

04. Modelagem com Funções

Princípios de Engenharia de Software (Texto em Elaboração)

Italo S. Vega

italo@pucsp.br

Faculdade de Estudos Interdisciplinares (FACEI)



PUC-SP

Pontifícia Universidade Católica de São Paulo



2022 Italo S. Vega

Sumário

4	Modelagem com Funções	4
4.1	Funções Computáveis	4
4.2	Propriedades de uma Função Computável	7
4.3	Funções Computáveis e Iterações	9
4.4	Resumo	13
4.5	Exercícios	14
	Referências	16



Richard Feynman (National Science Teachers Association, 1966) — Science is the belief in the ignorance of experts.

4 Modelagem com Funções



— Programas de computador implementam modelos de **ações computacionais**.

4.1 Funções Computáveis

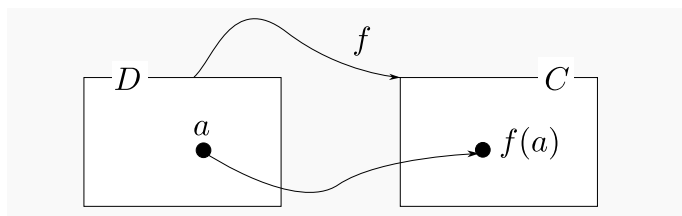
Função matemática Ao estudar o conceito de função matemática, Fubã encontrou a definição proposta por Jenkyns & Stephenson (2013), p. 51. Ele traduziu da seguinte maneira:

f é uma função do conjunto $D \neq \emptyset$ no conjunto C (escrito $f : D \rightarrow C$) se $f \subseteq D \times C$ onde cada $x \in D$ ocorre em, exatamente, um par ordenado de f .

O conjunto D é o domínio de f , enquanto C é o seu contradomínio (ou codomínio). Aqueles valores de C associados a D sob f formam o conjunto-imagem de f .



— Algo assim?



— Isso! Usaremos **dom** $f = D$ e **ran** $f \subseteq C$ para o domínio e a imagem de f .

Outra forma de se perceber a função f é como um conjunto de pares ordenados. A notação $f(x)$ refere-se ao valor correspondente à x de acordo com f :

$$f \triangleq \{ x : D \bullet x \mapsto f(x) \}$$

O símbolo \mapsto (*maplet*) enfatiza a ideia de f mapear x em um elemento $y \in \mathbf{ran} f$. Ao invés de (x, y) , escreve-se $x \mapsto y$.

4.1.1 Questão: função matemática

Contexto Considere o conjunto $g = \{1 \mapsto \text{FUBÃ}, 2 \mapsto \text{ESPEC}, 3 \mapsto \text{PROFE}\}$

Enunciado Assinale a alternativa contendo uma afirmação verdadeira:

1. $3 \in \text{ran } g$.
 2. g é uma função.
 3. $\text{dom } g = \{1, 2\}$.
 4. $g \cup \{2 \mapsto \text{PROFE}\}$ é uma função.
-

Função enumerável Um particular caso de função matemática f inclui aquelas sendo enumeráveis. Uma função é enumerável se for finita ou se existir um mapeamento um-para-um com \mathbb{N} . A função $g = \{1 \mapsto \text{FUBÃ}, 2 \mapsto \text{ESPEC}, 3 \mapsto \text{PROFE}\}$ ilustra uma função enumerável, uma vez que possui um conjunto finito de elementos.

4.1.2 Questão: arrays e funções enumeráveis

Contexto Considere a função $f \triangleq \{1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 9\}$. A imagem de 1 sob f é $f(1) = \{1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 9\}(1) = 1$. Essa função pode ser implementada pelo seguinte *array* em Python, lembrando que os índices começam em zero:

```
# CENÁRIO de implementação de "f"
f = [1, 4, 9] # implementação de função
print (f[0], f[1], f[2])
```

Enunciado Assinale a alternativa com uma afirmação **falsa**:

1. $f(3) = f[2]$.
 2. $\llbracket f(1) \rrbracket_{\text{Python}} = f[1]$.
 3. $\llbracket 2 + f(2) \rrbracket_{\text{Python}} = 2 + f[1]$.
 4. $\llbracket f(2)^2 \rrbracket_{\text{Python}} = f[1] * f[1]$.
-



— Podemos usar várias estruturas de dados para implementar uma função matemática enumerável em Python.

