1.4. Raciocínio matemático, indução e recursão

Para entendermos um texto matemático temos que compreender o que faz com que um argumento esteja matematicamente correcto, isto é, seja uma prova. Uma *prova* é uma demonstração de que alguma afirmação é verdadeira. Normalmente apresentamos provas escrevendo frases em Português misturadas com equações e símbolos matemáticos.

Quando é que um argumento matemático está correcto?

Um *teorema* é uma afirmação que se pode demonstrar ser verdadeira. Demonstra-se que um teorema é verdadeiro com uma sequência de afirmações que formam um argumento, chamada *prova*. Para construir provas, precisamos de métodos que nos permitam deduzir novas afirmações a partir de afirmações já comprovadas. As afirmações usadas numa prova incluem os *axiomas* ou *postulados* da teoria, as hipóteses do teorema a provar e teoremas previamente provados. As *regras de inferência* são as ferramentas para deduzir novas conclusões e ligam os diversos passos da prova. Recorde de 1.1 que um argumento do tipo

$$\begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_n \\ \hline \therefore B \end{array}$$

é uma regra de inferência se $A_1 \wedge A_2 \wedge \cdots \wedge A_n \rightarrow B$ for uma tautologia. Por exemplo, é a tautologia $(p \wedge (p \rightarrow q)) \rightarrow q$ que está por trás da regra *modus ponens*, como vimos no Capítulo 1.1. Vimos na altura, também, uma lista das regras de inferência mais usadas no raciocínio matemático:

Regras de inferência	**Tautologia**	**Nome**
$\begin{array}{l} p \\ p \rightarrow q \\ \hline \therefore q \end{array}$	$(p \wedge (p \rightarrow q)) \rightarrow q$	Modus ponens (MP)
$\begin{array}{l} \neg q \\ p \rightarrow q \\ \hline \therefore \neg p \end{array}$	$(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$	Modus tollens (MT)
$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$	$p \rightarrow (p \vee q)$	Adição (Ad)
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	$(p \wedge q) \rightarrow p$	Simplificação (S)
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	$((p \vee q) \wedge \neg p) \rightarrow q$	Silogismo disjuntivo (SD)
$\begin{array}{l} p \rightarrow q \\ q \rightarrow r \\ \hline \therefore p \rightarrow r \end{array}$	$((p \rightarrow q) \wedge (q \rightarrow r)) \rightarrow (p \rightarrow r)$	Silogismo hipotético (SH)

Qualquer argumento elaborado com regras de inferência diz-se *válido*. Quando todas as afirmações usadas num argumento válido são verdadeiras, podemos ter a certeza de chegar a uma conclusão correcta. No entanto, um argumento válido pode conduzir a conclusões incorrectas se uma ou mais proposições falsas são usadas no argumento. Por exemplo,

> "*Se* 101 *é divisível por* 3 *então* 101^2 *é divisível por* 9. 101 *é divisível por* 3. *Logo,* 101^2 *é divisível por* 9."

é um argumento válido (baseado na regra MP) mas a conclusão é falsa: 9 não divide $101^2 = 10\,201$.

Noutro tipo de falácias muito comum as conclusões estão incorrectas porque os argumentos não são válidos: apesar de aparentarem ser regras de inferência, na realidade não o são (baseiam-se em contingências e não em tautologias).

Exemplo 1. A proposição $((p \rightarrow q) \wedge q) \rightarrow p$ não é uma tautologia (é falsa quando p é falsa e q é verdadeira). No entanto, por vezes é usada como se fosse uma tautologia (este tipo de argumento incorrecto chama-se *falácia de afirmar a conclusão*):

> *Se resolver todos os exercícios destes apontamentos, então aprenderá matemática discreta. Aprendeu matemática discreta. Logo, resolveu todos os exercícios.*

(É claro que pode aprender matemática discreta sem precisar de resolver todos os exercícios destes apontamentos!)

Exemplo 2. A proposição $((p \rightarrow q \wedge \neg p) \rightarrow \neg q$ não é uma tautologia, pois é falsa quando p é falsa e q é verdadeira. É outro exemplo de proposição que por vezes é usada como regra de inferência em argumentos incorrectos (a chamada *falácia de negar a hipótese*):

> *Se resolver todos os exercícios destes apontamentos, então aprenderá matemática discreta. Não resolveu todos os exercícios. Logo, não aprendeu matemática discreta.*

(É claro que pode aprender matemática discreta sem ter resolvido todos os exercícios destes apontamentos!)

