MÉTODOS NUMÉRICOS PARA REPRESENTAÇÃO DE NÚMEROS E SOLUÇÃO DE EQUAÇÕES

Evandro Pedro Alves de Mendonça[a], **Marcelino José de Lima Andrade**[a].

[a] *Núcleo de Tecnologia (NTI), Universidade Federal de Pernambuco (UFPE), Campus Acadêmico do Agreste (CAA), Rodovia BR-104, km 59, S/N, Nova Caruaru, CEP. 55.014-900, Caruaru-PE, Brasil, http://www.ufpe.br/caa*

Palavras Chave: conversão de bases, erros, séries, truncamento, números primos, cálculo numérico.

Resumo. Ao trabalhar com números, usando computadores, há sempre problemas relacionados à representação numérica e à precisão/exatidão dos cálculos realizados. Este trabalho aborda esse tema evidenciando possíveis problemas que ocorrerão ao lidar com estas situações e como proceder em cada caso. No desenvolvimento do trabalho, foram utilizados métodos numéricos e o MATLAB para: converter números entre as bases decimal, binária e octal; solucionar equações usando aproximações; e realizar cálculos envolvendo números primos. Calculando numericamente o resultado da função e^{-x} por dois métodos diferentes e considerando 4 e 6 dígitos significativos para cada método, foram obtidos resultados muito distintos e o erro calculado em um deles foi de 5562%, enquanto que em outro, foi de apenas 0,55%. Portanto, após a análise dos resultados obtidos, puderam ser observados: métodos numéricos com mais exatidão e que têm custo computacional semelhante aos outros menos eficazes; problemas na conversão de bases e como resolvê-los; e a praticidade de implementar funções em MATLAB para realizar cálculos repetitivos como a conversão de bases ou o cálculo de números primos.

1 INTRODUÇÃO

Em todos os ramos das ciências exatas são utilizadas bases numéricas com o intuito de representar uma determinada quantidade através de símbolos, comumente chamados de algarismos. Cada base numérica tem um número correspondente de algarismos empregados. A base mais aplicada popularmente é a base decimal, cujos algarismos empregados são 0,1,2,3,4,5,6,7,8,9. Porém, existem diversas outras bases usadas no ramo científico.

Dentre os diversos tipos de base, a base binária é amplamente utilizada, principalmente na área computacional. Como o nome sugere, essa base possui dois algarismos de representação, que são 0 e 1. Através dessa representação, os computadores conseguem realizar operações complexas e criar informações como letras, palavras e textos. Outra base empregada em métodos computacionais é o sistema octal. Essa forma de representação possui 8 algarismos de representação, que são 0,1,2,3,4,5,6,7. Na informática, ele ainda é visto como uma saída mais compacta em relação ao sistema binário. Contudo, como já mostrado, a base decimal é mais utilizada normalmente, por isso, foram criados métodos para conversão de bases.

Porém, ao utilizar-se métodos computacionais para a realização de operações numéricas, é inevitável a presença de erros associados. Os erros são decorrentes de diversos procedimentos realizados automaticamente pelo computador, como o truncamento e arredondamento de números e transformações de bases numéricas. São várias as formas de representar esses erros, e cada uma recebe uma nomeação diferente, como por exemplo, o erro relativo. Por meio do estudo dos erros, pode-se minimiza-los de modos distintos.

Dentre os métodos para a redução dos erros, temos o emprego de algarismos significativos na operação. De acordo com a quantidade de dígitos significativos, podemos diminuir ou aumentar o erro. Quando se trata de números irracionais, pode-se aumentar ou diminuir o erro através da escolha adequada de uma série para representar tal número, assim como a determinação de termos da série.

2 EXERCÍCIOS PROPOSTOS

Segue, abaixo, a solução dos exercícios propostos sobre o tema.