Espec ilustra a sua afirmação usando um *array* de valores **String**. Deve-se lembrar que o domínio de índices de um *array* corresponde a um subconjunto de $\{0, 1, \dots, n-1\}$, com n sendo representado pela quantidade de elementos do valor *array*. No exemplo ilustrado pelo Espec, $\llbracket g(1) \rrbracket_{\text{Python}} = g[0]$:

```
# CENÁRIO de uma função em Python
# Função matemática:  $g = \{(1, \text{FUBA}), (2, \text{ESPEC}), (3, \text{PROFE})\}$ 
g = ["FUBA", "ESPEC", "PROFE"] # implementação de função
print (g[1])
```

Forma-lambda Outro particular caso de função origina o conceito de forma-lambda, definida por um algoritmo que calcula a imagem de um elemento do domínio. O esquema lógico para se declarar uma forma-lambda segue:

$$f \triangleq \lambda \text{ DECLARAÇÃO } | \text{ PREDICADO } \bullet \text{ EXPRESSÃO}$$

Após o símbolo λ são declarados os parâmetros da função, explicitando-se as variáveis e o tipo de cada uma delas. Separa-se a declaração e o predicado por uma barra vertical. Na ausência do predicado, omite-se a barra também. O predicado estabelece as condições que devem ser verdadeiras para a avaliação da expressão, escrita logo depois do separador \bullet . Quando se avalia a expressão, obtém-se a imagem dos valores dos parâmetros sob a forma-lambda.



— Alternativamente, então, a função $f \triangleq \{1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 9\}$ seria especificada como:

$$f \triangleq \lambda x : \mathbb{N} \mid 1 \leq x \leq 3 \bullet x^2$$

A forma-lambda de f possui apenas um parâmetro: x do tipo \mathbb{N} . O predicado $1 \leq x \leq 3$ estabelece que o valor x deve pertencer ao conjunto $\{1, 2, 3\}$. Uma vez verificada esta condição, faz sentido avaliar-se a expressão x^2 para conhecermos a imagem de x sob f .



— Consigo implementar a função f usando um método de Python:

```
# CENÁRIO de implementação da função lambda "f"
def f(x) : # declaração de parâmetro
    if (1 <= x and x <= 3) : # predicado
        return x*x # expressão que calcula a imagem de "x"

    raise ValueError(f"Inválido x= {x}")

# Chamada do método "f"
f(3)
```



— Perfeito! Com isso, sabemos que a função $f \triangleq \{1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 9\}$ pode ser implementada por um *array* ou por um método em Python.

4.1.3 Questão: definição de funções computáveis

Contexto Considere a seguinte definição da função g :

$$g = \{1 \mapsto \text{FUBA}, 2 \mapsto \text{ESPEC}, 3 \mapsto \text{PROFE}, 4 \mapsto \text{FE}\}$$

Enunciado Assinale a alternativa contendo uma afirmação verdadeira:

1. A função g encontra-se na forma-lambda.
2. A avaliação $g(5)$ produz um absurdo.
3. A imagem de $g(\sqrt{4})$ é FE.
4. $\text{dom } g = \text{ran } g$.

4.2 Propriedades de uma Função Computável



“Posso empregar estes conceitos para desenvolver um código que calcule as raízes do seguinte polinômio: $x^2 - 3x + 2$.”

Primeiro, Fubã usa o polinômio como a expressão de uma função lambda:

$$f_1 \triangleq \lambda x : \mathbb{R} \mid -10 \leq x \leq 10 \bullet x^2 - 3x + 2$$

