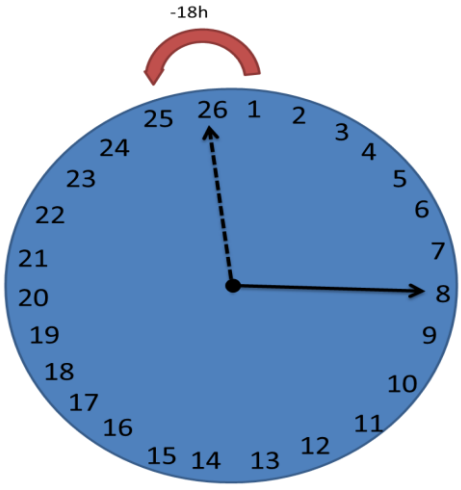
Fonte: O autor

Para números maiores que 26, lembrem-se sempre de encontrar o resto dele por 26 e olhar na tabela. Repare que alguns números não possuem inverso, isto ocorre porque o MDC dele e 26 é diferente de 1, por isso no app há uma notificação, onde o o MDC de A e 26 precisa ser 1.

Finalmente, já sabemos como foi gerado a Chave para Leitura A (vamos denotá-la aqui por A_D chave para decifrar), seguindo no nosso exemplo, em que escolhemos $A = 3$, provavelmente ele deve ter te retornado, segundo a tabela, uma chave para leitura $A_D = 9$. Para encontrar a Chave para Leitura B_D, basta realizar a multiplicação de $A_D.B$ visto que nossa função inversa é $f^{-1}(x) = A^{-1}.x - A^{-1}.B$. Lembre-se $A^{-1} = A_D$.

Assim, $A_D.B = 9.2 = 18$. Caso o valor tivesse ultrapassado 26, iriamos novamente procurar o resto dele por 26. Porém, observe que o segundo membro da função inversa é negativo, logo seria -18 o resultado correto, porém o programa deve ter lhe retornado 8. Isso ocorre devido ao fato de que os restos de uma divisão sempre são positivos, assim, se pensarmos quanto falta para 18 chegar a 26, iriamos perceber que é 8, exatamente o resto procurado. Pensemos de outra forma. Imagine o nosso relógio que marca 26 horas de uma dia, o relógio anda no sentido horário. Se pensarmos no menos como uma mudança de sentido (anti-horário), então o relógio andar -18h iria marcar 8h. Acompanhe na Figura 21:

Figura 21 Relógio de 26 horas



Fonte: O autor

Daí surge, então, as chaves que serão utilizadas para decifrar o código.

Tente sugerir aos alunos que tentem eles mesmos descobrir as Chaves de Leitura, com o auxílio da tabela e do relógio acima.

Como exemplo, façamos para $A = 7$ e $B = 10$.

Como vimos, pela tabela o inverso de 7 é 15. Assim, temos que $A_D = 15$. Para encontrarmos B_D devemos inicialmente fazer a operação $A_D.B$ neste caso 15.10, obtendo 150. Olhando para o resto da divisão de 150 por 26, temos 20, pois $150 = 5.26 + 20$, voltemos então 20h no sentido anti-horário e iremos determinar $B_D = 6$. Portando $A_D = 15$ e $B_D = 6$. Confira com o aplicativo.

Conclusão da Atividade

Ao término desta atividade, espera-se que os alunos tenham entendido conceitos introdutórios da aritmética modular, pode-se utilizar esta atividade como porta de entrada para o estudo de divisibilidade e congruência.

Alguns alunos já devem ter observado e talvez até clicado no botão compartilhar, que existe no aplicativo. Ele funciona simplesmente para enviar a mensagem que foi cifrada via algum aplicativo de conversação. Surge aí um questionamento interessante, digamos que você deseja enviar uma mensagem a alguém em uma situação um pouco turbulenta, onde ninguém pode ler está mensagem. Você percebeu que ao compartilhar a mensagem, para que seu colega leia, ele precisará da chave de leitura? Ou seja, teríamos que ter um canal superseguro para enviar essa chave a ele. Mas hora se tivéssemos esse canal, não teríamos necessidade de enviar a mensagem.

Eis então que acabamos de perceber que o nosso sistema de criptografia, não é tão eficiente quanto esperávamos, afinal as pessoas envolvidas nas trocas de mensagens, precisariam manter um certo contato em algum momento.

Bom, é aí que surge uma nova forma de criptografar, um método que irá acabar com este problema *A Criptografia de Chave Pública*.

4.4. SUGESTÃO II – Criptografia de Chave Pública

Duração:100 minutos/ 2 aulas

Objetivos: Entender o algoritmo de Criptografia de Chave Pública. Compreender o que é uma assinatura digital e qual a sua importância.

Série: A partir do 1ºAno do Ensino Médio

Pré-requisitos: Não há

Mas o que seria chave pública?

Nesta atividade, queremos que o aluno compreenda o conceito de chave pública, através do aplicativo *CodeClass*. Não iremos avançar nesta atividade sobre como funciona o aplicativo nem quais formas de fazê-lo, simplesmente queremos que o aluno entenda a essência do funcionamento da criptografia, onde os dois envolvidos na comunicação, em momento algum precisarão de um canal seguro para passar informação.

ATIVIDADE 1

Para a atividade, precisamos de algumas chaves que serão utilizadas na dinâmica. Como o foco sobre a geração das chaves será dado na proposta seguinte, abaixo seguem alguns exemplos de chave que poderão ser utilizados.

Figura 22 Sugestões de Chaves para Atividade

	CHAVE 1	CHAVE 2	CHAVE 3	CHAVE 4	CHAVE 5	CHAVE 6
(n,Publica)	(33,3)	(187,3)	(209,7)	(95,5)	(1243,3)	(649,3)
(n,Privada)	(33,7)	(187,107)	(209,103)	(95,29)	(1243,747)	(649,387)

	CHAVE 7	CHAVE 8	CHAVE 9	CHAVE10	CHAVE 11	CHAVE 12
(n,Publica)	(1003,3)	(493,3)	(1711,3)	(319,3)	(1067,7)	(1649,5)
(n,Privada)	(1003,619)	(493,299)	(1711,1083)	(319,187)	(1067,823)	(1649,1229)

Fonte: O autor

É interessante, talvez, deixar todas as chaves públicas expostas, para que um aluno saiba um a do outro. Como normalmente as turmas possuem mais de 20 alunos, talvez seja interessante dividi-los em grandes grupos, pois dois alunos não podem compartilhar uma mesma chave. No sistema é possível gerar quantas chaves forem necessárias. Peça-os que guardem com cuidado as suas chaves privadas e não mostrem ao colega.