Que métodos podemos usar para elaborar argumentos matemáticos correctos?

Se p e q são proposições, já vimos no Capítulo 1.1 como se define o valor lógico das proposições "não p", "p e q", " p ou q" e "se p então q". Esta última ("se p então q" ou "p implica q") é uma afirmação *condicional* com *hipótese* p e *conclusão* q. A sua *contraposta* é a afirmação "se não q então não p" e a sua *recíproca* é "se q então p". A seguinte tabela de verdade mostra que uma condicional e a sua contraposta são equivalentes:

p	q	$\neg q$	$\neg p$	$p \rightarrow q$	$\neg q \rightarrow \neg p$
V	V	F	F	V	V
V	F	V	F	F	F
F	V	F	V	V	V
F	F	V	V	V	V

Como a maioria dos teoremas utilizados na prática da matemática são implicações, as técnicas para provar implicações são muito importantes.

Façamos então uma breve digressão, com exemplos, pelas diversas técnicas de prova usadas pelos matemáticos.

(1) Verificação exaustiva. Algumas proposições podem ser provadas por verificação exaustiva de um número finito de casos.

Exemplo 1. *Existe um número primo entre 890 e 910.*

Prova. Verificando exaustivamente descobrirá que o 907 é primo. □

Exemplo 2. *Cada um dos números 288, 198 e 387 é divisível por 9.*

Prova. Verifique que 9 divide cada um desses números. □

É claro que uma proposição enunciada para um número infinito de casos não poderá ser provada directamente por verificação exaustiva (por mais casos que consigamos verificar nunca conseguiremos verificar todos...). Por exemplo, se tentarmos comprovar, com a ajuda do `Maple`, a famosa **conjectura de Goldbach**, que afirma que

qualquer inteiro par maior do que 2 pode-se escrever como soma de dois primos

não encontraremos nenhum *contra-exemplo* (isto é, um exemplo que refute a conjectura). Mesmo assim, não poderemos garantir que a conjectura é verdadeira para qualquer inteiro par maior do que 2 (tal como ninguém o conseguiu fazer até hoje!).

Procedimento que calcula a decomposição de um inteiro par na soma de dois primos:

```
> Goldbach := proc(p::integer)
>   local i,j,terminou,next_i_valor;
>   terminou := false;
>   i := 0; j := 0;
>   while not terminou do
>      next_i_valor := false;
>      i := i+1; j :=i;
>      while not next_i_valor do
>         if ithprime(i) + ithprime (j) = p then
>            printf(`%d pode ser expresso como %d+%d\n`,p,ithprime(i),ithprime(j));
>            terminou := true;
>            next_i_valor := true;
>         fi;
>         j := j+1;
>         if ithprime(j) >= p then
>            next_i_valor := true
>         fi;
```

```
>       od;
>    od;
> end:
```

```
> Goldbach(156456);
```

```
156456 pode ser expresso como 19+156437
```

Procedimento mais automático, que realiza o cálculo para todos os inteiros pares entre n e m:

```
> ManyGoldbach := proc(startval::integer,finalval::integer)
>    local i;
>    for i from max(4,startval) to finalval do
>        if i mod 2 = 0 then Goldbach(i);
>        fi;
>    od;
> end:
```

Verificação da conjectura para os inteiros pares entre 4 e 50:

```
> ManyGoldbach(4,50);
```

```
4 pode ser expresso como 2+2
6 pode ser expresso como 3+3
8 pode ser expresso como 3+5
10 pode ser expresso como 3+7
12 pode ser expresso como 5+7
14 pode ser expresso como 3+11
16 pode ser expresso como 3+13
18 pode ser expresso como 5+13
20 pode ser expresso como 3+17
22 pode ser expresso como 3+19
24 pode ser expresso como 5+19
26 pode ser expresso como 3+23
28 pode ser expresso como 5+23
30 pode ser expresso como 7+23
32 pode ser expresso como 3+29
34 pode ser expresso como 3+31
36 pode ser expresso como 5+31
38 pode ser expresso como 7+31
40 pode ser expresso como 3+37
42 pode ser expresso como 5+37
44 pode ser expresso como 3+41
46 pode ser expresso como 3+43
48 pode ser expresso como 5+43
50 pode ser expresso como 3+47
```

