UNIVERSIDADE PAULISTA – UNIP
Instituto de Ciência e Tecnologia - ICET

Sistemas de Informação
Campus Tatuapé

ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

Criptografia

André Dames de Oliveira Pereira D7621E-9

Bárbara Nunes Malhone N3754E-0

Djair Barral Pires D75AID-0

Gabriel de Oliveira Feba D74ABD-4

Jemerson Santos dos Reis D75AJG-2

Juliana Vieira de Carvalho N34558-6

Kauan Isaías de Morais D82AGF-0

Turma: SI1P33/SI2P33

SÃO PAULO
2018

UNIVERSIDADE PAULISTA – UNIP
Instituto de Ciência e Tecnologia - ICET

Sistemas de Informação
Campus Tatuapé

ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

Criptografia

Pesquisa do Curso de Sistemas de Informação, apresentado à Universidade Paulista, referente as Atividades Práticas Supervisionadas, como requisito de avaliação.

Prof. Orientador: Luiz Machi Lozano

SÃO PAULO
2018

SUMÁRIO

1. INTRODUÇÃO

O projeto tem como objetivo mostrar a importância da segurança das informações contidas na internet e no sistema global. Através de pesquisas vimos que há uma grande falha nos sistemas e nas redes de comunicação através da internet, dando espaço para ataques cibernéticos por meio de hackers e crackers, podendo haver perdas e exposições de informações e documentos sigilosos e secretos.

As pesquisas realizadas neste projeto nos trazem à reflexão sobre os métodos de segurança em todas as informações desde que sejam de grande ou pequena importância. O trabalho tem como objetivo criar um programa que criptografa e descriptografa mensagens usando o método RSA.

A RSA tem como método utilizar multiplicação de números primos e divisões modulares para cifrar a mensagem ou informações que o usuário envia para outro usuário. Tendo assim um fluxo de informações e dados com maior segurança. Este projeto teve pesquisas elaboradas através de sites de tecnologia e do Google acadêmico.

2. OBJETIVO

Durante o desenvolvimento desta pesquisa serão apresentadas informações sobre criptografia, seus tipos e sua importância. Com base nestas informações selecionaremos uma dentre as estudadas na pesquisa para que seja feito, de forma prática, um software estruturado em C que converta, com o uso de programação lógica, as informações compreensíveis em informações incompreensíveis a qualquer um que não tenha a chave correta.

3. INFORMAÇÃO

A informação é um conjunto de dados e conhecimentos organizados, que pode fazer referência a um determinado acontecimento, fato ou fenômeno. Com a evolução da sociedade e da tecnologia, a informação tornou-se muito mais do que simplesmente, “quantas ovelhas temos nosso cercado” ou “quem é nosso avô”. A partir dos dados convertidos em informações conseguimos melhorar, preservar e salvar vidas. Isto nos faz perceber o quanto é importante a sua confiabilidade e a sua veracidade em nosso dia-a-dia. Aqui estão algumas definições de informação importante para nós:

- A informação se configura em um recurso que confere significado a realidade mediante seus códigos e o conjunto de dados. Ela é capaz de dar origem ao desenvolvimento do pensamento humano.
- Ela também consiste em resolver problemas e tomar decisões, com base no uso coerente deste conhecimento adquirido através dela. Desta forma, quanto mais precisa ela seja, melhor será a comunicação.
- Nas áreas da informática e da tecnologia, a informação pode ser o agrupamento dos dados processados em um computador e que são capazes de gerar resultados para um determinado projeto.

Desde as épocas mais antigas tinha-se hábito de armazenar informações em argilas, tábuas e paredes, mas com o passar tempo percebemos que, ter as informações corretas nos fazia ganhar guerras, batalhas e dinheiro. Porém, a falta de segurança dessas informações fazia o oposto, assim os antigos perceberam a necessidade de sua proteção e começaram a criar maneiras de transformar algo legível para todos em algo legível apenas para alguns, essa prática foi chamada de criptografia.

4. CRIPTOGRAFIA

Com origem grega (kriptós – secreto) e (gráfein – escrita), a criptografia nada mais é do que cifrar um dado texto por meio de símbolos abreviaturas, de modo que fique ilegível para quem não tem acesso ao código criado. O objetivo principal da criptografia, é a confidencialidade, uma boa criptografia tem que garantir que a mensagem enviada seja lida apenas pelo seu destinatário. É uma técnica já antiga, usada por gregos e espartanos em troca de mensagens importantes. Podemos dividir a história da criptografia em dois períodos, a criptografia clássica, e a criptografia moderna. A fase clássica, vai dos povos antigos, idade média, até a segunda guerra mundial.

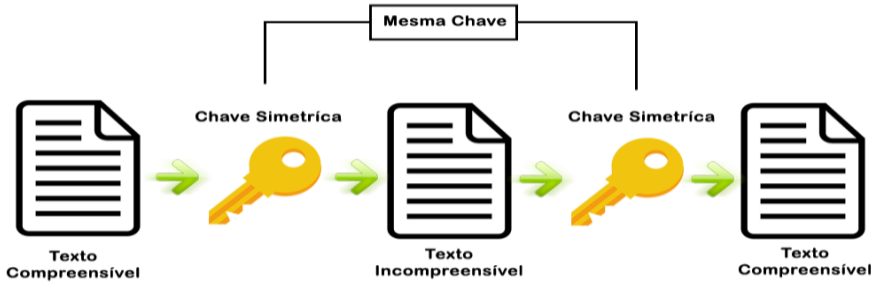
Existem duas formas básicas para se criptografar uma mensagem: usando códigos ou uma criptografia. Os códigos são apenas substituídos por uma letra ou por outra palavra, um código manipula o significado das mensagens, já uma cifra consiste na alteração da disposição de uma mensagem. Um exemplo, o modo que a palavra é cifrada como, "janela" para "kbofmb", em que é apenas substituída pela próxima letra do alfabeto, essa é a criptografia da cifra de cesar, recebe este nome pois era o método utilizado por Júlio Cesar (100 a.c – 44 a.c) para enviar mensagens confidenciais militares.