Inicialmente, os alunos foram divididos em trios. Primeiro precisamos que eles compreendam como funciona a encriptação neste sistema e quem é o emissor e o receptor da mensagem. Emissor quem escreve, receptor quem recebe.

Peça aos alunos que abram o aplicativo CodeClass. Em seguida cliquem no botão CRIPTOGRAFIA RSA depois CRIPTOGRAFAR MENSAGEM.

Na nova tela, aparecem alguns campos que precisam ser preenchidos. Neles solicitam chaves públicas e privadas. Nesse momento somente 2 alunos do trio irão participar. Um aluno irá pegar a chave pública do colega para enviar uma mensagem a ele. Por exemplo, no trio de João, Ana e Pedro, digamos que eles estão respectivamente com as Chaves 1, 2 e 3, da Figura 22. João mandará uma mensagem a Ana, então logo ele deve preencher os campos Chave Pública do Receptor $n = 187$ e $Cpub = 3$, que são as chaves de públicas de Ana, que qualquer um pode ter acesso.

Em seguida, ele deve escrever a mensagem nos 10 campos disponíveis abaixo colocando uma letra, em maiúsculo, em cada campo. Nesta primeira parte, não há necessidade ainda de assinar, pois este não é o nosso foco por enquanto. Finalmente ao apertar o botão CRIPTOGRAFAR será gerada uma sequência numérica abaixo, que são os números que serão enviados a Ana.

É importante que sejam respeitados os espaçamentos entre os números por exemplo, no nosso caso enviando a Ana a mensagem O L A Q U E R I D A, os números obtidos foram:

$$9-45-1-51-98-125-35-168-64-1$$

Esse espaçamento é fundamental para que Ana consiga decifrar a mensagem.

Agora, João deve anotar a mensagem num papel e enviar a Ana, caso prefira, ele pode escrever o nome dele no campo Autor da mensagem e compartilhar utilizando a função do próprio aplicativo. Porém a ideia neste momento é que João entregue a mensagem a Pedro e que ele entregue a Ana, nesse meio tempo, Pedro deve tentar descobrir qual a mensagem que foi enviada com a função DECIFRAR MENSAGEM.

A função de decifrar do aplicativo, é parecida com a função de cifrar. O receptor da mensagem, no caso Ana, irá digitar a sequência numérica recebida, cada bloco numérico em um campo como na foto abaixo. Em seguida digitará sua chave privada, neste caso, $n = 187, Cprivada = 107$. Observe que as chaves privadas são senhas e nunca devem ser expostas. Os números serão alterados, e para finalizar, basta que Ana pressione o botão CONVERTER PARA TEXTO, para que ela possa ter a mensagem traduzida.

É interessante perceber que Pedro pode tentar várias combinações para descobrir a mensagem, talvez por tentativa ele consiga, porém ele não tem nem noção de quantos caracteres tem a senha de Ana. Pode-se aproveitar o momento para fazer uma previsão de quantas tentativas ele iria precisar.

Assinando uma mensagem

Qual a razão de se assinar uma mensagem? Esta resposta encontramos pelo simples fato de reconhecermos a importância de garantir que o receptor não seja enganado por um falso emissor. Suponhamos, com base na nossa proposta anterior: O que impede Pedro de escrever uma mensagem criptografada para Ana e assinar como se fosse João? Por termos uma chave pública, na prática, qualquer pessoa teria o nosso endereço e poderia nos enviar qualquer mensagem. Imagine um falso banco nos enviando falsos boletos, como eu saberia se é realmente do meu banco, ou não?

Portanto, peça aos alunos que abram o aplicativo, cliquem em CRIPTOGRAFIA RSA em seguida CRIPTOGRAFAR MENSAGEM. Peçam que realizem o mesmo procedimento, da parte, vamos exemplificar. Peça que João envie a mesma mensagem para Ana, do mesmo modo que procedeu anteriormente, mas agora antes de clicar em criptografar, ele deve marcar a caixa Assinar Mensagem.

Na sequência, irá aparecer dois campos solicitando o valor de n e $Cpriv$. João deve inserir nestes campos a sua senha de leitura, sua chave privada. Não há problema em o fazer, pois os campos são do tipo senha preservando o segredo da mensagem.

Ao mesmo tempo, Pedro também deve escrever uma mensagem a Ana, e assiná-la. Como Pedro não conhece as credenciais privada de João, ele terá que utilizar as suas próprias chaves privadas, ou não suar nenhuma.

Agora, tanto Pedro, quanto João devem transmitir a sua mensagem a Ana. Caso eles queiram saber sobre a sua identidade, talvez torne o processo até mais interessante. Por exemplo, tanto Pedro quanto João, preenchem o campo Autor como João, assim, Ana não saberia qual o verdadeiro João.

Quando Ana receber as mensagens, ela deverá ir na tela para DECIFRAR MENSAGEM, lá irá proceder do mesmo modo que antes, porém deverá marcar o campo, MENSAGEM ASSINADA e lá colocar as credenciais públicas (Chave Pública) de João. Nesse momento, se ela fizer o mesmo processo de leitura com as mensagens de João e de Pedro, ela somente irá conseguir ler a mensagem do verdadeiro João. Provavelmente a outra nem será executada pelo aplicativo, podendo inclusive gerar erros, “bugando” ou até mesmo fechando a aplicação, pois o programa não conseguirá interpretar as mensagens.

4.5. Aplicação do *CodeClass*

No dia 01 de Outubro de 2019, foi realizada uma oficina no Colégio Estadual Governador Luiz Viana Filho, na cidade que Guanambi-BA, com o intuito de apresentar o aplicativo *CodeClass* e algumas das propostas metodológicas que foram fundamentadas e sugeridas. Nesta oficina, tivemos 13 colaboradores, entre professores e estudantes. Dentre eles, três professores da área de ciências da natureza e matemática e os demais, alunos do 1º Ano do Ensino Médio, sendo um da turma 1ºAM, um da turma 1ºBM e oito da turma 1ºDM, tidas regulares nesta instituição no turno matutino.

A oficina foi realizada no período da tarde, com início às 14 horas e término às 16 horas e 40 minutos. Como não havia tempo hábil disponível para realização de

todas as propostas sugeridas neste trabalho, foi elaborada uma proposta que agregasse um pouco de cada um dos temas sugeridos e que focasse mais na utilização do aplicativo e explanação teórica dos conceitos que regem a criptografia de modo prático. Deste modo, podemos realizar um teste das funcionalidades do sistema e se o mesmo é adequado para ser utilizado pelos estudantes.