(2) Prova de condicionais. A maioria dos teoremas que se provam em matemática são condicionais. Começamos por assumir que a hipótese é verdadeira. Depois tentamos encontrar uma proposição que resulte da hipótese e/ou factos conhecidos. Continuamos deste modo até chegarmos à conclusão.

Exemplo 3. *Se m é ímpar e n é par então $m-n$ é ímpar.*

Prova. Suponhamos que m é ímpar e n é par. Então $m = 2k+1$ e $n = 2l$ para alguns inteiros k e l. Portanto, $m - n = 2k + 1 - 2l = 2(k - l) + 1$, que é um inteiro ímpar. □

Exemplo 4. *Se n é ímpar então n^2 é ímpar.*

Prova. Suponhamos que n é ímpar. Então $n = 2k + 1$ para algum inteiro k. Portanto, $n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1$, que é um inteiro ímpar pois $2k^2 + 2k$ é um inteiro. □

Exemplo 5. *Se n^2 é par então n é par.*

Prova. A contraposta desta afirmação é "se n é ímpar então n^2 é ímpar", que é verdadeira pelo Exemplo anterior. □

Teste (1 minuto). Prove as recíprocas dos Exemplo 4 e 5.

(3) Prova de equivalências ("se e só se"; abreviadamente "sse"). Uma proposição da forma "p se e só se q" significa "p implica q" e "q implica p". Portanto, é preciso apresentar duas provas. Por vezes, estas provas podem ser escritas como uma só prova na forma "p sse r sse s sse ... sse q", onde cada "... sse ..." é conclusão evidente da informação anterior.

Exemplo 6. *n é par se e só se $n^2 - 2n + 1$ é ímpar.*
Prova.

n é par	sse	$n = 2k$ para algum inteiro k	(definição)
	sse	$n - 1 = 2k - 1$ para algum inteiro k	(álgebra)
	sse	$n - 1 = 2(k - 1) + 1$ para algum inteiro $k - 1$	(álgebra)
	sse	$n - 1$ é ímpar	(definição)
	sse	$(n - 1)^2$ é ímpar	(Exemplo 4 e Teste)
	sse	$n^2 - 2n + 1$ é ímpar	(álgebra) □

Muitas vezes um teorema afirma que determinadas proposições $p_1, p_2, \ldots, p_n$ são equivalentes, isto é, $p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n$ (o que assegura que as n proposições têm a mesma tabela de verdade). Uma maneira de provar o teorema é usar a tautologia

$$(p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n) \leftrightarrow ((p_1 \rightarrow p_2) \wedge (p_2 \rightarrow p_3) \wedge \cdots \wedge (p_n \rightarrow p_1))$$

que assegura a equivalência

$$(p_1 \leftrightarrow p_2 \leftrightarrow \cdots \leftrightarrow p_n) \equiv ((p_1 \rightarrow p_2) \wedge (p_2 \rightarrow p_3) \wedge \cdots \wedge (p_n \rightarrow p_1)).$$

Exemplo 7. *Para cada inteiro n as proposições seguintes são equivalentes:*

(i) *n é ímpar.*

(ii) *n^2 é ímpar.*

(iii) *$n^2 - 2n + 1$ é par.*

Prova. Para provar a equivalência das três asserções, basta provar que as implicações (i)→(ii), (ii)→(iii) e (iii)→(i) são verdadeiras:
(i)→(ii): Provada no Exemplo 4.