Como resultado da sua implementação de f na forma de um método, Fubã desenvolve o seguinte código em Python:

```
# CENÁRIO de implementação da função lambda "f1"
def f1 (x) : # declaração de parâmetro
    if (-10.0 <= x <= 10.0) : # predicado
        return x*x - 3*x + 2 # expressão que calcula a imagem de "x"

    raise ValueError(f"Inválido x= {x}")

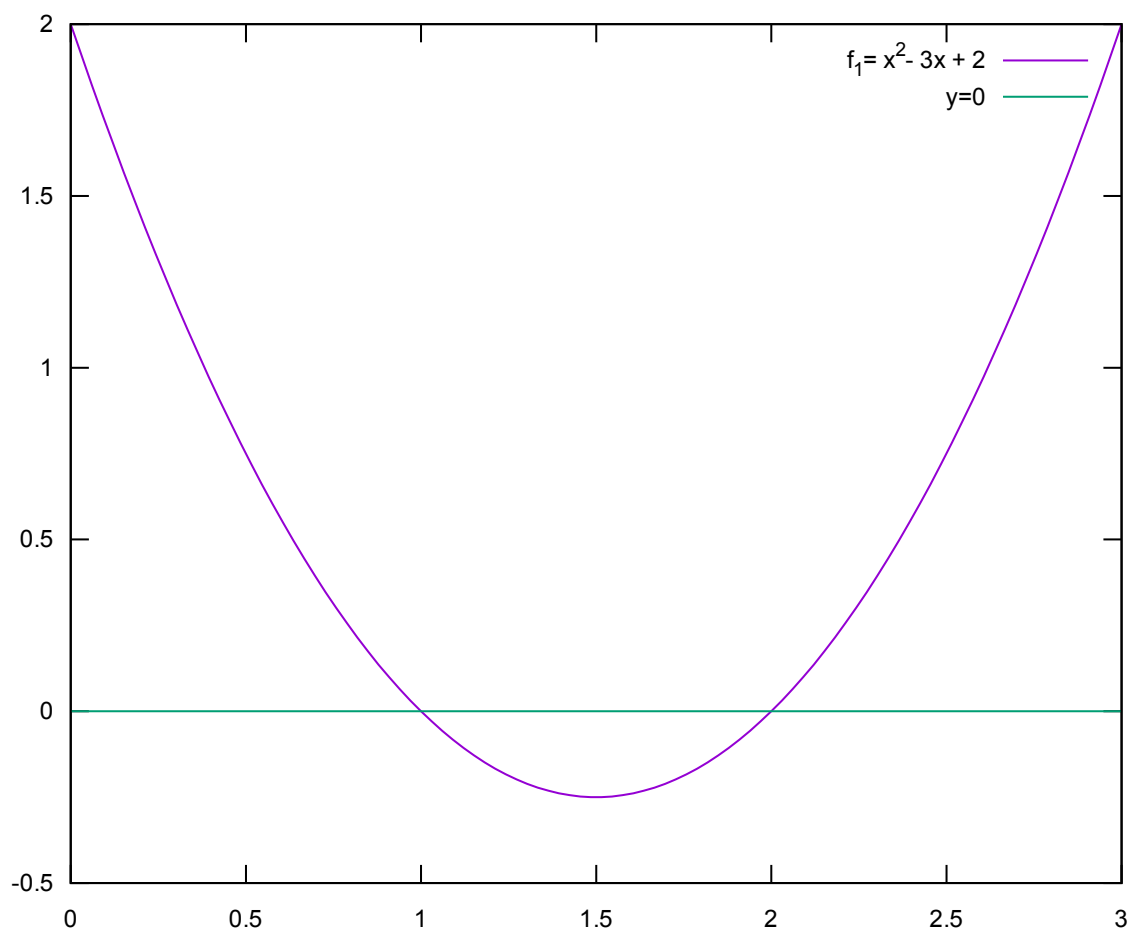
# Chamada do método "f"
print (f1(0), f1(0.5), f1(1.5), f1(2.5), f1(3))
```

Alguns dos elementos de f_1 encontram-se na tabela a seguir:

$x \mapsto$	$f_1(x)$
0	2
0.5	0.75
1.5	-0.25
2.5	0.75
3	2

4 Modelagem com Funções

Com base nestes elementos de f_1 , Fubã desenha o seguinte gráfico e percebe que os valores $x_1 = 1$ e $x_2 = 2$ são as raízes de f_1 , tornando verdadeira a condição $f_1(x_1) = 0 \wedge f_1(x_2) = 0$:



— Se eu usar a fórmula quadrática, saberei os valores do domínio x tal que $f_1(x) = 0$!