4.5.1. A Oficina

Para que pudéssemos avaliar a atividade, foi elaborado um questionário subdivido em seis partes, e que se encontra disponível no Anexo A. Este questionário foi preenchido pelos participantes no decorrer da atividade. Cada parte é precedida de uma proposta diferente relacionada a criptografia, associada ao que queremos avaliar naquele instrumento.

A Parte I, trata dos conhecimentos prévios que o participante possui sobre os conteúdos que serão abordados e foram respondidas antes do início da proposta.

A Parte II do instrumento, trata do funcionamento do Sistema de Criptografia Linear e visa observar se os participantes foram capazes de conseguir utilizar este sistema criptográfico. Por meio deste, também conseguiremos observar possíveis causas de insucesso se este for o caso. Na Parte III, objetivamos avaliar se o conceito de chave pública foi compreendido pelos participantes. A quarta e quinta partes relacionam-se com a criptografia RSA, bem como a utilização da assinatura RSA. Caso os participantes tenham alguma dificuldade em utilizar o sistema, este instrumento também servirá de registro para possíveis falhas.

Finalmente a última parte do questionário, a Parte VI, possui perguntas que direcionam a coleta de informações a respeito da impressão do usuário quanto a interface do sistema, o qual iremos investigar opinião do utilizador, bem como suas impressões quanto a usabilidade do *CodeClass*.

Parte I

A primeira parte buscou coletar informações quanto ao conhecimento prévio dos voluntários. Foi observado que dos treze avaliados, todos desconheciam sobre a Cifra de César, apenas três em algum momento já ouviu falar sobre criptografia, mas questionados oralmente sobre onde eles podiam exemplificar aplicações deste tema,

o geral não soube. Um dos participantes alegou já conhecer superficialmente o tema, por ter iniciado um curso técnico em Informática, mas não deu continuidade.

Uma coisa que chamou bastante atenção neste estágio inicial, foi todos os integrantes possuírem aparelhos celulares, acessarem diariamente as redes sociais e até mesmo efetuarem transações financeiras no aparelho, mas mesmo assim nunca se questionaram sobre como é tratada a segurança destes dados.

Parte II

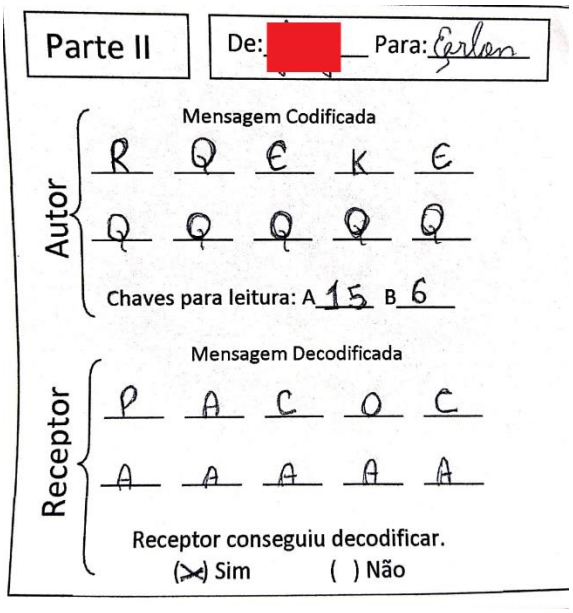
O início da oficina tratou da importância da criptografia para nossa era digital, em decorrência da conscientização da necessidade de escondermos informações e quem pode se beneficiar ou se prejudicar caso algo indevido seja exposto.

Em seguida foi apresentada a *Cifra de César* e como César fazia para utilizá-la à sua época. O mais interessante foi fazê-los perceber como a matemática se relaciona com está cifra por meio de uma função afim levando um número em outro. Afinal, como a maioria dos alunos, eram do 1ºAno Regular, o conteúdo de funções é amplamente trabalhado nesta série de acordo a BNCC [4]. Finalmente, foi apresentado o passo a passo para se executar esta codificação no *CodeClass*.

A atividade proposta foi cada um escolher dois parâmetros, seja A, seja B, uma mensagem de, no máximo, dez caracteres e enviá-la ao colega mais próximo, juntamente com a Chave para Leitura. Porém, essa informação não podia ser entregue diretamente, devendo antes passar pelo mediador da oficina que tentaria de todas as formas ler as informações trocadas. A ideia era que eles percebessem que as mensagens não estavam seguras visto que a senha para decodificar se encontrava no papel.

Como forma de auxiliá-los e de facilitar o acompanhamento em caso de erro ou distorções, foram utilizados cartões para que preenchessem com a mensagem codificada e enviassem a um colega.

Figura 23 Cartão da Parte II Criptografia Linear



Fonte: O autor

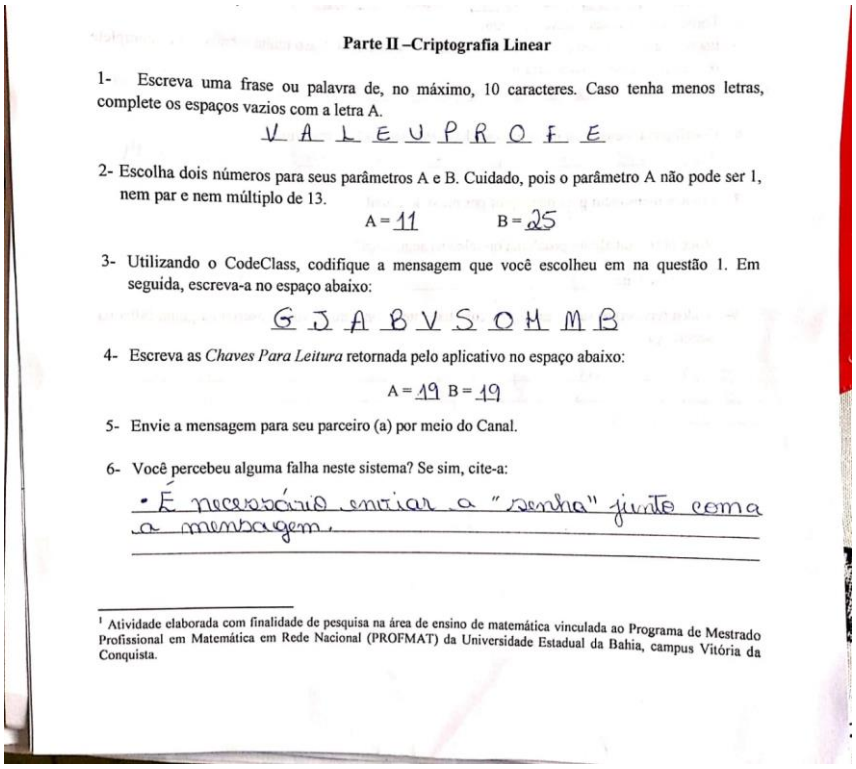
Neste cartão, o autor da mensagem iria escrever uma mensagem codificada pelo aplicativo e enviá-la para o receptor, juntamente com a chave para decodificá-la. Em seguida o receptor caso conseguisse decifrar a mensagem com o aplicativo deveria escrevê-la na parte inferior destinada a isto.