O estudo da quebra de criptografias, foi a gênese para a criação de computadores digitais, computadores que deram origem a algoritmos de criptografia. Tais algoritmos usavam conjunto de bits aleatórios (caracteres), para acessar essas mensagens criptografadas é necessário o uso de chaves. Essas chaves contem a mensagem criptografada e só o algoritmo correto consegue desembaralhar a mensagem. São separadas em dois modelos simétricas e assimétricas.

4.1 Chaves Simétricas

Chaves simétricas, nesse modelo a mesma chave é usada para codificar e descodificar. Quando compartilhada entre o usuário e o receptor, pode ser interceptada, e assim criando uma vulnerabilidade. São menos complexas, então necessita de menos processamento e pouco poder computacional.

Imagem 1 – Ilustração de chave simétrica

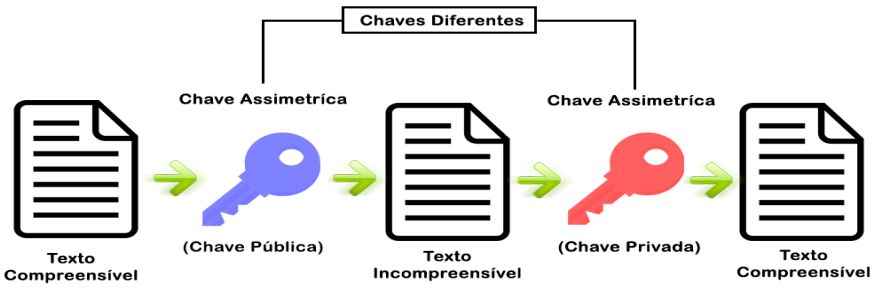


Do autor, 2018.

4.2 Chaves Assimétrica

Chaves assimétrica, nesse modelo utiliza duas chaves, uma pública que codifica a mensagem e outra privada que descodifica. Portanto, menor vulnerabilidade, pois somente a chave pública é divulgada, a chave privada fica segura com o receptor. São mais complexas, então necessita de um poder computacional maior e mais tempo de processamento.

Imagem 2 – Ilustração de chave assimétrica



Fonte: Do autor, 2018.

5. CRIPTOGRAFIA SIMÉTRICA RC4

Desenvolvida em 1987 pelo mesmo desenvolvedor da criptografia RSA, Ronald Rivest, O algoritmo criptográfico RC4, atualmente, não é considerado um dos melhores métodos criptográficos pelos adeptos a criptografia e pode se converter em um sistema muito inseguro em algumas aplicações. Porém num contexto prático, alguns sistemas baseados em RC4 podem ser seguros o suficiente. (B.SVAITE, D,2016).

O RC4 possui duas funcionalidades básicas:

- **KSA**

  (“Key Scheduler Algorithm” ou “Algoritmo Escalonador de Chaves”) Este, por meio de uma permutação pseudoaleatória é responsável por gerar uma chave secreta que será usada para encriptar e decriptar. O retorno desta chave não varia em relação ao tempo, dependendo apenas do valor de entrada, por esta característica ela é chamada de pseudoaleatória.

- **PRGA**

  ("Pseudo Random Generation Algorithm" ou “Algoritmo de Geração Pseudoaleatória”) O PRGA é o responsável por encriptar a mensagem propriamente dita a partir da chave obtida no KSA. Basicamente ela consiste em realizar operações de Ou-exclusivo(XOR) entre a permutação da chave obtida e a mensagem a ser encriptada, obtendo-se a mensagem cifrada. Sabemos por meio da lógica booleana que a operação XOR é uma operação do tipo simétrica, e, se utilizada a mesma permutação do processo de encriptação, no processo de decriptação será gerada a mensagem original novamente.

Este método criptográfico já foi um dos mais utilizados, por não ser afetado por ataques direcionados aos algoritmos de bloco e sua velocidade na nas operações de encriptação e decriptação eram o seu diferencial e, apesar de sua simplicidade e insegurança foi amplamente utilizada principalmente na proteção de redes sem fio. O protocolo WEP, pioneiro em relação a proteção de redes sem fio, utiliza o RC4 como método criptográfico e é, ainda hoje, utilizado em muitas residências mesmo já estando obsoleto no quesito segurança. Foi amplamente utilizado também no

protocolo SSL/TLS que é o responsável pela segurança do tráfego de informações pela internet, como por exemplo a proteção de sites e e-mails. (SALAS,2011).

Porém, desde a sua publicação na lista Cipherpunks, em 1994, começaram a surgir vários ataques criptoanalíticos, mas em 2001 é que foi demonstrado que a criptografia RC4 possuía uma vulnerabilidade no KSA, e que, protocolos de segurança que utilizassem este tipo de criptografia como o WEP, seriam facilmente atacados. Por esta razão, desde o início de 2016, os maiores navegadores como o Chrome, EDGE, Mozilla Firefox entre outros passaram a descontinuar o RC4 e, todos os servidores HTTPS que só suportavam o RC4 pararam de funcionar nestes navegadores. (PAIM, R)

6. CRIPTOGRAFIA ASSIMÉTRICA RSA

A criptografia RSA foi criada por três grandes professores do MIT (Massachusetts Institute of Technology): Ronald Rivest, Adi Shamir e Leonard Adleman (que avaliava as concepções matematicamente). Eles estavam em busca de um método de criptografia de chave assimétrica mais eficaz e com mais segurança. No ano de 1977, os três cientistas conseguiram registrar a patente e atualmente a RSA é a criptografia mais utilizada no mundo todo.

Nessa criptografia são utilizadas uma chave pública e uma privada. A chave pública pode ser repassada livremente para qualquer pessoa que queira enviar uma mensagem criptografada para quem “criou” a chave. No entanto, se uma pessoa tentar descriptografar a mensagem, ela só terá sucesso se possuir a chave privada, que é mantida em segredo.

As chaves (pública e privada) que compõem a criptografia RSA são basicamente formadas através da multiplicação de dois números primos. O resultado gerado dessa operação será público, porém a segurança da RSA é tão eficaz que se o resultado for um número muito grande, são necessários anos para fatorar o número e encontrar os números primos que foram utilizados para formar a multiplicação. (OLIVEIRA, 2010).

