Segurança da Informação. Quarta - Manhã. 1001.
Integrantes: Leandro Nascimento de Paula (202304190615), Lucas Vinicius Silveira (202402390562) e Luis Gustavo Lobato De Melo (202404346285).

Algoritmos de Criptografia: Implementação e Análise

De acordo com as orientações repassadas pelo professor, foram feitas pesquisas acerca do tema de criptografia a fim de melhor compreender o assunto e, ao final, se realizou a implementação de um algoritmo funcional de criptografia, juntamente com sua análise e testes.

CONCEITOS BÁSICOS

Criptografia

É o processo de transformação de dados em um formato ilegível chamado texto cifrado, para garantir confidencialidade e segurança durante a transmissão ou armazenamento de dados.

Texto Claro

É o texto original que precisa ser protegido.

Texto Cifrado

É o resultado do processo de criptografia, o texto transformado que não é facilmente legível sem a chave de decodificação.

Chave Criptográfica

É um valor usado pelos algoritmos de criptografia para criptografar e/ou descriptografar os dados. A segurança da criptografia geralmente depende da segurança desta chave.

APLICAÇÕES

A criptografia no ambiente digital pode ser aplicada de diversas formas, sendo a mais popularizada delas nas trocas de mensagens entre usuários em aplicativos de conversação. Além disso, ela também é aplicada no armazenamento de arquivos, transações eletrônicas, assinaturas de contratos, entre outros. Ou seja, onde há troca de dados entre usuários ou necessidade de proteger um item de acesso por cibercriminosos, ela pode ser adotada. Exemplos**:** Segurança de Dados em Comunicações, Dados em Armazenamento, Sistemas de Pagamento, Sistemas de Arquivos e Autenticação e Assinaturas Digitais.

TIPOS DE CRIPTOGRAFIA

Criptografia Simétrica

Utiliza a mesma chave para criptografar e descriptografar os dados. Exemplos incluem AES (Advanced Encryption Standard), DES (Data Encryption Standard), e 3DES.

Criptografia Assimétrica (ou Criptografia de Chave Pública)

Usa um par de chaves, uma pública e uma privada. A chave pública é compartilhada para criptografar os dados, enquanto a chave privada é mantida em segredo e usada para descriptografar. Exemplos incluem RSA, ElGamal e ECC (Elliptic Curve Cryptography)

Hashing

Não é estritamente criptografia, mas é usado para garantir a integridade dos dados. Transforma dados em um valor hash fixo e único. Exemplos incluem SHA (Secure Hash Algorithm) e MD5.

EXEMPLOS DE ALGORITMO DE CRIPTOGRAFIA

Cifra de Substituição Simples

Este método substitui cada letra do texto original por outra letra do alfabeto, de acordo com um padrão predefinido. Por exemplo, a letra "A" pode ser substituída por "Q", "B" por "Z" e assim por diante. A chave neste caso é a tabela de correspondência entre as letras originais e suas substitutas. A exemplo disso, a cifra de César é um algoritmo onde as letras são deslocadas para esquerda por 3 unidades das letras do alfabeto, para decifrar, basta aplicar o deslocamento inverso.

Cifra de Vigenère

A cifra de Vigenère é uma extensão da cifra de César, mas em vez de usar um único deslocamento para todas as letras, utiliza uma palavra-chave. Cada letra da mensagem é deslocada de acordo com a letra correspondente na palavra-chave. Se a palavra-chave for mais curta do que a mensagem, ela é repetida. Isso torna a cifra de Vigenère mais resistente à análise de frequência do que a cifra de César simples.

RSA (Rivest-Shamir-Adleman)

RSA é um algoritmo de criptografia assimétrica amplamente utilizado para comunicação segura na Internet. Ele envolve o uso de duas chaves: uma pública e uma privada. A chave pública é usada para criptografar a mensagem, enquanto a chave privada é usada para decifrá-la. A segurança do RSA baseia-se na dificuldade de fatorar números primos muito grandes

AES (Advanced Encryption Standard)

AES é um algoritmo de criptografia simétrica que substituiu o antigo padrão DES (Data Encryption Standard). Ele opera em blocos de dados de 128 bits e suporta chaves de 128, 192 e 256 bits. AES é altamente eficiente e amplamente utilizado em aplicativos que exigem alto desempenho de criptografia, como segurança de rede e proteção de dados.

