Seguidamente vamos determinar valores de b (em termos de a e n) para os quais a congruência $aX \equiv b \pmod{n}$ tem solução.

Se $a = 0$ esta congruência tem solução x se e só se $n \mid b$, e, neste caso, qualquer $x \in \mathbb{Z}$ satisfaz a congruência.
Se $x \in \mathbb{Z}$ satisfaz a conguência $aX \equiv b \pmod{n}$, então qualquer outro inteiro $x_1 \equiv x \pmod{n}$ a satisfaz também. Desta forma, ao encontar a solução x encontramos um conjunto de inteiros congruentes com x módulo n que satisfazem $aX \equiv b \pmod{n}$. Apesar disso, dizemos que encontrámos "só" uma solução, no sentido de que todas estas soluções são congruentes entre si módulo n.

A resolução de uma congruência linear $aX \equiv b \pmod{n}$ está intimamente relacionada com a resolução de equações diofantinas lineares em $\mathbb{Z}$. De facto,

$$aX \equiv b \pmod{n} \Leftrightarrow n \mid b - aX \Leftrightarrow \exists Y \in \mathbb{Z} : aX + nY = b$$

Já sabemos como determinar todas as soluções de uma equação diofantina linear, caso esta seja resolúvel. Para conguências temos:

Teorema

Sejam n um natural, a, b inteiros e $d = \mathrm{mdc}(a, n)$. A congruência $aX \equiv b \pmod{n}$ tem solução se e só se $d \mid b$. Nesse caso, existem d soluções distintas módulo n da forma

$$X \equiv \frac{xb + nk}{d} \pmod{n},$$

onde $k \in \{0, 1, \ldots, d-1\}$ e $x \in \mathbb{Z}$ é tal que $d = xa + yn$, para algum $y \in \mathbb{Z}$.

Demonstração. Temos $aX \equiv b \pmod{n} \Leftrightarrow \exists Y \in \mathbb{Z} : aX + nY = b$. Esta é uma equação diofantina linear que sabemos ter solução inteira se e só se $d \mid b$. Segue-se que $aX \equiv b \pmod{n}$ tem solução se e só se $d \mid b$. Sabemos também que, nesse caso, as soluções são pares (X, Y) onde X é da forma $X = \frac{xb+kn}{d}$, para alguns $k \in \mathbb{Z}$ e $x, y \in \mathbb{Z} : d = xa + yn$.
Duas soluções $\frac{xb+kn}{d}$ e $\frac{xb+k'n}{d}$ são congruentes módulo n se e só se $n \mid (\frac{xb+kn}{d} - \frac{xb+k'n}{d}) \Leftrightarrow n \mid \frac{n(k-k')}{d} \Leftrightarrow d \mid (k - k') \Leftrightarrow k \equiv k' \pmod{d}$. Logo existem d soluções distintas módulo n dadas por $X \equiv \frac{xb+nk}{d} \pmod{n}$, onde $k \in \{0, 1, \ldots, d-1\}$.

Exemplo

Vamos estudar as soluções da congruência linear $15X \equiv 66 \pmod 9$. Como $\mathrm{mdc}(15, 9) = 3$, $3 \mid 66$ e $3 = 2 \cdot 15 + (-3) \cdot 9$, temos três soluções distintas módulo 9 da forma

$$X \equiv \frac{2 \cdot 66 + 9k}{3} \pmod 9 \equiv 44 + 3k \pmod 9 \equiv 8 + 3k \pmod 9$$

para $k \in \{0, 1, 2\}$. Assim, $X \equiv 8 \pmod 9$ ou $X \equiv 2 \pmod 9$ ou $X \equiv 5 \pmod 9$ são todas as soluções.

Partindo do teorema anterior é fácil escrever um algoritmo que determine todas as soluções de uma congruência linear da forma $aX \equiv b \pmod n$, caso exista alguma.