7. CRIPTOGRAFIA QUÂNTICA

A criptografia quântica é a criptografias que utilizam princípios da Mecânica Quântica para que seja garantida uma comunicação segura no sistema. Com ela, o que manda a mensagem criptografada e o que recebe podem criar chaves secretas para criptografar e descriptografar códigos ou mensagens.

A criptografia quântica se destaca de outros métodos criptográficos por não precisar de comunicações secretas prévias, ela permite a detecção de intrusos e se mantêm segura mesmo que o intruso tem acesso ilimitado. Na verdade, ela é totalmente segura, exceto quando o intruso consegue remover e inserir mensagens no canal de transmissão(poder ler , tirar a mensagem, copiá-la e reenviá-la). Atualmente, essa técnica criptográfica seria mais segura para se usar, pois se baseia em leis da física, enquanto as que são usadas atualmente, se asseguram os dados com base em funções matemáticas que apenas com mais poder de processamento ou tempo podem ser quebradas.

Uma boa observação que a criptografia quântica só vai ser utilizada para produzir e distribuir as chaves, e não para transmitir a mensagem. A chave que é gerada pode ser utilizada com qualquer algoritmo de criptografia escolhido. O algoritmo mais usado associado com a criptografia quântica é o one-time pad, pois ele tem comprovação de uma segurança perfeita quando usado com qualquer chave escolhida e do mesmo tamanho que a mensagem. (RIEZNIK, A. A.; RIGOLIN,2005).

8. APLICAÇÃO DA CRIPTOGRAFIA ASSIMÉTRICA RSA

8.1 Criação da chave pública

1) Para a criação das chaves, é necessário que sejam escolhidos dois números aleatórios (**p** e **q**) e que sejam primos

$$\mathbf{p = 19 \text{ e } q = 43}$$

2) Fazer a multiplicação de **p** e **q**, encontrando o valor de **n**

$$\mathbf{n = p * q}$$

$$\mathbf{n = 19 * 43}$$

$$\mathbf{n = 817}$$

3) Utilizar a “função totiente de Euller” na variável **n**

$$\mathbf{\varphi(n) = \varphi(p) * \varphi(q)}$$

$$\mathbf{\varphi(n) = (p - 1) * (q - 1)}$$

$$\mathbf{\varphi(817) = (19 - 1) * (43 - 1)}$$

$$\mathbf{\varphi(817) = 756}$$

4) Encontrar um número (**e**) que seja maior que 1 e menor que o resultado obtido em **φ** (o número precisa ser primo também, chamado de co-primo neste caso)

$$\mathbf{1 < e < \varphi(n) = 1 < e < 756}$$

$$\mathbf{1 < 17 < 756}$$

Sendo assim, a chave pública, neste caso, é igual a **817** e **17**.

8.2 Tabela ASCII

A tabela ASCII (American Standard Code for Information Interchange) foi criada por Robert W. Bemer com o intuito de utilizar em todas as máquinas uma representação de números alfanuméricos. Antigamente, cada computador utilizava uma regra diferente para codificar, após essa criação, o sistema ASCII passou a ser utilizado em todos.

Para se cifrar a mensagem usando RSA, se é necessário criar uma tabela associando o alfabeto à números. É comum a utilização da tabela ASCII, porém é possível utilizar qualquer tabela que possua esse modelo (letras x números). (SUGAI,2015).

8.3 Cifrando com RSA

Para se cifrar a mensagem usando o RSA é preciso primeiro entender sobre a aritmética modular. Primeiramente, devemos lembrar, que o modulo é o resto de uma divisão inteira. Partindo desse princípio temos:

X:Y= D

X mod Y = R

Logo X = D*Y+R

Onde X é o divisor, Y o dividendo, D o resultado da divisão inteira e R o resto. Essa formula nos auxilia ao entender a divisão inteira de divisores menores que o dividendo por exemplo:

5:33=0

5 mod 33 =5

Logo 5=0*33+5

Para a cifragem usamos a nossa chave pública e nosso conjunto numérico, e a tabela ASCII ou qualquer outra feita por você. Vamos utilizar nesse exemplo a seguinte tabela:

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

12	13	14	15	16	17	18	19	20	21	22	23
M	N	O	P	Q	R	S	T	U	V	X	Z

A formula é a seguinte

c = n ^ p mod x

Onde C será nosso caractere criptografado, n o número correspondente a ele na tabela, p a nossa chave pública e x o tamanho do conjunto.

Vamos cifrar a palavra BOLO.

Usando os valores obtidos anteriormente temos então:

Para B

C= 2^17 mod 817 = 352

Para O

C= 14^17 mod 817 = 167

Para L

C= 11^17 mod 817 = 729

Para O

C= 14^17 mod 817 = 167

Temos então a palavra cifrada:

352 167 729 167

8.4 Criação da chave privada

Para se obter a chave privada, é preciso primeiro entender o conceito de inverso multiplicativo, e aplica-lo a nossa chave pública e no euller já obtidos. Ao multiplicar um número pelo seu inverso multiplicativo, obtém-se a identidade multiplicativa do seu respectivo conjunto. Ao trabalhar com o m.d.c. a identidade de nosso conjunto será sempre 1 pois o mínimo múltiplo comum entre números primos é 1.

Logo e * (e^⁻1) = 1

Para se obter o inverso multiplicativo modular, que é o usado no caso da RSA, utiliza-se o algoritmo de Euclides estendido da seguinte forma:

Dividiremos **e** por **φ(n)** utilizando os valores anteriores tem-se:

Div 1

17 : 756 = 0 com resto 17

Após isso dividimos o **divisor** anterior pelo resto até chegarmos em resto 1.