A Figura 23, foi um exemplo de mensagem que um dos participantes enviou. Nesta etapa todos conseguiram se comunicar. Um dos receptores não conseguiu decodificar a mensagem pois, o autor colocou as chaves de leitura invertidas, mas verificado o problema, o mesmo teve êxito na execução da atividade.

Muito alunos perceberam que havia algo errado em ter que enviar a senha junto, comentando em alguns casos. Dentre algumas falas, foi registrado, em anotações de campo a de um estudante:

“Ué, professor. Enviar essa senha tá meio estranho”
“E se alguém roubar a senha no caminho?”

Figura 24 Resposta de um aluno à Parte II



Parte II –Criptografia Linear

1- Escreva uma frase ou palavra de, no máximo, 10 caracteres. Caso tenha menos letras, complete os espaços vazios com a letra A.

V A L E U P R O F E

2- Escolha dois números para seus parâmetros A e B. Cuidado, pois o parâmetro A não pode ser 1, nem par e nem múltiplo de 13.

A = 11 B = 25

3- Utilizando o CodeClass, codifique a mensagem que você escolheu em na questão 1. Em seguida, escreva-a no espaço abaixo:

G J A B V S O H M B

4- Escreva as *Chaves Para Leitura* retornada pelo aplicativo no espaço abaixo:

A = 19 B = 19

5- Envie a mensagem para seu parceiro (a) por meio do Canal.

6- Você percebeu alguma falha neste sistema? Se sim, cite-a:

• É necessário enviar a "senha" junto com a mensagem.

[1] Atividade elaborada com finalidade de pesquisa na área de ensino de matemática vinculada ao Programa de Mestrado Profissional em Matemática em Rede Nacional (PROFMAT) da Universidade Estadual da Bahia, campus Vitória da Conquista.

Fonte: O autor

A Figura 24 mostra a resposta de um dos participantes à Parte II do questionário.

Parte III

Após todos conseguirem terminar a atividade proposta, foi feita uma dinâmica para tentarmos apresentar o conceito de chave pública. Na dinâmica, dois alunos foram escolhidos para trocar informações através de uma caixa. Porém, essa caixa não poderia ir sem um cadeado, pois no meio do caminho havia um terceiro aluno que tentaria abri-la. Assim, eles deveriam bolar uma estratégia para trocar as caixas, mantendo o conteúdo em segurança.

Os participantes, de maneira coletiva, tentaram bolar um plano para conseguir cumprir a atividade, mas sem êxito. Houve então a necessidade da interferência do mediador, que sugeriu colocar os dois cadeados na caixa. Neste momento um dos alunos conseguiu dar uma sugestão que completasse o desafio.

A solução deste problema, consistem em enviar a caixa com a mensagem dentro, trancada com o primeiro cadeado. Ao receber a caixa, o receptor irá colocar o seu cadeado na caixa e enviá-la de volta para o primeiro usuário, que irá retirar o seu cadeado e novamente enviá-la para o receptor. Que poderá finalmente abrir o último cadeado e revelar o seu conteúdo.

Nos questionários percebemos que somente um dos participantes não conseguiu compreender a funcionamento desta estratégia. Marcando que seriam necessárias mais que três “viagens da caixa” para o envio da informação.

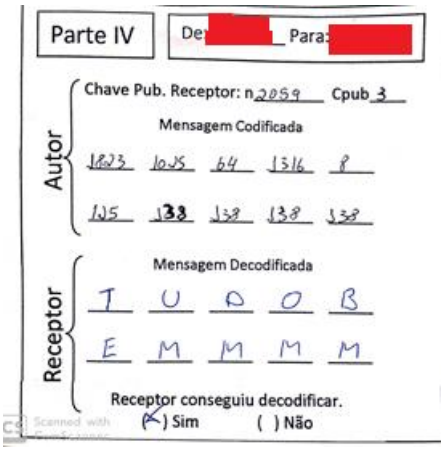
Parte IV

O objetivo da Parte IV, era simular um sistema de troca de mensagens pelo canal utilizando o sistema RSA. Primeiramente utilizamos o aplicativo para gerar as chaves públicas e privadas de cada um dos alunos presentes. Estas chaves foram inseridas nos questionários para que após a finalização da atividade pudéssemos testas.

Verificamos que todas as chaves foram geradas com sucesso, porém um dos participantes não conseguiu utilizar primos maiores que 17 para geração de suas chaves. O aplicativo era finalizado toda vez que este tentava acrescentar números maiores. Ainda não conseguimos identificar o erro que aconteceu unicamente em um aparelho da marca Samsung modelo J5 Prime.

Após cada um ter gerado as suas chaves, todas as chaves públicas foram divulgadas no quadro, para facilitar o processo de envio. Em seguida foram solicitados a escrever a mensagem ao colega no cartão correspondente para que fosse enviada através do Canal. Quando todos terminaram de redigir sua mensagem, o ministrante se passando pelo Canal, recolheu todos os cartões para entregá-los aos respectivos destinatários que iriam realizar as suas respectivas decodificações. Na Figura 25, segue um exemplo dos cartões enviados.

Figura 25 Troca de mensagens utilizando Criptografia RSA. Parte IV



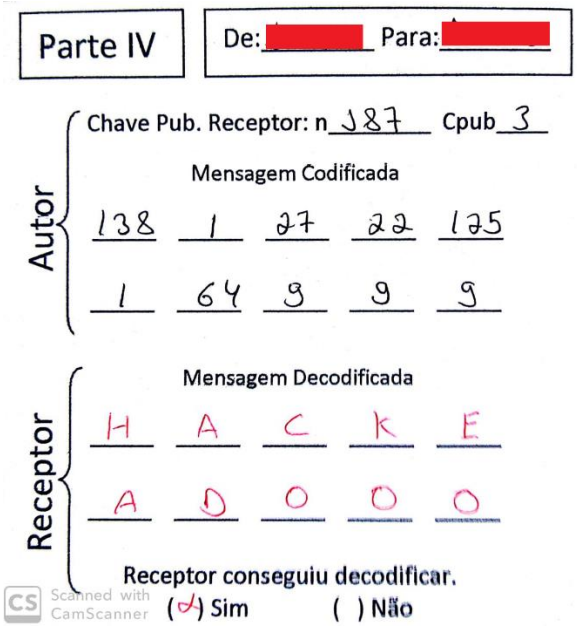
Fonte: O autor

Nesta etapa todos os alunos conseguiram efetuar a leitura de suas mensagens de maneira correta, porém alguns não a registraram no cartão, mantendo somente a decodificação no aparelho.