Algoritmo (Soluções de uma congruência linear)

INPUT: $a, b \in \mathbb{Z}$ e $n \in \mathbb{N}$ tais que $aX \equiv b \pmod{n}$

$d \leftarrow \mathrm{mdc}(a, n)$

Determinar inteiros x e y tais que $d = xa + yn$

if $d \mid b$ then

as soluções módulo n são $X = \frac{xb+nk}{d}$, com $k = 0, 1, \ldots, d-1$

else

OUTPUT: não há soluções

end

OUTPUT: todos os inteiros $X \pmod{n}$ que satisfazem a congruência $aX \equiv b \pmod{n}$

Exercício

Construa uma função GAP que determine todas as soluções módulo n da congruência linear $aX \equiv b \pmod{n}$, dados os inteiros a, b, n, com $n > 0$. Seguidamente use a implementação obtida para determinar as soluções de $15X \equiv 66 \pmod{9}$.

Aplicações
Detecção de erros

Uma das aplicações da aritmética modular está relacionada com a detecção de erros nos sistemas de identificação, como, por exemplo, nos códigos de barras.
Os números de identificação são usados em diversas situações, podendo ocorrer erros no seu registo, leitura ou transmissão.
Surge, desta forma, a necessidade de conceber sistemas de identificação que permitam detectar os erros ocorridos no decurso de qualquer um desses passos.
A experiência revela que a generalidade dos erros cometidos são de dois tipos: erros num só algarismo - estes são a grande maioria - e erros de troca de dois algarismos consecutivos, tornando-se a detecção destes tipos de erros fundamental.

A aritmética modular é muitas vezes utilizada para atribuir um dígito extra aos números de identificação, chamado o *dígito de controlo* ou *check digit*, com o objectivo de detectar erros.

Exemplo

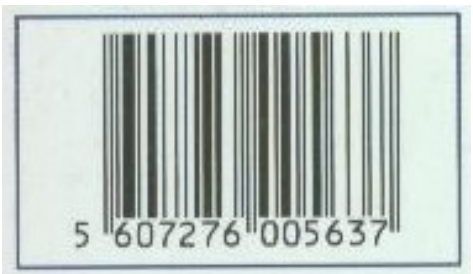
No sistema EAN-13 (European Article Number) os números de identificação do código de barras têm 13 algarismos. Os primeiros 7 algarismos identificam o produtor (destes, os 2 ou 3 primeiros identificam o país de origem, por exemplo, 560 é o código atribuído a Portugal), os 5 algarismos seguintes identificam o produto no catálogo da empresa produtora. O último algarismo é o algarismo de controlo e é este que permite ao leitor óptico confirmar se leu o número correctamente.
Um código de barras EAN-13 em que os algarismos do número de identificação são, por esta ordem, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13} satisfaz a condição seguinte:

$$a_1 + 3a_2 + a_3 + 3a_4 + a_5 + 3a_6 + a_7 + 3a_8 + a_9 + 3a_{10} + a_{11} + 3a_{12} + a_{13} \equiv 0 \pmod{10}$$

Esta condição permite determinar univocamente o algarismo de controlo a partir dos outros doze,

$$a_{13} \equiv -(a_1 + 3a_2 + a_3 + 3a_4 + a_5 + 3a_6 + a_7 + 3a_8 + a_9 + 3a_{10} + a_{11} + 3a_{12}) \pmod{10}$$

Verifiquemos que o código de barras da figura seguinte satisfaz a condição acima referida:



$(5 + 3 \cdot 6 + 0 + 3 \cdot 7 + 2 + 3 \cdot 7 + 6 + 3 \cdot 0 + 0 + 3 \cdot 5 + 6 + 3 \cdot 3 + 7) \pmod{10} \equiv 110 \pmod{10} \equiv 0 \pmod{10}$

O sistema EAN permite detectar erros num só algarismo e erros de troca de dois algarismos consecutivos, desde que a diferença entre estes não seja 5 (ou -5).

Existem outros sistemas análogos ao EAN, por exemplo o UPC (Universal Product Code) estando a principal diferença no número de algarismos utilizado. Para mais informação ver: http://www.gs1.org/.