2.1 1ª questão

Converta os seguintes números na base 2 para a base 10:

a)

$$\begin{aligned}(10110111)_2 &= 1\times2^7+0\times2^6+1\times2^5+1\times2^4+0\times2^3+1\times2^2+1\times2^1+1\times2^0\\ &= 128+0+32+16+0+4+2+1=\mathbf{(183)_{10}}\end{aligned}$$

b)

$$\begin{aligned}(10101{,}01101)_2 &= 1\times2^4+0\times2^3+1\times2^2+0\times2^1+1\times2^0\\ &+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}+0\times2^{-4}+1\times2^{-5}\\ &=16+0+4+0+1+0+0{,}25+0{,}125+0+0{,}03125=\mathbf{(21{,}40625)_{10}}\end{aligned}$$

c) 0,011011

$$\begin{aligned}(0{,}011011)_2 &= 0\times2^{-1}+1\times2^{-2}+1\times2^{-3}+0\times2^{-4}+1\times2^{-5}+1\times2^{-6}\\ &=0{,}25+0{,}125+0+0{,}03125+0{,}015625=\mathbf{(0{,}421875)_{10}}\end{aligned}$$

2.2 2ª questão

Converta os seguintes números na base 10 para a base 2:

a) 789

$$
\begin{aligned}
789/2 &= 394 \rightarrow resto\ (1)\\
394/2 &= 197 \rightarrow resto\ (0)\\
197/2 &= 98 \rightarrow resto\ (1)\\
98/2 &= 49 \rightarrow resto\ (0)\\
49/2 &= 24 \rightarrow resto\ (1)\\
24/2 &= 12 \rightarrow resto\ (0)\\
12/2 &= 6 \rightarrow resto\ (0)\\
6/2 &= 3 \rightarrow resto\ (0)\\
3/2 &= 1 \rightarrow resto\ (1)
\end{aligned}
$$

$$(789)_{10} = (\mathbf{1100010101})_{\mathbf{2}}$$

b) 74,0926

Parte inteira:

$$
\begin{aligned}
74/2 &= 37 \rightarrow resto\ (0)\\
37/2 &= 18 \rightarrow resto\ (1)\\
18/2 &= 9 \rightarrow resto\ (0)\\
9/2 &= 4 \rightarrow resto\ (1)\\
4/2 &= 2 \rightarrow resto\ (0)\\
2/2 &= 1 \rightarrow resto\ (0)
\end{aligned}
$$

$$(74)_{10} = (1001010)_2$$

Parte fracionária:

$$
\begin{aligned}
0{,}0926 \times 2 &= 0{,}1852\\
0{,}1852 \times 2 &= 0{,}3704\\
0{,}3704 \times 2 &= 0{,}7408\\
0{,}7408 \times 2 &= 1{,}4816\\
0{,}4816 \times 2 &= 0{,}9632\\
0{,}9632 \times 2 &= 1{,}9264\\
0{,}9264 \times 2 &= 1{,}8528\\
0{,}8528 \times 2 &= 1{,}7056\\
0{,}7056 \times 2 &= 1{,}4112\\
0{,}4112 \times 2 &= 0{,}8224\\
0{,}8224 \times 2 &= 1{,}6448
\end{aligned}
$$

O procedimento continua indefinidamente, porém foi definido que pararia após a décima iteração quando fosse encontrado o primeiro bit 1. Com isso, temos a seguinte aproximação para a parte fracionária do número em base decimal:

$$(0{,}0926)_{10} = (0{,}00010111101)_2$$

Assim, temos a seguinte conversão:

$$(74{,}0926)_{10} = (\mathbf{1001010{,}00010111101})_{\mathbf{2}}$$

No Anexo 1, é mostrada uma planilha em que algumas fórmulas foram inseridas para que seja feita a conversão da parte fracionária de um número em base decimal em uma parte fracionária na base binária. Com essa ferramenta, é possível obter facilmente muito mais acurácia.

c) 0,048911

$$\begin{aligned}
0{,}048911 \times 2 &= \color{red}{0}{,}097822 \\
0{,}097822 \times 2 &= \color{red}{0}{,}195644 \\
0{,}195644 \times 2 &= \color{red}{0}{,}391288 \\
0{,}391288 \times 2 &= \color{red}{0}{,}782576 \\
0{,}782576 \times 2 &= \color{red}{1}{,}565152 \\
0{,}565152 \times 2 &= \color{red}{1}{,}130304 \\
0{,}130304 \times 2 &= \color{red}{0}{,}260608 \\
0{,}260608 \times 2 &= \color{red}{0}{,}521216 \\
0{,}521216 \times 2 &= \color{red}{1}{,}042432 \\
0{,}042432 \times 2 &= \color{red}{0}{,}084864 \\
0{,}084864 \times 2 &= \color{red}{0}{,}169728 \\
0{,}169728 \times 2 &= \color{red}{0}{,}339456 \\
0{,}339456 \times 2 &= \color{red}{0}{,}678912 \\
0{,}678912 \times 2 &= \color{red}{1}{,}357824
\end{aligned}$$