Nesta etapa, foi inserida também uma jogada surpresa por parte do ministrante. Sem que os alunos soubessem, enquanto os autores estavam redigindo as mensagens, foram confeccionados cartões falsos pelo mediador, que seriam trocados na hora da entrega da seguinte forma.:

A aluno A, enviou para o aluno B uma certa mensagem. O mediador havia previamente confeccionado um cartão falso do aluno A para o B e efetuou a substituição do cartão verdadeiro pelo falso no momento da entrega. Após a decodificação, eles foram surpreendidos pela mensagem da Figura 26.

Figura 26 Cartão falso, confeccionado pelo mediador.



Fonte: O autor

Os participantes começaram a questionar sobre a segurança e debatendo sobre o que poderia ter acontecido. Sozinhos chegaram a conclusão de que os cartões haviam sido alterados. Neste momento um aluno questionou:

“ Como eu vou ter certeza que foi o “aluno A” que enviou a mensagem para mim? ”

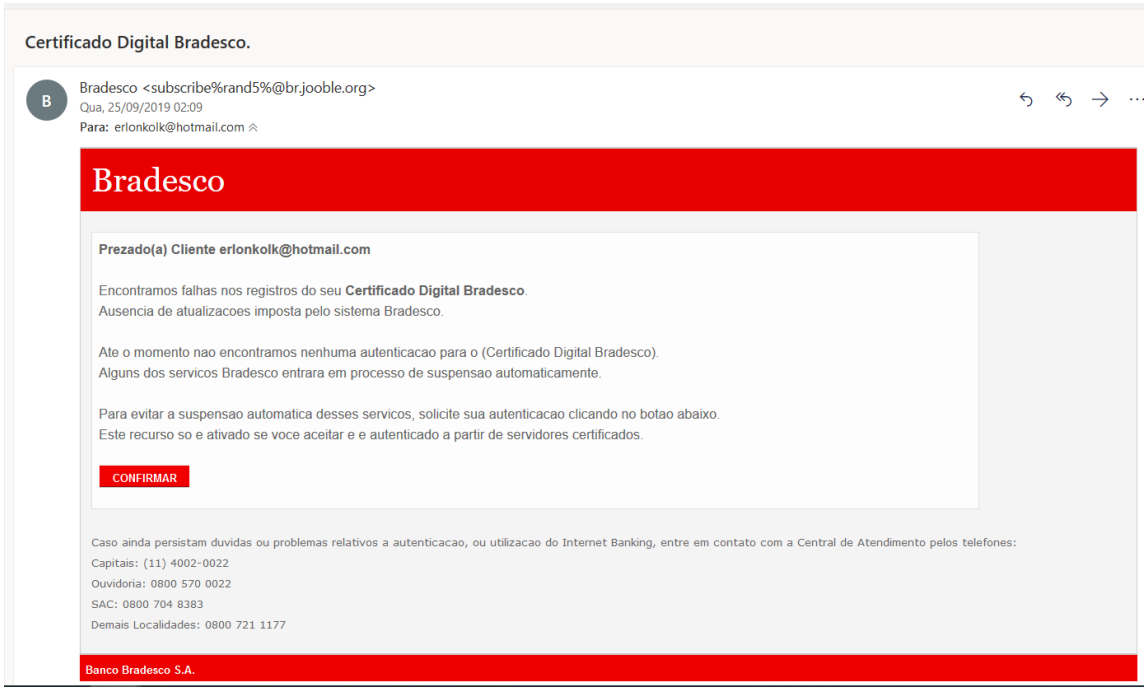
Aluno B

Parte V

A Parte V e a última que envolve a utilização do aplicativo, veio para responder o questionamento do aluno ao final da Parte IV. Como ter a certeza da autenticidade do remetente?

Foram apresentados durante a atividade, exemplos de situações práticas onde necessitamos verificar quem de fato é o remente, como nos casos dos e-mails. Foi apresentado o seguinte e-mail da Figura 27.

Figura 27 E-mail falso



Fonte: O autor

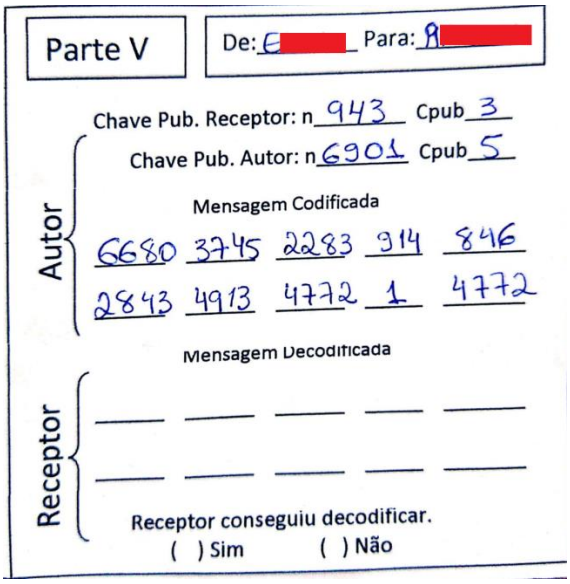
Após a explanação, os participantes perceberam como é importante no meio digital garantir a veracidade de uma informação, afinal caso acreditássemos que o e-mail era de fato do Banco Bradesco, poderíamos colocar a integridade do nosso computador em risco pois, muito provavelmente este link levaria a algum Malware que infectasse nossa máquina.

A dinâmica desta última parte, consistia em realizar a codificação de uma mensagem, utilizando também a função de Assinatura do aplicativo CodeClass. Esta etapa exigia um cuido a mais dos alunos, pois qualquer erro de digitação por parte do autor, iria inviabilizar o processo de decodificação.

O processo não ocorreu da maneira devida, pois o aplicativo apresentou um erro na decodificação, que os alunos não conseguirem contornar e isto acabou interferindo negativamente nesta etapa final.

O erro apresentado em todos os aparelhos, foi identificado como sendo na programação do sistema, após uma pequena alteração que havia sido feita no aplicativo. A imagem da Figura 28, é a codificação da mensagem T O M E C U I D A D, que o participante E enviou ao participante R, utilizando a função assinatura.

Figura 28 Troca de mensagens entre E e R utilizando assinatura.