Exemplo

O Sistema ISBN (International Standard Book Number) é um sistema de identificação numérica de livros, CD-Roms e publicações em braille. Cada editora identifica os seus livros com um número $a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}$ de 10 dígitos, servindo os primeiros 9 algarismos (divididos por 3 secções de comprimento variável) para identificar o livro e sendo a_{10} o dígito de teste. A primeira secção de algarismos identifica o país ou um agrupamento geográfico de editoras, a segunda secção identifica uma empresa editora particular desse grupo e a terceira identifica o número do livro dentro do catálogo da empresa editora. Este sistema, em 1 de Janeiro de 2007, passará a ter 13 dígitos. Em http://www.isbn.org/ encontra-se mais informação sobre este sistema.

A sequência de dígitos $a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10}$ do sistema ISBN verifica:

$$10a_1 + 9a_2 + 8a_3 + 7a_4 + 6a_5 + 5a_6 + 4a_7 + 3a_8 + 2a_9 + 1a_{10} \equiv 0 \pmod{11}$$

(onde a_{10} é substituído por 10 no caso de ser X). Assim, o dígito de controlo é dado por

$$a_{10} \equiv -(10a_1 + 9a_2 + 8a_3 + 7a_4 + 6a_5 + 5a_6 + 4a_7 + 3a_8 + 2a_9) \pmod{11}$$

O ISBN 972-23-2780-1 identifica o livro "O homem que sabia contar" de Malba Tahan. "972" é o código atribuido a Portugal e "23" é a identificação da editora Editorial Presença, que catalogou o livro com o número "2780".
O dígito de controlo "1" é dado por
$-(10 \cdot 9 + 9 \cdot 7 + 8 \cdot 2 + 7 \cdot 2 + 6 \cdot 3 + 5 \cdot 2 + 4 \cdot 7 + 3 \cdot 8 + 2 \cdot 0) \pmod{11} \equiv -263 \pmod{11} \equiv 1 \pmod{11}$.

No cálculo do dígito de controlo do ISBN existe a possibilidade de $a_{10} = 10$, pois 10 é um resto possível na divisão por 11. Quando isto acontece, usa-se o símbolo X (10 na numeração romana).

Exercício

Construa em GAP uma função que determine o dígito de controlo de um sistema de identificação ISBN dados os primeiros 9 algarismos.

O sistema ISBN permite detectar erros num só algarismo e erros de troca de dois algarismos, mesmo que não sejam consecutivos.
O dígito de controlo do Bilhete de Identidade é calculado pelo mesmo algoritmo do ISBN, com a diferença de que não utiliza o símbolo "X" quando o algarismo de controlo é 10, mas sim o "0".
Existem vários exemplos de sistemas de identificação modulares. Estes apresentam em geral várias limitações que podem ser ultrapassadas usando Teoria de Grupos... Sugere-se a leitura do artigo de Jorge Picado (consta da bibliografia desta disciplina).

Sistema Criptográfico RSA

Uma outra aplicação da aritmética modular está relacionada com a Criptografia, ciência que se ocupa das comunicações secretas. A procura de formas que possibilitem estabelecer comunicações emissor-receptor sem que terceiros tenham acesso à informação transmitida é uma preocupação antiga, inicialmente, associada a comunicações militares. Actualmente, a confidencialidade e a autenticidade das comunicações estabelecidas via Internet assumem uma enorme importância.
Os sistemas criptográficos, utilizados para codificar/descodificar mensagens, dizem-se simétricos ou tradicionais quando utilizam uma única chave, que serve tanto para codificar como para descodificar. A criptografia assimétrica (mais recente) utiliza uma chave para codificar e outra para decodificar as mensagens, pelo que o conhecimento da chave de codificação não basta para conseguir descodificar, o que torna os sistemas assimétricos mais seguros.
O sistema criptográfico RSA é assimétrico.
A sigla RSA vem do nome dos matemáticos R. Rivest, A. Shamir e L. Adleman que descrevem este sistema (ver bibliografia).

O resultado seguinte, estabelecido à custa do Teorema de Euler, é a base do sistema RSA.