IMPLEMENTAÇÃO DE ALGORITMO DE CRIPTOGRAFIA E DESCRIPTOGRAFIA

Após revisar os conceitos pesquisados, a equipe optou pela implementação de um algoritmo de substituição similar à Criptografia de César em Python, porém com uma chave (deslocação de caractere) escolhida pelo usuário no intervalo de 1 à 25, com a diferença de que o código aplicado diferencia letras maiúsculas e minúsculas, e também números inteiros, cada um em suas tabelas. As limitações pré definidas dessa implementação impossibilitam o uso de caracteres especiais, logo, até mesmo palavras acentuadas são proibidas pelo sistema.

```python
import string

criptografado = ""
descriptografado = ""

código = (input("Digite uma frase sem caracteres especiais ou acentos para ser criptografada. Você pode usar letras e números:\n"))

while any((i in string.punctuation or ord(i) > 192) for i in código) or not código.strip(): # Check
  código = input("Digite uma frase válida para ser criptografada, sem caractere especial ou acentos:\n")

shiftKey = input("Insira um número inteiro de 1 à 25, sem espaços:\n") # Key

while not shiftKey.isnumeric() or int(shiftKey) < 1 or int(shiftKey) > 25: # Check
  shiftKey = input("Insira um número inteiro válido de 1 à 25, sem espaços:\n")

shiftPrime = int(shiftKey)
shiftLetra = shiftPrime
shiftNumeral = shiftPrime % 9 if shiftPrime > 9 else shiftPrime
```

```
Digite uma frase sem caracteres especiais ou acentos para ser criptografada. Você pode usar letras e números:
Implementação 1 - Leandro
Digite uma frase válida para ser criptografada, sem caractere especial ou acentos:
Implementacao 1 Leandro
Insira um número inteiro de 1 à 25, sem espaços:
26
Insira um número inteiro válido de 1 à 25, sem espaços:
25
```

Nessa primeira parte do programa são declaradas as variáveis de string vazias “criptografado” e “descriptografado”. Após isso, o “código” que será criptografado é pedido como input de usuário, mais abaixo, é feito uma checagem desse input, averiguando se ele possui caracteres especiais, acentos ou é apenas uma string vazia, determinando se o usuário errou as exigências. Da mesma forma, logo abaixo é feito o input da key de criptografia, aqui chamada de “shiftKey” e, assim como feita com o “código”, também é feita uma checagem, porém aqui é apenas analisado se o usuário digitou um número inteiro e se esse número está dentro do intervalo de 1 à 25. Por fim, a variável da key é atribuída como inteiro à outra variável “shiftPrime”, e a partir dessa nova variável é que obtemos a deslocação de letra e número em “shiftLetra” e “shiftNumeral”.

Nesse detalhe, a deslocação alfabética é a mesma que o usuário digita como key, entretanto, a variação numeral segue outra lógica, pois, se tratando de um sistema de computador, todos os caracteres e suas tabelas são baseados na tabela geral ASCII, então a key escolhida pelo usuário corre o risco de ir além do endereço estabelecido para numerais. Portanto, foi feito um esquema de repetição onde todo número-key que exceda o intervalo de 0 à 9 (ex: 10) volta a repetir a partir da posição de 0, com o seu valor restante percorrendo o resto do intervalo de posições estabelecido (Note que aqui o “0” é tratado como posição 1 na tabela, fazendo com que ela tenha, efetivamente, 10 posições ao todo). Contudo, da forma aplicada, caso o valor retorne para o mesmo caractere numérico do “código” a ser criptografado, ele irá pular para o próximo número, a fim de evitar que não seja criptografado.

```
Digite uma frase sem caracteres especiais ou acentos para ser criptografada. Você pode usar letras e números:
9
Insira um número inteiro de 1 à 25, sem espaços:
10
Seu código criptografado: 9
Seu código descriptografado: 9
```

(O que aconteceria se o valor não pulasse o caractere original. É como se não houvesse criptografia.)

```
Digite uma frase sem caracteres especiais ou acentos para ser criptografada. Você pode usar letras e números:
9
Insira um número inteiro de 1 à 25, sem espaços:
10
Seu código criptografado: 0
Seu código descriptografado: 9
```

(Como foi implementado para evitar burlas na criptografia.)