Div 2

756 : 17 = 44 com resto 8

Div 3

17 : 8 = 2 com resto 1

Agora seguimos para a parte estendida do algoritmo:

Isolamos o resto da divisão da seguinte forma:

X = D*Y+**R**

R = (1*X) - (D*Y)

Div 1

17= 0*756+17

17 = (1*17)-(0*756)

Div 2

756 = 44*17+8

8 = (1*756)-(44*17)

Na **Div 1** temos 17 = 1*17 logo usaremos essa **Div** como divisor da **Div 2,** substituindo na equação temos:

8= (1*756) - 44* (**(1*17) - (0*756)**)

Agora distribuímos a multiplicação:

8= (1*756) – (44*17) – (0*756)

Somamos os múltiplos comuns:

(**Res 1)**

8= (1*756)-(44*17)

Agora isolamos o resto da **Div 3:**

17 = 2*8+1

1= (1*17)-(2*8)

Como em **res 1** temos que 8 = (1*756) - (44-17), utilizamos essa expressão como divisor da nossa **div 3:**

1= (1*17) – 2* ((1*756) – (44*17))

Distribuímos a multiplicação:

1= (1*17) – (2*756) - (88*17)

Agora somamos os múltiplos comuns:

1 = (**89***17)-(2*756)

Pronto o inverso multiplicativo de nosso e(17) é 89 agora só decifrar nossa mensagem. Vale lembrar que no algoritmo para computador, usamos uma função while para satisfazer a equação atribuindo ++ a chave pública até obtermos o valor da mesma, tornando mais fácil o processo acima.

8.5 Decifrando em RSA

Agora que temos nossa chave privada em mãos, basta apenas repetir nossa exponenciação modular, usando a chave privada dessa forma:

m = c^k mod n

m = c^89 mod 817

Utilizando a mensagem criptografada antes:

352 167 729 167

Para 352:

m= 352 ^ 89 mod 817

m= 2

Para 167:

m= 167 ^ 89 mod 817

m= 14

Para 729:

m= 729 ^ 89 mod 817

m= 11

Para 167:

m= 167 ^ 89 mod 817

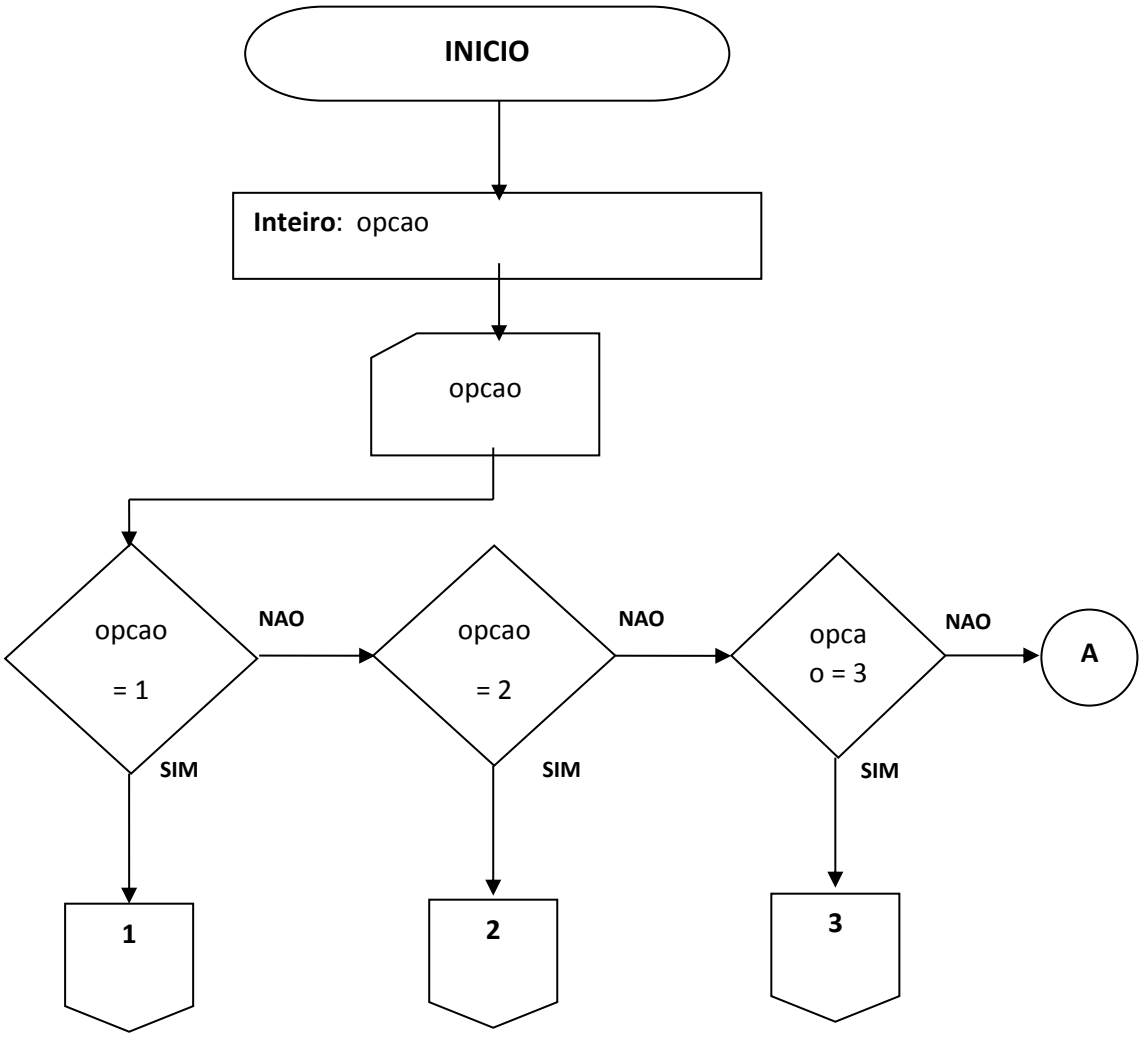
m= 14

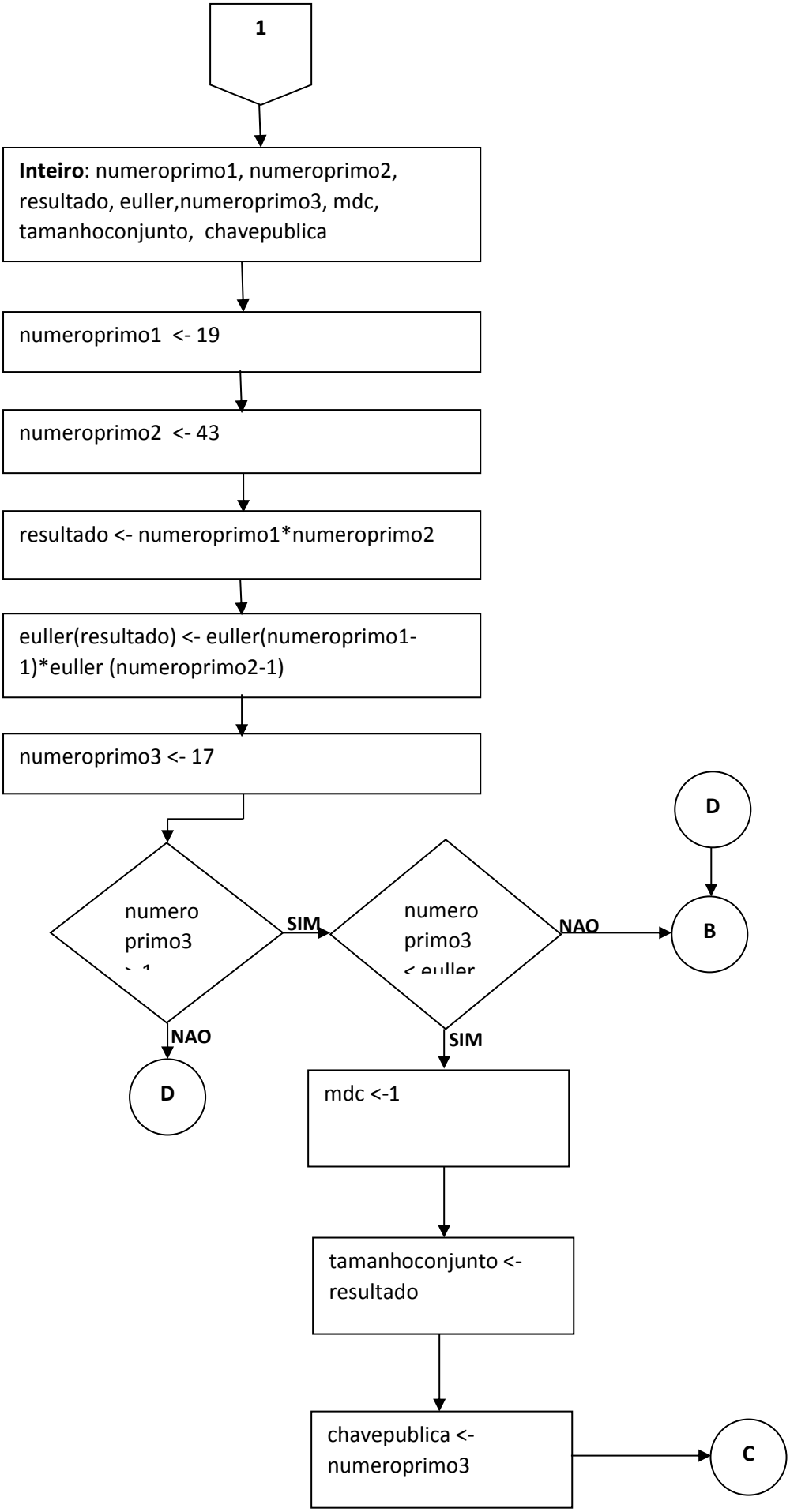
Temos agora a mensagem decifrada

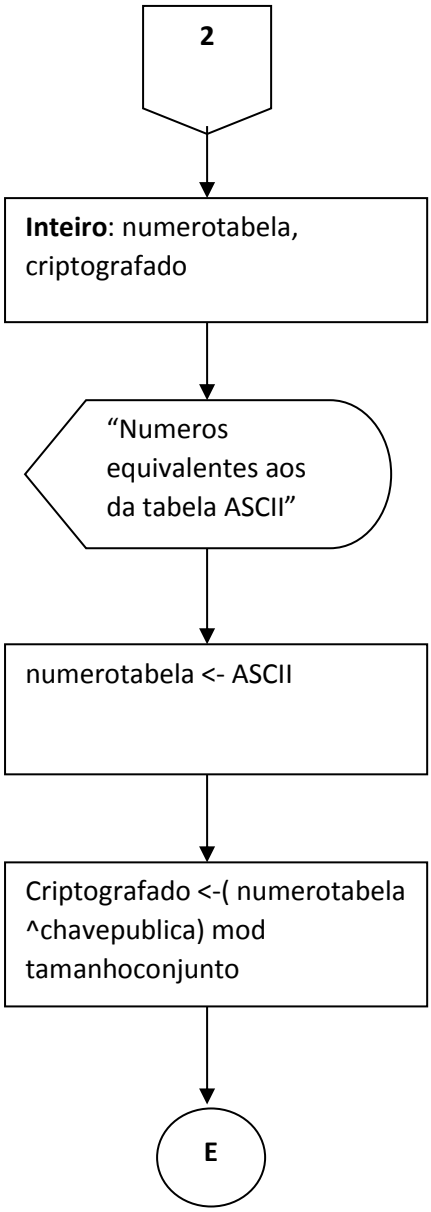
2 14 11 14

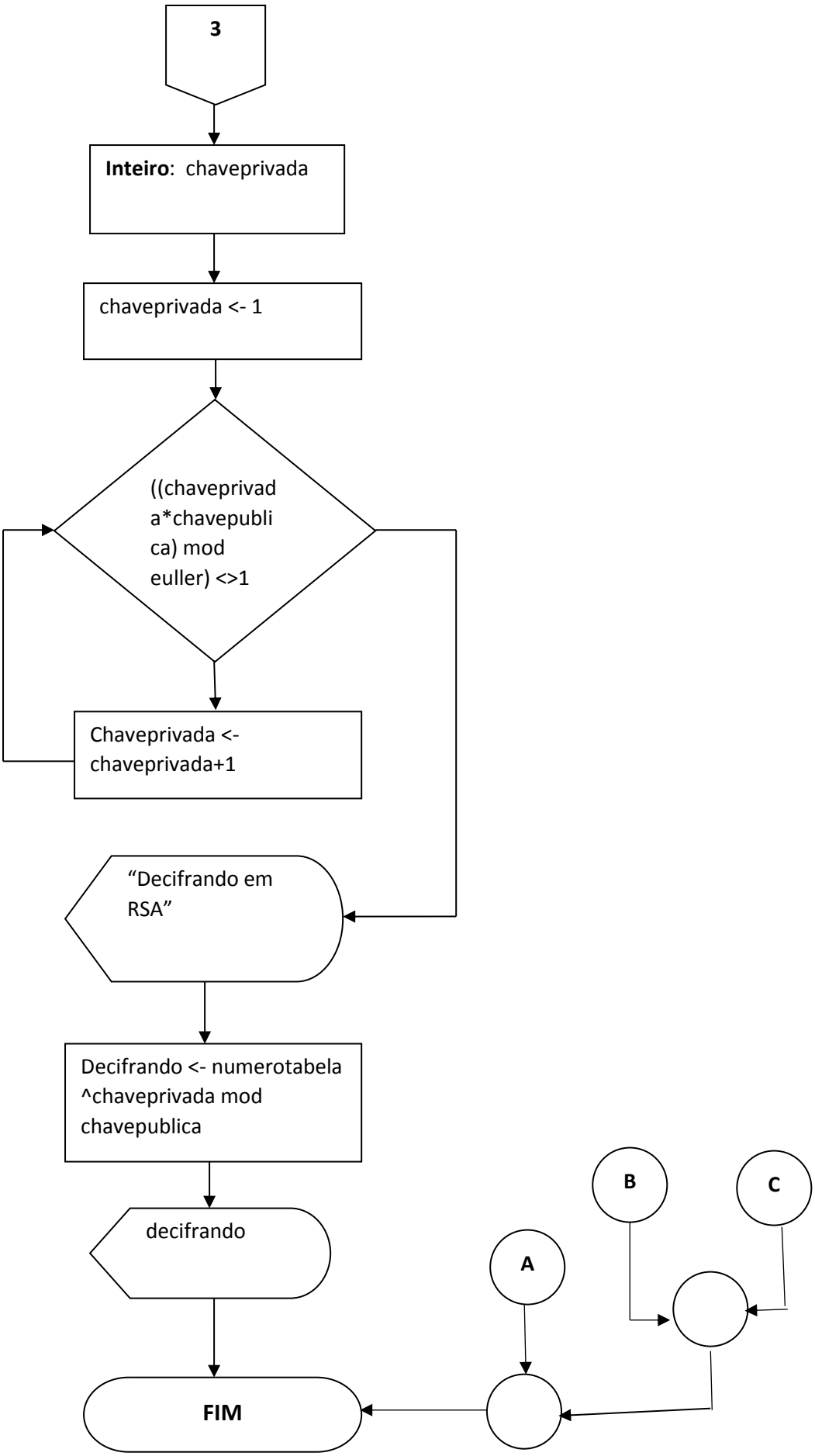
Utilizando a tabela montada acima temos BOLO.

9. FLUXOGRAMA




1
Inteiro: numeroprimo1, numeroprimo2, resultado, euller,numeroprimo3, mdc, tamanhoconjunto, chavepublica
numeroprimo1 <- 19
numeroprimo2 <- 43
resultado <- numeroprimo1*numeroprimo2
euller(resultado) <- euller(numeroprimo1-1)*euller (numeroprimo2-1)
numeroprimo3 <- 17
numero primo3
SIM
numero primo3
NAO
D
B
NAO
SIM
D
mdc <-1
tamanhoconjunto <- resultado
chavepublica <- numeroprimo3
C


2
Inteiro: numerotabela, criptografado
"Numeros equivalentes aos da tabela ASCII"
numerotabela <- ASCII
Criptografado <-(numerotabela ^chavepublica) mod tamanhoconjunto
E


3
Inteiro: chaveprivada
chaveprivada <- 1
((chaveprivad a*chavepubli ca) mod euller) <>1
Chaveprivada <- chaveprivada+1
“Decifrando em RSA”
Decifrando <- numerotabela ^chaveprivada mod chavepublica
decifrando
FIM
A
B
C

10. CÓDIGOS DO PROGRAMA DE CRIPTOGRAFIA RSA EM C

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <math.h>
#include <time.h>
// função para calcular pontencia e modulo do big number.(a ^ e % n).
long calcular(long long a, long long e, long long n){
    long long A = a, P = 1, E = e;
    while(1){
        if(E == 0)
            return P;