Proposição

Sejam m um número natural e a um inteiro primo com m. Sendo c e d números naturais, tem-se que, se $cd \equiv 1 \pmod{\varphi(m)}$, então $a^{cd} \equiv a \pmod{m}$.

Demonstração. Se $cd \equiv 1 \pmod{\varphi(m)}$, então $cd = 1 + t\varphi(m)$, para um certo inteiro não negativo t. Assim, utilizando o Teorema de Euler, temos

$$a^{cd} = a^{1+t\varphi(m)} = a \cdot a^{t\varphi(m)} = a \cdot (a^{\varphi(m)})^t \equiv a \pmod{m}$$

Portanto, $a^{cd} \equiv a \pmod{m}$. □

Na prática, m vai aparecer como um produto de dois primos p e q e um modo de garantir que a é primo com m é fazer com que p e q sejam maiores que a. Devido ao resultado seguinte, consequência do Pequeno Teorema de Fermat, esse cuidado adicional ($\mathrm{mdc}(a, m) = 1$) não vai ser necessário.

Proposição

Sejam p e q primos distintos e a um inteiro. Seja $m = pq$. Se c e d são inteiros tais que $cd \equiv 1 \pmod{\varphi(m)}$, então $a^{cd} \equiv a \pmod{m}$.

Demonstração. Como $cd \equiv 1 \pmod{\varphi(m)}$, tem-se $cd = 1 + t\varphi(m)$, para um certo inteiro não negativo t. Suponhamos agora que $\mathrm{mdc}(a, p) = 1$. Notando que $\varphi(m) = (p-1)(q-1)$ e usando o Pequeno Teorema de Fermat,

$$a^{cd} = a^{1+t\varphi(m)} = a \cdot a^{t\varphi(m)} = a \cdot (a^{p-1})^{t(q-1)} \equiv a \cdot (1)^{t(q-1)} \equiv a \pmod{p},$$

logo $p \mid (a^{cd} - a)$.
Se $\mathrm{mdc}(a, p) \neq 1$, então $p \mid a$ e, portanto, também $p \mid (a^{cd} - a)$.
De modo análogo podemos concluir que $q \mid (a^{cd} - a)$.
Assim, uma factorização de $a^{cd} - a$ envolve tanto os primos p e q, logo $a^{cd} - a$ é divisível por pq, ou seja, $a^{cd} \equiv a \pmod{m}$. □

Para codificar uma mensagem, utilizando o sistema RSA, o primeiro passo é transformá-la numa sequência de números (pode-se, por exemplo, converter um texto numa sequência de números utilizando a tabela ASCII - *American Standard Code for Information Interchange*).
Usando a notação dos resultados anteriores, façamos $m = pq$, sendo p e q dois números primos. Dados dois números inteiros c (de *codificação*) e d (de *descodificação* tais que $cd \equiv 1 \pmod{\varphi(m)}$, isto é, $cd \equiv 1 \pmod{(p-1)(q-1)}$, podemos codificar um número x, com $x < m$ inteiro, utilizando

$$C : x \longrightarrow x^c \pmod{m}.$$

Os resultados dados asseguram que a função

$$D : y \longrightarrow y^d \pmod{m}.$$

é a função de descodificação, pois

$$D(C(x)) = x^{cd} \pmod{m} \equiv x \pmod{m}.$$

Na prática, um certo Receptor que pretenda possibilitar a Emissores que lhe enviem mensagens codificadas deve escolher primos distintos p e q ("grandes") e um valor c ("pequeno") tal que $\mathrm{mdc}(c,(p-1)(q-1))=1$. Depois, deve dar a conhecer os valores c e m (que constituem a *chave pública*) aos Emissores interessados. Um Emissor que pretenda enviar uma mensagem secreta ao Receptor usa a chave pública do Receptor para codificar a sua mensagem.
Só o Receptor, que conhece a factorização de $m=pq$ e consequntemente pode calcular o inverso, d, de c módulo $(p-1)(q-1)$, consegue descodificar a mensagem.
Como a função codificadora e os seus parâmetros são públicos, este tipo de sistemas criptográficos dizem-se de *chave pública*. O parâmetro d diz-se a *chave privada*.
A segurança do método RSA baseia-se no facto de, para valores de m suficientemente grandes, decompor m como produto de dois primos distintos p e q ser muito difícil (com os algoritmos e computadores actuais pode levar milhões de anos). Por outro lado, existem métodos eficientes para encontrar um número grande cuja probabilidade de não ser primo é negligenciável.