(ii)→(iii): Se n^2 é ímpar então n^2+1 é par. Como $2n$ é sempre par, então n^2-2n+1 é par.
(iii)→(i): No Exemplo 6 provámos que se n é par então n^2-2n+1 é ímpar. A implicação (iii)→(i) é a sua contraposta, pelo que também é verdadeira. □

(4) Prova por contradição (redução ao absurdo). Como vimos em 1.1, uma proposição que é sempre falsa é chamada uma *contradição*. Por exemplo, "p e não p" é uma contradição para qualquer proposição p. Vimos também que da tabela de verdade da implicação decorre que "se p então q" é equivalente a "se não q então não p"; isto é ainda equivalente a "p e não q implica falso":

p	q	$p \to q$	$\neg p$	$\neg q$	$\neg q \to \neg p$	$p \wedge \neg q$	$p \wedge \neg q \to$ F
V	V	V	F	F	V	F	V
V	F	F	F	V	F	V	F
F	V	V	V	F	V	F	V
F	F	V	V	V	V	F	V

Portanto temos aqui diversas maneiras equivalentes de demonstrar uma implicação $p \to q$. Por exemplo, da última equivalência ficamos a saber que para provar "se p então q" é suficiente provar "p e não q implica falso", ou seja, assumir p e $\neg q$ e depois argumentar de modo a chegar a uma contradição (afirmação falsa). Chama-se a esta técnica de demonstração *prova por contradição* (ou *por redução ao absurdo*).

Exemplo 8. *Se n^2 é ímpar então n é ímpar.*

Prova. Suponhamos, *por absurdo*, que n^2 é ímpar e n é par. Então $n = 2k$ para algum inteiro k pelo que $n^2 = (2k)^2 = 4k^2 = 2(2k^2)$ é também um inteiro par. Chegamos assim à conclusão que n^2 é simultaneamente um inteiro par e ímpar, o que é uma contradição. □

Exemplo 9. *Se $2|5n$ então n é par.*

Prova. Suponhamos, *por absurdo*, que $2|5n$ e n é ímpar. Então $5n = 2d$ para algum inteiro d e $n = 2k+1$ para algum inteiro k. Juntando tudo obtemos $2d = 5n = 5(2k+1) = 10k+5$. Logo $5 = 2d - 10k = 2(d-5k)$, o que é uma contradição pois afirma que 5 é um número par! □

Exemplo 10. *$\sqrt{2}$ é um número irracional.*

Prova. Suponhamos, *por absurdo*, que $\sqrt{2}$ é racional. Então $\sqrt{2} = \frac{p}{q}$ para algum par de inteiros p e q (podemos assumir que a fracção p/q está já escrita na sua forma reduzida, isto é, $\mathrm{mdc}(p,q) = 1$). Elevando ao quadrado ambos os membros da igualdade anterior obtemos $2q^2 = p^2$, pelo que p^2 é par. Então, pelo Exemplo 5, p é par. Sendo p par, é claro que p^2 é um múltiplo de 4, ou seja, $p^2 = 4k$ para algum inteiro k. Consequentemente, $2q^2 = 4k$, isto é, $q^2 = 2k$ é par, pelo que q também é par. Chegámos aqui a uma contradição: p e q são pares mas $\mathrm{mdc}(p,q) = 1$. □

Exercício. Identifique, nas provas realizadas nas duas secções anteriores (1.2 e 1.3), as técnicas de demonstração aí utilizadas.

(5) Prova por indução matemática. A que é igual a soma dos n primeiros inteiros positivos ímpares? Para $n = 1, 2, 3, 4, 5$ tem-se

$$\begin{aligned}
1 &= 1,\\
1+3 &= 4,\\
1+3+5 &= 9,\\
1+3+5+7 &= 16,\\
1+3+5+7+9 &= 25.
\end{aligned}$$

Destes valores particulares é razoável conjecturar[18] que a soma, para qualquer n, deverá ser igual a n^2. O *método de indução matemática* permite-nos provar facilmente que esta conjectura está correcta[19]. Trata-se de um método muito potente para provar asserções deste tipo, enunciadas sobre o conjunto $\mathbb{N}$ dos naturais. Baseia-se na observação óbvia que todo o subconjunto não vazio de $\mathbb{N}$ tem um elemento mínimo e no seguinte:

Base do Princípio de Indução Matemática. *Seja $S \subseteq \mathbb{N}$, $1 \in S$ e suponhamos que $k \in S$ implica $k+1 \in S$. Então $S = \mathbb{N}$.*