        else if(E%2 != 0){
            P = (A * P)%n;
            E =
             (E-1)/2;
        }
        else{
            E = E/2;
        }
        A = (A*A)%n;
    }
}


// função para sorteio.
    int sorteioDePrimo(int limite){
        // declaração de variavel com valores primos para sortear.
        int primos[172] =
{2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,1
31,137,139,149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,233,239,241,251,25
7,263,269,271,277,281,283,293,307,311,313,317,331,337,347,349,353,359,367,373,379,383,389,397
,401,409,419,421,431,433,439,443,449,457,461,463,467,479,487,491,499,503,509,521,523,541,547,
557,563,569,571,577,587,593,599,601,607,613,617,619,631,641,643,647,653,659,661,673,677,683,6
91,701,709,719,727,733,739,743,751,757,761,769,773,787,797,809,811,821,823,827,829,839,853,85
7,859,863,877,881,883,887,907,911,919,929,937,941,947,953,967,971,977,983,991,997,1009,1013,1
019,1021};
        // processamento do sorteio.
        int auxiliar = (rand() % 172);
```

```
        int numeroPrimo = primos[auxiliar];

        while(numeroPrimo >= limite) {
            auxiliar = (rand() % 172);
            numeroPrimo = primos[auxiliar];
        }
        return numeroPrimo;
            }
void criarChaves(){
    // declaração de variaveis.
    int chavePrivada, chavePublica, tamanhoDoConjunto, euller, contador;
    int numeroPrimoUm, numeroPrimoDois;