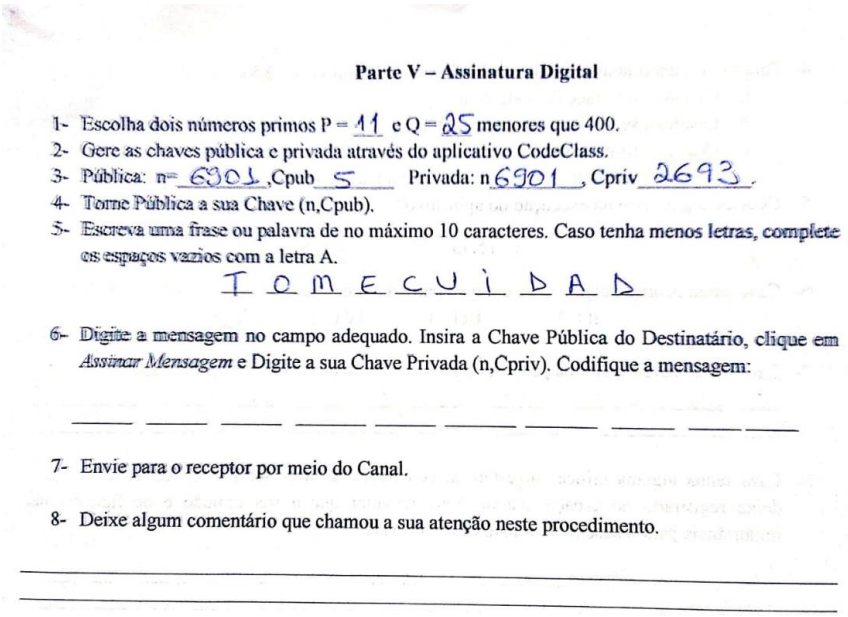


Fonte: O autor

Após comparar os questionários, conseguimos identificar as chaves públicas e privadas de ambos participantes, para que pudéssemos simular o processo e entender a origem do erro.

O participante E que é autor da mensagem, possuía Chave Pública: $n = 6901$ e $Cpub = 5$ e Chave Privada: $n = 6901$ e $Cpriv = 2693$, como pode ser verificado na Figura 29.

Figura 29 Chaves do Autor E



Parte V – Assinatura Digital

1- Escolha dois números primos P = 11 e Q = 25 menores que 400.
2- Gere as chaves pública e privada através do aplicativo CodeClass.
3- Pública: n= 6901 ,Cpub 5 Privada: n 6901 , Cpriv 2693 .
4- Torne Pública a sua Chave (n,Cpub).
5- Escreva uma frase ou palavra de no máximo 10 caracteres. Caso tenha menos letras, complete os espaços vazios com a letra A.

T O M E C U I D A D

6- Digite a mensagem no campo adequado. Insira a Chave Pública do Destinatário, clique em *Assinar Mensagem* e Digite a sua Chave Privada (n,Cpriv). Codifique a mensagem:

7- Envie para o receptor por meio do Canal.

8- Deixe algum comentário que chamou a sua atenção neste procedimento.

Fonte: O autor

Já o usuário R, possuía a Chave Públicas e Privadas respectivamente $n = 946, Cpub = 3$ e $n = 943, Cpriv = 587$.

A codificação feita por E, foi correta, porém, na decodificação, quando R inseriu os dados no aplicativo juntamente com a sua Chave Privada e a Chave Pública do autor, obteve o resultado como da Figura 30

Figura 30 Resultado apresentado na decodificação

Fonte: O autor

Observe que os números apresentados deveriam ser menores que 26, para que fosse possível converter a mensagem em texto. Concluímos então que o sistema não codificou corretamente a mensagem. Porém caso pressionássemos novamente o botão DECODIFICAR MENSAGEM, iríamos perceber que a mensagem seria decodificada corretamente.

Figura 31 Mensagem após segunda tentativa de decodificação

Fonte: O autor

Porém ainda apresentado erro, pois o botão de conversão para texto não apareceu, tornando necessária uma conversão manual utilizando a correspondência da Figura 5. Com isso, a etapa proposta não obteve o sucesso esperado, mas foi explanado aos participantes sobre o problema e realizado um exemplo coletivamente de decodificação com assinatura.

Parte VI

Finalmente a última parte do questionário tratava sobre as percepções do participante a respeito da oficina e do aplicativo. Buscamos coletar informações a respeito da dinâmica da oficina, interface e usabilidade do CodeClass.

Devido a finalização da oficina prevista para as 16:30 e ter se estendido até às 16:40, muitos alunos tiveram de sair apressadamente deixando parte da sexta parte incompleta.

Dos que avaliaram o aplicativo, pudemos perceber que no geral, avaliaram positivamente a interface e boa usabilidade. Porém o ponto que mais chamou a atenção, foi a o erro na última tarefa. O aplicativo nesta etapa recebeu notas mais baixas na avaliação.

De um modo geral, pudemos perceber que os participantes avaliaram positivamente a atividade e a forma como estes conteúdos, que eram desconhecidos por muitos, foram trabalhados. Os docentes envolvidos acharam a oficina *“bem interativa e dinâmica”*.

Julgaram também que a oficina deveria ser dividida em mais seções afim de diluir as informações passadas, dando tempo para: “*digerir todas as informações*” que se faziam presentes.

5. RESULTADOS E DISCUSSÕES

Tentaremos apresentar agora um paralelo de todo o processo que foi desenvolvido desde o início deste projeto até o momento, apresentado as falhas e acertos que obtivemos durante toda a execução do mesmo.

1.1. Confecção do *CodeClass*

Durante o desenvolvimento do aplicativo, pudemos perceber a necessidade de compreender processos mais avançados de programação, para que seja possível reduzir o tempo de execução de muitas tarefas que o CodeClass executa. Acreditamos que o aplicativo ainda é muito primitivo neste ponto, pois faz-se necessário ainda a execução de tarefas repetidas para obter certo resultado.

Além disso, gostaríamos de tornar a entrada de informações mais intuitiva, de modo que fosse mais fácil para o usuário inserir os dados na plataforma. Imagine você simplesmente poder a mensagem em um único campo e o sistema conseguir identificar cada caractere independente de ser maiúscula ou minúscula. Estas serão provavelmente melhorias nas próximas atualizações do aplicativo do CodeClass.

Um outro processo que identificamos falhas, foi na geração das chaves. Por mais que funcione, o tempo de execução aparenta ainda ser extenso. Visto que para números relativamente pequenos a partir 600, não conseguimos concluir a tarefa pela demora na execução. Isto se dá pelo fato do algoritmo utilizado na resolução da Equação Diofantina, necessária na geração das chaves, não ser eficiente, vide Anexo 2.