Ou seja, um código “9”, quando usado da key de deslocação de valor 10, iria ser criptografado como “9”, já que 10 excederia o intervalo, retornaria a posição 0 e

seu valor restante chegaria ao 9. Da forma implementada, um código “9” que utiliza da key 10 irá retornar a posição 0 e, quando chegar ao 9, irá pular o caractere, tendo como resultado 0.

A lógica explicada foi apenas para definir a key de deslocamento, o programa continua mais abaixo.

```
##CRIPTOGRAFIA
for i in range(len(código)):
  shiftLoop = shiftLetra if código[i].isalpha() else shiftNumeral

  #1. Caractere de letra maiúscula ou minúscula que foge da tabela
  if ord(código[i]) + shiftLoop > 90 and código[i].isupper():
    criptografado += chr(ord(código[i]) + shiftLoop - 26)
  elif ord(código[i]) + shiftLoop > 122 and código[i].islower():
    criptografado += chr(ord(código[i]) + shiftLoop - 26)

  #2. Caractere de número que foge da tabela
  elif ord(código[i]) + shiftLoop > 57 and código[i].isnumeric():
    criptografado += chr(ord(código[i]) + shiftLoop - 10)

  #3. Espaços em branco
  elif código[i].isalnum() == False:
    criptografado += código[i]

  #4. Caractere de letra ou número que não foge da tabela
  else:
    criptografado += chr(ord(código[i]) + shiftLoop)

print(f"Seu código criptografado: {criptografado}")
```

A lógica da criptografia é iniciada dentro de um loop for que toma como range o conteúdo do “código”. Dentro do loop, é criada uma variável que é usada unicamente dentro dessa estrutura; “shiftLoop” toma o valor das outras variáveis shift, dependendo de qual caractere está sendo criptografado na vez. Mais abaixo, há a primeira condicional de substituição, sua lógica de checagem e resultante é adotada em praticamente todas as outras condicionais de criptografia e descriptografia desse programa, com pequenas variações.

A estrutura segue o padrão: verifica se o endereço do caractere analisado somado ao valor de “shiftLoop” vai além do que é delimitado para sua categoria na tabela ASCII e a qual delas ele pertence. Se a condicional for verdadeira, o caractere analisado se desloca para a posição correspondente à soma do valor de

“shiftLoop” menos o número necessário para que retorne ao início da sua tabela e é adicionado na variável “criptografado”.

ASCII control characters		
00	NULL	(Null character)
01	SOH	(Start of Header)
02	STX	(Start of Text)
03	ETX	(End of Text)
04	EOT	(End of Trans.)
05	ENQ	(Enquiry)
06	ACK	(Acknowledgement)
07	BEL	(Bell)
08	BS	(Backspace)
09	HT	(Horizontal Tab)
10	LF	(Line feed)
11	VT	(Vertical Tab)
12	FF	(Form feed)
13	CR	(Carriage return)
14	SO	(Shift Out)
15	SI	(Shift In)
16	DLE	(Data link escape)
17	DC1	(Device control 1)
18	DC2	(Device control 2)
19	DC3	(Device control 3)
20	DC4	(Device control 4)
21	NAK	(Negative acknowl.)
22	SYN	(Synchronous idle)
23	ETB	(End of trans. block)
24	CAN	(Cancel)
25	EM	(End of medium)
26	SUB	(Substitute)
27	ESC	(Escape)
28	FS	(File separator)
29	GS	(Group separator)
30	RS	(Record separator)
31	US	(Unit separator)
127	DEL	(Delete)

ASCII printable characters					
32	space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

(Tabela ASCII e a posição de cada caractere)

A lógica por trás da soma e subtração é praticamente a mesma vista na definição de “shiftNumeral”, onde, caso um valor vá além do intervalo delimitado, ele retorna a posição inicial. Nesse sentido, o último dos caracteres numéricos e alfabéticos maiúsculos e minúsculos está, respectivamente, nas posições 57, 90 e 122 da tabela.