    // processamento de criação das chaves.
            numeroPrimoUm = sorteioDePrimo(1024);
            numeroPrimoDois = sorteioDePrimo(1024);
            tamanhoDoConjunto = numeroPrimoUm * numeroPrimoDois;
            euller=(numeroPrimoUm -1)*(numeroPrimoDois-1);
            chavePublica = sorteioDePrimo(euller);
            chavePrivada=1;
            while(((chavePrivada*chavePublica)%euller)!=1){
                    chavePrivada++;
            }
            printf("Tamanho do conjunto: %d.\n", tamanhoDoConjunto);
            printf("Chave Pública: %d.\n", chavePublica);
            printf("Chave Privada: %d.\n", chavePrivada);
}
// função para descriptografar e criptografar
void criptografar(int op) {
    // declaração de variáveis.
    int chave, tamanhoDoConjunto, retornoMensagem[1024];
    int contador;
    char mensagem[1024];

    // processamento da entrada de chaves para opcão escolhida.
    printf("Digite a chave pública: ");
```

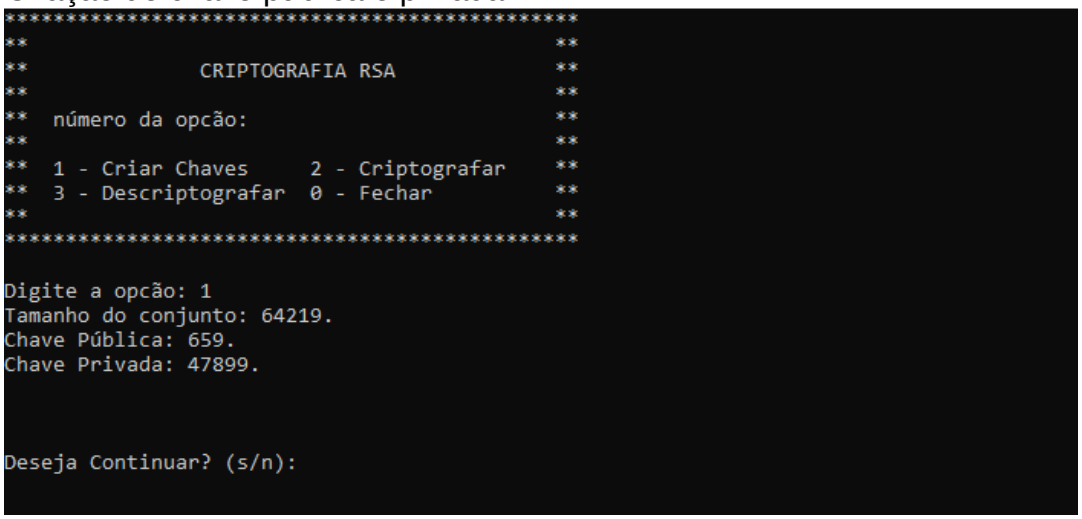
```
    scanf("%d", &chave);
    printf("Digite o tamanho do conjunto: ");
    scanf("%d", &tamanhoDoConjunto);
    printf("\n");

    // entrada da mensagem.
    printf("Digite a mensagem(max 1024 caracteres): \n\n");
    printf("---> ");
    scanf( "\n");
    fgets(mensagem, 1024, stdin);

    // processamento e saída da encriptação ou decriptação da mensagem.
    printf("\n\n");
    printf("Mensagem com a chave %d e %d!\n", chave, tamanhoDoConjunto);
    printf("---> ");
    for(contador = 0; mensagem[contador] != '\0'; contador++) {
            retornoMensagem[contador] = calcular(mensagem[contador], chave, tamanhoDoConjunto);
            printf("%d&", retornoMensagem[contador]);
    }
}
void decriptar(int op) {
    // declaração de variáveis.
    int chave, tamanhoDoConjunto, retornoMensagem[4096];
    int contador;
    int contadorRetorno = 0;
    char mensagem[4096];

    // processamento da entrada de chaves para opcão escolhida.
    printf("Digite a chave privada: ");
    scanf("%d", &chave);
    printf("Digite o tamanho do conjunto: ");
    scanf("%d", &tamanhoDoConjunto);
    printf("\n");
```

```
    // entrada da mensagem.
    printf("Digite a mensagem encripritada(max 4096 caracteres): \n\n");
    printf("---> ");
    scanf( "\n");
    fgets(mensagem, 4096, stdin);
    // processamento e saída da encriptação ou decriptação da mensagem.
    printf("\n\n");
    printf("Mensagem com a chave %d e %d!\n", chave, tamanhoDoConjunto);
    printf("---> ");
    int auxiliar = 0;
    for(contador = 0; mensagem[contador] != '\0'; contador++) {
        if(mensagem[contador] == '&') {
            retornoMensagem[contador] = calcular(auxiliar, chave, tamanhoDoConjunto);
            auxiliar = 0;
            printf("%c", retornoMensagem[contador]);
        } else if(auxiliar > 0) {
            auxiliar = (auxiliar * 10) + (mensagem[contador] - 48); // Na tabela ASCII números começam
apartir do 48.
        } else {
            auxiliar = (mensagem[contador] - 48); // Na tabela ASCII números começam apartir do 48.
        }
    }
}
int main(int argc, char* argv[]) {
    setlocale(LC_ALL,"portuguese");
    srand((unsigned)time(NULL));

    int opcao=5;
    char continuar;