Prova. Suponhamos, por absurdo, que $S \neq \mathbb{N}$. Então $\mathbb{N} - S \neq \varnothing$ logo tem um elemento mínimo m. Como $1 \in S$ e $m \notin S$, então $m > 1$. Portanto, $m - 1$ é um natural e pertence a S (pois m é o mínimo de $\mathbb{N} - S$). Logo, por hipótese, $(m-1)+1 \in S$, isto é, $m \in S$. Chegamos assim a uma contradição: $m \notin S$ e $m \in S$. Em conclusão, $S = \mathbb{N}$. □

Princípio de Indução Matemática (PIM). *Seja $P(n)$ uma proposição para cada $n \in \mathbb{N}$. Para provar que $P(n)$ é verdadeira para qualquer $n \in \mathbb{N}$ basta:*

(1) (Passo inicial) *Mostrar que $P(1)$ é verdadeira.*

(2) (Passo indutivo) *Mostrar que a implicação $P(n) \rightarrow P(n+1)$ é verdadeira para qualquer $n \in \mathbb{N}$.*

Prova. Suponhamos que os dois passos foram provados. Seja $S = \{n \mid P(n) \text{ é verdadeira}\}$. O passo inicial garante que $1 \in S$; por outro lado, o passo indutivo garante que $k \in S$ implica $k+1 \in S$. Então, pela Base do PIM, podemos concluir que $S = \mathbb{N}$. □

No passo indutivo, $P(n)$ chama-se *hipótese de indução*.

Exemplo 11. *Para qualquer natural n, a soma dos n primeiros inteiros positivos ímpares é igual a n^2.*

Prova. Seja $P(n)$ a proposição $1+3+5+\cdots+(2n-1) = n^2$. $P(1)$ é claramente verdadeira: $1 = 1^2$. Assumindo que $P(n)$ é verdadeira, provemos que $P(n+1)$ é verdadeira. O membro esquerdo de $P(n+1)$ é:

$$\begin{aligned}
1+3+5+\cdots+(2n-1)+(2n+1) &= (1+3+5+\cdots+(2n-1))+(2n+1)\\
&= n^2+(2n+1) && \text{(hip. indução)}\\
&= (n+1)^2 && \text{(álgebra)}
\end{aligned}$$

[18] Veja www.mat.uc.pt/~picado/ediscretas/somatorios/Matematica_sem_palavras_files/soma_impares.html.

[19] Assim como todas as outras em www.mat.uc.pt/~picado/ediscretas/somatorios.

que é o membro direito de $P(n+1)$. Portanto, $P(n+1)$ é verdadeira e, pelo PIM, segue que $P(n)$ é verdadeira para qualquer n. □

Exemplo 12. *Seja* $f : \mathbb{N} \to \mathbb{N}$ *definida por*

$$f(n) = \begin{cases} 1 & \textit{se } n = 1 \\ f(n-1) + n^2 & \textit{se } n \neq 1. \end{cases}$$

Então $f(n) = \dfrac{n(n+1)(2n+1)}{6}$ *para qualquer* n.

Prova. Seja $P(n)$ a proposição $f(n) = n(n+1)(2n+1)/6$. Como $f(1) = 1$ e $1(1+1)(2+1)/6 = 1$, então $P(1)$ é verdadeira. Assumindo que $P(n)$ é verdadeira, provemos que $P(n+1)$ é verdadeira. O membro esquerdo de $P(n+1)$ é:

$$\begin{aligned} f(n+1) &= f(n+1-1) + (n+1)^2 && \text{(definição de } f\text{)} \\ &= f(n) + (n+1)^2 && \text{(álgebra)} \\ &= \tfrac{n(n+1)(2n+1)}{6} + (n+1)^2 && \text{(hip. indução)} \\ &= \tfrac{(n+1)(2n^2+7n+6)}{6} && \text{(álgebra)} \\ &= \tfrac{(n+1)(n+2)(2n+3)}{6} && \text{(álgebra)} \\ &= \tfrac{(n+1)((n+1)+1)(2(n+1)+1)}{6} && \text{(álgebra)} \end{aligned}$$

que é o membro direito de $P(n+1)$. Portanto, $P(n+1)$ é verdadeira e, pelo PIM, segue que $P(n)$ é verdadeira para qualquer n. □

O PIM também funciona para proposições $P(n)$ com $n \in \{a, a+1, \ldots\}$. Neste caso, o elemento mínimo é a, pelo que basta modificar o passo inicial para "mostrar que $P(a)$ é verdadeira".