Pelo mesmo critério de parada da letra b), obtém-se a seguinte conversão:

$$(0{,}048911)_{10} = (\mathbf{0{,}00001100100001})_{\mathbf{2}}$$

2.3 3ª questão

Converta os seguintes números na base 8 para a base 10:

a) 67,45

$$\begin{aligned}(67{,}45)_8 &= 6 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} + 5 \times 8^{-2} = 48 + 7 + 0{,}5 + 0{,}078125 \\ &= (\mathbf{55{,}578125})_{\mathbf{10}}\end{aligned}$$

b) 3,0163

$$\begin{aligned}(3{,}0163)_8 &= 3 \times 8^0 + 0 \times 8^{-1} + 1 \times 8^{-2} + 6 \times 8^{-3} + 3 \times 8^{-4} \\ &= 3 + 0 + 0{,}015625 + 0{,}01171875 + 0{,}000732421875 \\ &= (\mathbf{3{,}028076171875})_{\mathbf{10}}\end{aligned}$$

c) 0,1361

$$\begin{aligned}(0{,}1361)_8 &= 0 \times 8^0 + 1 \times 8^{-1} + 3 \times 8^{-2} + 6 \times 8^{-3} + 1 \times 8^{-4} \\ &= 0 + 0{,}125 + 0{,}046875 + 0{,}01171875 + 0{,}000244140625 \\ &= (\mathbf{0{,}183837890625})_{\mathbf{10}}\end{aligned}$$

2.4 4ª questão

Para esta questão, usaremos apenas o truncamento dos números usando 4 e 6 dígitos significativos, ou seja, o arredondamento não será utilizado. Serão feitos cálculos utilizando a série de Taylor para a função e^{-x} com 12 termos. O objetivo é calcular o valor de e^{-5} através desses dois métodos e comparar com o valor real $6{,}7379469991 \times 10^{-3}$.

- Usando 4 dígitos significativos:
  - Primeira abordagem:

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} + \frac{x^6}{6!} - \frac{x^7}{7!} + \frac{x^7}{8!} - \frac{x^9}{9!} + \frac{x^{10}}{10!} - \frac{x^{11}}{11!}$$

$$e^{-5} = 1 - 5 + \frac{5^2}{2!} - \frac{5^3}{3!} + \frac{5^4}{4!} - \frac{5^5}{5!} + \frac{5^6}{6!} - \frac{5^7}{7!} + \frac{5^8}{8!} - \frac{5^9}{9!} + \frac{5^{10}}{10!} - \frac{5^{11}}{11!}$$

$$e^{-5} = 1{,}000 - 5{,}000 + \frac{25{,}00}{2{,}000} - \frac{125{,}0}{6{,}000} + \frac{625{,}0}{24{,}00} - \frac{3125}{120{,}0} + \frac{1{,}562 \times 10^4}{720{,}0}$$

$$- \frac{7{,}812 \times 10^4}{5040} + \frac{3{,}906 \times 10^5}{4{,}032 \times 10^4} - \frac{1{,}953 \times 10^6}{3{,}628 \times 10^5} + \frac{9{,}765 \times 10^6}{3{,}628 \times 10^6} - \frac{4{,}882 \times 10^7}{3{,}991 \times 10^7}$$

$$\begin{gathered} e^{-5} = 1{,}000 - 5{,}000 + 12{,}50 - 20{,}83 + 26{,}04 - 26{,}04 \\ +21{,}69 - 15{,}50 + 9{,}687 - 5{,}383 + 2{,}691 - 1{,}223 \end{gathered}$$

$$e^{-5} = -4{,}000 - 8{,}330 + 0{,}000 + 6{,}190 + 4{,}304 + 1{,}468$$

$$e^{-5} = -12{,}33 + 6{,}190 + 5{,}772 = -6{,}140 + 5{,}772$$

$$\boldsymbol{e^{-5} = -0{,}3680 = -3{,}680 \times 10^{-1}}$$

  - Segunda abordagem:

$$e^{-x} = \frac{1}{e^x} = \frac{1}{1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} + \frac{x^7}{8!} + \frac{x^9}{9!} + \frac{x^{10}}{10!} + \frac{x^{11}}{11!}}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{1 + 5 + \frac{5^2}{2!} + \frac{5^3}{3!} + \frac{5^4}{4!} + \frac{5^5}{5!} + \frac{5^6}{6!} + \frac{5^7}{7!} + \frac{5^8}{8!} + \frac{5^9}{9!} + \frac{5^{10}}{10!} + \frac{5^{11}}{11!}}$$

$$e^{-5} = \frac{1}{e^5}$$

$$= \begin{pmatrix} 1{,}000 + 5{,}000 + \frac{25{,}00}{2{,}000} + \frac{125{,}0}{6{,}000} + \frac{625{,}0}{24{,}00} + \frac{3125}{120{,}0} + \frac{1{,}562 \times 10^4}{720{,}0} \\ + \frac{7{,}812 \times 10^4}{5040} + \frac{3{,}906 \times 10^5}{4{,}032 \times 10^4} + \frac{1{,}953 \times 10^6}{3{,}628 \times 10^5} + \frac{9{,}765 \times 10^6}{3{,}628 \times 10^6} + \frac{4{,}882 \times 10^7}{3{,}991 \times 10^7} \end{pmatrix}^{-1}$$

$$e^{-5} = \frac{1}{e^5} = \begin{pmatrix} 1{,}000 + 5{,}000 + 12{,}50 + 20{,}83 + 26{,}04 + 26{,}04 \\ +21{,}69 + 15{,}50 + 9{,}687 + 5{,}383 + 2{,}691 + 1{,}223 \end{pmatrix}^{-1}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{6{,}000 + 33{,}33 + 52{,}08 + 37{,}19 + 15{,}07 + 3{,}914}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{39{,}33 + 89{,}27 + 18{,}98} = \frac{1}{128{,}6 + 18{,}98} = \frac{1}{147{,}5}$$

$$\boldsymbol{e^{-5} = \frac{1}{e^5} = 6{,}779 \times 10^{-3}}$$

- Usando 6 dígitos significativos:
  - Primeira abordagem:

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} + \frac{x^6}{6!} - \frac{x^7}{7!} + \frac{x^7}{8!} - \frac{x^9}{9!} + \frac{x^{10}}{10!} - \frac{x^{11}}{11!}$$

$$e^{-5} = 1 - 5 + \frac{5^2}{2!} - \frac{5^3}{3!} + \frac{5^4}{4!} - \frac{5^5}{5!} + \frac{5^6}{6!} - \frac{5^7}{7!} + \frac{5^8}{8!} - \frac{5^9}{9!} + \frac{5^{10}}{10!} - \frac{5^{11}}{11!}$$

$$e^{-5} = 1{,}00000 - 5{,}00000 + \frac{25{,}0000}{2{,}00000} - \frac{125{,}000}{6{,}00000} + \frac{625{,}000}{24{,}0000} - \frac{3125{,}00}{120{,}000}$$
$$+ \frac{15625{,}0}{720{,}000} - \frac{78125{,}0}{5040{,}00} + \frac{390625}{40320{,}0} - \frac{1{,}95312 \times 10^6}{362880} + \frac{9{,}76562 \times 10^6}{3{,}62880 \times 10^6} - \frac{4{,}88281 \times 10^7}{3{,}99168 \times 10^7}$$

$$e^{-5} = 1{,}00000 - 5{,}00000 + 12{,}5000 - 20{,}8333 + 26{,}0416 - 26{,}0416$$
$$+21{,}7013 - 15{,}5009 + 9{,}68812 - 5{,}38227 + 2{,}69114 - 1{,}22324$$

$$e^{-5} = -4{,}00000 - 8{,}33330 + 0{,}00000 + 6{,}20040 + 4{,}30585 + 1{,}46790$$

$$e^{-5} = -12{,}3333 + 6{,}20040 + 5{,}77375 = -6{,}13290 + 5{,}77375 = \boldsymbol{-0{,}359150}$$
$$= \boldsymbol{-3{,}59150 \times 10^{-1}}$$