    while(opcao != 0){
                // cabeçalho.
                printf("*********************************************\n");
                printf("**                                         **\n");
                printf("**            CRIPTOGRAFIA RSA             **\n");
                printf("**                                         **\n");
```

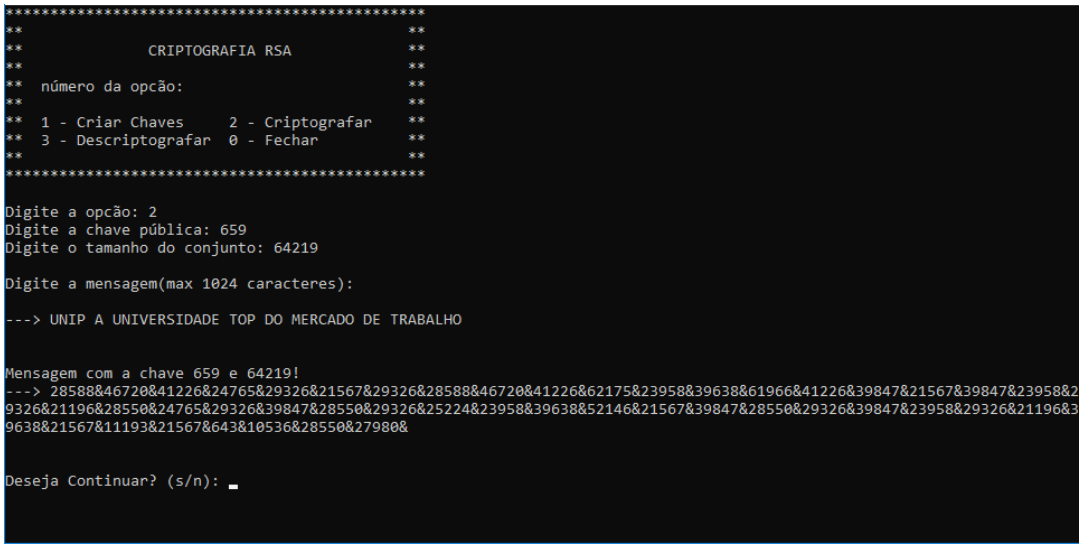
```
            printf("**  número da opcão:                    **\n");
            printf("**                                      **\n");
            printf("**  1 - Criar Chaves     2 - Criptografar    **\n");
            printf("**  3 - Descriptografar  0 - Fechar          **\n");
            printf("**                                      **\n");
            printf("**********************************************\n\n");
            printf("Digite a opcão: ");
            scanf("%d", &opcao);
            // valida opcão.
               while(opcao < 0 || opcao > 3) {
               printf("Digite uma opcão válida!\n");
               printf("Digite a opcão: ");
               scanf("%d", &opcao);
            }
            // encaminha entrada de opcão.
            if(opcao == 1) {
               criarChaves();
               scanf("%c", &continuar);
            }else if(opcao == 2) {
               criptografar(opcao);
            }else if(opcao == 3) {
               decriptar(opcao);
            }else if(opcao == 0){
                  break;
                  }
            printf("\n\n\nDeseja Continuar? (s/n): ");
            scanf("%c",&continuar);
            if((continuar == 's') || (continuar == 'S')){
                  system("cls");
                  }else{
                        opcao = 0;
            }
         }

   return 0;
}
```

11. PRINT DAS TELAS

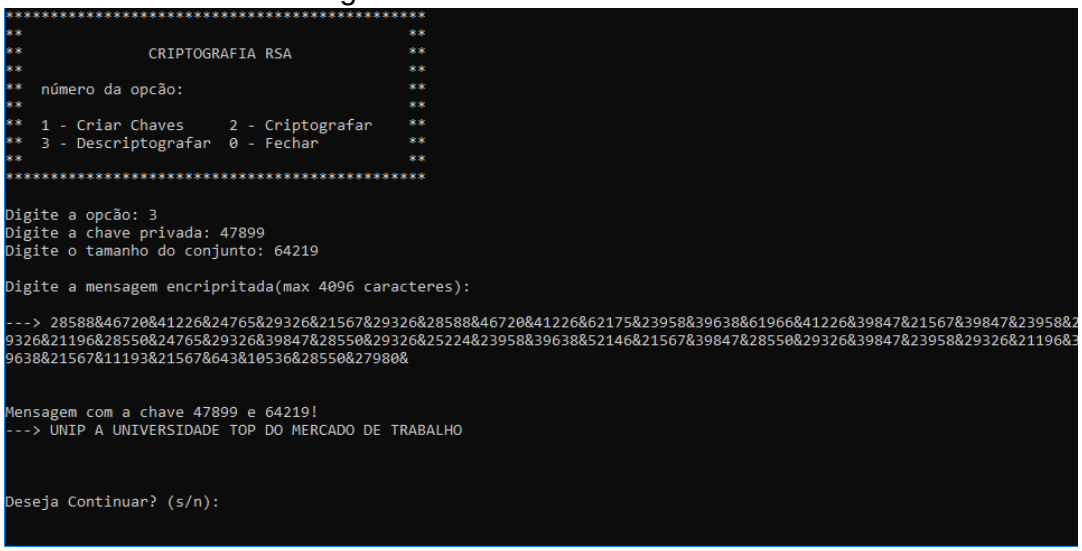
Criação de chave pública e privada:



Criptografando a mensagem:



Decodificando a mensagem:



BIBLIOGRAFIA

B.SVAITE, D. RC4 - adeus ao favorito do TLS e outros. **Cifra Extrema**, 21 Setembro 2016. Disponivel em: <http://www.cifraextrema.com/single-post/2016/09/21/RC4---adeus-ao-favorito-do-TLS-e-outros>. Acesso em: 15 out. 2018.

OLIVEIRA, F. Entendendo (de verdade) a criptografia RSA. **LAMBDA3**, 10 Dezembro 2012. Disponivel em: <https://www.lambda3.com.br/2012/12/entendendo-de-verdade-a-criptografia-rsa/>. Acesso em: 16 out. 2018.

PAIM, R. R. WEP, WPA e EAP. **Gta UFRJ**. Disponivel em: <https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2011_2/rodrigo_paim/downloads/trabalho.pdf>. Acesso em: 15 out. 2018.

RIEZNIK, A. A.; RIGOLIN,. Introducao a Criptografia Quantica. **Scielo**, p. 517 - 526, 2005. Disponivel em: <http://www.scielo.br/pdf/%0D/rbef/v27n4/a04v27n4.pdf>. Acesso em: 14 out. 2018.

SALAS, M. P. Análise Crítica da Implementação da Cifra RC4 no Protocolo WEP. **ResearchGate**, Setembro 2011. Disponivel em: <http://www.students.ic.unicamp.br/~ra089053/pdf/1WEP.pdf>. Acesso em: 15 out. 2018.

SUGAI, A. O que é o código ASCII e pra que serve? **TechTudo**, 15 Fevereiro 2015. Disponivel em: <https://www.techtudo.com.br/noticias/noticia/2015/02/o-que-e-o-codigo-ascii-e-para-que-serve-descubra.html>. Acesso em: 16 out. 2018.