Teste. Usando indução matemática, prove:

(a) A fórmula $1 + 2 + 3 + \cdots + (n-1) = \frac{n^2-n}{2}$ que usámos na página 40.

(b) Os números harmónicos $H(n)$ satisfazem a desigualdade $H(2^n) > \frac{n}{2}$.

Em algumas situações pode ser difícil definir um objecto de modo explícito e ser mais fácil defini-lo em função dele próprio. A este processo chama-se *recursão* e pode ser usado para definir sequências, sucessões e conjuntos. Por exemplo, a sequência das potências de 2 pode ser definida explicitamente por $a_n = 2^n$ para $n = 0, 1, 2, \ldots$, mas também pode ser definida recursivamente pelo primeiro termo $a_0 = 1$ e pela regra que permite definir um termo à custa dos anteriores: $a_{n+1} = 2a_n$ para $n = 0, 1, 2, \ldots$.

Portanto, podemos definir uma função sobre os inteiros não negativos

- especificando o valor da função em 0 e
- dando uma regra que permita calcular o seu valor num inteiro a partir dos valores em inteiros menores.

Tal definição chama-se uma *definição recursiva* ou *indutiva*.

Teste (1 minuto). Encontre uma definição recursiva da função factorial

$$f(n) = n! = n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1.$$

As definições recursivas são também utilizadas frequentemente para definir conjuntos. Nesse caso, especifica-se um colecção inicial de elementos como pertencendo ao conjunto que se pretende definir, e depois especificam-se as regras de construção dos elementos do conjunto a partir de elementos que já se sabe estarem no conjunto. Foi o que fizemos, quando logo na Secção 1.1 definimos deste modo o conjunto das fórmulas bem formadas do cálculo proposicional. Os conjuntos definidos deste modo ficam bem definidos e os teoremas sobre eles podem ser provados usando a definição recursiva.

Exemplo. Seja S o conjunto definido recursivamente por

- $3 \in S$,
- Se $x \in S$ e $y \in S$ então $x + y \in S$.

Mostre que S é o conjunto dos inteiros positivos divisíveis por 3. (Assume-se implicitamente neste tipo de definições que um elemento só pertence a S se puder ser gerado usando as duas regras na definição de S.)

Prova. Seja C o conjunto de todos os inteiros positivos divisíveis por 3. Para provar a igualdade $C = S$ temos que verificar as inclusões $C \subseteq S$ e $S \subseteq C$.

$C \subseteq S$: Provemos por indução matemática que todo o inteiro positivo divisível por 3 pertence a S. Para isso seja $P(n)$ a proposição "$3n \in S$". O passo inicial $P(1)$ é verdadeiro pela primeira regra da definição recursiva. Para estabelecer o passo indutivo, assumimos que $P(n)$ é verdadeira, ou seja, que $3n \in S$. Mas então, como 3 também pertence a S, pela segunda regra da definição recursiva, $3n + 3 = 3(n + 1)$ também está em S.

$S \subseteq C$: Basta mostrar que as regras de definição de S só geram elementos que estão contidos em C. A primeira é evidente: $3 \in C$. Quanto à segunda, se $x, y \in S$ são divisíveis por 3 então $x + y$ também é divisível por 3, o que completa a prova. □

Encontramos muitas vezes algoritmos recursivos quando pretendemos resolver problemas discretos. Um algoritmo chama-se *recursivo* se resolve um problema reduzindo-o a uma instância do mesmo problema com input mais pequeno.

Uma definição recursiva exprime o valor de uma função num inteiro positivo em termos dos valores da função em inteiros mais pequenos. Isto significa que podemos ter sempre um algoritmo recursivo para calcular o valor de uma função definida por recursão. Por exemplo:

Teste. Especifique um algoritmo recursivo para calcular a potência a^n de um número real a para qualquer inteiro não negativo n.