O próximo bloco de condicional apenas checa se o caractere é alfanumérico. Se não for, ele apenas adiciona à variável “criptografado". Por fim, caso nenhuma das condicionais anteriores forem verdadeiras, o programa apenas desloca o caractere no valor de “shiftLoop” e o adiciona à variável “criptografado”. Após isso, há o print da variável “criptografado”, mostrando como ficou a frase criptografada pelo usuário.

Como dito anteriormente, as mesmas lógicas de condicionais se repetem de forma similar na parte de descriptografia. Logo, a explicação se delimita apenas em evidenciar as diferenças.

```
##DESCRIPTOGRAFIA
for i in range(len(criptografado)):
  shiftLoop = shiftLetra if criptografado[i].isalpha() else shiftNumeral

  #1. Caractere de letra maiúscula ou minúscula que foge da tabela
  if ord(criptografado[i]) - shiftLoop < 65 and criptografado[i].isupper():
    descriptografado += chr(ord(criptografado[i]) - shiftLoop + 26)
  elif ord(criptografado[i]) - shiftLoop < 97 and criptografado[i].islower():
    descriptografado += chr(ord(criptografado[i]) - shiftLoop + 26)

  #2. Caractere de número que foge da tabela
  elif ord(criptografado[i]) - shiftLoop < 48 and criptografado[i].isnumeric():
    descriptografado += chr(ord(criptografado[i]) - shiftLoop + 10)

  #3. Espaços em branco
  elif criptografado[i].isalnum() == False:
    descriptografado += criptografado[i]

  #4. Caractere de letra ou número que não foge da tabela
  else:
    descriptografado += chr(ord(criptografado[i]) - shiftLoop)

print(f"Seu código descriptografado: {descriptografado}")
```

Mais uma vez o código inicia a partir de um loop for ao range do conteúdo de uma variável, nesse caso, “criptografado”. Além disso, a mesma variável “shiftLoop” se mantém, juntamente com sua lógica.

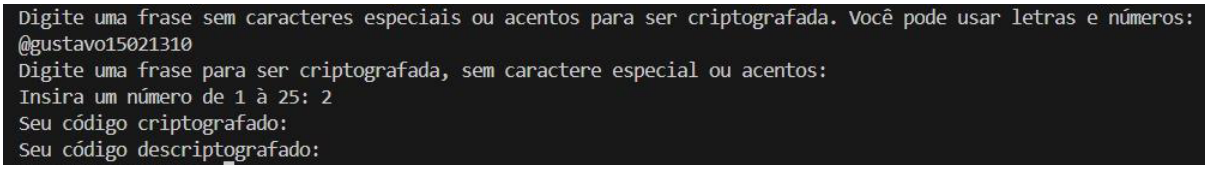
No primeiro e segundo dos blocos são feitas checagens inversamente proporcionais àquelas dos mesmos na criptografia, contudo, a parte de checagem das tipagens se mantém. Ou seja, é verificado se o endereço do caractere subtraído ao valor de “shiftLoop” é aquém do intervalo estabelecido e qual tipo de caractere é analisado. Se verdadeiro, o caractere se desloca para posição obtida através da subtração por “shiftLoop” e soma do valor que o fará retornar à tabela, assim, sendo adicionado à variável “descriptografado”.

O bloco 3 não se altera, apenas adicionando à variável “descriptografado”, e, semelhantemente ao parágrafo anterior, o bloco 4 apenas adiciona à variável o caracteres deslocado do valor subtraído de “shiftLoop”. Ao fim, é exibido para o usuário seu código descriptografado.

```
Digite uma frase sem caracteres especiais ou acentos para ser criptografada. Você pode usar letras e números:
Implementacao em python
Insira um número inteiro de 1 à 25, sem espaços:
14
Seu código criptografado: Wadzsasbhoqoc sa dmhvcb
Seu código descriptografado: Implementacao em python
```