Vamos seguidamente considerar alguns exemplos. Para o efeito, faremos uso do GAP. Chamamos a atenção para as funções `INT_CHAR` e `CHAR_INT` que permitem fazer a correspondência entre caracteres e os respectivos códigos ASCII.

Exemplo

Sejam $p = 19$ e $q = 23$. Então, $m = 19 \cdot 23 = 437$ e $\varphi(m) = 18 \cdot 22 = 396$. Seja $c = 7$, temos $\mathrm{mdc}(c, \varphi(m)) = 1$ e $c < \varphi(m)$. Para determinar o inverso de $c = 7$ módulo $\varphi(m) = 396$ vamos utilizar o GAP.

```
gap> d := 7^-1 mod 396;
283
```

Então, $d = 283$.
Um receptor R, para permitir que lhe enviem mensagens codificadas torna pública a chave $c = 7$; $m = 437$ e guarda para si a chave $d = 283$.

Vamos supor que um emissor E pretende enviar ao receptor A o seguinte texto confidencial: "Estou no Porto". Com recurso ao GAP

```
gap> L:="Estou no Porto";
"Estou no Porto"
gap> LN:=List(L,i->INT_CHAR(i));
[ 69,115,116,111,117,32,110,111,32,80,111,114,116,111 ]
```

o texto numa é transformado sequência de números:
069; 115; 116; 111; 117; 032; 110; 111; 032; 080; 111; 114; 116; 111.
Depois, calcula-se $a^7 \pmod{437}$, para cada número a da lista anterior.

```
gap> LC:=List(LN,a->a^7 mod 437);
[ 69,115,185,245,59,257,374,245,257,329,245,114,185,245 ]
gap>
```

A sequência
069, 115, 185, 245, 059, 257, 374, 245, 257, 329, 245, 114, 185, 245
representa a mensagem codificada e pronta a enviar.

O receptor R, ao receber a mensagem vai descodificá-la, primeiro calculando $y^{283} \pmod{437}$, para cada número y da lista que recebeu e, depois, fazendo corresponder a cada número obtido o caracter da tabela previamente combinada.

```
gap> LC:=[ 69, 115, 185, 245, 59, 257, 374, 245, 257, 329, 245,
gap> LD:=List(LC,y->y^283 mod 437);
[ 69, 115, 116, 111, 117, 32, 110, 111, 32, 80, 111, 114, 116, 1
gap> LD:=List(LD,i->CHAR_INT(i));
"Estou no Porto"
gap>
```

A seguir apresenta-se o mesmo texto a ser transmitido, mas são usados outros primos.

```
gap> p := 257;;q := 263;;
gap> m := p*q;
67591
gap> c := 11;;
gap> d := 11^-1 mod Phi(m);
12195
gap>  L:="Estou no Porto";;
gap> LN:=List(L,i->INT_CHAR(i));
[ 69,115,116,111,117,32,110,111,32,80,111,114,116,111 ]
gap> LC:=List(LN,x->x^11 mod 67591);
[ 39487,3108,11259,9744,54697,63864,15881,9744,63864,8412,9744,
  7092,11259,9744 ]
gap> LD:=List(LC,y->y^d mod m);
[ 69,115,116,111,117,32,110,111,32,80,111,114,116,111 ]
gap> LD:=List(LD,i->CHAR_INT(i));
"Estou no Porto"
```

Pode construir exemplos envolvendo números e mensagens bem maiores...

Para o efeito deve começar por obter o ficheiro "rsa_algebra_CC.g" a partir da página da disciplina.

O ficheiro "rsa_exemplo_CC.g", também disponível na página da disciplina, contém um exemplo.