Solução. A definição recursiva de a^n diz-nos que $a^{n+1} = a \times a^n$ e $a^0 = 1$ (condição inicial). Então podemos fazer:

```
procedure potencia(a : número real ≠ 0, n : inteiro não negativo)
if n = 0 then potencia(a, n) := 1
else potencia(a, n) := a * potencia(a, n − 1)
```

Também não é difícil especificar um algoritmo recursivo para o cálculo do máximo divisor comum de dois inteiros:

```
procedure mdc(a, b : inteiros não negativos com a < b)
if a = 0 then mdc(0, b) := b
else mdc(a, b) := mdc(b mod a, a)
```

Há outro modo de calcular a função potência $f(n) = a^n$ a partir da sua definição recursiva: em vez de reduzir sucessivamente o cálculo a inteiros mais pequenos, podemos começar com o valor da função em 1 e aplicar sucessivamente a definição recursiva para encontrar os valores da função em números sucessivamente maiores. Tal procedimento diz-se *iterativo*. Por outras palavras, para calcular a^n usando um processo iterativo, começamos em 1 e multiplicamos sucessivamente por cada inteiro positivo $\leq n$:

```
procedure potencia iterativa(a : número real ≠ 0, n : inteiro positivo)
x := 1
for i := 1 to n
   x := a * x
{x é a^n}
```

Terminamos com mais um exemplo, desta vez usando o `Maple`. A famosa **conjectura de Collatz** (ou conjectura "$3x+1$"), que até hoje ninguém conseguiu provar, afirma que se pegarmos num inteiro x arbitrário, e se iterarmos sucessivamente a função

$$f(x) = \begin{cases} \dfrac{x}{2} & \text{se } x \text{ é par} \\ \\ 3x+1 & \text{se } x \text{ é ímpar} \end{cases}$$

chegaremos inevitavelmente ao inteiro 1 ao cabo de um número finito de passos.

```
procedure Collatz(n : inteiro positivo)
x := n
while x ≠ 1 do
   if x mod 2 = 0 then x := x/2
   else x := 3x + 1
```

Procedimento Maple que define a função de Collatz:

```
> Collatz := proc(n::integer)
>    if type(n,even) then
>       n/2;
>    else
>       3*n+1;
>    fi;
> end:
```

Procedimento que itera a função até que o valor obtido seja 1: (incluímos um contador dessas iterações para vermos quanto tempo demoram as iterações a estabilizar no 1; como não temos a certeza de estabilizarem sempre no 1, impomos um limite superior — razoavelmente grande — no número de iterações a efectuar)

```
> IC := proc(raiz::integer)
>    local sentinela, contador;
>    contador := 0;
>    sentinela := raiz;
>    while sentinela <> 1 and contador < 1000^1000 do
>        sentinela := Collatz(sentinela);
>        contador := contador + 1;
>    od;
>    RETURN(contador);
> end:
```

Verificação da conjectura para os primeiros 200 inteiros:

```
> seq(IC(i), i=1..200);
```

```
 0, 1, 7, 2, 5, 8, 16, 3, 19, 6, 14, 9, 9, 17, 17, 4, 12, 20, 20, 7, 7, 15, 15, 10,
23, 10, 111, 18, 18, 18, 106, 5, 26, 13, 13, 21, 21, 21, 34, 8, 109, 8, 29, 16, 16,
16, 104, 11, 24, 24, 24, 11, 11, 112, 112, 19, 32, 19, 32, 19, 19, 107, 107, 6, 27,
27, 27, 14, 14, 14, 102, 22, 115, 22, 14, 22, 22, 35, 35, 9, 22, 110, 110, 9, 9, 30,
30, 17, 30, 17, 92, 17, 17, 105, 105, 12, 118, 25, 25, 25, 25, 25, 87, 12, 38, 12,
100, 113, 113, 113, 69, 20, 12, 33, 33, 20, 20, 33, 33, 20, 95, 20, 46, 108, 108,
108, 46, 7, 121, 28, 28, 28, 28, 28, 41, 15, 90, 15, 41, 15, 15, 103, 103, 23, 116,
116, 116, 23, 23, 15, 15, 23, 36, 23, 85, 36, 36, 36, 54, 10, 98, 23, 23, 111, 111,
111, 67, 10, 49, 10, 124, 31, 31, 31, 80, 18, 31, 31, 31, 18, 18, 93, 93, 18, 44,
18, 44, 106, 106, 106, 44, 13, 119, 119, 119, 26, 26, 26, 119, 26
```