ANÁLISE DE ERROS E SEGURANÇA

Não foram encontrados erros substanciais nos testes de segurança. As limitações do programa quanto a certos tipos de caracteres também foram testadas, mas nada fora do estabelecido foi encontrado, visto que o programa faz checagem. Contudo, um integrante da equipe se deparou com um erro incomum e que não se conseguiu replicar em nenhum outro momento, onde foi aceito o uso de caractere especial e não houve a criptografia do código. Devido a falha nas tentativas de replicar o erro, esse caso foi julgado como apenas uma anormalidade singular do interpretador de Python, uma vez que o programa se comportou da maneira estabelecida todas as outras vezes em que se tentou usar o mesmo input ou similar.



BIBLIOGRAFIA

- Conteúdo Digital SAVA
- Stallings, W. (2015). *Criptografia e segurança de redes: princípios e práticas*. (6a ed.) Pearson. https://plataforma.bvirtual.com.br

CÓDIGO

```
import string

criptografado = ""
descriptografado = ""

código = (input("Digite uma frase sem caracteres especiais ou acentos para ser
criptografada. Você pode usar letras e números:\n")) # O que será criptografado

while any((i in string.punctuation or ord(i) > 192) for i in código) or not
código.strip(): # Check
  código = input("Digite uma frase válida para ser criptografada, sem caractere
especial ou acentos:\n")

shiftKey = input("Insira um número inteiro de 1 à 25, sem espaços:\n") # Key

while not shiftKey.isnumeric() or int(shiftKey) < 1 or int(shiftKey) > 25: #
Check
  shiftKey = input("Insira um número inteiro válido de 1 à 25, sem espaços:\n")

shiftPrime = int(shiftKey)
shiftLetra = shiftPrime
```

```
shiftNumeral = shiftPrime % 9 if shiftPrime > 9 else shiftPrime
# A cada iteração, o módulo % retorna qualquer valor abaixo de 9; acima, ele
retorna restos a partir de 0. Evita a repetição do mesmo número.

##CRIPTOGRAFIA
for i in range(len(código)):
  shiftLoop = shiftLetra if código[i].isalpha() else shiftNumeral

  #1. Caractere de letra maiúscula ou minúscula que foge da tabela
  if ord(código[i]) + shiftLoop > 90 and código[i].isupper():
          criptografado += chr(ord(código[i]) + shiftLoop - 26)
  elif ord(código[i]) + shiftLoop > 122 and código[i].islower():
          criptografado += chr(ord(código[i]) + shiftLoop - 26)

  #2. Caractere de número que foge da tabela
  elif ord(código[i]) + shiftLoop > 57 and código[i].isnumeric():
          criptografado += chr(ord(código[i]) + shiftLoop - 10)

  #3. Espaços em branco
  elif código[i].isalnum() == False:
          criptografado += código[i]

  #4. Caractere de letra ou número que não foge da tabela
  else:
          criptografado += chr(ord(código[i]) + shiftLoop)

print(f"Seu código criptografado: {criptografado}")

##DESCRIPTOGRAFIA
for i in range(len(criptografado)):
  shiftLoop = shiftLetra if criptografado[i].isalpha() else shiftNumeral

  #1. Caractere de letra maiúscula ou minúscula que foge da tabela
  if ord(criptografado[i]) - shiftLoop < 65 and criptografado[i].isupper():
          descriptografado += chr(ord(criptografado[i]) - shiftLoop + 26)
  elif ord(criptografado[i]) - shiftLoop < 97 and criptografado[i].islower():
          descriptografado += chr(ord(criptografado[i]) - shiftLoop + 26)

  #2. Caractere de número que foge da tabela
  elif ord(criptografado[i]) - shiftLoop < 48 and criptografado[i].isnumeric():
          descriptografado += chr(ord(criptografado[i]) - shiftLoop + 10)

  #3. Espaços em branco
  elif criptografado[i].isalnum() == False:
          descriptografado += criptografado[i]

  #4. Caractere de letra ou número que não foge da tabela
  else:
          descriptografado += chr(ord(criptografado[i]) - shiftLoop)

print(f"Seu código descriptografado: {descriptografado}")
```