Fubã cria a função h com base na fórmula quadrática para calcular as duas soluções em \mathbb{R} da equação quadrática $x^2 - 3x + 2 = 0$:

$$h \triangleq \lambda a, b, c : \mathbb{R} \mid a \neq 0 \wedge \Delta > 0 \bullet (x_1, x_2)$$

com

$$\begin{aligned} \Delta &= b^2 - 4ac \\ x_1 &= \frac{-b + \sqrt{\Delta}}{2a} \\ x_2 &= \frac{-b - \sqrt{\Delta}}{2a} \end{aligned}$$

Na sua definição de h , a quantidade Δ determina as propriedades das raízes, sendo conhecida por discriminante. Fubã também considera a seguinte interpretação ao produzir o seu código:

$$\llbracket \mathbb{R} \rrbracket_{Python} = \text{double}$$



— Ao executar esse código de h conseguirei calcular as raízes de f_1 :

```
# CENÁRIO de implementação da fórmula quadrática
import math
def h(a, b, c) : # declaração
    if (a != 0) : # predicado
        delta = b*b - 4*a*c;
        if (delta > 0) : # predicado
            x1 = (-b + math.sqrt(delta)) / (2*a)
            x2 = (-b - math.sqrt(delta)) / (2*a)
            return [x1, x2] # expressão

        raise ValueError ("Sem soluções reais!")
    raise ValueError ("a não pode ser zero!")

# CENÁRIO de avaliação de função
x = h(1,-3,2)
print (x[0], x[1])
```

4.3 Funções Computáveis e Iterações

Fubã intrigou-se com uma observação a respeito de alguns números irracionais, como π (que também é transcendental) e $\sqrt{2}$. Eles ilustram o conceito de números computáveis, sendo definidos por um algoritmo. Weisstein (2022) apresenta um resultado importante a esse respeito:

Any computable function can be incorporated into a program using while-loops (i.e., “while something is true, do something else”). For-loops (which have a fixed iteration limit) are a special case of while-loops, so computable functions could also be coded using a combination of for- and while-loops.

O método de Newton-Raphson ilustra muito bem esse resultado. Obtém-se a solução numérica de equações por aproximações sucessivas quando se utiliza aquele método. Para calcular a raiz quadrada de um número n , por exemplo, utilizam-se as regras da seguinte lógica:

$$\begin{aligned}\text{início} &\triangleq \wedge n = 2 \\ &\wedge \epsilon = 0.001 \\ &\wedge r = n\end{aligned}$$

$$\begin{aligned}\text{passo} &\triangleq \wedge \left| r - \frac{n}{r} \right| > r \cdot \epsilon \\ &\wedge r' = \frac{r + \frac{n}{r}}{2}\end{aligned}$$



— Consigo produzir um código para implementar essa lógica:

```
# CENÁRIO de implementação de Newton-Raphson
# Raiz quadrada de 'n' com precisão 'EPS'
n = 2.0
EPS = 0.001
r = n
# o predicado inicial "início" é verdadeiro
while (abs(r - n / r) > r * EPS) :
    r = (n / r + r) / 2.0
    print (abs(r - n / r))
    # a ação "passo" é verdadeira
# a ação "passo" é verdadeira
print (n , EPS , f"{r:.3f}" )
```



— Com esses conceitos, será que não conseguimos avançar no modelo do quebra-cabeça Batalha Naval?

4.3.1 Questão: modelo dos navios de uma frota

Contexto Em um quebra-cabeça Batalha Naval¹ torna-se necessário representar as coordenadas de cada um dos cinco navios de uma frota. Cada navio foi codificado como um elemento do seguinte conjunto:

$$\text{Navios} \triangleq \{N_1, N_2, N_3, N_4, N_5\}$$

Enunciado Assinale a alternativa contendo uma correta implementação desse conjunto em Python:

1. `Navios = ['N1', 'N2', 'N3', 'N4']` com `type(Navios) == list`

¹<https://www.conceptispuzzles.com/index.aspx?uri=puzzle/battleships/history>

2. Navios = {1, 2, 3, 4, 5} com `type(Navios) == set`
 3. Navios = [True, False, 'N3', 'N4', 'N5'] com `type(Navios) == list`
 4. Navios = {"N1", "N2", 3, True, 'N5'} com `type(Navios) == set`
-

4.3.2 Questão: modelo das posições dos navios de uma frota

Contexto O conjunto de posições de navios, no que lhe concerne, foi modelada como:

$$\text{Pos} \triangleq \{ \forall x, y : \mathbb{Z} \mid 1 \leq x, y \leq 5 \bullet (x, y) \}$$

Espera-se comparar um elemento de Pos com outro. Por exemplo, é verdade que $(3, 1) = (3, 1)$.

