Números Inteiros e Criptografia, 2020.2

Trabalho Final[†]

Submeta as soluções de todas as questões do trabalho até 10 de junho às 9:00 salvando os arquivos na sua pasta chamada "Trabalho Final" no Google Drive

Em todas as questões que envolverem codificação (incluindo a sua implementação do RSA), usaremos a tabela de correspondência entre números e símbolos dada na última página deste PDF.

Lembre-se: você pode usar tudo o que foi visto em aula, em listas anteriores, ou mesmo qualquer questão do trabalho para responder outras questões (mesmo que você não tenha feito a questão que está citando!), desde que você seja claro na sua citação do que está usando. A única exceção é a Questão 6, na qual você não pode usar o item **b** na solução do item **a**, a não ser que você faça o item **b**.

Como sempre, justifique todas as questões.

Questão 1. Prove que a equação $x^{41} + 81x + 41y^{15} = 197$ não possui soluções com $x, y \in \mathbb{Z}$. (*Dica*: Suponha, para uma prova por contradição, que a equação tenha uma solução com x, y inteiros. Encontre um módulo p primo conveniente e reduza ambos os lados da equação (mod p), encontrando uma situação impossível.)

Questão 2.

a. Sejam $n \geq 1$ um natural, $\ell_0, \ell_1, \ldots, \ell_{n-1} \in \mathbb{N}$ e $x, y \in \mathbb{Z}$.
Prove que se

$$x \equiv y \pmod{\ell_0 \cdot \ell_1 \cdots \ell_{n-1}}$$

então

$$\text{para todo } i < n \text{ temos } \quad x \equiv y \pmod{\ell_i}.$$

(*Dica*: Lembre-se de que a definição de $x \equiv y \pmod{z}$ fala sobre *divisibilidade*.)

b. Mostre que a recíproca do item **a** não é sempre verdadeira.

c. Sejam $n \geq 1$ um natural, $\ell_0, \ell_1, \ldots, \ell_{n-1} \in \mathbb{N}$ e $x, y \in \mathbb{Z}$.
Suponha que os ℓ_i sejam primos entre si, i.e., suponha que se $i \neq j$ com $i, j < n$, então $\text{mdc}(\ell_i, \ell_j) = 1$. Prove que neste caso vale a recíproca do item **a**, isto é, prove que se

$$\text{para todo } i < n \text{ temos } \quad x \equiv y \pmod{\ell_i}.$$

então

$$x \equiv y \pmod{\ell_0 \cdot \ell_1 \cdots \ell_{n-1}}$$

(*Dica*: Indução!)

[†]Publicado em 27/5; modificado em 2/6 (nova data para submissão).

Questão 3. Chamamos um natural ímpar $n > 1$ de *pseudoprimo de Miller–Rabin para a base b* se n é composto mas o teste de Miller–Rabin para n com base b é inconclusivo.

Mostre que se um número n é pseudoprimo de Miller–Rabin para a base b, então n é um pseudoprimo de Fermat para esta mesma base.

Questão 4. A mensagem `40, 743, 680, 40, 714, 397` foi codificada pelo método RSA usando módulo $n = 871$ e expoente $e = 317$. Além disso, sabe-se que $\phi(n) = 792$. Encontre a mensagem original **sem fatorar** n ou $\phi(n)$.

Questão 5. O expoente $e = 2$ nunca deveria ser usado como expoente público no RSA. Por quê?

Questão 6.

a (TCR, versão simplificada)**.** Sejam ℓ_1 e ℓ_2 naturais coprimos e $n = \ell_1 \cdot \ell_2$.

Prove que, para quaisquer inteiros a e b, o sistema

$$\begin{cases} x \equiv a \pmod{\ell_1} \\ x \equiv b \pmod{\ell_2} \end{cases}$$

tem *uma única* solução em módulo n, e esta solução é dada por

$$x \equiv (a \cdot \ell_2 \cdot \ell_2') + (b \cdot \ell_1 \cdot \ell_1') \pmod{n},$$

onde ℓ_1' é o inverso de ℓ_1 em módulo ℓ_2 e ℓ_2' é o inverso de ℓ_2 em módulo ℓ_1.

b (TCR, versão completa)**.** Suponha que $\ell_1, \ell_2, \ldots, \ell_k$ sejam primos entre si, i.e., $\text{mdc}(\ell_i, \ell_j) = 1$ para todos $1 \leq i < j \leq k$, e seja $n = \prod_{i=1}^{k} \ell_i$.

Prove que, para quaisquer inteiros $a_1, a_2, \ldots, a_k$, o sistema

$$\begin{cases} x \equiv a_1 \pmod{\ell_1} \\ x \equiv a_2 \pmod{\ell_2} \\ \quad \vdots \\ x \equiv a_k \pmod{\ell_k} \end{cases}$$

possui *uma única solução* módulo n, e esta solução é dada por

$$x \equiv \sum_{i=1}^{k} (a_i \cdot m_i \cdot m_i') \pmod{n},$$

onde para cada i com $1 \leq i \leq k$ definimos

$$m_i = \frac{n}{\ell_i} = \prod_{\substack{j \in \{1,2,\ldots,k\} \\ j \neq i}} \ell_j$$

$$m_i' = \text{o inverso multiplicativo de } m_i \text{ em módulo } \ell_i.$$

Questão 7 (Aceleração de descriptação no RSA com o TCR)**.** Uma das aplicações práticas do Teorema Chinês dos Restos é para acelerar a etapa de descriptação de mensagens. O procedimento é o seguinte: ao gerar seu módulo público $n = pq$, expoente público e e expoente privado d, o usuário também calcula e guarda os seguintes valores:

- o inverso de p módulo q
- o inverso de q módulo p
- a forma reduzida de d módulo $p - 1$
- a forma reduzida de d módulo $q - 1$.