  - Segunda abordagem:

$$e^{-x} = \frac{1}{e^x} = \frac{1}{1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} + \frac{x^7}{8!} + \frac{x^9}{9!} + \frac{x^{10}}{10!} + \frac{x^{11}}{11!}}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{1 + 5 + \frac{5^2}{2!} + \frac{5^3}{3!} + \frac{5^4}{4!} + \frac{5^5}{5!} + \frac{5^6}{6!} + \frac{5^7}{7!} + \frac{5^8}{8!} + \frac{5^9}{9!} + \frac{5^{10}}{10!} + \frac{5^{11}}{11!}}$$

$$= \left(\begin{array}{c} 1{,}00000 + 5{,}00000 + \frac{25{,}0000}{2{,}00000} + \frac{125{,}000}{6{,}00000} + \frac{625{,}000}{24{,}0000} + \frac{3125{,}00}{120{,}000} + \frac{15625{,}0}{720{,}000} \\ + \frac{78125{,}0}{5040{,}00} + \frac{390625}{40320{,}0} + \frac{1{,}95312 \times 10^6}{362880} + \frac{9{,}76562 \times 10^6}{3{,}62880 \times 10^6} + \frac{4{,}88281 \times 10^7}{3{,}99168 \times 10^7} \end{array} \right)^{-1}$$

$$e^{-5} = \frac{1}{e^5} = \left(\begin{array}{c} 1{,}00000 + 5{,}00000 + 12{,}5000 + 20{,}8333 + 26{,}0416 + 26{,}0416 \\ +21{,}7013 + 15{,}5009 + 9{,}68812 + 5{,}38227 + 2{,}69114 + 1{,}22324 \end{array} \right)^{-1}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{6{,}00000 + 33{,}3333 + 52{,}0832 + 37{,}2022 + 15{,}0703 + 3{,}91438}$$

$$e^{-5} = \frac{1}{e^5} = \frac{1}{39{,}3333 + 89{,}2854 + 18{,}9846} = \frac{1}{128{,}618 + 18{,}9846} = \frac{1}{147{,}602}$$

$$\boldsymbol{e^{-5} = \frac{1}{e^5} = 6{,}77497 \times 10^{-3}}$$

Temos, portanto, os seguintes resultados obtidos:

	1ª abordagem	**2ª abordagem**	**Valor real**
4DS	-3,680×10^{-1}	6,779×10^{-3}	6,7379469991×10^{-3}
6DS	-3,59150×10^{-1}	6,77497×10^{-3}	

Tabela 1: Valores obtidos nos métodos aplicados

Assim, podemos calcular os erros relativos segundo a seguinte fórmula:

$$Erro\ relativo = E_r = \frac{|valor\ real - valor\ calculado|}{|valor\ real|} = \frac{|x - \bar{x}|}{|x|}$$

- 4DS (1ª abordagem):
$$E_{r1} = \frac{|6{,}7379469991 \times 10^{-3} - (-3{,}680 \times 10^{-1})|}{|6{,}7379469991 \times 10^{-3}|} = 55{,}62 = \mathbf{5562}\%$$

- 4DS (2ª abordagem):
$$E_{r2} = \frac{|6{,}7379469991 \times 10^{-3} - (6{,}779 \times 10^{-3})|}{|6{,}7379469991 \times 10^{-3}|} = 0{,}006093 = \mathbf{0{,}6093}\%$$

- 6DS (1ª abordagem):
$$E_{r3} = \frac{|6{,}7379469991 \times 10^{-3} - (-3{,}59150 \times 10^{-1})|}{|6{,}7379469991 \times 10^{-3}|} = 54{,}3025 = \mathbf{5430{,}25}\%$$

- 6DS (2ª abordagem):
$$E_{r2} = \frac{|6{,}7379469991 \times 10^{-3} - (6{,}77497 \times 10^{-3})|}{|6{,}7379469991 \times 10^{-3}|} = 0{,}00549470 = \mathbf{0{,}549470}\%$$

Portanto, temos a seguinte tabela com os erros relativos:

	1ª abordagem	**2ª abordagem**
4DS	5562%	0,6093%
6DS	5430,25%	0,549470%

Tabela 2: Erros relativos

Assim, fica evidente que a melhor forma de calcular e^{-5} segundo os métodos utilizados é com 6 dígitos significativos e usando a segunda abordagem.

2.5 5ª questão

O algoritmo implementado consta no Anexo 2.

2.6 6ª questão

O algoritmo implementado consta no Anexo 3.

3 CONCLUSÃO

Após a análise dos resultados, conclui-se que:

A compreensão do funcionamento das bases numéricas leva a entender como os cálculos são processados nas máquinas e, visto isso, é possível entender também os erros associados às transformações entre bases, em que, por exemplo, informações podem ser perdidas ao realizar truncamento ou arredondamento. Tudo isso está sempre limitado à capacidade de processamento, armazenamento e endereçamento da máquina que executa essas tarefas, isto é, dependendo das tolerâncias envolvidas na aplicação que está sendo desenvolvida, é necessário escolher o método e a máquina corretas para manter a exatidão dos cálculos dentro dos valores esperados.

A forma como determinada função é calculada pode gerar resultados muito distintos e, a depender do método numérico empregado para realizar os cálculos, a um custo computacional equiparável, é possível ter resultados muito mais aproximados do valor real em um número de iterações pequeno.

Criar funções em MATLAB ou qualquer outro software científico pode ser uma ferramenta muito útil, uma vez que poupa o trabalho de ter que fazer o mesmo código várias vezes. Para o caso em que foi desenvolvido nesse trabalho, há formas mais eficientes de calcular, por exemplo, números primos ou fazer conversão de inteiro para decimal, porém, se for algo mais específico, dependendo do projeto, não há forma prática já implementada disponível e, assim, será necessário criar códigos específicos para aquela aplicação.

REFERÊNCIAS

Chapra, S. C., e Canale, R. P. *Métodos Numéricos para Engenharia*. 5ª edição. Porto Alegre: AMGH, 2011.

Gilat, A., e Subramaniam, V. *Métodos Numéricos para Engenheiros e Cientistas: uma introdução com aplicações usando o MATLAB*. Porto Alegre: Bookman, 2008.

ANEXO 1

Planilha com a qual foi realizada a conversão de base decimal para binária (apenas a parte fracionária).

Fórmulas usadas:

- Células da Coluna D: “=C2*2”;
- Células da Coluna E: “=SE(D2=1;1;SE(D2<1;0;1))”;
- Células da Coluna C: “=SE(E2=0;D2;D2-1)”.

Parte fracionária inicial	Iteração	Parte fracionária da iteração atual	x2	Bits
0,0926	1	0,0926	0,1852	0
	2	0,1852	0,3704	0
	3	0,3704	0,7408	0
	4	0,7408	1,4816	1
	5	0,4816	0,9632	0
	6	0,9632	1,9264	1
	7	0,9264	1,8528	1
	8	0,8528	1,7056	1
	9	0,7056	1,4112	1
	10	0,4112	0,8224	0
	11	0,8224	1,6448	1
	12	0,6448	1,2896	1
	13	0,2896	0,5792	0
	14	0,5792	1,1584	1
	15	0,1584	0,3168	0
	16	0,3168	0,6336	0
	17	0,6336	1,2672	1
	18	0,2672	0,5344	0
	19	0,5344	1,0688	1
	20	0,0688	0,1376	0

Tabela 3: Questão 2, letra b), parte fracionária

Parte fracionária inicial	Iteração	Parte fracionária da iteração atual	x2	Bits
0,048911	1	0,048911	0,097822	0
	2	0,097822	0,195644	0
	3	0,195644	0,391288	0
	4	0,391288	0,782576	0
	5	0,782576	1,565152	1
	6	0,565152	1,130304	1
	7	0,130304	0,260608	0
	8	0,260608	0,521216	0
	9	0,521216	1,042432	1
	10	0,042432	0,084864	0
	11	0,084864	0,169728	0
	12	0,169728	0,339456	0
	13	0,339456	0,678912	0
	14	0,678912	1,357824	1
	15	0,357824	0,715648	0
	16	0,715648	1,431296	1
	17	0,431296	0,862592	0
	18	0,862592	1,725184	1
	19	0,725184	1,450368	1
	20	0,450368	0,900736	0

Tabela 4: Questão 2, letra c)

ANEXO 2

somaprimos.m

```
function [ SP ] = somaprimos(num) %declaracao da função conforme questão
%Esta função soma os números primos entre 100 e 500 que sejam menores que 'num'.
%Obs.: 'num' deve ser um número maior que 100 e menor que 500
%Exemplo: somaprimos(110) -> ans = 420
    if((num<=100)||(num>=500)) %validação inicial
        disp('Argumento inválido! Número deve ser maior que 100 e menor que 500.');
        SP = 'Erro';
        return
    end
    %inicialização de variáveis
    SP = 0;
    naoeprimo = 0;%0 = falso; 1 = verdadeiro.

    %aqui faremos o teste de numeros primos, ou seja, faremos divisoes
    %do número por outros numeros menores que ele, exceto: 1, pois não
    %se faz necessário, uma vez que todo número é divisivel por 1; e o
    %2, pois numeros divisíveis por 2 (pares) não são primos.
    for i=101:2:(num-1) %anda de 2 em 2 ate num-1 (para não incluir o próprio num)
        naoeprimo = 0; %restaura o valor do sinalizador.
        for j = 3:2:(i/2) %testa divisão de i por 3 até i/2
            if mod(i,j) == 0 %teste da divisão inteira
                naoeprimo = 1; %sinaliza que i não é primo
            end
        end
        if naoeprimo == 0 %nesse caso, i e primo
            SP = SP + i; %acrescenta i na soma
        end
    end
end
```

Figura 1: Código da questão 5

ANEXO 3

real_binario.m

```
function Bi = real_binario(num)
%Transforma um número real positivo em um binario.
%Exemplo:
%real_binario(10)
%ans = 1 0 1 0
%Obs.: o argumento dessa função deve estar entre 0 e 999999.99

    if(num<0||num>999999.99)%validação inicial para intervalo
        disp('Valor inválido! O argumento deve estar entre 0 e 999999.99');
        return
    elseif (num == 0)||(num==1)
        Bi = num;
    else
        %conversão da parte inteira de 'num'
        int = fix(num); %atribui a parte inteira de 'num' a 'int'
        frac = num - int;%atribui a parte fracionária de 'num' a 'frac'
        Bi = [];%inicialização da variável de saída

        while (fix(int/2)>=1)%condição de parada: valor inteiro do quociente < 1
            %i recebe o resto da divisão inteira de int por 2, que só pode
            %assumir 2 valores diferentes, 0 ou 1, ou seja, i é um bit.
            i = mod(int,2);

            %o i da iteração atual é inserido no início do vetor Bi
            Bi = [i Bi];

            %atualização do valor de int
            int = fix(int/2);
        end
        Bi = [int Bi];%insere o último quociente, que é o bit mais significativo.

```

Figura 2: Código da questão 6 (parte 1)

```
        %conversão da parte fracionária
        %critério de parada: ao menos dez iterações e até encontrar o
        %primeiro bit 1, salvo quando a parte fracionária for completamente
        %convertida antes desse critério.
        if frac > 0 %testa se há parte fracionária a ser convertida
            tamanho_inteiro = size(Bi);%tamanho do vetor somente com a parte inteira
            tamanho_total = 0;
            %inicialização de variáveis
            cont = 0;
            bit = 0;

            while(cont < 10 || bit ~= 1)
                frac = frac*2;
                if (frac == 1)
                    bit = 1; %frac*2 é exatamente igual a uma parte inteira
                    Bi = [Bi bit];%adiciona o bit ao final do vetor Bi
                    break;%termina o laço pois a parte fracionária foi completamente convertida
                elseif(frac > 1)
                    bit = 1; %frac*2 superou uma parte inteira
                    frac = frac - 1;%remove a parte inteira novamente
                else
                    bit = 0; %frac*2 não atingiu uma parte inteira
                end
                Bi = [Bi bit]; %adiciona o bit ao final do vetor Bi
                cont = cont + 1; %incremento de variável de controle
            end
            tamanho_total = size(Bi);%tamanho do vetor após adicionar a parte fracionária

            %Para que o usuário identifique quais bits representam a parte
            %inteira e quais representam a parte fracionária, exibimos estes
            %dados. Os bits mais a esquerda são os da parte inteira.
            fprintf("\nTamanho da parte inteira:     %d bits",tamanho_inteiro(2));
            fprintf("\nTamanho da parte fracionária: %d bits\n",(tamanho_total(2)-tamanho_inteiro(2)));
        end
    end
end
```

Figura 3: Código da questão 6 (parte 2)