Enunciado Assinale a alternativa contendo uma expressão em Python que produza `True`:

1. `(3,1) == [3,1]`
 2. `(3,1) in [[1,1],[2,2],[3,1],[4,4],[5,5]]`
 3. `(3,1) in {(1,1),(2,2),(3,1),(4,4),(5,5)}`
 4. `type({(1,1),(2,2),(3,1),(4,4),(5,5)}) == tuple`
-

4.3.3 Questão: coordenadas dos navios de uma frota

Contexto Finalmente, decidiu-se usar uma função `coord` que mapeia cada elemento do conjunto Navios em uma imagem em Pos, com uma restrição. Dois navios não podem estar na mesma posição. Um exemplo `coord` seria:

$$\text{coord} \triangleq \{N_1 \mapsto (1, 1), N_2 \mapsto (2, 2), N_3 \mapsto (3, 3), N_4 \mapsto (4, 4), N_5 \mapsto (5, 5)\}$$

Enunciado Assinale a alternativa contendo uma afirmação verdadeira:

1. $N_{10} : \text{Navios}$.
 2. $(0, 0) : \text{Pos}$.
 3. $\exists n : \text{Navios} \wedge \text{coord}(n) = (2, 2)$
 4. $\exists n_1, n_2 : \text{Navios} \wedge n_1 \neq n_2 \wedge \text{coord}(n_1) = \text{coord}(n_2)$
-



— Puxa! Posso codificar a função `coord` em Python usando

```
# CENÁRIO de implementação das coordenadas dos navios
coord = { "N1" : (1, 1), "N2" : (2, 2), "N3" : (3, 3),
          "N4" : (4, 4), "N5" : (5, 5) }
```



— Isso! E a aplicação da função `coord` em um navio $n \in \text{Navios}$ fica bem fácil...

Por exemplo, de acordo com a definição de `coord`, a imagem do navio N_1 é $(1, 1)$. Ou seja:

$$\text{coord}(N_1) = (1, 1)$$

Em Python:

```
# CENÁRIO de aplicação da função "coord"
coord["N1"] == (1, 1)
```

Contagem de elementos de uma função Especifica-se empolga e projeta uma outra função que calcula a quantidade de navios que se encontram em uma determinada coordenada x . A ideia é simples. Ele coleta as posições de todos os navios que se encontram na coordenada x . Em seguida, ele retorna a quantidade de posições coletadas.



— Se eu considerar o conjunto de Navios e a função `coord`, defino a função `dica` da seguinte forma:

$$\text{dica} \triangleq \{ \forall n : \text{Navios} \mid \text{coord}(n)_1 = x \bullet \text{coord}(n) \}$$



— Nossa! Consigo implementar essa lógica facilmente em Python!

```
# CENÁRIO da contagem de navios em uma coordenada X
# VARIÁVEIS de estado
Navios = {'N1', 'N2', 'N3', 'N4', 'N5'}
coord = {"N1" : (1, 1), "N2" : (2, 2), "N3" : (3, 3),
         "N4" : (4, 4), "N5" : (5, 5)}
# Função de cálculo da dica
def dica (x) :
    k = []
    for n in Navios :
        (xn, yn) = coord[n]
        if x == xn :
            k.append (coord[n])
    return len(k)
# Dica de quantos navios encontram-se na coordenada "x=1"
dica (1)
```

4.3.4 Questão: quantidade de elementos de um conjunto-função

Contexto Em uma determinada configuração do quebra-cabeça Batalha Naval, os navios foram posicionados nas seguintes coordenadas:

$$\text{coord} \triangleq \{N_1 \mapsto (1, 1), N_2 \mapsto (1, 2), N_3 \mapsto (1, 3), N_4 \mapsto (4, 4), N_5 \mapsto (5, 5)\}$$