Acreditamos que estes processos podem ser otimizados, com o auxílio de algum colaborador na área de computação. Sabemos também da limitação da plataforma em que o mesmo foi desenvolvido. E por isso ainda é um desafio conseguirmos encontrar meios de contornar certas demandas.

Um resultado que já obtivemos em umas das atualizações feitas antes mesmo da utilização do aplicativo, foi a alteração do algoritmo que faz a verificação dos primos.

Na tela de geração dos primos, utilizamos um algoritmo apresentado em um vídeo do projeto Matemática com Aplicativos, que pode ser acessado em [13], que utiliza o Teorema de Wilson, na sua confecção. Para geração dos primos, ele se mostra bastante eficiente, visto que é possível armazenar o valor anterior na memória e reutilizá-lo para determinar o próximo primo. Porém quando tivemos que montar uma rotina para verificar a primalidade de certo número, este algoritmo não se mostrou tão eficiente quanto a força bruta (testar número por número até obtermos um divisor ou não). Visto que este teste deve ser feito somente com os ímpares até a raiz do primo que desejamos verificar.

1.2. **Oficina**

Na oficina podemos perceber como a criptografia ainda é pouco difundida nos meios públicos. Mais impressionantes ainda é observar como a segurança na internet é pouco discutida. O número de ataques cibernéticos só tem aumentado no Brasil, chegando a dobrar em 2018 segundo reportagem do Canaltech disponível em [15]. Ainda assim, percebemos o quanto que as pessoas pouco sabem ou não se interessam pelam tema. Quando realizávamos a divulgação da oficina, poucas pessoas apresentaram interesse no assunto.

Durante as oficinas, percebemos que um ponto que foi bastante comentado pelos participantes foi a Criptografia Linear. Percebemos que eles gostaram bem mais desta do que da RSA. Talvez por ser mais simples de se entender os exemplos da Cifra de César que ilustra bastante o tema. Ela foi crucial também para poderem comparar o que difere a Linear da RSA quanto a ter a chave pública exposta.

Encontramos então uma aplicação para as Funções Afim, juntamente com a Aritmética Modular, abordada normalmente em livros do Ensino Médio. Além disso, o fato dos alunos poderem participar da atividade com o celular a todo momento em seu porte, os motivou a querer prestar mais atenção na proposta.

6. CONSIDERAÇÕES FINAIS

A criptografia busca utilizar diversos métodos não só matemáticos para esconder informações, como foi o caso da Citale Espartana. Exatamente por isso, é uma porta de entrada para tratar de diversos assuntos presentes no currículo não só do ensino médio, como também fundamental.

Não precisamos montar um sistema complexo, ou mesmo superseguro para se trabalhar criptografia dentro da sala de aula. O que precisamos é encontrar formas de torná-la acessível e prática. Podemos criar diversas técnicas de se criptografar uma mensagem utilizando muitos conteúdos da disciplina de matemática, as temáticas que abordamos aqui são apenas uma pontinha de um leque vasto e diversificado de sistemas.

Uma busca rápida pelo repositório de dissertações do PROFMAT, já é possível identificar quase 70 trabalhos que visam trazer aplicações de criptografia no ensino médio. Estes, buscam aplicar alguns conteúdos como matrizes, resíduos quadráticos, recorrências e funções. Isso mostra que é possível trazer para sala de aula, fatos que sejam realmente relevantes para o aluno. É possível dar um sentindo e uma aplicação prática ao emaranhado de números e operações matemáticas.

Além disso, torna-se possível despertar um senso crítico nos alunos, instigando-os a discutir se é possível “criar” criptografias com uma função afim, mas o porquê de não o fazê-la com a função quadrática. Será que podemos modificar a função do segundo grau para que possamos utilizá-la?

Alguns métodos ainda, são muito menos falados, mas poderíamos ainda citar o Criptex supostamente elaborado por Leonardo da Vinci que Dan Brow apresenta em seu livro e que posteriormente virou filme *O Código da Vinci* [16]. Um cofre, que necessita de uma senha para revelar seu conteúdo e, se forçado libera uma substância em seu interior que destrói a mensagem. É a Geometria e a Engenharia juntas para esconder segredos e aumentar ainda mais a gama do que pode ser trabalhado.

Montamos uma proposta que ainda tem muito a amadurecer e ser aperfeiçoada, e que foi bem aceita pelos estudantes. Acreditamos ainda, que seja possível aguçar ainda mais o interesse, se além de apresentarmos o sistema,

trabalharmos também a programação por traz do aplicativo. As oportunidades são inúmeras, resta agora tornarmos a criptografia cada vez mais popular.

REFERÊNCIAS

[1] COUTINHO S. C. **Números Inteiros e Criptografia RSA.** Rio de Janeiro: IMPA, 2011. (Coleção Matemática e Aplicações)

[2] SINGH, S. **O livro dos códigos**. Record, Rio de Janeiro. Tradução de Jorge Calife, 2001.

[3] BAHIA SECRETARIA DA EDUCAÇÃO. **Orientações curriculares para o ensino médio área: matemática** / Secretaria da Educação. – Salvador: Secretaria da Educação, 2015. Disponível em <http://escolas.educacao.ba.gov.br/sites/default/files/private/midiateca/documentos/2017/matematica.pdf>. Acessado dia 03/10/2019.

[4] BRASIL. **Base Nacional Comum Curricular.** Brasília. MEC/SEF, 2019. Disponível em : http://basenacionalcomum.mec.gov.br/images/BNCC_EI_EF_110518_versaofinal_site.pdf> Acessado em: 03/10/2019

[5] SANTOS J. P. O. **Introdução à Teoria dos Números.** -3 ed.- Rio de Janeiro: IMPA, 2018.

[6] HEFEZ, A. **Aritmética.** Rio de Janeiro: SBM, 2016.

[7] MARTINEZ, F. B. et al. **Teoria dos Números: um passeio com primos e outros números familiares pelo mundo inteiro**. Rio de Janeiro: IMPA, 2018.

[8] COMPUTAÇÃO NA ESCOLA **Quem Somos.** Disponível em <http://www.computacaonaescola.ufsc.br/?page_id=1656.> Acessado em 13/08/2019).

[9] PROJETO COMPUTAÇÃO NA ESCOLA **Página do Projeto**. Disponível em: <http://www.computacaonaescola.ufsc.br/> .Acessado em 13/08/2019 .

[10] MIT APP INVENTOR II Acesso a Plataforma. Disponível em: <http://ai2.appinventor.mit.edu/> Acessado em : 19/09/2019.