Lembrando que a tarefa básica na etapa de descriptação é, ao receber um bloco encriptado m, calcular a forma reduzida da potência modular $m^d \mod pq$, explique como usar o Teorema Chinês dos Restos (e o Pequeno Teorema de Fermat) e os dados calculados acima para tornar essa tarefa mais fácil (e explique em que sentido a tarefa fica mais fácil).

Questão 8. Três pares (n, e) de chaves públicas do RSA,

$$(32333464105158123139761850953950 3, 3),$$

$$(37554017468380006506803029920135 1, 5),$$

e $(42265968263874274411577354568970 1, 5)$

foram geradas usando somente 5 números primos distintos no total. Encontre a fatoração em primos de algum dos módulos públicos.

Questão 9. Implemente o RSA em Python! Sua implementação deve ter (pelo menos) os seguintes componentes.

a. Uma função para gerar números primos. Sua função deve receber como entrada um natural n e gerar um número (provavelmente) primo p satisfazendo $10^n < p < 10^{n+2}$, sorteando p aleatoriamente no intervalo desejado e rodando 10 testes de Miller–Rabin com bases b aleatórias no intervalo $1 < b < p - 1$. (Naturalmente, p só deve ser aceito como provavelmente primo se todos os testes forem inconclusivos.)

b. Uma função chamada `gera_chaves` (por favor use este nome) para gerar chaves do RSA. Sua função deve usar sua função da letra **a** para gerar primos p e q, cada um com aproximadamente 50 algarismos, e retornar:

- $n = pq$
- algum número e inversível módulo $\phi(n) = (p - 1)(q - 1)$
- o inverso d de e módulo $\phi(n)$

Para uma solução realmente *completa*, sua função deve retornar também:

- p
- q

- o inverso de p módulo q
- o inverso de q módulo p
- a forma reduzida de d módulo $p-1$
- a forma reduzida de d módulo $q-1$.

c. Uma função chamada `encriptar` (por favor use este nome) que recebe como entrada uma string `texto` e números `n` e `e`, e retorna uma lista de números que seja uma sequência válida dos blocos numéricos resultantes da encriptação do `texto` com chave pública de módulo `n` e expoente `e`.

d. Uma função chamada `descriptar` (por favor use este nome) que recebe como entrada uma lista `blocos` e números `n` e `d`, e retorna a string resultante da descriptação da sequência de blocos usando a chave privada de módulo `n` e expoente `d`.

Para uma solução realmente *completa*, implemente a versão rápida de descriptação, usando a Questão 7 e os valores adicionais retornados pela função `gera_chaves`.

Use a transformação de símbolos em números dados na tabela ao final deste documento; você encontra a tabela em versões de dicionários de Python, um para conversão de símbolos em números e outra na direção contrária, em

`https://www.hugonobrega.com/codigo.py`

Como todos usaremos a mesma tabela de conversão, faremos uma troca de mensagens encriptadas ao vivo durante a aula teórica de 10 de junhogg (começando às 9:00)! A participação é livre e não conta para a avaliação.

Teste suas funções!

Por exemplo, se você usou `gera_chaves` e obteve `n`, `e`, `d` como chaves pública e privada, então você deve obter, no interpretador do Python:

```
>>> descriptar(encriptar('reticências sempre ajudam',n,e),n,d)
'reticências sempre ajudam'
```

cód.	símb.
111	0
112	1
113	2
114	3
115	4
116	5
117	6
118	7
119	8
121	9
122	=
123	+
124	−
125	/
126	*
127	a
128	b
129	c
131	d
132	e
133	f
134	g
135	h
136	i
137	j
138	k
139	l

cód.	símb.
141	m
142	n
143	o
144	p
145	q
146	r
147	s
148	t
149	u
151	v
152	w
153	x
154	y
155	z
156	á
157	à
158	â
159	ã
161	é
162	ê
163	í
164	ó
165	ô
166	õ
167	ú
168	ç
169	A

cód.	símb.
171	B
172	C
173	D
174	E
175	F
176	G
177	H
178	I
179	J
181	K
182	L
183	M
184	N
185	O
186	P
187	Q
188	R
189	S
191	T
192	U
193	V
194	W
195	X
196	Y
197	Z
198	Á
199	À

cód.	símb.
211	Â
212	Ã
213	É
214	Ê
215	Í
216	Ó
217	Ô
218	Õ
219	Ú
221	Ç
222	,
223	.
224	!
225	?
226	;
227	:
228	_
229	(
231	)
232	"
233	#
234	$
235	%
236	@
237	(espaço)
238	(nova linha)