Supondo-se que:

$$\text{Navios} \triangleq \{N_1, N_2, N_3, N_4, N_5\}$$

O cálculo da quantidade de navios em uma coordenada x é feito pela seguinte função:

$$\text{dica} \triangleq |\{ \forall n : \text{Navios} \mid \text{coord}(n)_1 = x \bullet \text{coord}(n) \}|$$

Enunciado Assinale a alternativa contendo uma afirmação verdadeira:

1. $\text{dica}(1) = 3$.
 2. $\text{dica}(2) = 2$.
 3. $\text{dica}(3) = 1$.
 4. $\text{dica}(4) = 4$.
-

4.4 Resumo



1. Função matemática como um conjunto de pares onde o primeiro elemento de cada par aparece uma única vez.
2. **dom** é o conjunto domínio de uma função.
3. **ran** é o conjunto imagem de uma função.
4. Funções computáveis são enumeráveis.
5. Expressões-lambda são usadas na definição de funções computáveis.
6. Funções computáveis são implementadas por iterações **while**.

4.5 Exercícios

4.5.1 Ano Bissexto

Contexto Para decidir se um determinado ano a é ou não bissexto, costuma-se utilizar uma lógica baseada na seguinte função (Sedgewick & Wayne, 2017):

$$\text{bis} = \lambda a : \mathbb{N} \mid (a \bmod 4 = 0 \wedge a \bmod 4 \neq 100) \vee (a \bmod 4 = 400) \bullet \text{true}$$

Enunciado Assinale a alternativa que descreve partes da regra corretamente:

1. a é bissexto se for divisível por 400.
2. Para ser bissexto, a deve ser divisível por 4.
3. Se a não for divisível por 100 ele é bissexto.
4. O ano a é bissexto se for divisível por 4, por 100 e por 400.

4.5.2 Ano Bissexto em Python

Contexto O mesmo do exercício anterior.

Enunciado

- a. Implemente a função `bis` em Python.
- b. Mostre que `bis(2016) = true`.

4.5.3 Cálculo da Raiz Quadrada [*]

Contexto O método de Newton-Raphson para calcular a raiz quadrada de um número n baseia-se nas seguintes regras:

$$\begin{aligned} \text{início} &\triangleq \wedge n = 2 \\ &\wedge \epsilon = 0.001 \\ &\wedge r = n \end{aligned}$$

$$\begin{aligned} \text{passo} &\triangleq \wedge \left| r - \frac{n}{r} \right| \leq r\epsilon \\ &\wedge r' = \frac{r + \frac{n}{r}}{2} \end{aligned}$$

Enunciado

1. Reescreva a lógica de modo que os valores de n e de ϵ sejam informados interativamente.
2. Implemente a lógica do item anterior em Python.

4.5.4 Crescimento Populacional [**]

Contexto Suponha que uma população seja representada por um valor entre 0 (extinta) e 1 (máximo sustentável)—(Sedgewick & Wayne, 2017, p. 89). Se a população no instante t é x então ela passará para $rx(1 - x)$ no instante seguinte, com r denotando a ideia de parâmetro de fecundidade. Ele controlará a taxa de crescimento a população.

Enunciado

1. Escreva o predicado inicial e a fórmula de ação para a lógica de análise populacional.
2. Desenvolva um código em Python que implemente a lógica do item anterior.
3. Se $x = 0.01$, calcule os resultados de iteração do modelo para diferentes valores de r .
4. Para quais valores de r a população se estabiliza em $x = 1 - \frac{1}{r}$?
5. O que se pode afirmar a respeito da população quando $r \in \{3.5, 3.8, 5\}$?

Referências

- Jenkyns, T., & Stephenson, B. (2013). *Fundamentals of Discrete Math for Computer Science: A Problem-Solving Primer*. Springer-Verlag.
- Sedgewick, R., & Wayne, K. (2017). *Introduction to Programming in Java: An Interdisciplinary Approach*. (A.-W. Professional, Org.) (2º ed).
- Weisstein, E. W. (2022, março 17). Computable Function. (M. W. W. Resource, Org.). Recuperado de <https://mathworld.wolfram.com/ComputableFunction.html>