[11] LEMOS M. **Criptografia, Números Primos e Algoritmos.** Rio de Janeiro: IMPA, 2009. (Publicações Matemáticas)

[12] ALVES E. A. W. **Aplicativo CodeClass para andriod.** CodeClass.apk – Disponível em: <https://drive.google.com/open?id=1ZIGebePmMCXP6gXMrBTfBFWfoQGQXa-j> ou Disponível em: <https://is.gd/lniSDC> (encurtador de url). Acessado em: 01/10/2019

[13] BARBOSA. M. A. **Números Primos - Teorema de Wilson - MIT App Inventor.** Disponível em: https://www.youtube.com/watch?v=vyROeH5b7kg. Acessado em: 10/09/2019.

[14] BARBOSA. M. A. **Desenvolvendo Aplicativos Para Dispositivos Móveis Através Do Mit App Inventor 2 Nas Aulas De Matemática**. 2016. 144 f. Dissertação (Mestrado em Matemática) - PROFMAT, UESC, Bahia.

[15] WAKKA W. **Número de ataques cibernéticos no Brasil quase que dobrou em 2018.** Disponível em: <https://canaltech.com.br/seguranca/numero-de-ataques-ciberneticos-no-brasil-quase-que-dobrou-em-2018-119600/> Acessado em: 30/09/2019

[16] BROW D., **The Da Vinci code.** First edition ed. New York: Doubleday, 2003.

ANEXO A

Criptografia e o CodeClass Uma Proposta Metodológica[6]

Você é: Estudante ( ) Professor ( )

Parte I – Grau de Conhecimento

1- Você sabe o que é criptografia?
( ) Sim ( ) Não
2- Caso tenha marcado sim na Pergunta 1, responda: Você entende a importância da criptografia na nossa era digital?
( ) Sim ( ) Não
3- Já ouviu falar do termo *Sistema de Chave Pública*?
( ) Sim ( ) Não
4- Você conhece a *Cifra de César*?
( ) Sim ( ) Não

Parte II – Criptografia Linear

1- Escreva uma frase ou palavra de, no máximo, 10 caracteres. Caso tenha menos letras, complete os espaços vazios com a letra A.

___ ___ ___ ___ ___ ___ ___ ___ ___ ___

2- Escolha dois números para seus parâmetros A e B. Cuidado, pois o parâmetro A não pode ser 1, nem par e nem múltiplo de 13.

A = ___ B = ___

3- Utilizando o CodeClass, codifique a mensagem que você escolheu em na questão 1. Em seguida, escreva-a no espaço abaixo:

___ ___ ___ ___ ___ ___ ___ ___ ___ ___

4- Escreva as *Chaves Para Leitura* retornada pelo aplicativo no espaço abaixo:

A = ___ B = ___

5- Envie a mensagem para seu parceiro (a) por meio do Canal.

---

[6] Atividade elaborada com finalidade de pesquisa na área de ensino de matemática vinculada ao Programa de Mestrado Profissional em Matemática em Rede Nacional (PROFMAT) da Universidade Estadual da Bahia, campus Vitória da Conquista.

6- Você percebeu alguma falha neste sistema? Se sim, cite-a:

_____
_____
_____
_____

Parte III – Criptografia Chave Pública

1- Você compreendeu o princípio de Codificação por Chave Pública?
   ( ) Sim. ( ) Um pouco, ainda tenho algumas dúvidas. ( ) Não Compreendi.
2- Quantas viagens a caixa teve que realizar para atingirmos o objetivo?
   ( ) 1 ( ) 2 ( ) 3 ( ) 4+
3- E quando tornamos a chave do cadeado pública, quantas viagens teve que realizar?
   ( ) 1 ( ) 2 ( ) 3 ( ) 4+

Parte IV – Criptografia RSA

1- Escolha dois números primos P = ___ e Q = ___ menores que 400.
2- Gere as chaves públicas e privadas por meio do aplicativo CodeClass.
3- Pública: n=_____,Cpub_____ Privada: n_____, Cpriv_____
4- Torne Pública a sua Chave (n,Cpub).
5- Escreva uma frase ou palavra de no máximo 10 caracteres. Caso tenha menos letras, complete os espaços vazios com a letra A.

___ ___ ___ ___ ___ ___ ___ ___ ___ ___

1- Codifique a mensagem utilizando a Chave Pública do Destinatário

_____ _____ _____ _____ _____ _____ _____ _____ _____

2- Envie a mensagem para o receptor Por meio do Canal.

3- Você observou algum problema ou falha na segurança?

( ) Sim. Cite_____ ( ) Não

4- Todos receberam suas mensagens corretamente? Comente se você observou alguma falha na segurança.

_____
_____
_____
_____

Parte V – Assinatura Digital

1- Escolha dois números primos P = ___ e Q = ___ menores que 400.
2- Gere as chaves pública e privada através do aplicativo CodeClass.
3- Pública: n=_____,Cpub_____ Privada: n_____, Cpriv_____.
4- Torne Pública a sua Chave (n,Cpub).
5- Escreva uma frase ou palavra de no máximo 10 caracteres. Caso tenha menos letras, complete os espaços vazios com a letra A.

___ ___ ___ ___ ___ ___ ___ ___ ___ ___

6- Digite a mensagem no campo adequado. Insira a Chave Pública do Destinatário, clique em *Assinar Mensagem* e Digite a sua Chave Privada (n,Cpriv). Codifique a mensagem:

_____ _____ _____ _____ _____ _____ _____ _____ _____

7- Envie para o receptor por meio do Canal.

8- Deixe algum comentário que chamou a sua atenção neste procedimento.

_____
_____
_____

Parte VI – Sobre o Aplicativo

1- Você acredita que conseguiu compreender os principais conceitos que envolvem a criptografia?

( ) Compreendi alguns conceitos ( ) Compreendi a maioria dos conceitos

( ) Compreendi todos os conceitos ( ) Não compreendi muito bem

2- Das Partes II, III, IV, V da oficina, qual lhe chamou mais atenção? E por quê?

II ( ) III ( ) IV ( ) V ( )

_____
_____
_____
_____

3- Você sentiu a necessidade de compreender como são feitas algumas etapas dos processos de codificação e decodificação?

( ) Não ( ) Sim. Qual? _____

4- Para as perguntas abaixo, avalie de 1 a 5, onde 1 é muito ruim e 5 excelente.
5- Quanto a interface do aplicativo ____.
6- Quanto a velocidade de processamento ___.
7- Quanto a forma como as telas estão dispostas ___.
8- Quanto a erros que apareceram durante a realização das atividades ___.

ANEXO B

Parte do algoritmo de geração de chaves do Sistema RSA