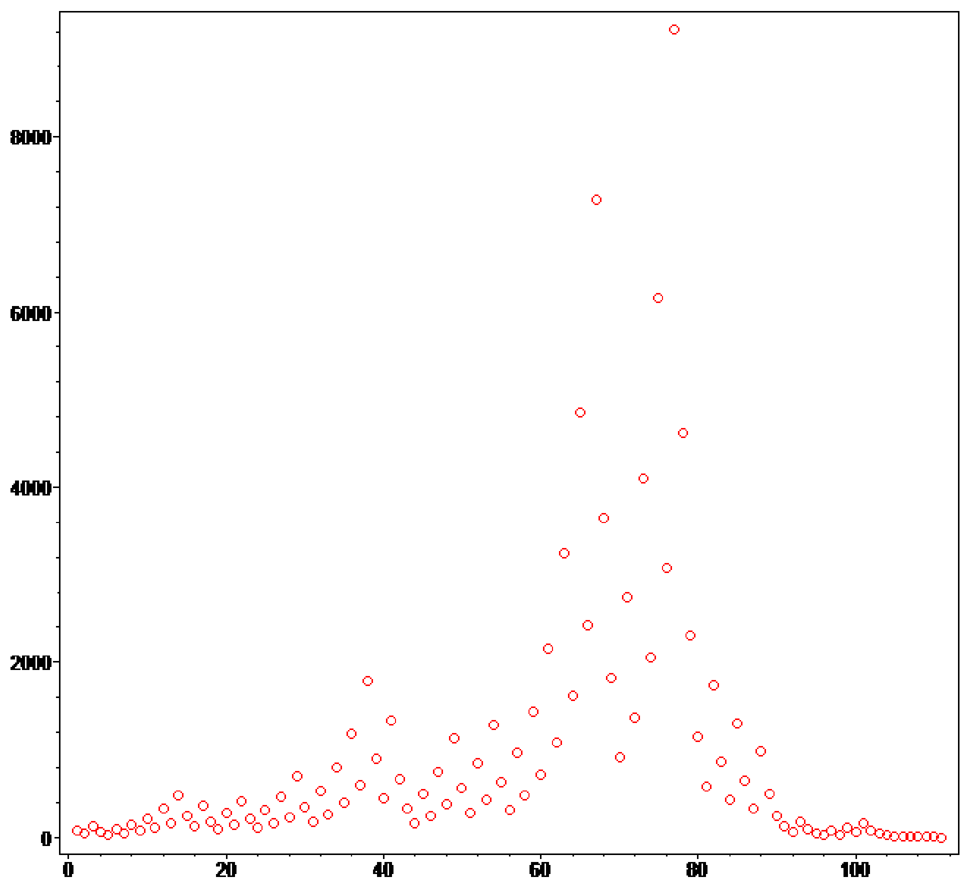
Cada número indica o número de iterações necessárias para levar a função até 1. Por exemplo, para $n = 3$ são necessárias 7 iterações e para $n = 27$ são necessárias 111 iterações. Calculemos explicitamente estas últimas:

Procedimento que calcula a sequência $n, f(n), f(f(n)), \ldots, 1$:

```
> sCollatz := proc(n::integer)
>    local x;
>    x := n;
>    while x<>1 do
>       print(x);
>       x := Collatz(x);
>    od;
> end:
>
> sCollatz(27);
```

27 82 41 124 62 31 94 47 142 71 214 107 322 161 484 242 121 364 182 91 274 137 412 206 103 310 155 466 233 700 350 175 526 263 790 395 1186 593 1780 890 445 1336 668 334 167 502 251 754 377 1132 566 283 850 425 1276 638 319 958 479 1438 719 2158 1079 3238 1619 4858 2429 7288 3644 1822 911 2734 1367 4102 2051 6154 3077 9232 4616 2308 1154 577 1732 866 433 1300 650 325 976 488 244 122 61 184 92 46 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1

Desde o valor inicial 27 até 1 a função atinge um pico na 77^a iteração com o valor 9232. Graficamente:



$n = 14$: