



Novo  
Ensino  
Suplementar

# Módulo I

Programação Estruturada  
Combinatória e Probabilidade  
Funções Elementares e Matrizes

Professores:  
Programação Estruturada: Juliano Genari  
Combinatória e Probabilidade: Cícero Calcheiros  
Funções Elementares e Matrizes: Vinicius Guardiano

Coordenadores:  
Krerley Oliveira  
Ana Carla de Carvalho Correia  
Wagner Ranter



stone

# Conteúdo

<b>I</b>	<b>Programação Estruturada</b>	<b>1</b>
<b>1</b>	<b>Primeiro programa</b>	<b>2</b>
1.1	Algoritmos . . . . .	2
1.2	Linguagens de programação . . . . .	3
1.3	Python + Colab/Jupyter . . . . .	4
1.4	Primeiro programa . . . . .	5
<b>2</b>	<b>Fundamentos de programação</b>	<b>7</b>
2.1	Variáveis . . . . .	7
2.1.1	Tipos de dados . . . . .	7
2.1.3	Constantes . . . . .	9
2.2	Operadores aritméticos . . . . .	9
2.2.2	Operadores aplicados em strings . . . . .	11
2.3	Operadores de atribuição . . . . .	11
2.4	Conversão de tipos . . . . .	12
2.5	Entrada de dados . . . . .	12
2.6	Formatação de strings . . . . .	13
<b>3</b>	<b>Estruturas condicionais</b>	<b>15</b>
3.1	Operadores relacionais e lógicos . . . . .	15
3.2	Tabela verdade . . . . .	16
3.2.3	Estrutura IF/ELSE . . . . .	17
<b>4</b>	<b>Estruturas de repetição</b>	<b>20</b>
4.1	Laço while . . . . .	20
4.2	Laço for . . . . .	21
4.2.2	Função range . . . . .	22
4.2.4	Função enumerate . . . . .	23
4.3	Comando break . . . . .	24
4.4	Comando continue . . . . .	25
<b>6</b>	<b>Sequências</b>	<b>26</b>
6.1	Tipos de sequências . . . . .	26
6.1.1	Listas . . . . .	26
6.1.3	Tuplas . . . . .	26
6.1.5	Strings . . . . .	27
6.2	Operadores aplicados em sequências . . . . .	27
6.3	Indexação de sequências . . . . .	28
6.3.1	Operador de fatiamento . . . . .	29
6.3.2	Usos básicos do operador fatiamento . . . . .	30
6.3.3	Usos avançados do operador fatiamento . . . . .	31
6.4	Listas . . . . .	31
6.4.1	Adicionando itens a listas . . . . .	31
6.4.2	Removendo itens de listas . . . . .	32
6.4.5	Percorrendo listas . . . . .	34

<b>7</b>	<b>Strings</b>	<b>35</b>
7.1	Comparação de strings . . . . .	35
7.1.2	Comparação de substrings . . . . .	35
7.2	Métodos lower() e upper() . . . . .	36
7.3	Pesquisa em strings . . . . .	38
7.3.1	Método find() . . . . .	38
7.3.4	Método count() . . . . .	39
7.4	Quebra e junção de strings . . . . .	39
7.4.1	Método split() . . . . .	39
7.4.3	Método join() . . . . .	40
7.5	Método replace() . . . . .	41
7.5.2	Método strip() . . . . .	42
7.6	Métodos de validação . . . . .	42
<b>9</b>	<b>Dicionários</b>	<b>44</b>
9.1	Iterando em dicionários . . . . .	44
9.2	Combinando dicionários . . . . .	45
9.3	Deletando itens de dicionários . . . . .	47
9.4	Copiando dicionários . . . . .	47
<b>10</b>	<b>Funções</b>	<b>48</b>
10.1	Funções com parâmetros . . . . .	48
10.1.1	Parâmetros posicionais . . . . .	49
10.1.3	Parâmetros de palavra chave . . . . .	49
10.2	Funções com retorno . . . . .	49
10.3	Escopo de variáveis . . . . .	50
10.3.1	Variáveis locais . . . . .	50
10.3.2	Variáveis globais . . . . .	51
10.3.3	Modificando variáveis globais dentro de uma função . . . . .	51
10.4	Recursão . . . . .	52
<b>11</b>	<b>Bibliotecas</b>	<b>54</b>
11.1	Importando bibliotecas . . . . .	54
11.2	Bibliotecas nativas . . . . .	55
11.3	Mais bibliotecas . . . . .	57
<b>12</b>	<b>NumPy, parte 1</b>	<b>61</b>
12.1	Arrays NumPy . . . . .	61
12.2	Operações com Arrays . . . . .	61
12.3	Arrays multidimensionais . . . . .	63
12.4	Indexação e fatiamento . . . . .	65
12.4.2	Indexação booleana . . . . .	66
12.5	Transposição e Reformatação . . . . .	67
<b>13</b>	<b>NumPy, parte 2</b>	<b>69</b>
13.1	Funções Universais . . . . .	69
13.2	Funções de Álgebra Linear . . . . .	70
13.3	Funções de Operações de Conjunto . . . . .	71
13.4	Geração de números pseudoaleatórios . . . . .	72
<b>14</b>	<b>Captura de erros, manipulação de arquivos e simulações</b>	<b>74</b>
14.1	Captura de erros . . . . .	74
14.1.4	Identificando exceções . . . . .	76

14.2	Leitura e escrita de arquivos . . . . .	77
14.2.2	Estrutura with . . . . .	77
14.3	Simulações de probabilidades . . . . .	78
14.3.1	Simulando lançamento de moeda . . . . .	78
14.3.3	Simulando lançamento de dados . . . . .	78
14.3.5	Problema de Monty Hall . . . . .	79
<b>15</b>	<b>Pandas, parte 1</b>	<b>81</b>
15.1	Series . . . . .	81
15.1.2	Indexando Series . . . . .	82
15.2	DataFrame . . . . .	82
15.2.2	Acessando DataFrames . . . . .	83
15.3	Filtrando DataFrames . . . . .	84
15.4	Métodos de Series (ou colunas) . . . . .	86
15.5	Iterando em DataFrames . . . . .	87
15.6	Aplicação de funções . . . . .	88
<b>16</b>	<b>Pandas, parte 2</b>	<b>90</b>
16.1	Leitura e escrita de DataFrames . . . . .	90
16.2	Plotagem . . . . .	91
16.3	Outros métodos de manipulação . . . . .	93
<b>II</b>	<b>Combinatória e Probabilidade</b>	<b>95</b>
<b>17</b>	<b>Permutações com e sem repetições; permutações circulares</b>	<b>96</b>
17.1	Permutações . . . . .	97
17.2	Permutações circulares . . . . .	99
17.3	Exercícios . . . . .	100
<b>18</b>	<b>Combinações a argumentos combinatórios; combinações com repetição; contagem</b>	<b>101</b>
	<b>Dupla</b>	<b>101</b>
18.1	Combinações . . . . .	101
18.2	Combinações com repetições . . . . .	102
18.3	Exercícios Propostos . . . . .	103
<b>19</b>	<b>Princípio de Inclusão-Exclusão</b>	<b>105</b>
19.1	Exercícios propostos . . . . .	107
<b>20</b>	<b>Lemas de Kaplansky</b>	<b>109</b>
20.1	Exercícios propostos . . . . .	111
<b>21</b>	<b>Números Binomiais</b>	<b>112</b>
21.1	Exercícios Propostos . . . . .	115
<b>22</b>	<b>Princípio das Gavetas</b>	<b>116</b>
22.1	Exercícios Propostos . . . . .	118
<b>23</b>	<b>Espaços de Probabilidade</b>	<b>119</b>
23.1	Exercícios Propostos . . . . .	121
<b>24</b>	<b>Probabilidade Condicional</b>	<b>123</b>
24.1	Exercícios Propostos . . . . .	125

<b>25 Variáveis Aleatórias</b>	<b>127</b>
<b>26 Esperança</b>	<b>129</b>
26.1 Exercícios Propostos . . . . .	130
<b>27 Enunciado das Leis dos grandes números</b>	<b>132</b>
27.1 Exercícios propostos . . . . .	133
<b>28 Enunciado do Teorema Central do Limite</b>	<b>135</b>
28.1 Exercícios Propostos . . . . .	135
 <b>III Funções Elementares e Matrizes</b>	 <b>137</b>
<b>29 Noções de Lógica e Indução Finita</b>	<b>138</b>
29.1 Proposição . . . . .	138
29.2 Negação . . . . .	138
29.3 Conectivos . . . . .	139
29.4 Condicionais . . . . .	140
29.5 Tautologias . . . . .	142
29.6 Proposições logicamente falsas . . . . .	143
29.7 Relação de implicação . . . . .	143
29.8 Relação de equivalência . . . . .	143
29.9 Sentenças abertas . . . . .	143
29.10 Quantificadores . . . . .	144
29.11 Como negar proposições . . . . .	144
29.12 Princípio de Indução Finita . . . . .	146
29.13 Exercícios . . . . .	149
<b>30 Conjuntos e Noções Básicas de Funções</b>	<b>150</b>
30.1 Operações com conjuntos . . . . .	150
30.2 Descrição de um conjunto . . . . .	151
30.2.2 Conjuntos dos números . . . . .	152
30.3 Descrição de intervalos . . . . .	154
30.4 Produto cartesiano (par ordenado) . . . . .	155
30.5 Noções básicas de funções . . . . .	156
30.6 Gráfico de uma função . . . . .	158
30.7 Função injetora, sobrejetora e bijetora . . . . .	159
30.8 Função composta . . . . .	161
30.9 Função inversa . . . . .	162
30.10 Exercícios . . . . .	165
<b>31 Funções Reais e Funções Elementares</b>	<b>166</b>
31.1 Sinais de uma função . . . . .	166
31.2 Zeros de uma função . . . . .	167
31.3 Função crescente e função decrescente . . . . .	169
31.4 Função constante . . . . .	171
31.5 Função afim . . . . .	172
31.6 Exercícios . . . . .	174
<b>32 Funções Quadráticas e Problemas de Otimização</b>	<b>175</b>
32.1 Definições e Preliminares . . . . .	175

32.2 A imagem da função quadrática . . . . .	177
32.2.4 Sinais de uma função quadrática . . . . .	178
32.3 Gráfico da função quadrática . . . . .	180
32.4 Problemas de Otimização . . . . .	184
32.5 Exercícios . . . . .	185
<b>33 Função Modular</b>	<b>186</b>
33.1 Definições Básicas . . . . .	186
33.2 A função modular . . . . .	187
33.3 Uma interpretação geométrica do módulo . . . . .	189
33.4 Exercícios . . . . .	193
<b>34 Polinômios e Equações Polinomiais</b>	<b>194</b>
34.1 Definições Básicas . . . . .	194
34.1.4 Relação entre coeficientes e raízes de polinômios . . . . .	195
34.2 Divisão de polinômios . . . . .	198
34.2.1 Método da Chave . . . . .	198
34.2.5 Método de Briot-Ruffini . . . . .	201
34.2.8 Multiplicidade de uma raiz . . . . .	202
34.2.10 Raízes Racionais . . . . .	203
34.3 Exercícios . . . . .	204
<b>35 Função Exponenciais e Equações Exponenciais</b>	<b>205</b>
35.1 Definições de Potenciação e Radiciação . . . . .	205
35.2 Funções Exponenciais . . . . .	208
35.2.4 Equações Exponenciais . . . . .	211
35.3 Exercícios . . . . .	213
<b>36 Logaritmos</b>	<b>214</b>
36.1 Definição de Logaritmo . . . . .	214
36.2 Propriedades dos Logaritmos . . . . .	214
36.3 Equações Logarítmicas . . . . .	217
36.4 Exercícios . . . . .	218
<b>37 Matrizes</b>	<b>219</b>
37.1 Operações com matrizes . . . . .	220
37.2 A relação de matrizes com sistemas lineares . . . . .	222
37.3 Método para encontrar $A^{-1}$ . . . . .	225
37.4 Classificação de sistemas lineares . . . . .	226
37.5 Exercícios . . . . .	230
<b>38 Determinantes</b>	<b>231</b>
38.1 Propriedades dos determinantes $2 \times 2$ . . . . .	231
38.2 Determinante visto como área . . . . .	233
38.3 Determinantes de ordens maiores . . . . .	235
38.4 Propriedades dos determinantes $n \times n$ . . . . .	239
38.4.5 Determinante de Vandermonde . . . . .	244
38.5 Sistemas Lineares e Determinantes . . . . .	245
38.6 Exercícios . . . . .	250

# I

## Programação Estruturada

# 1 | Primeiro programa

## §1.1 Algoritmos

Um **algoritmo** é uma sequência de instruções bem definidas e ordenadas que levam à resolução de um determinado problema. Pode-se dizer que um algoritmo é um conjunto de passos a se seguir para resolver um problema específico.

Vamos dar alguns exemplos de algoritmos simples para ilustrar melhor:

**Exemplo 1.1.1.** Algoritmo para encontrar o maior número em uma lista:

1. Ler a lista de números.
2. Armazenar o primeiro número da lista.
3. Percorrer a lista, comparando cada número com o número armazenado.
4. Se o número atual for maior que armazenado, substituir o número armazenado pelo número atual.
5. Ao chegar no final da lista o número armazenado será o maior da lista.

**Exemplo 1.1.2.** Algoritmo para fazer um bolo de cenoura com cobertura de chocolate:

1. Pré-aqueça o forno a 180°C.
2. Descasque e rale duas cenouras médias e reserve.
3. Em uma tigela grande, misture a farinha, o açúcar, o fermento em pó, o bicarbonato de sódio, a canela e o sal.
4. Adicione os ovos, o óleo e a baunilha à mistura de farinha e misture bem até ficar homogêneo.
5. Adicione as cenouras raladas à massa e misture novamente.
6. Despeje a massa em uma forma untada e enfarinhada.
7. Asse no forno pré-aquecido por 30 a 35 minutos, ou até que um palito inserido no centro do bolo saia limpo.
8. Retire o bolo do forno e deixe esfriar por cerca de 10 minutos na forma.
9. Enquanto o bolo esfria, prepare a cobertura de chocolate: derreta o chocolate em banho-maria ou no microondas, acrescente o creme de leite e misture até obter uma mistura homogênea.
10. Desenforme o bolo e cubra com a cobertura de chocolate.
11. Sirva o bolo de cenoura com cobertura de chocolate decorado com nozes picadas ou raspas de chocolate, se desejar.

Estes são apenas alguns exemplos de algoritmos simples. Algoritmos mais complexos podem



ser construídos a partir de *combinações de instruções simples*. A chave para um bom algoritmo é que ele deve ser preciso, eficiente e fácil de entender e implementar.

## §1.2 Linguagens de programação

As **linguagens de programação** são utilizadas para escrever códigos que os computadores conseguem entender e executar. São utilizadas para criar programas, aplicativos e sistemas. Existem diversas linguagens de programação, cada uma com suas características e finalidades específicas. Alguns exemplos de linguagens de programação são:

- **Python:** linguagem de programação *interpretada*, fácil de aprender e utilizada em diversas áreas, desde ciência de dados até desenvolvimento web;

```
1 mensagem = "Olá, mundo!"
2 print(mensagem)
```

- **Java:** linguagem de programação orientada a objetos, utilizada no desenvolvimento de aplicativos móveis, sistemas bancários, jogos, entre outros;

```
1 public class OlaMundo {
2     public static void main(String[] args) {
3         String mensagem = "Ola, mundo!";
4         System.out.println(mensagem);
5     }
6 }
```

- **C:** linguagem de programação de baixo nível, utilizada no desenvolvimento de sistemas operacionais, compiladores, dispositivos embarcados, entre outros;

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Ola, mundo!\n");
5     return 0;
6 }
```

- **JavaScript:** linguagem de programação interpretada, utilizada no desenvolvimento de páginas web interativas e aplicativos móveis;

```
1 let mensagem = "Ola, mundo!";
2 console.log(mensagem);
```

- **Ruby:** linguagem de programação de alto nível, utilizada em desenvolvimento web, automação e testes;

```
1 mensagem = "Ola, mundo!"
2 puts mensagem
```

- **PHP:** linguagem de programação interpretada, utilizada no desenvolvimento de sites dinâmicos e sistemas de gerenciamento de conteúdo.

```
1 <?php
2     $mensagem = "Ola, mundo!";
3     echo $mensagem;
4 ?>
```

Cada linguagem de programação possui sua própria sintaxe, conjunto de regras e palavras-chave que são utilizadas para escrever códigos. A escolha da linguagem de programação a ser utilizada depende da finalidade do projeto e das habilidades e preferências do programador.

### Classificação de linguagens de programação

Linguagens de programação podem ser classificadas de diversas maneiras:

- **Nível de Abstração:** as linguagens de programação podem ser classificadas em linguagens de baixo nível, que são mais próximas da linguagem de máquina, e linguagens de alto nível, que são mais próximas da linguagem humana.
- **Paradigma de Programação:** as linguagens de programação podem ser classificadas com base no paradigma de programação que elas usam. Por exemplo, as linguagens de programação orientadas a objetos (OO) enfatizam a criação de objetos que encapsulam dados e comportamentos, enquanto as linguagens de programação funcional se concentram em funções puras e imutabilidade.
- **Tipagem:** as linguagens de programação podem ser classificadas como tipadas ou não tipadas. As linguagens de programação tipadas exigem que todas as variáveis tenham um tipo de dados definido, enquanto as linguagens de programação não tipadas permitem que os tipos de dados das variáveis sejam inferidos pelo compilador ou interpretador.
- **Execução:** as linguagens de programação podem ser classificadas como compiladas ou interpretadas. As linguagens compiladas, como C, C++, Java e Swift, são traduzidas para linguagem de máquina antes da execução, enquanto as linguagens interpretadas, como Python, Ruby e JavaScript, são traduzidas em tempo de execução. Embora essa distinção seja importante, muitas linguagens modernas usam técnicas que combinam as duas abordagens, como o Java, que é uma linguagem compilada, mas usa a máquina virtual Java para interpretar o bytecode gerado pelo compilador.

Essas são apenas algumas das principais classificações de linguagens de programação. Cada classificação tem seus próprios subgrupos, e muitas linguagens de programação se enquadram em várias categorias.

## §1.3 Python + Colab/Jupyter

**Python** é uma das linguagens de programação mais populares do mundo, e isso se deve em grande parte à sua simplicidade, eficiência e versatilidade. Essas características tornam o Python uma excelente escolha para iniciantes em programação, mas também o tornam uma opção poderosa para profissionais experientes. Além disso, o Python tem uma grande comunidade ativa de desenvolvedores, o que significa que existem muitos recursos disponíveis, como bibliotecas, frameworks e módulos, que tornam a programação mais fácil e eficiente. Neste curso, utilizaremos o Python como linguagem principal para ensinar os conceitos fundamentais de programação, como tipos de dados, variáveis, expressões, condicionais e repetições. Além disso, o Python é amplamente utilizado em áreas como ciência de dados, inteligência artificial e aprendizado de

máquina, o que significa que o conhecimento adquirido neste curso pode ser facilmente aplicado em outros contextos.

Nesse curso, aprenderemos Python utilizando o **Google Colab** como ambiente de desenvolvimento. O Colab é um serviço do Google para criar e executar notebooks **Jupyter**, é uma ferramenta permite criar e compartilhar documentos que contêm código, visualizações e texto explicativo. Ele é fácil de usar e tem suporte para várias linguagens de programação, incluindo Python. Uma das principais vantagens do Jupyter é a sua flexibilidade, pois permite que o aluno execute o código em blocos, o que facilita o aprendizado e o entendimento de cada etapa do processo de programação. Além disso, o Jupyter é gratuito e pode ser acessado online ou instalado localmente. No entanto, caso prefira, o aluno pode instalar suas próprias ferramentas necessárias, como um compilador e um editor de texto, para desenvolver seus códigos.

An example: visualizing data in the notebook ✨

Below is an example of a code cell. We'll visualize some simple data using two popular packages in Python. We'll use [NumPy](#) to create some random data, and [Matplotlib](#) to visualize it.

Note how the code and the results of running the code are bundled together.

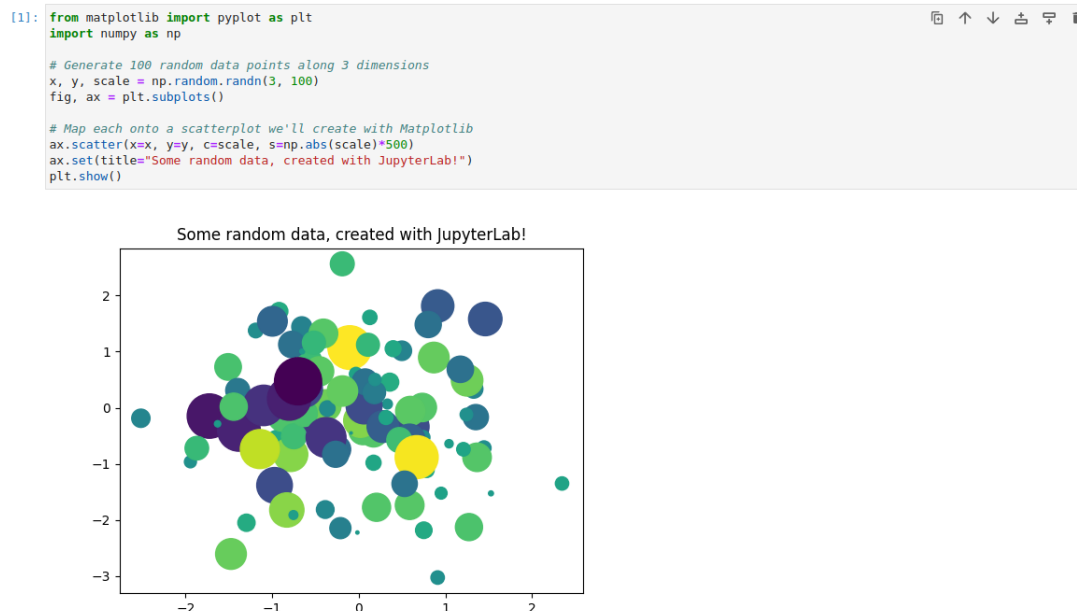


Figura 1.1: Código Python com descrição textual e visualização de dados rodando em um notebook Jupyter.

Além de permitir a execução de códigos, os notebooks Jupyter também permitem a visualização de dados na forma de imagens, planilhas (datasets) e plotagens. Isso é especialmente útil para o aprendizado de análise de dados e criação de visualizações, pois o aluno pode explorar e entender melhor os dados utilizados em cada etapa do processo. Com isso, o Jupyter se torna uma ferramenta essencial para o desenvolvimento de projetos e trabalhos relacionados à programação e análise de dados.

## §1.4 Primeiro programa

Para escrever seu primeiro programa em Python, você pode utilizar o Jupyter Notebook, que já está disponível para você:

1. Abra o Jupyter Notebook.
2. Crie um novo notebook.

3. Na primeira célula, escreva o seguinte código:

```
print("Olá, mundo!")
```

4. Rode a célula utilizando o atalho **Shift + Enter**.
5. Pronto! O seu primeiro programa em Python foi executado com sucesso.

Esse programa é muito simples, mas já é um bom começo para quem está iniciando na programação. Ele utiliza a função **print** para exibir a mensagem "Ola, mundo!" na tela. Essa é uma tradição entre os programadores e serve para testar se o ambiente de programação está configurado corretamente.

# 2

## Fundamentos de programação

Os fundamentos da programação são essenciais para aqueles que estão iniciando seus estudos em desenvolvimento de software. Para criar programas em qualquer linguagem de programação, é necessário compreender conceitos básicos, como variáveis, constantes, tipos de dados, operadores aritméticos e de atribuição, entrada de dados e conversão de tipos. Esses conceitos são cruciais para a compreensão da lógica da programação. Com o conhecimento desses fundamentos, é possível avançar em estudos mais avançados em programação e desenvolver soluções mais complexas e sofisticadas.

### §2.1 Variáveis

Variáveis são utilizadas para armazenar valores na memória do computador. As variáveis possuem um nome que serve para identificá-las, e um valor que pode ser alterado ao longo da execução do programa. O valor armazenado na variável pode ser de diferentes tipos, como números, texto, listas, entre outros. As variáveis são essenciais para a construção de algoritmos e programas, pois permitem que os valores sejam armazenados e manipulados de maneira dinâmica durante a execução do programa. Para declarar uma variável em Python, basta escolher um nome para ela e atribuir um valor, utilizando o sinal de igual `=`. Por exemplo:

```
1 | idade = 30
2 | nome = "João"
```

Dessa forma, a variável `idade` armazena o valor `30`, e a variável `nome` armazena a string (sequência de caracteres) `Joao`.

#### Nomes válidos para variáveis

Em Python, os nomes de variáveis devem seguir algumas regras para serem válidos. Eles podem consistir em letras (maiúsculas e minúsculas), dígitos e sublinhados e devem começar com uma letra ou sublinhado. É importante lembrar que nomes de variáveis são sensíveis a maiúsculas e minúsculas, ou seja, `variavel` e `Variavel` são considerados nomes de variáveis diferentes. Além disso, existem algumas palavras reservadas em Python que não podem ser usadas como nomes de variáveis, como `if`, `while`, `for`, `and`, entre outros. É uma boa prática usar nomes de variáveis descritivos e significativos para tornar o código mais legível e fácil de entender.

Nomes válidos:

- `idade`
- `_nome_completo`
- `total_de_vendas`
- `primeiraNota`

Nomes inválidos:

- `2a`
- `saldo da conta`
- `@usuario`
- `nome-completo`

#### §2.1.1 Tipos de dados

Em Python, os tipos de dados são dinâmicos, o que significa que o interpretador infere o tipo da variável a partir do valor atribuído a ela. Dessa forma, não é necessário declarar explicitamente o

tipo das variáveis, como em outras linguagens de programação.

Os tipos de dados disponíveis em Python incluem:

- **int**: Números inteiros, como `42`.
- **float**: Números de ponto flutuante, como `3.14`.
- **bool**: Valores booleanos, `True` ou `False`.
- **str**: Sequências de caracteres, como `"Hello, world!"`.
- **list**: Listas, como `[1, 2, 3]`.
- **tuple**: Tuplas, como `(1, 2, 3)`.
- **set**: Conjuntos, como `{1, 2, 3}`.
- **dict**: Dicionários, como `{"fruta": "maçã", "cor": "vermelha"}`.

**Exemplo 2.1.2.** Variáveis de diferentes tipos:

```
1  # Exemplos de variáveis em Python
2
3  x = 42  # x é um inteiro
4  y = 3.14 # y é um número de ponto flutuante
5  z = True # z é um booleano
6  s = "Olá, mundo!" # s é uma string
7  lista = [1, 2, 3, 4] # lista é uma lista de inteiros
8  tupla = ("a", "b", "c") # tupla é uma tupla de strings
9  conjunto = {1, 2, 3, 4, 5} # conjunto é um conjunto de inteiros
10 dicionario = {"nome": "João", "idade": 30, "cidade": "São Paulo"}
    # dicionário é um dicionário de strings e inteiro
```

Como mencionado, o tipo de cada variável é inferido pelo interpretador do Python automaticamente. Isso significa que o programador não precisa declarar explicitamente o tipo da variável, tornando o processo de programação mais simples e eficiente. Além disso, essa característica permite que as variáveis sejam reatribuídas a um valor de um tipo diferente sem precisar alterar a declaração da variável.

### Estruturas homogêneas e heterogêneas

As estruturas de dados **list**, **tuple**, **set** e **dict** em Python podem armazenar objetos de qualquer tipo, incluindo outros objetos compostos como listas, tuplas e dicionários. Essas estruturas de dados podem ser heterogêneas, ou seja, podem armazenar diferentes tipos de objetos, ou homogêneas, ou seja, podem armazenar objetos de apenas um tipo.

Por exemplo, uma **lista** em Python pode conter uma combinação de tipos de objetos, como inteiros, strings e outros tipos compostos:

```
minha_lista = [1, "dois", 3.0, [4, 5, 6]]
```

Já uma **tupla** em Python pode conter uma combinação de tipos de objetos ou objetos de um único tipo:

```
minha_tupla = (1, "dois", 3.0, [4, 5, 6])
```

Um **conjunto (set)** em Python só pode conter objetos imutáveis de tipos únicos, como inteiros ou strings:

```
meu_set = {1, 2, 3, 4, 5}
```

Um **dicionário** em Python armazena pares de chave-valor, onde as chaves podem ser de qualquer tipo imutável, como inteiros, strings ou tuplas, e os valores podem ser qualquer tipo de objeto:

```
meu_dict = {'chave1': 100, 'chave2': "valor2", 3: [1, 2, 3]}
```

A capacidade de armazenar objetos de diferentes tipos em uma única estrutura de dados é uma das características poderosas de Python, permitindo que os programadores criem estruturas de dados flexíveis e expressivas.

### §2.1.3 Constantes

Em Python, não existe uma declaração explícita de constantes como em outras linguagens de programação, mas é possível definir valores constantes através do uso de variáveis nomeadas em maiúsculas. Por convenção, esses nomes são escritos em letras maiúsculas, e não devem ser modificados durante a execução do programa. O uso de constantes ajuda a tornar o código mais legível, e ajuda a evitar erros causados por mudanças inesperadas nos valores dessas variáveis. Por exemplo, podemos definir uma constante para o valor de pi da seguinte forma:

```
PI = 3.14159
```

Dessa forma, se algum trecho do código precisar utilizar o valor de pi, podemos utilizar a variável PI em vez de escrever o valor literal diretamente no código, o que torna o código mais legível e fácil de manter.

## §2.2 Operadores aritméticos

Os operadores aritméticos permitem realizar operações matemáticas simples. Em Python são eles:

- Adição **+**: soma dois valores;
- Subtração **-**: subtrai o segundo valor do primeiro;
- Multiplicação **\***: multiplica dois valores;
- Divisão **/**: divide o primeiro valor pelo segundo;
- Divisão inteira **//**: divide o primeiro valor pelo segundo e retorna apenas a parte inteira do resultado, desprezando os caracteres após a virgula;
- Resto da divisão **%**: retorna o resto da divisão do primeiro valor pelo segundo;
- Exponenciação **\*\***: eleva o primeiro valor à potência do segundo.

**Exemplo 2.2.1.** Utilizando operadores aritméticos em Python:

```
1 | # Declaração de variáveis
2 | a = 10
3 | b = 3
```

```
4
5 # Operações aritméticas
6 soma = a + b
7 subtracao = a - b
8 multiplicacao = a * b
9 divisao = a / b
10 divisao_inteira = a // b
11 resto_divisao = a % b
12 exponenciacao = a ** b
13
14 # Impressão dos resultados
15 print("Soma:", soma) # Soma: 13
16 print("Subtração:", subtracao) # Subtração: 7
17 print("Multiplicação:", multiplicacao) # Multiplicação: 30
18 print("Divisão:", divisao) # Divisão: 3.3333333333333335
19 print("Divisão inteira:", divisao_inteira) # Divisão inteira: 3
20 print("Resto da divisão:", resto_divisao) # Resto da divisão: 1
21 print("Exponenciação:", exponenciacao) # Exponenciação: 1000
```

Este código declara as variáveis **a** e **b** com os valores **10** e **3**, respectivamente. Em seguida, realiza as operações aritméticas entre elas e armazena os resultados em variáveis distintas. Por fim, imprime os resultados das operações na tela.

### Operações matemáticas avançadas

Além das operações aritméticas básicas é possível realizar cálculos matemáticos mais complexos utilizando bibliotecas como a **math** e o **numpy**. A biblioteca **math** contém funções matemáticas padrão, como exponenciação, logaritmo, seno e cosseno, entre outras. O **numpy** é uma poderosa ferramenta para trabalhar com matrizes e vetores, e também possui funções para as principais operações matemáticas, sendo muito utilizada em áreas como a análise de dados e a inteligência artificial. Com o **numpy**, é possível realizar operações aritméticas com matrizes, calcular estatísticas, realizar operações de álgebra linear, entre outras funcionalidades. Veja um exemplo simples de utilização do **numpy**:

```
1 import numpy as np
2
3 a = np.array([1, 2, 3])
4 b = np.array([4, 5, 6])
5
6 c = a + b
7
8 print(c)
```

Na primeira linha do código, a biblioteca NumPy é importada e renomeada como **np**. Em seguida, as variáveis **a** e **b** são criadas como arrays do numpy com os valores **[1, 2, 3]** e **[4, 5, 6]**, respectivamente. A variável **c** é criada como a soma de **a** e **b**. Por fim, o resultado é impresso no console usando a função **print()**, que exibe o valor de **c**. A saída do programa seria um novo array contendo os elementos **[5, 7, 9]**, que são o resultado da soma dos elementos correspondentes de **a** e **b**. O exemplo ilustra o uso básico do **numpy** para realizar operações matemáticas com arrays. Mais funções e recursos da biblioteca serão explorados posteriormente.



### §2.2.2 Operadores aplicados em strings

Em Python, é possível utilizar operadores aritméticos em strings para realizar operações como concatenação e repetição. O operador de concatenação é o símbolo de adição `+`, que permite juntar duas ou mais strings em uma só. Já o operador de repetição é o asterisco `*`, que possibilita repetir uma string múltiplas vezes:

**Exemplo 2.2.3.** Operadores concatenação `+` e repetição `*`:

```
1  # Define duas strings
2  s1 = "Olá"
3  s2 = "Mundo"
4
5  # Concatena as duas strings utilizando o operador de soma
6  s3 = s1 + s2
7  print(s3) # Saída esperada: "OláMundo"
8
9  # Repete a string 's1' três vezes utilizando o operador de
   multiplicação
10 s4 = s1 * 3
11 print(s4) # Saída esperada: "OláOláOlá"
```

Estes operadores também podem ser utilizados em algumas situações com outros tipos de variáveis como listas, tais usos serão abordados posteriormente.

### §2.3 Operadores de atribuição

Os operadores de atribuição em Python são utilizados para armazenar um valor em uma variável. O operador de atribuição mais simples, que já foi visto, é o sinal de igual `=`, que é utilizado para atribuir um valor a uma variável. Além do operador de atribuição padrão, Python também oferece operadores compostos de atribuição que permitem que a variável receba um valor e faça uma operação aritmética ao mesmo tempo. Por exemplo, o operador `+=` atribui à variável o valor da soma dela com outro valor. Outros operadores compostos incluem `-=`, `*=`, `/=`, `//=` e `%=`.

**Exemplo 2.3.1.** Uso de operadores de atribuição:

```
1  x = 5 # atribui o valor 5 à variável x
2  y = 2
3  x += y # equivale a x = x + y; o valor de x passa a ser 7
4  y *= 3 # equivale a y = y * 3; o valor de y passa a ser 6
5  z = 10
6  z %= 3 # equivale a z = z % 3; o valor de z passa a ser 1
```

Observe que os operadores compostos de atribuição utilizados foram `+=`, `*=` e `%=`, que simplificam a escrita de expressões como `x = x + y` ou `y = y * 3`. O uso desses operadores simplifica o uso e pode tornar o código mais legível e conciso.

## §2.4 Conversão de tipos

A conversão de tipos é o processo de transformar um valor de um tipo de dado em outro tipo de dado. Em Python, é possível converter os tipos de dados básicos, como `int`, `float` e `str`, usando funções de conversão incorporadas, como `int()`, `float()` e `str()`. A conversão de tipos é útil quando precisamos trabalhar com diferentes tipos de dados em uma operação ou quando precisamos apresentar os dados em um formato específico. Por exemplo, se tivermos uma variável que armazena um valor numérico como uma string, podemos convertê-la em um valor inteiro usando a função `int()` para realizar cálculos matemáticos com ela.

### Exemplo 2.4.1. Conversão de tipos:

```
1  # Define duas variáveis iniciais de tipos diferentes
2  a = 2
3  b = "3"
4
5  # Imprime os tipos das variáveis iniciais
6  print("a é do tipo", type(a))
7  print("b é do tipo", type(b))
8
9  # Converte a variável 'b' de string para inteiro e soma com a
   # variável 'a'
10 c = a + int(b)
11
12 # Imprime o resultado da soma
13 print("O resultado da soma é", c)
14
15 # Converte a variável 'a' de inteiro para float e divide por 'c'
16 d = float(a) / c
17
18 # Imprime o resultado da divisão
19 print("O resultado da divisão é", d)
```

Neste código, primeiro definimos duas variáveis, **a** e **b**, com tipos diferentes: **a** é um inteiro, e **b** é uma string contendo o número 3. Em seguida, convertemos a variável **b** de string para inteiro usando a função `int()`, e a adicionamos à variável **a**, que continua sendo um inteiro.

Depois disso, convertemos a variável **a** de inteiro para float usando a função `float()`, e a dividimos pelo resultado anterior, armazenado em **c**. Por fim, imprimimos os resultados das operações e os tipos das variáveis usadas em cada etapa.

## §2.5 Entrada de dados

A entrada de dados é uma tarefa comum na programação, pois é preciso receber informações do usuário para executar operações. Em Python, podemos utilizar a função `input()` para obter entrada de dados do usuário por meio do teclado. O resultado da função `input()` é sempre uma string.

Por exemplo, suponha que queremos pedir ao usuário para inserir seu nome e, em seguida, exibir uma mensagem de boas-vindas. Podemos fazer isso da seguinte maneira:

**Exemplo 2.5.1.** Entrada simples:

```
1 # Pede que o usuário digite um valor para a variável "nome"
2 nome = input("Digite seu nome: ")
3
4 # Imprime uma mensagem na tela utilizando a variável "nome"
5 print("Bem-vindo,", nome)
```

Nesse exemplo, a função `input()` é utilizada para solicitar que o usuário digite um valor para a variável `nome`. Em seguida, a mensagem "Bem vindo, " e a variável `nome` são passadas para a função `print()`.

É importante lembrar que o valor retornado pela função `input()` é sempre uma string. Se quisermos realizar operações aritméticas ou comparar valores numéricos, devemos converter a entrada do usuário em um tipo numérico apropriado usando as funções `int()` ou `float()`, por exemplo.

**Exemplo 2.5.2.** Operações com resultados de entradas:

```
1 # Pede que o usuario digite um valor para a variavel "idade"
2 idade = int(input("Digite sua idade: "))
3
4 # Calcula o ano em que o usuario fara 100 anos
5 ano = 2023 + (100 - idade)
6
7 # Imprime uma mensagem na tela utilizando a variavel "ano"
8 print("Voce fara 100 anos em " + str(ano))
```

Nesse exemplo, a função `input()` é utilizada para solicitar que o usuário digite um valor para a variável `idade`. Como esse valor será utilizado em operações matemáticas, é necessário converter o tipo de dado para inteiro utilizando a função `int()`. Em seguida, o ano em que o usuário fará 100 anos é calculado com base na idade fornecida e armazenado na variável `ano`. Por fim, a mensagem "Você fará 100 anos em " concatenada com o valor de ano convertido para string com a função `str()` é impressa na tela com a função `print()`.

## §2.6 Formatação de strings

Em Python, a formatação de strings é uma técnica que permite inserir valores de variáveis em uma string de forma dinâmica. Existem duas maneiras comuns de formatar strings em Python: usando o método `format()` e usando as f-strings (ou formatted string literals).

Usando o método `format()`, podemos inserir valores de variáveis em uma string, indicando onde queremos que eles apareçam usando chaves `{}`. Por exemplo:

**Exemplo 2.6.1.** Formatação de strings com o método `format()`

```
1 # Pergunta ao usuario seu nome e idade
2 nome = input("Digite seu nome: ")
3 idade = int(input("Digite sua idade: "))
4
```

```
5 # Imprime uma string formatada utilizando o metodo format() com
   o nome e idade informados
6 print("Ola, meu nome e {} e eu tenho {} anos.".format(nome,
   idade))
```

Este código retorna a string com o valor da variável **nome** inserido na posição do primeiro {} e o valor da variável **idade** inserido na posição do segundo {}.

Outra maneira mais recente e mais simples de formatar strings em Python é usando f-strings. F-strings permitem que as variáveis sejam inseridas diretamente na string, precedidas pelo caractere f:

**Exemplo 2.6.2.** Formatação de strings com uma f-string:

```
1 # Pergunta ao usuário seu nome e idade
2 nome = input("Digite seu nome: ")
3 idade = int(input("Digite sua idade: "))
4
5 # Imprime uma string formatada utilizando uma f-string com o nome
   e idade informados
6 print(f"Olá, meu nome é {nome} e eu tenho {idade} anos.")
```

O resultado seria o mesmo que no exemplo anterior.

Usando tanto o método `format()` quanto as f-strings, podemos formatar strings de diversas maneiras, incluindo especificar o tipo de dado, a precisão de números decimais e outras opções de formatação. Para especificar o tipo de dado, pode-se utilizar dentro do {} os códigos de tipo **d** para inteiro, **f** para ponto flutuante e **s** para string. Já para especificar a precisão de um número de ponto flutuante, pode-se utilizar a sintaxe `:.Nf`, onde **N** é o número de casas decimais desejado.

**Exemplo 2.6.3.** Formatando um ponto flutuante com 3 casas decimais:

```
1 num = 3.141592
2 # Formatando com o format():
3 print("O valor de pi é aproximadamente {:.3f}".format(num))
4 # Ou com f-string:
5 print(f"O valor de pi é aproximadamente {num:.3f}")
```

A saída desse código seria:

**O valor de pi é aproximadamente 3.142**

Note que a sintaxe `:.3f` foi utilizada para especificar a precisão do número de ponto flutuante.

# 3 | Estruturas condicionais

Estruturas condicionais são utilizadas na programação para criar lógicas que permitem a execução de diferentes blocos de código dependendo de certas condições. Elas são fundamentais para a tomada de decisões em um programa, permitindo que o software possa escolher entre diferentes caminhos a serem seguidos, de acordo com as entradas e variáveis que ele recebe. As estruturas condicionais são amplamente utilizadas em todas as linguagens de programação, e o conhecimento de como utilizá-las é fundamental para a escrita de programas funcionais e eficientes. Em geral, as estruturas condicionais são compostas por uma ou mais condições, que são testadas por meio de operadores relacionais e lógicos, e por um ou mais blocos de código a serem executados em caso de sucesso ou falha da condição testada.

## §3.1 Operadores relacionais e lógicos

Os operadores relacionais em Python são utilizados para comparar dois valores e retornar um valor booleano, que pode ser **True** ou **False**. Os operadores relacionais são:

- `==` (igual a)
- `!=` (diferente de)
- `<` (menor que)
- `<=` (menor ou igual a)
- `>` (maior que)
- `>=` (maior ou igual a)

Já os operadores lógicos são utilizados para combinar valores booleanos e retornar um novo valor booleano. Os operadores lógicos em Python são:

- **and** (e lógico)
- **or** (ou lógico)
- **not** (negação lógica)

O operador **and** retorna **True** se ambos os valores comparados forem **True**, e **False** caso contrário. Já o operador **or** retorna **True** se pelo menos um dos valores comparados for **True**, e **False** caso contrário. O operador **not** inverte o valor booleano de um operando.

**Exemplo 3.1.1.** Se queremos verificar se uma pessoa é maior de idade e tem carteira de motorista, podemos escrever:

```
| idade >= 18 and possui_cnh == True
```

Isso retornará **True** se a pessoa atender a ambas as condições, e **False** caso contrário. Também podemos usar parênteses para controlar a ordem das operações. Por exemplo:

```
(idade >= 18 and possui_cnh == True) or (idade >= 21 and
    possui_carro == True)
```

Isso retornará **True** se a pessoa tiver mais de 18 anos e CNH, ou mais de 21 anos e carro, e **False** caso contrário.

Os operadores relacionais e lógicos são utilizados para criar expressões booleanas e condições em estruturas de decisão e repetição.

### Operadores binários

Além dos operadores relacionais e lógicos, existem também os operadores binários em Python. Esses operadores são utilizados para realizar operações entre bits. Os principais operadores binários em Python são:

- **&** (AND): Retorna o resultado da operação "E" bit a bit (bitwise) entre dois operandos.
- **|** (OR): Retorna o resultado da operação "OU" bit a bit (bitwise) entre dois operandos.
- **^** (XOR): Retorna o resultado da operação "OU Exclusivo" bit a bit (bitwise) entre dois operandos.
- **~** (NOT): Retorna o resultado da operação de negação bit a bit (bitwise) do operando.
- **>>** (Right Shift): Desloca os bits do operando para a direita em uma quantidade especificada.
- **<<** (Left Shift): Desloca os bits do operando para a esquerda em uma quantidade especificada.

Os operadores **&**, **|**, **^** e **~** também podem ser aplicados em arrays booleanas do NumPy, permitindo a realização de operações element-wise entre esses arrays. Por exemplo, podemos utilizar o operador **&** para comparar dois arrays booleanos e retornar um terceiro array com o resultado da operação AND elemento a elemento.

## §3.2 Tabela verdade

A **tabela verdade** é usada para determinar a lógica de um circuito ou expressão lógica.

**Exemplo 3.2.1.** A tabela verdade para o operador lógico **and** é a seguinte:

A	B	A and B
True	True	True
False	True	False
True	False	False
False	False	False

Tabela 3.1: Tabela verdade do operador "E"(AND)

Nessa tabela, **A** e **B** são os valores de entrada do operador lógico **and**, e **A and B** é o resultado da operação. Os valores **True** e **False** representam, respectivamente, verdadeiro e falso.

A tabela no exemplo mostra que, quando ambos **A** e **B** são verdadeiros, o resultado da operação

**and** é verdadeiro. Em todas as outras combinações, o resultado é falso. Essa informação pode ser usada para construir expressões lógicas mais complexas.

**Exemplo 3.2.2.** Vamos ver a tabela verdade da expressão na sessão anterior:

```
(idade >= 18 and possui_cnh) or (idade >= 21 and possui_carro)
```

idade	possui_cnh	possui_carro	Resultado
<18	False	False	False
<18	False	True	False
<18	True	False	False
<18	True	True	False
>=18 and <21	False	False	False
>=18 and <21	False	True	False
>=18 and <21	True	False	True
>=18 and <21	True	True	True
>=21	False	False	False
>=21	False	True	True
>=21	True	False	True
>=21	True	True	True

Tabela 3.2: Tabela da Verdade da expressão.

### §3.2.3 Estrutura IF/ELSE

A estrutura if/else é uma das mais importantes na programação, pois permite ao programador controlar o fluxo de execução do código, realizando ações diferentes de acordo com uma condição específica. Isso permite que o programa possa tomar decisões, como realizar uma ação ou não, imprimir uma mensagem de erro ou de sucesso, entre outras possibilidades. A estrutura if/else é essencial para que o programa possa se adaptar a diferentes situações, tornando-o mais dinâmico e flexível. Sem essa estrutura, o programa seria limitado a realizar sempre as mesmas ações, sem poder responder a condições específicas.

Em Python a sintaxe básica é:

```
1  if condição:
2      # faça algo se a condição for verdadeira
3  else:
4      # faça algo se a condição for falsa
```

**Exemplo 3.2.4.** Programa que verifica se um número é par ou ímpar e imprime na tela uma mensagem correspondente:

```
1  numero = 10
2
3  if numero % 2 == 0:
4      print("0 numero e par")
5  else:
6      print("0 numero e impar")
```

### Indentação de código

Indentação é um conceito fundamental em programação, que consiste na organização do código por meio de espaços em branco. A indentação é importante porque torna o código mais legível e facilita a compreensão da estrutura do programa.

No Python, a indentação é obrigatória para indicar a estrutura do código, em vez de utilizar chaves ou outras formas de delimitadores de blocos. Isso significa que a indentação é usada para definir a estrutura de condicionais, loops, funções e classes.

A indentação pode ser realizada por meio de espaços ou de tabulações (tecla tab do teclado) para indicar níveis de hierarquia no código. No entanto, a escolha de um padrão específico pode variar de acordo com a preferência pessoal ou com as diretrizes de codificação de uma equipe ou organização.

No Python a PEP 8 é a guia de estilo de código mais utilizada e ela recomenda o uso de 4 espaços para cada nível de indentação, em vez de tabulações. Isso se deve ao fato de que a tabulação pode variar de tamanho dependendo das configurações de cada editor, o que pode levar a inconsistências no código e dificultar a leitura e a manutenção do mesmo. O uso de espaços ao invés de tabulações garante que a indentação seja exibida corretamente em diferentes plataformas e editores de texto.

Quando existe mais de uma condição a ser verificada podemos utilizar o comando **elif** (else if) formando a estrutura if/elif/else. A sintaxe básica é a seguinte:

```
1  if condição1:
2      # faça algo se a condição 1 for verdadeira
3  elif condição2:
4      # faça algo se a condição 1 for falsa e a condição 2 for
      verdadeira
5  else:
6      # faça algo se todas as condições anteriores forem falsas
```

**Exemplo 3.2.5.** Programa que verifica se um número é positivo, negativo ou zero e imprime na tela uma mensagem correspondente:

```
1  numero = -5
2
3  if numero > 0:
4      print("O número é positivo")
5  elif numero < 0:
6      print("O número é negativo")
7  else:
8      print("O número é zero")
```

No exemplo acima, a primeira condição verifica se o número é maior que zero. Se a condição for verdadeira, o programa imprime na tela a mensagem "O número é positivo". Caso contrário, o programa passa para a próxima condição, que verifica se o número é menor que zero. Se essa condição for verdadeira, o programa imprime na tela a mensagem "O número é negativo". Caso contrário, ou seja, se todas as condições anteriores forem falsas, o programa imprime na tela a mensagem "O número é zero".



**Exemplo 3.2.6.** O comando `elif` pode ser reutilizado quantas vezes necessário para cobrir todas as condições que se deseja avaliar:

```
1  nota = float(input("Digite a nota do aluno: "))
2
3  if nota >= 9:
4      print("A")
5  elif nota >= 8:
6      print("B")
7  elif nota >= 7:
8      print("C")
9  elif nota >= 6:
10     print("D")
11 else:
12     print("F")
```

Nesse exemplo, a estrutura `elif` é usada várias vezes para verificar a nota do aluno e atribuir uma letra correspondente. O uso do `elif` permite que a verificação seja feita em etapas, sem precisar repetir várias vezes a mesma condição. Além disso, é possível adicionar quantos `elif` forem necessários para cobrir todas as faixas de notas desejadas.

Isso permite que o código seja mais conciso e organizado, evitando a necessidade de aninhar vários blocos `if` dentro de outros, por exemplo. Além disso, o uso do `elif` torna o código mais fácil de ser lido e entendido por outros programadores, uma vez que a lógica do código fica mais clara e objetiva.

### Estrutura `match/case`

Em situações onde uma grande quantidade de condições precisam ser avaliadas várias linguagens de programação possuem a estrutura `switch/case` que permite avaliar múltiplas condições em um único bloco de código. Até a versão 3.9 do Python, não havia uma estrutura `switch/case` nativa na linguagem. Em vez disso, a recomendação era usar uma sequência de `if/elif/else` para avaliar várias condições. No entanto, essa abordagem pode tornar o código menos legível e mais complexo à medida que o número de condições aumenta.

Com a versão 3.10 do Python, foi introduzida uma nova estrutura chamada `match/case`. Ela funciona de maneira semelhante ao `switch/case` em outras linguagens de programação, permitindo avaliar várias condições em um único bloco de código. A sintaxe básica é a seguinte:

```
1  match variavel:
2      case valor_1:
3          # código a ser executado se a variável for igual a valor_1
4      case valor_2:
5          # código a ser executado se a variável for igual a valor_2
6      ...
7      case valor_n:
8          # código a ser executado se a variável for igual a valor_n
9      case _:
10         # código a ser executado se a variável não corresponder a
            nenhum dos valores anteriores
```

# 4 | Estruturas de repetição

As estruturas de repetição, também conhecidas como laços ou loops, são fundamentais na programação, permitindo que um conjunto de instruções seja executado repetidamente até que uma condição seja atendida.

Em outras palavras, elas permitem automatizar tarefas que seriam muito trabalhosas, demoradas ou em muitas situações até impossíveis de serem feitas manualmente. Com as estruturas de repetição, podemos executar o mesmo bloco de código várias vezes sem precisar repeti-lo manualmente.

Existem duas formas de estruturas de repetição: o laço `while` e o laço `for`. Cada uma dessas estruturas tem suas próprias peculiaridades e é adequada para diferentes tipos de tarefas. É importante saber como e quando usar cada um deles para escrever um código eficiente e elegante.

## §4.1 Laço `while`

O laço `while` é uma estrutura de repetição em Python que permite executar um bloco de código enquanto uma determinada condição é verdadeira. O bloco de código dentro do laço é repetido continuamente até que a condição se torne falsa.

A sintaxe básica do laço `while` é:

```
1 while condicao:
2     # bloco de código a ser executado enquanto a condição for
    verdadeira
```

O laço `while` é muito útil quando não se sabe quantas vezes o bloco de código precisará ser executado, pois ele executa até que a condição definida seja atingida. Ele também pode ser utilizado para criar um loop infinito, caso a condição nunca seja falsa.

**Exemplo 4.1.1.** Imprimir os números de 1 a 5.

```
1 # Define a variável x como 1
2 x = 1
3
4 # Enquanto x for menor ou igual a 5, repita o bloco de código
    abaixo
5 while x <= 5:
6     # Imprime o valor atual de x
7     print(x)
8     # Incrementa o valor de x em 1
9     x += 1
```

Nesse exemplo, o laço `while` é utilizado para imprimir os números de 1 a 5. A condição definida é que o valor de `x` seja menor ou igual a 5. O bloco de código é repetido enquanto a condição for verdadeira, incrementando o valor de `x` em cada iteração.

**Exemplo 4.1.2.** Autenticação de usuário

```
1  # Solicita ao usuário a digitação da senha
2  senha = input("Digite a senha: ")
3
4  # Enquanto a senha digitada não for igual a "12345", o loop será
   executado
5  while senha != "12345":
6      senha = input("Senha incorreta. Digite novamente: ")
7
8  # Se a senha digitada for igual a "12345", a mensagem "Senha
   correta. Acesso liberado!" será exibida
9  print("Senha correta. Acesso liberado!")
```

Nesse exemplo, o laço **while** é utilizado para solicitar uma senha ao usuário. Enquanto a senha digitada for diferente de "12345", o bloco de código dentro do laço será executado novamente, solicitando que o usuário digite a senha novamente. Quando a senha correta for digitada, o laço é encerrado e a mensagem de "acesso liberado" é exibida.

## §4.2 Laço for

O laço **for** é uma estrutura de repetição que permite iterar sobre um objeto iterável, como uma lista, tupla, dicionário, conjunto ou string, executando o bloco de código para cada elemento.

Sua sintaxe básica é:

```
1  for elemento in iteravel:
2      # código a ser executado
```

Onde **elemento** é uma variável que assume um valor a cada iteração e **iterável** é o objeto que está sendo iterado.

O laço **for** é muito útil em situações em que precisamos executar um bloco de código para cada elemento de uma sequência, como por exemplo:

- Percorrer uma lista e realizar uma operação em cada elemento;
- Iterar sobre as chaves ou valores de um dicionário;
- Gerar uma sequência numérica com a função `range()`.

**Exemplo 4.2.1.** Percorre uma lista de números armazenando os números pares:

```
1 # Define uma lista com os números de 1 a 10
2 numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
3 # Define uma lista vazia para armazenar os números pares
4 numeros_pares = []
5
6 # Itera em todos os valores da lista
7 for n in numeros:
8     # Se o número for par, adiciona a numeros_pares
9     if (n % 2) == 0:
10         numeros_pares.append(n)
11
12 # Imprime o resultado
13 print("Os números pares na lista são: ", numeros_pares)
```

### §4.2.2 Função range

Em outras linguagens de programação, como C e Java, o laço **for** é geralmente utilizado com base em um contador. Ou seja, é necessário especificar um valor inicial, uma condição de parada e um incremento do contador em cada iteração. Já em Python, o laço **for** é utilizado principalmente para percorrer elementos de um objeto iterável, como listas, tuplas e dicionários. Nesse caso, não é necessário especificar um contador ou a condição de parada, já que o próprio Python realiza o controle das iterações. Além disso, o laço **for** do Python é capaz de percorrer elementos de forma mais simples e elegante, tornando o código mais legível e fácil de entender.

Para comportamentos semelhantes ao **for** em outras linguagens de programação, o Python usa a função **range()**, que cria uma sequência de números que pode ser iterada no laço **for**. É importante notar que o **range** não inclui o último número especificado em sua sequência.

A sintaxe básica da função **range** é a seguinte:

```
range(inicio, fim, incremento)
```

- O parâmetro **inicio** é o valor inicial da sequência.
- O parâmetro **fim** é o valor final da sequência (não incluso).
- O parâmetro **incremento** é o valor que será somado a cada iteração.

Os parâmetros **inicio** e **incremento** são opcionais, o **inicio** por padrão tem o valor 0 e o **incremento** por padrão é definido como 1.

Por exemplo, a chamada **range(5)** retorna a sequência [0, 1, 2, 3, 4], e a chamada **range(1, 10, 2)** retorna a sequência [1, 3, 5, 7, 9].

No contexto de um loop **for**, a função **range** é comumente utilizada para definir a quantidade de iterações do loop. Por exemplo:

**Exemplo 4.2.3.** Imprime os números de 1 a 5:

```
1 # Imprime os números de 1 a 5
2 for i in range(1, 6):
3     print(i)
```

Nesse exemplo, o loop executa 5 iterações, e em cada iteração a variável `i` recebe o valor da sequência gerada pela função `range`.

Note que para a sequência ir até o número 5 precisamos passar 6 como `fim` já que o valor dele não é incluído na sequência.

#### §4.2.4 Função `enumerate`

A função `enumerate` é utilizada para iterar sobre uma sequência (por exemplo, uma lista ou uma tupla) juntamente com um contador, que é incrementado a cada iteração. A função retorna um objeto do tipo `enumerate`, que pode ser convertido para uma lista de tuplas contendo o índice e o valor de cada item da sequência.

A sintaxe da função é a seguinte:

```
enumerate(iterable, start=0)
```

- O parâmetro `iterable` é a sequência que se deseja iterar.
- O parâmetro `start` é opcional e indica o valor inicial do contador. O seu valor padrão é 0.

**Exemplo 4.2.5.** Usando a função `enumerate`:

```
1 # Cria uma lista com quatro nomes
2 nomes = ['Ana', 'Carlos', 'João', 'Maria']
3 # Loop for que percorre a lista de nomes utilizando a função
  enumerate para obter o índice e o valor de cada elemento
4 for i, nome in enumerate(nomes):
5     # Imprime uma mensagem com o índice e o nome correspondente
6     print(f'0 índice {i} corresponde ao nome {nome}')
```

Saída:

```
0 índice 0 corresponde ao nome Ana
0 índice 1 corresponde ao nome Carlos
0 índice 2 corresponde ao nome Joao
0 índice 3 corresponde ao nome Maria
```

No exemplo acima, a função `enumerate` é usada para iterar sobre a lista de nomes juntamente com um contador que começa em 0. A cada iteração do loop, o valor da variável `i` é incrementado e recebe o valor do índice atual da lista, enquanto o valor da variável `nome` recebe o valor atual da lista.

#### Atribuição simultânea de mais de uma variável

No Python, é possível fazer atribuições simultâneas de mais de uma variável utilizando a vírgula para separar os valores. Isso significa que podemos atribuir valores a diversas variáveis em uma única linha de código.

```
x, y, z = 1, 2, 3
```

Nesse exemplo, estamos atribuindo o valor 1 para a variável `x`, o valor 2 para a variável `y` e o valor 3 para a variável `z`.

Outro exemplo:

```
a, b = b, a
```

Nesse exemplo, estamos trocando os valores das variáveis `a` e `b` utilizando a atribuição simultânea. Isso é possível porque, antes da atribuição, o valor de `b` é atribuído à variável `a` e o valor de `a` é atribuído à variável `b`.

A função `enumerate` retorna a cada iteração uma tupla contendo dois valores: o índice e o valor correspondente na iteração. No exemplo 4.2.5, `enumerate(nomes)` retorna uma sequência de tuplas (`índice, valor`) para cada elemento na lista `nomes`. Em seguida, essas tuplas são descompactadas na variável `i` e `nome` usando a atribuição simultânea, permitindo que ambos sejam usados no loop `for`.

## §4.3 Comando break

O comando `break` é um controle de fluxo que permite interromper a execução de um loop (`for` ou `while`) antes que ele termine naturalmente. Quando o comando `break` é executado dentro do loop, o fluxo do programa é desviado imediatamente para a próxima instrução depois do loop, ignorando todas as outras iterações que ainda restariam.

Isso pode ser útil quando se quer parar um loop antes de percorrer toda a sequência de elementos. Por exemplo, imagine que se está percorrendo uma lista de números e deseja parar o loop assim que encontrar o primeiro número negativo. Nesse caso, pode-se utilizar o comando `break` para interromper o loop assim que a condição for satisfeita.

**Exemplo 4.3.1.** Encontrando o número negativo:

```
1 # Cria uma lista de números
2 numeros = [1, 2, 3, -4, 5, 6]
3
4 # Itera em todos os números da lista
5 for num in numeros:
6     # Caso o número atual seja negativo, imprime uma mensagem e
        # sai do loop
7     if num < 0:
8         print(f"Encontrou número negativo: {num}")
9         break
10    print(f"Processando número: {num}")
```

Neste exemplo, o loop percorre a lista de números e, ao encontrar o primeiro número negativo (-4), o comando `break` é executado, interrompendo o loop e exibindo a mensagem "Encontrou número negativo: -4". Note que, como o loop foi interrompido, ele não processou os números 5 e 6 da lista.

**Exemplo 4.3.2.** Utilizando o **break** para sair de um loop infinito:

```
1  # Repete o bloco enquanto o usuário não digitar sair
2  while True:
3      # Lê uma entrada do usuário
4      entrada = input("Digite um valor (ou 'sair' para encerrar): ")
5      # Se o usuário digitar sair, interrompe o loop
6      if entrada == 'sair':
7          break
8      # Imprime a entrada do usuário
9      print(f'Você digitou: {entrada}')
10
11 print('Fim do programa.')
```

Neste exemplo, o programa continuará lendo valores de entrada do usuário até que ele digite "sair". Quando isso acontecer, o **break** é executado, o que interrompe o loop e o programa imprime "Fim do programa."

## §4.4 Comando continue

O comando **continue** é utilizado em laços de repetição para pular a iteração atual e seguir para a próxima. Ou seja, quando o comando **continue** é encontrado dentro de um laço de repetição, as instruções que estão abaixo dele são ignoradas para a iteração atual e o programa segue para a próxima iteração do laço.

O **continue** é muito útil quando se quer pular um determinado elemento da iteração, mas sem sair completamente do laço.

**Exemplo 4.4.1.** Imprimindo somente os números ímpares de uma lista:

```
1  # Cria uma lista de números
2  numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
3
4  # Itera em todos os números
5  for numero in numeros:
6      # Se o número for par, pula a execução do resto do bloco
7      if numero % 2 == 0:
8          continue
9      # Imprime o número
10     print(numero)
```

Nesse exemplo, estamos percorrendo a lista de números e imprimindo somente os números ímpares. Quando encontramos um número par, utilizamos o comando **continue** para pular essa iteração antes do **print** e seguir para a próxima.

# 6 | Sequências

Sequências em Python são um dos principais tipos de dados usados. Uma sequência é uma coleção ordenada de elementos que podem ser de diferentes tipos, como números, strings ou outras sequências. Em Python, existem três tipos principais de sequências: listas, tuplas e strings.

As listas são mutáveis e podem ser alteradas ou atualizadas, enquanto as strings e tuplas são imutáveis e não podem ser alteradas depois de criadas. As sequências em Python permitem armazenar e manipular dados de maneira eficiente, como acessar elementos individuais, fatias de elementos, ordenar, filtrar, etc. Conhecer bem as sequências e como manipula-las é fundamental para quem deseja aprender programação em Python.

## §6.1 Tipos de sequências

### §6.1.1 Listas

As listas em Python são estruturas de dados que permitem armazenar uma coleção de valores em sequência, separados por vírgulas e delimitados por colchetes. Uma das principais vantagens das listas é a sua flexibilidade, já que elas podem ser editadas, fatiadas e expandidas de maneira muito simples.

Além disso, elas aceitam qualquer tipo de elemento, desde números, strings, até outras listas e objetos mais complexos. É possível inclusive ter elementos de tipos diferentes na mesma lista. Essa característica é muito útil em muitas aplicações, como por exemplo quando se quer armazenar informações de diferentes tipos sobre um mesmo objeto.

Outra vantagem das listas é a sua capacidade de expansão dinâmica, ou seja, elas podem ser adicionadas ou removidas elementos a qualquer momento, de forma a atender às necessidades do programa em execução.

#### Exemplo 6.1.2. Exemplos de listas

```
1 # Lista de inteiros
2 lista_inteiros = [1, 2, 3, 4, 5]
3
4 # Lista de strings
5 lista_strings = ['banana', 'maçã', 'abacaxi']
6
7 # Lista com elementos de tipos diferentes
8 lista_diferentes_tipos = ['python', 3.14, True]
```

### §6.1.3 Tuplas

As tuplas em Python são estruturas semelhantes às listas, porém com uma diferença fundamental: são imutáveis, ou seja, uma vez que uma tupla é criada, não é possível alterar o seu conteúdo. As tuplas são definidas com valores separados por vírgula entre parênteses, mas os parênteses podem ser omitidos em algumas situações.

Assim como as listas, as tuplas aceitam qualquer tipo de elemento, inclusive mais de um tipo na mesma tupla. Embora não possam ser modificadas, as tuplas podem ser acessadas e manipuladas de diversas formas, permitindo operações como concatenação, fatiamento e indexação.



As tuplas são úteis em situações onde é necessário garantir que o conteúdo não seja modificado acidentalmente ou intencionalmente, como em funções que retornam múltiplos valores, por exemplo.

**Exemplo 6.1.4.** Exemplos de tuplas

```
1 # tupla de inteiros
2 tupla_inteiros = (1, 2, 3, 4, 5)
3
4 # tupla de strings
5 tupla_strings = ('banana', 'maçã', 'abacaxi')
6
7 # tupla com elementos de tipos diferentes
8 tupla_diferentes_tipos = ('python', 3.14, True)
```

### §6.1.5 Strings

Strings são sequências de caracteres que representam texto e são definidas entre aspas simples ('...') ou duplas ("..."). As strings são imutáveis, ou seja, uma vez criadas, seus valores não podem ser alterados. No entanto, é possível criar uma nova string a partir de uma já existente por meio de operações como concatenação e formatação.

Além disso, as strings suportam uma série de métodos que permitem manipular e analisar seu conteúdo, como busca, substituição, transformação de caixa e separação em substrings.

As strings são amplamente utilizadas em Python para representar palavras, frases, nomes de arquivos, entre outros tipos de texto. A manipulação de strings será discutida em mais detalhes na próxima aula.

**Exemplo 6.1.6.** Exemplos de strings

```
1 # string de letras
2 string_letras = 'abcde'
3
4 # string de números
5 string_numeros = '12345'
6
7 # string com caracteres especiais
8 string_especiais = '$#@!'
```

## §6.2 Operadores aplicados em sequências

Os operadores aplicados em sequências em Python são:

- **Operador de Concatenação (+):** utilizado para unir duas sequências do mesmo tipo em uma única sequência.
- **Operador de Repetição (\*):** utilizado para repetir uma sequência um determinado número de vezes.

- **Operador de Pertencimento (in):** utilizado para verificar se um elemento pertence a uma sequência.

Esses operadores podem ser aplicados em sequências como strings, listas e tuplas em Python. Por exemplo, podemos utilizar o operador de concatenação para unir strings ou listas, o operador de repetição para gerar sequências repetidas, o operador de pertinência para verificar se um elemento está contido em uma lista, e o operador de comparação para comparar o tamanho de duas sequências.

**Exemplo 6.2.1.** Exemplos de uso dos operadores em sequências

```
1 # Operador concatenação
2 lista1 = [1, 2, 3]
3 lista2 = [4, 5, 6]
4 lista_concatenada = lista1 + lista2
5 print(lista_concatenada)
6 # Saída: [1, 2, 3, 4, 5, 6]
7
8 # Operador repetição
9 tupla = (1, 2, 3)
10 tupla_repetida = tupla * 3
11 print(tupla_repetida)
12 # Saída: (1, 2, 3, 1, 2, 3, 1, 2, 3)
13
14 # Operador pertencimento
15 palavra = "python"
16 if "p" in palavra:
17     print("A palavra contém a letra 'p'")
18 # Saída: A palavra contém a letra 'p'
```

## §6.3 Indexação de sequências

A indexação é uma operação muito comum em sequências em Python. Ela é utilizada para acessar um elemento específico dentro de uma sequência, como uma lista, uma tupla ou uma string.

A indexação em Python começa com o número 0 para o primeiro elemento da sequência, e vai incrementando de 1 para cada elemento subsequente. Por exemplo, na lista [1, 2, 3, 4], o elemento 1 tem índice 0, o elemento 2 tem índice 1, e assim por diante.

Para acessar um elemento de uma sequência, é preciso indicar o seu índice dentro dos colchetes que delimitam a sequência. Por exemplo, se quisermos acessar o segundo elemento da lista [1, 2, 3, 4], utilizamos a indexação com o número 1:

```
1 lista = [1, 2, 3, 4]
2 segundo_elemento = lista[1]
3 print(segundo_elemento) # Saída: 2
```

A indexação também pode ser utilizada para acessar uma fatia (slice) da sequência. Para isso, é preciso indicar o índice inicial e final separados por dois pontos dentro dos colchetes. Por exemplo, se quisermos acessar os elementos da lista [1, 2, 3, 4] a partir do segundo elemento até o final, podemos utilizar a seguinte indexação:

```
1 lista = [1, 2, 3, 4]
2 fatiado = lista[1:]
3 print(fatiado) # Saída: [2, 3, 4]
```

Nesse exemplo, utilizamos o índice 1 como início da fatia e deixamos em branco o índice final, indicando que queremos acessar todos os elementos até o final da sequência.

Além de poder acessar os elementos de uma sequência através de índices positivos, também é possível utilizar índices negativos. Nesse caso, a contagem começa a partir do final da sequência, ou seja, o último elemento é representado pelo índice -1, o penúltimo pelo índice -2 e assim por diante:

```
1 lista = [1, 2, 3, 4]
2 penultimo_elemento = lista[-2]
3 print(penultimo_elemento) # Saída: 3
```

### §6.3.1 Operador de fatiamento

O operador de fatiamento é representado pelo símbolo de dois pontos (:), e é utilizado para selecionar uma porção de uma sequência, como uma lista, tupla ou string. Ele permite que você especifique um índice inicial e final, e também a quantidade de itens que devem ser pulados entre esses dois índices. O resultado é uma nova sequência que contém apenas os elementos selecionados.

A sintaxe básica do operador de fatiamento pode ser representada por:

```
sequencia[inicio:fim]
```

Onde **inicio** indica o índice inicial da sequência a ser fatiada e **fim** indica o índice final (não inclusivo). Caso **inicio** não seja informado, a fatia será iniciada a partir do primeiro elemento da sequência. Caso **fim** não seja informado, a fatia será finalizada no último elemento da sequência. Existe também a sintaxe com passo:

```
sequencia[inicio:fim:passo]
```

Onde **passo** indica o intervalo entre os índices a serem incluídos na fatia. O passo padrão quando **passo** não é especificado é 1. O valor do passo pode ser passo negativo, nesse caso a lista é acessada de forma invertida. É importante ressaltar que o operador de fatiamento retorna uma nova sequência contendo os elementos que estão dentro do intervalo especificado, sem modificar a sequência original.

#### Cópia de sequencias

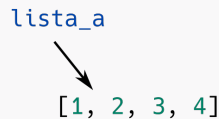
Uma maneira intuitiva de se copiar sequências seria simplesmente atribuir a sequência original a uma nova variável. Porém ao executarmos o código:

```
1 # Cria as listas
2 lista_a = [1, 2, 3, 4]
3 lista_b = lista_a
4
5 # Editar um elemento da lista b
6 lista_b[0] = 10
7
8 # Imprime o mesmo elemento da lista a
9 print(lista_a[0])
```

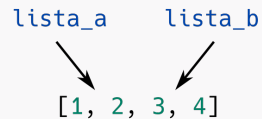
Observamos que a saída será **10** no lugar do esperado **1**, de alguma maneira a edição da `lista_b` afetou os valores da `lista_a`.

Isso ocorre porque ao atribuir uma sequência a uma nova variável, na verdade estamos criando uma referência para a sequência original, e não uma nova cópia. Assim, qualquer alteração feita em uma das variáveis irá afetar a outra.

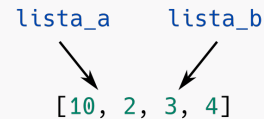
Após a linha 2:



Após a linha 3:



Após a linha 6:

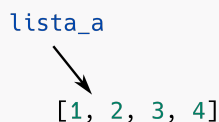


Para fazer uma cópia real da sequência, podemos utilizar o operador de fatiamento, por exemplo:

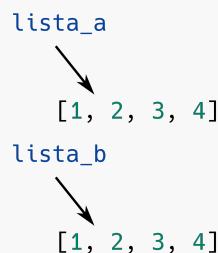
```
1 # Cria as listas
2 lista_a = [1, 2, 3, 4]
3 lista_b = lista_a[:]
4
5 # Editar um elemento da lista b
6 lista_b[0] = 10
7
8 # Imprime o mesmo elemento da lista a
9 print(lista_a[0])
```

Isso irá criar uma nova lista com os mesmos elementos da lista original, sem que as variáveis estejam relacionadas entre si:

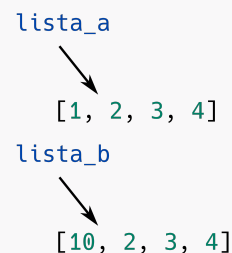
Após a linha 2:



Após a linha 3:



Após a linha 6:



O uso do operador fatiamento é uma das maneiras possíveis de se copiar sequências, existem outras.

### §6.3.2 Usos básicos do operador fatiamento

- Cópia

```
lista_b = lista_a[:]
```

- Acesso até um índice (aberto)

```
lista[:2] # Acessa os índices 0 e 1
```

- Acesso a partir de um índice (fechado)

```
lista[2:] # Acessa os índices 2, 3, ... ate o ultimo item
```

- Acesso entre dois índices

```
lista[1:4] # Acessa os índices 1 e 2
```

### §6.3.3 Usos avançados do operador fatiamento

- Acesso até um índice com passo

```
lista[:4:2] # Acessa os índices 0 e 2
```

- Acesso a partir de um índice com passo

```
lista[2::2] # Acessa os índices 2, 4, 6 ...
```

- Acesso entre dois índices com passo

```
lista[1:6:2] # Acessa os índices 1, 3 e 5
```

- Acesso somente com passo

```
lista[::3] # Acessa os índices 0, 3, 6 ...
```

- Acesso somente com passo negativo (inverte a lista)

```
lista[::-1] # Acessa os índices n-1, n-2, ..., 2, 1, 0
```

## §6.4 Listas

Além das operações de sequencias já discutidas, existem alguns métodos específicos a listas que permitem a sua manipulação.

### §6.4.1 Adicionando itens a listas

Em Python, é possível adicionar itens a uma lista de duas maneiras principais: usando o método `append()` ou com a concatenação de listas.

O método `append()` adiciona um novo elemento ao final da lista:

```
1 lista = [1, 2, 3]
2 lista.append(4)
3 print(lista) # Saída: [1, 2, 3, 4]
```

Podemos também adicionar novos elementos a lista utilizando o operador de concatenação/atribuição (`+=`):

```
1 lista = [1, 2, 3]
2 lista += [4]
3 print(list3) # Saída: [1, 2, 3, 4]
```

É importante lembrar que a concatenação de listas cria uma nova lista, enquanto o método `append()` adiciona um elemento à lista original.

### §6.4.2 Removendo itens de listas

O Python oferece duas maneiras de se remover um item de uma lista: utilizando o método `pop()` ou o comando `del`.

O método `pop()` remove o último elemento da lista por padrão e retorna o valor removido. É possível passar um índice como argumento para remover um elemento específico da lista. Veja um exemplo:

```
1 lista = ['a', 'b', 'c', 'd']
2 ultimo_elemento = lista.pop()
3 print(ultimo_elemento) # Saída: 'd'
4 print(lista) # Saída: ['a', 'b', 'c']
5
6 elemento_c = lista.pop(2)
7 print(elemento_c) # Saída: 'c'
8 print(lista) # Saída: ['a', 'b']
```

Já a palavra-chave `del` é usada para remover um item pelo índice. Diferente do `pop()`, o `del` não retorna o valor removido, apenas remove o item da lista. Veja um exemplo:

```
1 lista = ['a', 'b', 'c', 'd']
2 del lista[2]
3 print(lista) # Saída: ['a', 'b', 'd']
```

Tanto o `pop()` quanto o `del` podem ser utilizados para remover o último elemento da lista. No caso do `pop`, basta não passar nenhum argumento. Já no caso do `del`, basta passar o índice do último elemento (ou `-1`, caso a lista tenha um tamanho desconhecido). Veja um exemplo:

```
1 lista = ['a', 'b', 'c', 'd']
2 lista.pop()
3 print(lista) # Saída: ['a', 'b', 'c']
4
5 del lista[-1]
6 print(lista) # Saída: ['a', 'b']
```

#### Exemplo 6.4.3. Usando listas como filas

Uma fila é uma estrutura de dados em que o primeiro elemento a ser adicionado é o primeiro a ser removido, seguindo uma ordem de entrada conhecida como "primeiro a entrar, primeiro a sair" (FIFO, do inglês First-In-First-Out).

Listas também podem ser utilizadas como filas, onde o primeiro elemento adicionado será o primeiro a ser removido. Isso é feito utilizando o método `append()` para adicionar um elemento no final da lista e o método `pop(0)` para remover o primeiro elemento:

```
1 # Cria lista para armazenar a fila
```

```
2  fila = []
3
4  # Adiciona elementos
5  fila.append('a')
6  fila.append('b')
7  fila.append('c')
8
9  # Imprime a fila
10 print(fila) # Saída: ['a', 'b', 'c']
11
12 # Remove um elemento
13 primeiro = fila.pop(0)
14 print(primeiro) # Saída: 'a'
15 print(fila) # Saída: ['b', 'c']
```

Nesse exemplo, a lista `fila` é utilizada como fila. Os elementos 'a', 'b' e 'c' são adicionados na ordem utilizando o método `append()`. Em seguida, o primeiro elemento é removido utilizando o método `pop(0)` e o valor é armazenado na variável `primeiro`. A lista `fila` agora contém apenas os elementos 'b' e 'c'.

#### Exemplo 6.4.4. Usando listas como pilhas

Uma pilha é uma estrutura de dados em que o último elemento a ser adicionado é o primeiro a ser removido, seguindo uma ordem conhecida como "último a entrar, primeiro a sair" (LIFO, do inglês Last-In-First-Out).

Listas também podem ser utilizadas como pilhas, onde o último elemento adicionado será o primeiro a ser removido. Isso é feito utilizando o método `append()` para adicionar um elemento no final da lista e o método `pop()` para remover o último elemento:

```
1  # Cria lista para armazenar a pilha
2  pilha = []
3
4  # Adiciona elementos
5  pilha.append('a')
6  pilha.append('b')
7  pilha.append('c')
8
9  # Imprime a pilha
10 print(pilha) # Saída: ['a', 'b', 'c']
11
12 # Remove um elemento
13 ultimo = pilha.pop()
14 print(ultimo) # Saída: 'c'
15 print(pilha) # Saída: ['a', 'b']
```

Nesse exemplo, a lista `pilha` é utilizada como pilha. Os elementos 'a', 'b' e 'c' são adicionados na ordem utilizando o método `append()`. Em seguida, o último elemento é removido utilizando o método `pop()` e o valor é armazenado na variável `ultimo`. A lista `pilha` agora contém apenas os elementos 'a' e 'b'.

### §6.4.5 Percorrendo listas

Existem diferentes maneiras de percorrer uma lista em Python, dependendo do objetivo específico do código. Algumas das formas mais comuns são:

1. **Percorrendo por índices:** é possível utilizar o laço de repetição `for` juntamente com a função `range()` e a função `len()` para percorrer os índices respectivos ao comprimento de uma lista e utilizando seus índices como referência. Por exemplo:

```
1 lista = [10, 20, 30, 40, 50]
2 for i in range(len(lista)):
3     print(i, lista[i])
```

2. **Percorrendo por valores:** outra forma de percorrer uma lista é utilizando o laço de repetição `for` diretamente com seus valores. Por exemplo:

```
1 lista = [10, 20, 30, 40, 50]
2 for valor in lista:
3     print(valor)
```

3. **Percorrendo por índices e valores:** por fim, também é possível utilizar a função `enumerate()` em conjunto com o laço de repetição `for` para percorrer uma lista utilizando tanto seus índices quanto seus valores. Por exemplo:

```
1 lista = [10, 20, 30, 40, 50]
2 for i, valor in enumerate(lista):
3     print(i, valor)
```



# 7 | Strings

Uma string é uma sequência de caracteres em Python, que pode ser composta por letras, números, símbolos e espaços em branco. Em Python, as strings são representadas por uma cadeia de caracteres entre aspas simples ( ' ') ou duplas ( " ").

Neste capítulo, vamos aprender a manipular strings em Python. Veremos como acessar caracteres individuais em uma string, como percorrer uma string usando loops, como concatenar strings e como formatar strings para exibição de informações. Também abordaremos alguns métodos úteis para trabalhar com strings, como transformar uma string em maiúsculas ou minúsculas e como dividir uma string em substrings.

## §7.1 Comparação de strings

Ao trabalhar com strings em Python, é comum a necessidade de comparar duas strings para verificar se elas são iguais ou diferentes. Para isso, pode-se utilizar os operadores de comparação:

- O operador `==` verifica se duas strings são iguais.
- O operador `!=` verifica se duas strings são diferentes.

### Exemplo 7.1.1. Comparando strings

```
1  # Define 3 strings
2  string1 = "hello"
3  string2 = "world"
4  string3 = "hello"
5
6  # Verifica se string1 é igual a string2
7  if string1 == string2:
8      print("As strings são iguais.")
9  else:
10     print("As strings são diferentes.")
11
12 # Verifica se string1 é igual a string3
13 if string1 == string3:
14     print("As strings são iguais.")
15 else:
16     print("As strings são diferentes.")
```

Nesse exemplo, a primeira comparação entre as strings `string1` e `string2` resulta em "As strings são diferentes." pois elas não são iguais. Já a segunda comparação entre as strings `string1` e `string3` resulta em "As strings são iguais." pois elas são iguais.

### §7.1.2 Comparação de substrings

Substrings são partes de uma string que podem ser extraídas utilizando índices. Para obter uma substring em Python, pode-se utilizar a sintaxe de fatiamento: `string[inicio:fim]`. Onde "inicio" é o índice do primeiro caractere da substring e "fim" é o índice do último caractere + 1.

Por exemplo, para obter a substring "mundo" a partir da string "olá mundo", pode-se utilizar:

```
1 s = "olá mundo"
2 substring = s[4:]
3 print(substring) # Saída: "mundo"
```

Para utilizar substrings em comparações, pode-se simplesmente comparar a substring extraída com outra string:

```
1 s = "olá mundo"
2 substring = s[-6:]
3 if substring == "mundo!":
4     print("A substring é igual a 'mundo!'")
```

Outra opção é utilizar os métodos `startswith()` e `endswith()`, que verificam se uma string começa ou termina com uma determinada substring:

```
1 s = "olá mundo"
2 if s.startswith("olá"):
3     print("A string começa com 'olá!'")
4 if s.endswith("mundo"):
5     print("A string termina com 'mundo!'")
```

Esses métodos são especialmente úteis quando a comparação envolve apenas o início ou o fim da string.

Mais uma opção para comprar substrings é o operador `in`, que verifica se uma string é uma substring de outra string:

```
1 texto = "olá mundo"
2 if "olá" in texto:
3     print("Substring encontrada!")
4 else:
5     print("Substring nao encontrada!")
```

Nesse caso, a saída seria "Substring encontrada!", pois a string "Hello" está presente na string original.

A principal diferença entre os métodos `startswith()` e `endswith()` e o operador `in` é que os métodos verificam se uma string começa ou termina com determinado valor, enquanto o operador verifica se uma string contém determinado valor em qualquer posição.

## §7.2 Métodos `lower()` e `upper()`

Os métodos `lower()` e `upper()` são utilizados para converter as letras de uma string em letras minúsculas ou maiúsculas, respectivamente.

O método `lower()` retorna uma cópia da string original com todas as letras em minúsculas. Já o método `upper()` retorna uma cópia da string original com todas as letras em maiúsculas.

Esses métodos são úteis em várias situações, como na comparação de strings, pois nem sempre as letras de duas strings são iguais, mas a diferença está apenas na caixa (maiúscula ou minúscula) em que elas estão escritas. Com a conversão para letras maiúsculas ou minúsculas, é possível fazer a comparação sem se preocupar com essa diferença.

**Exemplo 7.2.1.** Comparando strings com lower()

```
1 # Define duas strings
2 string1 = "Hello"
3 string2 = "hello"
4
5 # Compara as duas utilizando o método lower
6 if string1.lower() == string2.lower():
7     print("As strings são iguais, ignorando as diferenças entre
8         maiúsculas e minúsculas.")
9 else:
10    print("As strings são diferentes.")
```

Nesse exemplo, a comparação entre as strings `string1` e `string2` resulta em "As strings são iguais, ignorando as diferenças entre maiúsculas e minúsculas." pois elas são iguais quando convertidas para letras minúsculas.

**Exemplo 7.2.2.** Tratando entradas de usuário para ignorar maiúsculas

```
1 # Inicializa uma lista vazia para armazenar os números
2 numeros = []
3
4 # Loop infinito para receber entradas do usuário
5 while True:
6     # Solicita que o usuário digite um número ou "sair" para
7     # finalizar
8     entrada = input("Digite um número ou 'sair' para sair: ")
9
10    # Se a entrada do usuário for "sair" (ignorando maiúsculas),
11    # o loop é encerrado
12    if entrada.lower() == 'sair':
13        break
14
15    # Converte a entrada do usuário para float e adiciona à lista
16    # de números
17    numeros.append(float(entrada))
18
19    # Calcula a média dos números utilizando as funções sum() e len()
20    media = sum(numeros) / len(numeros)
21    # Exibe a média dos números
22    print(f"A média dos números é: {media}")
```

Neste exemplo, a string "sair" é utilizada como critério para encerrar o loop que solicita ao usuário a digitação dos números. No entanto, para garantir que o programa também encerre o loop caso o usuário digite "Sair" ou "SAIR", por exemplo, é aplicado o método `lower()` à string digitada pelo usuário.

## §7.3 Pesquisa em strings

### §7.3.1 Método find()

O método `find()` é utilizado para buscar a primeira ocorrência de uma substring em uma string. Ele retorna a posição da primeira letra da substring na string original, ou -1 caso a substring não seja encontrada.

A sintaxe do método é a seguinte:

```
string.find(substring, inicio, fim)
```

Onde:

- **string**: a string original onde será feita a busca
- **substring**: a substring que será buscada na string original
- **inicio (opcional)**: a posição inicial onde a busca será iniciada (padrão é 0)
- **fim (opcional)**: a posição final onde a busca será encerrada (padrão é o final da string)

#### Exemplo 7.3.2. Uso do método `find()`

```
1 frase = "A vida é bela"
2 posicao = frase.find("bel")
3 print(posicao) # Saída: 9
```

Neste exemplo, a posição da primeira letra da substring "bel" na string original "A vida é bela" é 9.

É importante lembrar que o método `find()` retorna apenas a posição da primeira ocorrência da substring. Se houverem mais ocorrências da mesma substring na string original, elas não serão consideradas. Para buscar todas as ocorrências de uma substring na string original, pode-se utilizar o método `find()` dentro de um loop, atualizando o parâmetro **inicio** a cada iteração.

#### Exemplo 7.3.3. Busca de todas as ocorrências de uma substring

```
1 # String para realizada a busca
2 texto = "Hoje é um belo dia para programar, não é mesmo? Vamos
   programar juntos hoje!"
3 # String para ser buscada
4 substring = "programar"
5
6 # Inicializa a variável que vai armazenar a posição inicial da
   busca
7 posicao = 0
8 # Inicializa a variável que vai armazenar as posições das
   ocorrências
9 ocorrencias = []
10
11 # Faz a busca na string enquanto houver ocorrências
12 while True:
13     # procura a substring a partir da posição atual
```

```
14     posicao = texto.find(substring, posicao)
15
16     # se não encontrou mais nenhuma ocorrência, encerra o loop
17     if posicao == -1:
18         break
19
20     # adiciona a posição da ocorrência na lista
21     ocorrencias.append(posicao)
22
23     # atualiza a posição inicial para começar a busca da próxima
        ocorrência
24     posicao += 1
25
26 # Exibe as posições das ocorrências
27 print(ocorrencias)
```

Para encontrar todas as ocorrências da substring "programar" na string acima e armazená-las em uma lista, podemos usar o método `find()` em um loop `while`. O método `find()` retorna a posição da primeira ocorrência da substring na string ou -1 caso não encontre. Podemos utilizar o parâmetro de início para avançar a busca a partir da posição da última ocorrência encontrada.

### §7.3.4 Método `count()`

O método `count()` é utilizado para contar o número de ocorrências de uma substring em uma string. Ele recebe como parâmetro a substring a ser contada e retorna a quantidade de vezes que ela aparece na string.

#### Exemplo 7.3.5. Uso do método `count()`

```
1 texto = "O rato roeu a roupa do rei de Roma"
2 qtd_ocorrencias = texto.count('ro')
3 print(qtd_ocorrencias) # Saída: 2
```

Nesse exemplo, o método `count()` foi utilizado para contar quantas vezes a substring "ro" aparece na string "O rato roeu a roupa do rei de Roma". O resultado é 2, já que essa substring aparece duas vezes na string.

## §7.4 Quebra e junção de strings

Quebra e junção de strings em listas e em substrings é uma operação muito comum em programação. Em Python, essas operações podem ser realizadas utilizando os métodos `split()` e `join()`.

### §7.4.1 Método `split()`

O método `split()` divide uma string em uma lista de substrings com base em um separador especificado. O separador padrão é um único espaço ' '.

**Exemplo 7.4.2.** Uso do método `split()`

```
1 frase = "O rato roeu a roupa do rei de Roma."  
2 palavras = frase.split()  
3 print(palavras)  
4 # Saída: ['O', 'rato', 'roeu', 'a', 'roupa', 'do', 'rei', 'de',  
           'Roma.']
```

O código acima irá imprimir a lista de palavras da frase, onde cada elemento da lista é uma palavra separada por espaço. O método `split()` também pode ser utilizado com separadores diferentes do espaço:

```
1 frase = "banana,maçã,laranja,abacaxi,melancia"  
2 lista_frutas = frase.split(",")  
3 print(lista_frutas)  
4 # Saída: ['banana', 'maçã', 'laranja', 'abacaxi', 'melancia']
```

Neste exemplo, a string frase é quebrada em uma lista de substrings utilizando a vírgula como separador, através do método `split()` com parâmetro `","`. O resultado é a lista `lista_frutas` contendo cada fruta como um elemento da lista.

Note que o separador utilizado no método `split()` foi a vírgula `(",")` em vez do espaço `(" ")`, que é o separador padrão.

**Conversão de strings em lista**

É possível também converter uma string diretamente em uma lista utilizando a função `list()`. Cada caractere da string será considerado um elemento da lista. Por exemplo:

```
1 minha_string = "Python"  
2 minha_lista = list(minha_string)  
3 print(minha_lista)  
4 # Saída: ['P', 'y', 't', 'h', 'o', 'n']
```

É importante lembrar que, ao converter uma string em lista, cada elemento da lista será do tipo string. Se for necessário realizar operações matemáticas ou comparações numéricas com os caracteres da string, é necessário convertê-los para o tipo numérico correspondente utilizando funções como `int()` ou `float()`.

**§7.4.3 Método `join()`**

O método `join()` é utilizado para juntar uma sequência de strings em uma única string. Esse método recebe como parâmetro uma lista de strings (ou outra sequência de strings) e retorna uma única string contendo todas as strings da sequência original concatenadas, separadas por um separador especificado.

Sintaxe:

```
| string_final = separador.join(lista_de_strings)
```

Onde:

- **separador**: a string que será usada para separar cada item da lista na string final.
- **lista\_de\_strings**: a sequência de strings que será concatenada.

**Exemplo 7.4.4.** Uso do método `join`

```
1 | nomes = ['João', 'Maria', 'Pedro']
2 | nomes_string = ', '.join(nomes)
3 | print(nomes_string)
4 | # Saída: João, Maria, Pedro
```

Nesse exemplo, o método `join()` recebe a lista de nomes e a string `' , '` como separador, criando uma única string contendo os nomes concatenados, separados por vírgula e espaço.

**Exemplo 7.4.5.** Substituindo um caracter por outro em uma string

```
1 | # Definindo a string original
2 | string = "banana"
3 |
4 | # Convertendo a string em uma lista de caracteres
5 | lista_string = list(string)
6 |
7 | # Substituindo todos os "a" por "o"
8 | for i in range(len(lista_string)):
9 |     if lista_string[i] == "a":
10 |         lista_string[i] = "o"
11 |
12 | # Convertendo a lista de caracteres de volta para uma string
13 | string_modificada = "".join(lista_string)
14 |
15 | # Imprimindo a string modificada
16 | print(string_modificada)
```

Neste exemplo, a string original é "banana" e o objetivo é substituir todos os caracteres "a" por "o". Primeiro, a string é convertida em uma lista de caracteres utilizando a função `list()`. Depois, é feita uma iteração sobre a lista para substituir os caracteres "a" por "o". Finalmente, a lista de caracteres é convertida de volta para uma string utilizando o método `join()`, e a string modificada é impressa na tela. O resultado seria a string "bonono".

## §7.5 Método `replace()`

O método `replace()` é utilizado para substituir uma substring por outra em uma string. Sua sintaxe é a seguinte:

```
string.replace(substring_antiga, substring_nova, quantidade)
```

Onde:

- **string**: a string original;
- **substring\_antiga**: a substring que será substituída;
- **substring\_nova**: a substring que será colocada no lugar da antiga;
- **quantidade** (opcional): a quantidade de ocorrências que serão substituídas (padrão é todas).

**Exemplo 7.5.1.** Uso do método `replace()`

```
1 frase = "O rato roeu a roupa do rei de Roma"
2 nova_frase = frase.replace("r", "p")
3 print(nova_frase)
4 # Saída: O pato poueu a poupa do pei de Roma"
```

Nesse exemplo, a string "O rato roeu a roupa do rei de Roma" é atribuída à variável `frase`. Em seguida, é criada uma nova string (`nova_frase`) que é o resultado da substituição de todas as ocorrências da letra "r" por "p" na string original. O resultado impresso é a nova string: "O pato poueu a poupa do pei de Roma".

Note que o último R não foi substituído pois ele é um carácter minúsculo e somente substituímos os r minúsculos.

### §7.5.2 Método `strip()`

O método `strip()` é usado para remover caracteres específicos do início e/ou do final de uma string. O método retorna uma cópia da string original com os caracteres especificados removidos.

A sintaxe do método `strip()` é a seguinte:

```
| string.strip(caracteres)
```

Onde:

- `string` é a string original a ser manipulada.
- `caracteres` é uma string opcional contendo os caracteres que devem ser removidos do início e/ou do final da string. Se essa string não for especificada, serão removidos espaços em branco (incluindo tabulações e quebras de linha).

**Exemplo 7.5.3.** Uso do método `strip()`

```
1 texto = "  Olá, mundo!  "
2 texto = texto.strip()
3 print(texto)
4 # Saída: "Olá, mundo!"
```

Nesse exemplo, a string original contém espaços em branco no início e no final. O método `strip()` é usado para remover esses espaços em branco, deixando apenas o texto desejado. O resultado é a string "Ola, mundo!" sem espaços em branco adicionais.

## §7.6 Métodos de validação

Além dos métodos já discutidos as strings também possuem métodos de validação que retornam `True` ou `False` de acordo com o conteúdo da string, alguns desses são:

- `.isalnum()`: Checa se a string é alfanumérica (letras e números somente)
- `.isalpha()`: Checa se a string possui somente letras (inclui vogais acentuadas)
- `.isdigit()`: Checa se a string possui somente dígitos (números)



- `.isspace()`: Checa se a string possui apenas espaços em branco (espaços, tab, quebra de linha)
- `.isupper()`: Checa se a string possui somente caracteres maiúsculos.
- `.islower()`: Checa se a string possui somente caracteres minúsculos.

Mais métodos podem ser encontrados na documentação do Python: <https://docs.python.org/pt-br/3.11/library/stdtypes.html#string-methods>

# 9

## Dicionários

Dicionários são estruturas de dados em Python que permitem armazenar informações não ordenadas em pares de chave-valor. Ao contrário das listas, onde os elementos são indexados numericamente, os dicionários usam chaves únicas para identificar cada valor associado a ela. Dessa forma, eles são úteis quando precisamos acessar os valores de acordo com suas chaves, sem precisar passar por todos os elementos da estrutura. Os dicionários são mutáveis e podem ser modificados a qualquer momento, adicionando, removendo ou atualizando pares de chave-valor. Eles são amplamente utilizados em Python, especialmente para armazenar e manipular dados em formato de tabela.

Os dicionários em Python são definidos por meio de chaves e contêm uma coleção de pares chave-valor. Cada elemento do dicionário consiste em uma chave e seu respectivo valor, separados por dois pontos (:), e separados por vírgulas. A chave deve ser única e imutável, enquanto o valor pode ser de qualquer tipo de dados, incluindo outros dicionários.

A sintaxe básica de um dicionário em Python é:

```
nome_do_dicionario = {chave1: valor1, chave2: valor2, chave3: valor3}
```

Para acessar os valores do dicionário por meio das chaves, utilizamos colchetes com a chave logo após o dicionário:

```
nome_do_dicionario[chave]
```

### Exemplo 9.0.1. Criando, alterando e acessando dicionários

```
1 # Definição do dicionário com quantidade de frutas
2 qtde_frutas = {'banana': 2, 'maçã': 3, 'abacaxi': 1}
3
4 # Imprime a quantidade de bananas
5 print(qtde_frutas['banana']) # Saída: 2
6
7 # Adiciona 3 bananas
8 qtde_frutas['banana'] += 3
9
10 # Imprime a quantidade de bananas
11 print(qtde_frutas['banana']) # Saída: 5
```

Nesse exemplo, é criado um dicionário chamado `qtde_frutas` com três pares de chave-valor: `'banana': 2`, `'maçã': 3` e `'abacaxi': 1`. Em seguida, é impresso na tela o valor associado a chave `'banana'` e logo após é somado 3 ao valor associado à chave `'banana'` através do uso do operador de indexação `[]` e do operador de soma-atribuição `+=` e da atribuição de um novo valor. Por fim, o novo valor associado à chave `'banana'` é acessado através do mesmo operador de indexação `[]` e é impresso na tela.

## §9.1 Iterando em dicionários

Existem algumas maneiras de iterar em um dicionário em Python. As principais são:

- Iterar nas chaves: utilizando o método `keys()` do dicionário, é possível iterar sobre todas

as chaves do dicionário. Exemplo:

```
for chave in dicionario.keys():
```

Como esse é o modo padrão de iteração de um dicionário, o loop também pode ser escrito assim:

```
for chave in dicionario:
```

- Iterar nos valores: utilizando o método `values()` do dicionário, é possível iterar sobre todos os valores do dicionário. Exemplo:

```
for valor in dicionario.values():
```

- Iterar simultaneamente nas chaves e valores: utilizando o método `items()` do dicionário, é possível iterar sobre todos os pares chave-valor do dicionário. Exemplo:

```
for chave, valor in dicionario.items():
```

Lembrando que a ordem de iteração nos itens pode ocorrer em ordem diferente da que foram adicionados ao dicionário, já que os dicionários são estruturas de dados desordenadas em Python.

**Exemplo 9.1.1.** Diferentes formas de iterar em um dicionário:

```
1 meu_dict = {"chave1": "valor1", "chave2": "valor2", "chave3":  
2             "valor3"}  
3 # Iterando sobre as chaves do dicionário  
4 for chave in meu_dict:  
5     print(chave)  
6  
7 # Iterando sobre os valores do dicionário  
8 for valor in meu_dict.values():  
9     print(valor)  
10  
11 # Iterando sobre os itens do dicionário (chave e valor)  
12 for chave, valor in meu_dict.items():  
13     print(f'Chave: {chave}, Valor: {valor}')
```

## §9.2 Combinando dicionários

Para combinar dois dicionários em Python, podemos utilizar o método `update()`. Este método atualiza um dicionário com os elementos de outro dicionário, adicionando novos pares chave-valor ou atualizando os valores das chaves existentes.

A sintaxe básica para o uso do método `update()` é a seguinte:

```
dicionario1.update(dicionario2)
```

Onde `dicionario1` é o dicionário que será atualizado com os elementos de `dicionario2`.

**Exemplo 9.2.1.** Combinando dois dicionários

```
1 dicionario1 = {"maçã": 5, "banana": 3, "laranja": 2}
2 dicionario2 = {"uva": 4, "maçã": 2, "banana": 1}
3
4 # Combinando os dicionários
5 dicionario1.update(dicionario2)
6
7 # Imprimindo o dicionário combinado
8 print(dicionario1)
```

Neste exemplo, definimos dois dicionários `dicionario1` e `dicionario2`. Em seguida, utilizamos o método `update()` para combinar os dois dicionários, atualizando o `dicionario1` com os elementos do `dicionario2`. Por fim, imprimimos o dicionário combinado para verificar o resultado.

O resultado da execução do código será o seguinte:

```
1 {"maçã": 2, "banana": 1, "laranja": 2, "uva": 4}
```

**Combinando dois dicionários com chaves em comum**

Quando usamos o método `update()` para combinar dois dicionários, caso eles possuam uma chave em comum, o valor da chave do segundo dicionário será usado para sobrescrever o valor da chave no primeiro dicionário. Isso ocorre porque o método `update()` atualiza o primeiro dicionário com as chaves e valores do segundo dicionário, e se houverem chaves em comum, o valor do segundo dicionário prevalecerá sobre o primeiro. É importante estar ciente disso ao utilizar o método `update()`, para evitar a perda de informações importantes em nossos dicionários.

**Exemplo 9.2.2.** Combinando dois dicionários somando os valores

Para combinar dicionários contendo valores numéricos somando os valores, podemos iterar nos valores e chaves de segundo dicionário com o método `items()` e checar se cada chave existe no primeiro dicionário somando ou atribuindo o valor conforme necessário:

```
1 # Definindo os dicionários com quantidades de frutas
2 deposito1 = {"maçã": 5, "banana": 3, "laranja": 2}
3 deposito2 = {"uva": 4, "maçã": 2, "banana": 1}
4
5 # Itera nas quantidades de frutas no deposito2
6 for fruta, quantidade in deposito2.items():
7     # Checa se a chave existe no deposito1
8     if fruta in deposito1:
9         # Caso exista, soma os dois valores e atualiza o deposito1
10        deposito1[fruta] += quantidade
11    else:
12        # Caso não exista, adiciona ao deposito1
13        deposito1[fruta] = quantidade
14
15 # Imprime o dicionário combinado
```

```
16 | print(deposito1)
```

Neste exemplo, o valor numérico corresponde à quantidade de cada fruta disponível. O resultado do programa será um dicionário que combina os valores de quantidade de ambas as variáveis **deposito1** e **deposito2**, resultando em um novo dicionário contendo a quantidade total de cada fruta:

```
| {'maçã': 7, 'banana': 4, 'laranja': 2, 'uva': 4}
```

## §9.3 Deletando itens de dicionários

Para deletar um item de um dicionário em Python, podemos usar o comando **del**, passando a chave do item que queremos deletar como argumento. A sintaxe é a seguinte:

```
| del meu_dicionario[chave]
```

Essa linha de código irá remover o item do dicionário que tem a chave especificada. É importante notar que se a chave não existir no dicionário, um erro será lançado.

## §9.4 Copiando dicionários

De maneira similar a listas somente atribuir uma variável contendo um dicionário a uma nova variável não cria uma cópia desse dicionário:

```
1 | dict1 = {'a': 1, 'b': 2, 'c': 3}
2 | dict2 = dict1 # atribuição direta
3 | dict2['a'] = 10
4 | print(dict1['a']) # Saída: 10
```

Nesse caso, a variável **dict2** não é uma cópia de **dict1**, mas sim uma referência ao mesmo objeto. Isso significa que qualquer modificação feita em **dict1** será refletida em **dict2** e vice-versa.

```
1 | dict1 = {'a': 1, 'b': 2, 'c': 3}
2 | dict2 = dict(dict1) # cria uma copia usando a função dict()
3 | dict2['a'] = 10
4 | print(dict1['a']) # Saída: 1
```

Dessa forma, é criado um novo objeto dicionário a partir do objeto **dict1**, que é independente dele. Qualquer modificação feita em um dos dicionários não afetará o outro.

Vale ressaltar que, se o dicionário a ser copiado contiver objetos mutáveis (como listas), esses objetos não serão copiados automaticamente. Nesse caso, é necessário criar cópias desses objetos separadamente.

# 10 | Funções

Funções são blocos de código que executam uma tarefa específica e podem ser chamados em outras partes do programa para reutilização do código. Elas são úteis quando precisamos executar uma mesma sequência de comandos em diferentes partes do código, ou quando queremos separar o código em partes menores e mais fáceis de gerenciar (modularização).

A sintaxe básica de uma função em Python é a seguinte:

```
1 | def nome_da_funcao(parametros):  
2 |     # corpo da função  
3 |     return valor_de_retorno
```

Explicando cada elemento:

- **def**: palavra-chave que define uma nova função
- **nome\_da\_funcao**: nome dado à função, que deve seguir as mesmas regras de nomeação de variáveis
- **parametros**: argumentos opcionais da função que podem ser passados para dentro dela, múltiplos argumentos podem ser passados para uma função separados por vírgula.
- **:**: sinaliza o início do corpo da função
- **return**: palavra-chave que indica o valor de retorno da função (opcional)
- **valor\_de\_retorno**: valor retornado pela função (opcional)

**Exemplo 10.0.1.** Função simples que imprime uma mensagem na tela

```
1 | def say_hello():  
2 |     print("Olá, mundo!")
```

Essa função é definida usando a palavra-chave **def** seguida pelo nome da função e, entre parênteses, os parâmetros da função (nesse caso, não há nenhum). A linha termina com dois pontos. O corpo da função é indentado e contém a lógica da função (neste caso, imprimir uma mensagem na tela).

Para chamar a função, basta escrever seu nome seguido por parênteses:

```
| say_hello()
```

Isso imprimirá **Olá, mundo!** na tela.

## §10.1 Funções com parâmetros

As funções em Python também podem receber parâmetros, que são valores que podem ser passados para a função para que ela os utilize em sua execução. Existem dois tipos de parâmetros: posicionais e de palavra chave.

### §10.1.1 Parâmetros posicionais

Os parâmetros posicionais são aqueles que são passados na mesma ordem em que são definidos na declaração da função e não possuem valor padrão sendo portanto mandatórios na chamada da função.

**Exemplo 10.1.2.** Função de soma com parâmetros.

```
1 def soma(a, b):  
2     print(f"{a} + {b} = {a + b}")
```

Nessa função, os parâmetros **a** e **b** são posicionais. Ao ser chamada a função imprime na tela os números passados e o resultado.

### §10.1.3 Parâmetros de palavra chave

Os parâmetros de palavra chave são aqueles que são especificados pelo nome na chamada da função. Eles podem ser passados em qualquer ordem e possuem valor padrão e podem portanto ser omitidos na chamada. Parâmetros de palavra chave devem sempre ser declarados depois dos posicionais.

**Exemplo 10.1.4.** Função com parâmetro por palavra chave:

```
1 def mostrar_informacoes(nome, idade, cidade="Maceió"):  
2     print("Nome:", nome)  
3     print("Idade:", idade)  
4     print("Cidade:", cidade)
```

Nessa função, **nome** e **idade** são parâmetros posicionais e **cidade** é um parâmetro de palavra chave com valor padrão "São Paulo". Para chamá-la, podemos passar todos os parâmetros:

```
| mostrar_informacoes("Joao", 30, cidade="Rio de Janeiro")
```

Ou podemos omitir o valor do parâmetro com valor padrão:

```
| mostrar_informacoes("Maria", 25)
```

Nesse caso, o valor de **cidade** será "Maceió".

## §10.2 Funções com retorno

Funções com retorno são aquelas que, além de receberem parâmetros, também retornam um valor para quem as chamou. Isso permite que o valor calculado dentro da função seja utilizado em outras partes do código.

O valor de retorno pode ser qualquer tipo de dado em Python, como um número, uma string, uma lista, uma tupla ou até mesmo outro objeto.

Aqui está um exemplo de função que recebe dois números como parâmetros e retorna a soma entre eles:

**Exemplo 10.2.1.** Função de soma com retorno:

```
1 def soma(a, b):
2     resultado = a + b
3     return resultado
4
5 # Chamando a função e armazenando o resultado em uma variável
6 resultado_da_soma = soma(2, 3)
7
8 # Imprimindo o resultado da soma
9 print(resultado_da_soma) # Saída: 5
```

Podemos também utilizar a função diretamente em uma expressão:

```
print(soma(4, 5)) # Saída: 9
```

### Função com retorno de múltiplos valores

Uma função pode também retornar múltiplos valores utilizando uma tupla:

```
1 def calcular_media(nota1, nota2, nota3):
2     media = (nota1 + nota2 + nota3) / 3
3     if media >= 7:
4         situacao = 'Aprovado'
5     else:
6         situacao = 'Reprovado'
7     return media, situacao
8
9 # Chamando a função e armazenando os resultados em variáveis
   distintas
10 media_aluno, situacao_aluno = calcular_media(8, 7, 9)
11
12 # Imprimindo os resultados
13 print('Média:', media_aluno) # Saída: Média: 8.0
14 print('Situação:', situacao_aluno) # Saída: Situação: Aprovado
```

## §10.3 Escopo de variáveis

Quando uma variável é definida em uma função, ela só é visível dentro da função. Isso é conhecido como escopo local. Quando a variável é definida fora da função, ela pode ser acessada de qualquer lugar do programa. Isso é conhecido como escopo global.

### §10.3.1 Variáveis locais

As variáveis locais são definidas dentro de uma função e só podem ser acessadas dentro da função. Elas são criadas quando a função é chamada e são destruídas quando a função é concluída. Vamos dar uma olhada no exemplo a seguir:

```
1 # Define uma função com uma variável local
2 def minha_funcao():
```



```
3     x = 10
4     print("O valor de x dentro da função e: ", x)
5
6     # Chama a função
7     minha_funcao()
8
9     # Tenta imprimir a variável que definimos dentro da função
10    print("O valor de x fora da funcao e:", x)
```

A saída será:

```
1 0 valor de x dentro da função e: 10
2 NameError: name 'x' is not defined
```

Como podemos ver, a variável `x` só existe dentro da função `minha_funcao()`. Quando tentamos acessá-la fora da função, recebemos um erro.

### §10.3.2 Variáveis globais

As variáveis globais são definidas fora de uma função e podem ser acessadas de qualquer lugar do programa. Elas podem ser usadas dentro de funções, mas se elas forem modificadas dentro de uma função, a modificação só será refletida dentro da função. Vamos dar uma olhada no exemplo a seguir:

```
1 # Define uma variável fora da função
2 x = 10
3
4 # Define uma função que acessa a variável
5 def minha_funcao():
6     print("O valor de x dentro da funcao e:", x)
7
8 # Chama a função
9 minha_funcao()
10
11 # Imprime a variável
12 print("O valor de x fora da funcao e:", x)
```

A saída será:

```
1 0 valor de x dentro da função e: 10
2 0 valor de x fora da função e: 10
```

Como podemos ver, a variável `x` é definida fora da função e pode ser acessada dentro da função e fora dela.

### §10.3.3 Modificando variáveis globais dentro de uma função

Para modificar ou criar uma variável global dentro de uma função, precisamos usar a palavra-chave `global`. Vamos dar uma olhada no exemplo a seguir:

```
1 # Define uma variável fora da função
2 x = 10
3
```

```

4  # Define uma função que edita o valor da variável global
5  def minha_funcao():
6      global x
7      x = 20
8      print("0 valor de x dentro da funcao e:", x)
9
10 # Chama a função
11 minha_funcao()
12
13 # Imprime a variável
14 print("0 valor de x fora da funcao e:", x)

```

A saída será:

```

1  0 valor de x dentro da função e: 20
2  0 valor de x fora da função e: 20

```

Como podemos ver, a palavra-chave **global** é usada para indicar que estamos nos referindo à variável global `x` e não a uma variável local com o mesmo nome.

Este uso não é recomendado, sempre que possível deve se usar parâmetros e retornos para trafegar dados de dentro para fora de funções e vice-versa.

## §10.4 Recursão

Recursão é uma técnica em programação onde uma função chama a si mesma para resolver um problema. Ela é útil quando um problema pode ser quebrado em problemas menores e similares.

Para entendermos melhor o que são funções recursivas vamos observar a função fatorial, ela pode ser definida como:

$$fatorial(x) = x * (x - 1) * \dots * 2 * 1 \quad (10.1)$$

Logo o fatorial de  $x - 1$  será:

$$fatorial(x - 1) = (x - 1) * \dots * 2 * 1 \quad (10.2)$$

Substituindo 10.2 em 10.1 obtemos:

$$fatorial(x) = x * fatorial(x - 1) \quad (10.3)$$

Porém para a nossa definição ficar completa precisamos saber quando parar de calcular  $fatorial(x - 1)$  precisamos de uma *condição de parada*:

$$fatorial(x) = \begin{cases} x * fatorial(x - 1) & \text{se } x > 1 \\ 1 & \text{se } x = 1 \end{cases} \quad (10.4)$$

O que define a nossa função fatorial em termos dela mesma para todo  $x \in \mathbb{N}$ .

**Exemplo 10.4.1.** Fatorial com função recursiva

```
1 def fatorial(n):
2     if n == 1:
3         return 1
4     else:
5         return n * fatorial(n-1)
```

Nesta função, se  $n$  for igual a 1, ela retorna 1. Caso contrário, ela retorna  $n$  multiplicado pelo resultado da chamada recursiva da função fatorial para  $n-1$ .

**Exemplo 10.4.2.** Termo  $n$ -ésimo da sequência de Fibonacci

```
1 def fibonacci(n):
2     if n == 0:
3         return 0
4     elif n == 1:
5         return 1
6     else:
7         return fibonacci(n-1) + fibonacci(n-2)
```

Nesta função, se  $n$  for igual a 0, ela retorna 0. Se  $n$  for igual a 1, ela retorna 1. Caso contrário, ela retorna a soma dos dois termos anteriores da sequência de Fibonacci, calculados recursivamente.

É importante lembrar que a recursão pode facilmente levar loops infinitos. Ao definir uma função recursiva é importante sempre ter uma condição de parada e fazer com que os parâmetros se encaminhem para a condição de parada a cada chamada da função.

No exemplo 10.4.1 a condição de parada é retornar o valor **1** quando  $n$  chega no valor **1**, e o valor de  $n$  é subtraído um a cada chamada da função, garantindo que eventualmente ele chegue ao valor **1** para qualquer  $n$  positivo.

Já no exemplo 10.4.2 as condições de parada é retornar **0** ou **1** quando  $n$  chega nos valores de **0** e **1** respectivamente, para este exemplo precisamos de duas condições de parada pois para cada chamada da função precisamos chamar ela novamente para  $n-1$  e  $n-2$ , se não tivéssemos as duas condições de parada em sequência a execução não seria interrompida.

# 11 | Bibliotecas

A programação em Python é poderosa e flexível, e uma das principais razões para isso é a ampla variedade de bibliotecas disponíveis. Bibliotecas são coleções de código pré-escrito que fornecem funções e recursos adicionais para facilitar o desenvolvimento de software.

Nesta aula, exploraremos o uso de bibliotecas em Python. Veremos como importar bibliotecas em nossos programas, como usar as bibliotecas nativas do Python e também como utilizar outras bibliotecas de terceiros.

Ao importar uma biblioteca, ganhamos acesso a um conjunto de funcionalidades adicionais que podem ser usadas em nosso código. As bibliotecas nativas do Python já vêm instaladas com o Python e fornecem uma ampla gama de recursos úteis. Já as bibliotecas de terceiros são desenvolvidas por outras pessoas ou organizações e oferecem funcionalidades especializadas para atender a diferentes necessidades.

## §11.1 Importando bibliotecas

Para utilizar uma biblioteca em um programa Python, é necessário importá-la.

Existem algumas maneiras de importar uma biblioteca em Python. A forma mais comum é utilizando o comando **import**. Veja o exemplo abaixo:

```
import math
```

Nesse exemplo, estamos importando a biblioteca **math**. Agora, podemos utilizar as funções e constantes disponíveis nessa biblioteca em nosso código.

Outra forma de importar uma biblioteca é utilizando a declaração **from**. Com essa declaração, é possível importar apenas funções específicas ou elementos específicos de uma biblioteca. Veja o exemplo abaixo:

```
from math import sqrt, sin
```

Nesse exemplo, estamos importando apenas as funções **sqrt** e **sin** da biblioteca **math**. Dessa forma, não precisamos utilizar o nome da biblioteca antes de chamar essas funções em nosso código.

Além disso, é possível renomear uma biblioteca ou função ao importá-la, utilizando a palavra-chave **as**. Isso pode ser útil para deixar o código mais legível ou evitar conflitos de nomes. Veja o exemplo abaixo:

```
import math as m
```

Nesse exemplo, estamos importando a biblioteca **math**, mas renomeando-a para **m**. Agora, podemos utilizar as funções da biblioteca utilizando o nome **m** em vez de **math**.

Ao utilizar bibliotecas em seus programas Python, é importante ler a documentação da biblioteca para entender quais funções e recursos estão disponíveis e como utilizá-los corretamente.

**Exemplo 11.1.1.** Importando bibliotecas

```
1  # Importando a biblioteca inteira
2  import math
3
4  # Utilizando uma função da biblioteca inteira
5  raiz_quadrada = math.sqrt(25)
6  print(f"A raiz quadrada de 25 é {raiz_quadrada}")
7
8  # Importando apenas uma função específica
9  from random import randint
10
11 # Utilizando a função diretamente
12 numero_aleatorio = randint(1, 100)
13 print(f"Um número aleatório: {numero_aleatorio}")
14
15 # Importando uma biblioteca com um nome diferente
16 import numpy as np
17
18 # Utilizando uma função da biblioteca com o nome alterado
19 vetor = np.arange(1, 6)
20 print(f"Um vetor: {vetor}")
```

Nesse exemplo, primeiro importamos a biblioteca **math** inteira, em seguida, usamos a função **sqrt** para calcular a raiz quadrada de um número, observe que para chamar um função de uma biblioteca utilizamos o ponto (.) da mesma maneira que acessamos métodos de objetos:

```
| math.sqrt(25)
```

Depois, importamos apenas a função **randint** da biblioteca **random** e a usamos para gerar um número aleatório, neste caso como importamos a função específica não é preciso chamar no nome da biblioteca ante do nome da função.

```
| randint(1, 100)
```

Por fim, importamos a biblioteca **numpy** e a renomeamos para **np**. Utilizamos a função **arange** da biblioteca **numpy** para criar um vetor e o exibimos na tela, neste caso acessamos a função com o nome renomeado da biblioteca seguido do ponto e do nome da função.

```
| np.arange(1,6)
```

## §11.2 Bibliotecas nativas

As bibliotecas nativas são um conjunto de bibliotecas padrão que já vêm instaladas com o Python e oferecem uma ampla gama de funcionalidades. Algumas das bibliotecas nativas mais importantes incluem:

- **datetime**: Fornece classes e funções para manipulação de datas e horas.
- **math**: Fornece funções matemáticas avançadas, como trigonometria, logaritmos e funções

exponenciais. A **numpy** normalmente é utilizada no lugar da biblioteca **math**, pois apresenta mais funcionalidades e melhor performance.

- **random**: Fornece funções para geração de números aleatórios. A **numpy.random** é bastante utilizada no lugar da biblioteca **random**, pois apresenta mais funcionalidades e melhor performance.
- **time**: Fornece funções relacionadas ao tempo, como medição de tempo de execução.
- **os**: Fornece funções para interação com o sistema operacional, como manipulação de arquivos e diretórios.
- **sys**: Fornece funções e variáveis relacionadas ao interpretador Python, como argumentos de linha de comando e redirecionamento de entrada/saída.
- **json**: Fornece funções para trabalhar com o formato de dados JSON.
- **re**: Fornece suporte a expressões regulares para busca e manipulação de padrões em strings.
- **csv**: Fornece funcionalidades para leitura e escrita de arquivos CSV (Comma Separated Values).
- **urllib**: Fornece funções para trabalhar com URLs, como fazer requisições HTTP e baixar conteúdo da web.

Uma listagem completa das bibliotecas nativas pode ser encontrada em:

<https://docs.python.org/pt-br/3/library/>

#### Exemplo 11.2.1. Manipulando datas

```
1  from datetime import datetime
2
3  # Obtendo a data e hora atual
4  data_hora_atual = datetime.now()
5  print(data_hora_atual)
6
7  # Formatando uma data
8  data_formatada = data_hora_atual.strftime("%d/%m/%Y")
9  print(data_formatada)
10
11 # Calculando diferença de tempo
12 data_aniversario = datetime(1990, 5, 15)
13 diferenca_tempo = data_hora_atual - data_aniversario
14 print(diferenca_tempo.days)
```

Neste exemplo, importamos a classe **datetime** da biblioteca **datetime** usando a forma **from datetime import datetime**. Essa biblioteca fornece classes e funções para manipulação de datas e horas. No exemplo, utilizamos a classe **datetime** para obter a data e hora atual usando **datetime.now()**. Em seguida, formatamos a data no formato desejado usando o método **strftime**.

**Exemplo 11.2.2.** Medindo tempo

```
1  import time
2
3  # Atraso de 1 segundo
4  time.sleep(1)
5
6  # Medição de tempo de execução
7  inicio = time.time()
8  # Código a ser medido
9  # .....
10 #
11 fim = time.time()
12 tempo_execução = fim - inicio
13 print("Tempo de execução:", tempo_execução)
```

Neste exemplo, importamos a biblioteca `time` usando a forma `import time`. Essa biblioteca fornece funções relacionadas ao tempo. No exemplo, utilizamos a função `sleep(1)` para fazer um atraso de 1 segundo. Também utilizamos as funções `time.time()` para medir o tempo de execução de um trecho de código.

## §11.3 Mais bibliotecas

Existem diversas bibliotecas disponíveis em Python que são amplamente utilizadas na área de ciência de dados. Essas bibliotecas fornecem uma ampla gama de funcionalidades para análise, manipulação e visualização de dados. Algumas das bibliotecas mais populares para ciência de dados em Python são:

- **NumPy**: Biblioteca fundamental para computação científica em Python. Ela fornece estruturas de dados eficientes para lidar com arrays multidimensionais e funções matemáticas de alto desempenho.
- **Pandas**: Biblioteca que oferece estruturas de dados e ferramentas de análise de dados fáceis de usar. O **Pandas** permite manipular e analisar dados de forma eficiente, incluindo recursos para limpeza, filtragem, agregação e visualização.
- **Matplotlib**: Biblioteca de visualização de dados em 2D que permite criar gráficos estáticos, gráficos interativos e animações. É amplamente utilizada para criar gráficos de linhas, barras, dispersão, histogramas, entre outros.
- **Seaborn**: Biblioteca de visualização estatística que funciona em conjunto com o **Matplotlib**. Ela fornece uma interface de alto nível para criar gráficos estatísticos atraentes e informativos.
- **Scikit-learn**: Biblioteca de aprendizado de máquina em Python que oferece uma variedade de algoritmos e ferramentas para tarefas de aprendizado supervisionado e não supervisionado. É amplamente utilizada para tarefas como classificação, regressão, agrupamento e seleção de recursos.
- **TensorFlow**: Biblioteca de aprendizado de máquina de código aberto desenvolvida pelo Google. Ela permite criar e treinar modelos de aprendizado de máquina, especialmente modelos de redes neurais profundas, de forma eficiente e escalável.

- **Keras**: Biblioteca de aprendizado de máquina de alto nível que funciona em conjunto com o **TensorFlow**. Ela fornece uma interface simples e intuitiva para a construção e treinamento de redes neurais.
- **SciPy**: Biblioteca que fornece um conjunto de algoritmos e funções matemáticas de alto nível para ciência e engenharia. Ela abrange áreas como otimização, álgebra linear, processamento de sinais, estatística, entre outros.
- **Statsmodels**: Biblioteca que fornece modelos estatísticos e ferramentas para análise de dados. Ela inclui uma variedade de modelos de regressão, testes estatísticos e técnicas de análise exploratória de dados.
- **NLTK**: Biblioteca de processamento de linguagem natural que fornece ferramentas e recursos para tarefas como tokenização, stemming, lematização, marcação de partes do discurso, entre outros.

Essas são apenas algumas das bibliotecas disponíveis para ciência de dados em Python. Cada uma delas possui sua própria funcionalidade e pode ser combinada para realizar análises complexas e sofisticadas.

### Instalando bibliotecas de terceiros

Bibliotecas de terceiros precisam ser instaladas, a grande maioria delas é distribuída pelo o sistema de gerenciamento de pacotes padrão para Python, o **pip**. Ele permite que você instale, atualize e remova bibliotecas de terceiros de forma fácil e eficiente. O **pip** é uma ferramenta de linha de comando que vem pré-instalada com a maioria das distribuições do Python.

Para utilizá-la em linha de comando a sintaxe é:

```
| pip install nome_da_biblioteca
```

Ou:

```
| python -m pip install nome_da_biblioteca
```

As bibliotecas disponíveis para instalação via **pip** estão listadas no PyPI (Python Package Index). O PyPI é o repositório de pacotes Python mais popular, mas outros repositórios também podem ser utilizados pelo **pip**. Nele, você pode encontrar informações sobre diferentes bibliotecas, incluindo documentação, exemplos de uso, versões disponíveis e links para download.

O Google Colaboratory já possui instalado a grande maioria das bibliotecas de terceiros utilizadas para ciência de dados, porém caso seja necessário instalar mais alguma também é possível, como no exemplo:

```
| !pip install ktrain
```

#### Exemplo 11.3.1. Usando a biblioteca pandas

```
1 | import pandas as pd
2 |
3 | # Criando um DataFrame a partir de um dicionário
4 | data = {'Nome': ['João', 'Maria', 'Pedro', 'Ana'],
5 |         'Idade': [25, 30, 35, 28],
```



```
6         'Cidade': ['São Paulo', 'Rio de Janeiro', 'Belo
              Horizonte', 'Curitiba'])
7
8 df = pd.DataFrame(data)
9
10 # Imprimindo o DataFrame
11 print(df)
12
13 # Acessando colunas específicas do DataFrame
14 print(df['Nome'])
15
16 # Filtrando linhas do DataFrame com base em uma condição
17 filtro = df['Idade'] > 28
18 print(df[filtro])
19
20 # Adicionando uma nova coluna ao DataFrame
21 df['Sexo'] = ['Masculino', 'Feminino', 'Masculino', 'Feminino']
22 print(df)
23
24 # Realizando operações estatísticas no DataFrame
25 print(df.describe())
26
27 # Salvando o DataFrame em um arquivo CSV
28 df.to_csv('dados.csv', index=False)
```

Neste exemplo, importamos a biblioteca pandas e a renomeamos como `pd`. Em seguida, criamos um `DataFrame` a partir de um dicionário de dados. Demonstramos como acessar colunas específicas do `DataFrame`, filtrar linhas com base em uma condição, adicionar uma nova coluna ao `DataFrame` e realizar operações estatísticas. Por fim, salvamos o `DataFrame` em um arquivo CSV (tipo de arquivo muito utilizado para armazenar tabelas de dados) utilizando a função `to_csv()`. O pandas oferece uma ampla gama de funcionalidades para manipulação e análise de dados, facilitando o trabalho com conjuntos de dados estruturados.

#### Exemplo 11.3.2. Usando a biblioteca numpy

```
1 import numpy as np
2
3 # Criando um array unidimensional
4 arr1 = np.array([1, 2, 3, 4, 5])
5
6 # Imprimindo o array
7 print(arr1)
8 # Saída: [1 2 3 4 5]
9
10 # Criando um array bidimensional
11 arr2 = np.array([[1, 2, 3], [4, 5, 6]])
12
13 # Imprimindo o array
14 print(arr2)
```

```
15 # Saída:
16 # [[1 2 3]
17 #  [4 5 6]]
18
19 # Acessando elementos específicos do array
20 print(arr2[0, 1])
21 # Saída: 2
22
23 # Realizando operações matemáticas com o array
24 arr3 = np.array([1, 2, 3])
25 arr4 = np.array([4, 5, 6])
26
27 # Soma dos arrays
28 soma = arr3 + arr4
29 print(soma)
30 # Saída: [5 7 9]
31
32 # Produto dos arrays
33 produto = arr3 * arr4
34 print(produto)
35 # Saída: [4 10 18]
36
37 # Calculando a média dos elementos do array
38 media = np.mean(arr4)
39 print(media)
40 # Saída: 5.0
41
42 # Fazendo uma operação em todos os elementos do array
43 dobro = arr1 * 2
44 print(dobro)
45 # Saída: [ 2  4  6  8 10]
```

Neste exemplo, importamos a biblioteca NumPy e a renomeamos como np. Em seguida, criamos diferentes arrays usando a função np.array(). Demonstramos operações matemáticas básicas, como soma e multiplicação de arrays, cálculo da média e do desvio padrão, além de acesso e manipulação de elementos específicos do array. Também utilizamos a função np.random.rand() para gerar um array com valores aleatórios e a função np.mean() para calcular a média dos elementos de um array. Esses são apenas alguns exemplos de uso da biblioteca NumPy, que oferece uma ampla gama de funcionalidades para trabalhar com arrays e realizar cálculos numéricos eficientes.

# 12 | NumPy, parte 1

O NumPy é uma biblioteca fundamental para computação científica em Python. Ela fornece suporte para arrays multidimensionais e uma ampla gama de funções matemáticas que podem ser aplicadas a esses arrays.

Com o NumPy, é possível realizar operações matemáticas de maneira eficiente, manipular dados de forma eficaz e realizar cálculos complexos de maneira rápida. Ele também oferece recursos avançados, como transposição e reformatação de arrays, indexação e fatiamento, manipulação de dados booleanos, ordenamento e funções de conjunto.

O NumPy é amplamente utilizado em áreas como ciência de dados, processamento de imagens, simulações e modelagem matemática. É uma biblioteca essencial para qualquer pessoa que trabalhe com análise numérica e computação científica em Python.

## §12.1 Arrays NumPy

A biblioteca NumPy oferece suporte a arrays multidimensionais, que são estruturas de dados eficientes e poderosas para armazenar e manipular dados numéricos. Os arrays NumPy são homogêneos, ou seja, todos os elementos devem ser do mesmo tipo de dados. Eles fornecem uma maneira eficiente de realizar operações matemáticas em grande escala.

Para criar um array NumPy, podemos usar a função `numpy.array()` e fornecer uma lista de elementos.

```
array = np.array([1, 2, 3, 4, 5])
```

Além disso, o NumPy fornece uma série de funções e métodos para realizar operações com arrays, como cálculos estatísticos, ordenamento, reformatação, entre outros. Essas funcionalidades tornam o NumPy uma biblioteca essencial para computação científica e análise de dados em Python.

Arrays unidimensionais, como o exemplo acima, podem ser acessadas de maneira similar a listas:

```
print(array[2]) # Saída: 3
```

## §12.2 Operações com Arrays

É possível aplicar os operadores aritméticos do Python a arrays NumPy, a principal característica dessa aplicação é que os operadores são aplicados elemento a elemento nos arrays, ou seja, cada elemento do array é combinado com o elemento correspondente do outro array, resultando em um novo array com os resultados das operações.

Isso significa que, ao realizar uma operação aritmética entre dois arrays, como adição, subtração, multiplicação ou divisão, cada elemento do primeiro array será combinado com o elemento correspondente do segundo array para obter o resultado. Dessa forma, o tamanho dos arrays deve ser compatível para que a operação possa ser realizada adequadamente.

**Exemplo 12.2.1.** Operações aritméticas com arrays

```
1 import numpy as np
2
3 # Criando dois arrays
4 array1 = np.array([1, 2, 3])
5 array2 = np.array([4, 5, 6])
6
7 # Adição de arrays
8 resultado_soma = array1 + array2
9 print(resultado_soma)
10 # Saída: [5 7 9]
11
12 # Subtração de arrays
13 resultado_subtracao = array1 - array2
14 print(resultado_subtracao)
15 # Saída: [-3 -3 -3]
16
17 # Multiplicação de arrays
18 resultado_multiplicacao = array1 * array2
19 print(resultado_multiplicacao)
20 # Saída: [4 10 18]
21
22 # Divisão de arrays
23 resultado_divisao = array1 / array2
24 print(resultado_divisao)
25 # Saída: [0.25 0.4 0.5]
```

Nesse exemplo, cada operação é realizada elemento a elemento nos arrays `array1` e `array2`, resultando nos novos arrays:

- `resultado_soma`
- `resultado_subtracao`
- `resultado_multiplicacao`
- `resultado_divisao`

Essa capacidade de aplicar operadores aritméticos a arrays no NumPy facilita a realização de cálculos e operações matemáticas em conjuntos de dados de forma rápida e eficiente.

Também é possível aplicar os operadores aritméticos entre arrays e valores escalares. Neste caso o resultado é uma nova array, do mesmo tamanho da array original, onde cada elemento é o respectivo elemento da array original operado com o valor escalar.

**Exemplo 12.2.2.** Aplicando operadores entre arrays e escalares

```
1 import numpy as np
2
3 # Criando um array NumPy
4 array = np.array([1, 2, 3, 4, 5])
5
6 # Multiplicando o array por um valor escalar
7 result1 = array * 2
8
9 # Dividindo o array por um valor escalar
10 result2 = array / 3
11
12 # Somando o array com um valor escalar
13 result3 = array + 10
14
15 # Subtraindo um valor escalar do array
16 result4 = array - 2
17
18 # Imprimindo os resultados
19 print(result1) # Saída: [2, 4, 6, 8, 10]
20 print(result2) # Saída: [0.33333333, 0.66666667, 1.0,
21                  1.33333333, 1.66666667]
22 print(result3) # Saída: [11, 12, 13, 14, 15]
23 print(result4) # Saída: [-1, 0, 1, 2, 3]
```

Neste exemplo, criamos um array NumPy com os valores de 1 a 5. Em seguida, aplicamos operadores aritméticos para realizar operações entre o array e valores escalares.

No caso do **result1**, multiplicamos cada elemento do array por 2, resultando em [2, 4, 6, 8, 10]. No **result2**, dividimos cada elemento do array por 3. No **result3**, somamos 10 a cada elemento do array. E no **result4**, subtraímos 2 de cada elemento do array.

## §12.3 Arrays multidimensionais

Além de arrays unidimensionais, o NumPy também suporta arrays multidimensionais, que podem ter duas ou mais dimensões. Esses arrays são essenciais para representar dados em forma de matrizes ou tensores, comumente utilizados em problemas de ciência de dados e computação científica.

Para criar um array multidimensional, podemos passar uma lista de listas como argumento para a função `numpy.array()`. Cada lista interna representa uma linha da matriz e deve ter o mesmo número de elementos.

```
| array2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

Podemos acessar elementos específicos de um array multidimensional usando índices. Os índices seguem a notação [linha, coluna], em que a primeira dimensão refere-se às linhas e a segunda dimensão refere-se às colunas.

```
| print(array2d[0, 1]) # Acessando a primeira linha e segunda coluna (2)
```

Para acessar mais dimensões é só colocar os índices necessários separados por vírgula. O formato de uma array pode ser acessado pelo atributo **shape**:

```
array2d.shape # (3, 3)
```

Este atributo armazena uma tupla onde cada posição é o comprimento da array no respectivo eixo.

#### Exemplo 12.3.1. Operações com arrays multidimensionais

```
1 # Define dois arrays
2 array1 = np.array([[1, 2], [3, 4]])
3 array2 = np.array([[5, 6], [7, 8]])
4
5 # Soma dos arrays
6 resultado_soma = array1 + array2
7 print(resultado_soma)
8
9 # Multiplicação dos arrays
10 resultado_multiplicacao = array1 * array2
11 print(resultado_multiplicacao)
```

Primeiramente, são criados dois arrays multidimensionais **array1** e **array2** utilizando a função **numpy.array()**. Em seguida, são realizadas operações de soma e multiplicação com esses arrays.

Os exemplos 12.2.1 e 12.3.1 ilustram como as operações aritméticas são aplicadas elemento a elemento em arrays no NumPy. Essa capacidade de realizar operações com eficiência em grandes conjuntos de dados é uma das principais vantagens do uso do NumPy para computação científica e análise de dados.

#### Mais formas para criação de arrays NumPy

Além da função **np.array()**, existem várias maneiras de criar arrays NumPy, algumas delas são:

- **numpy.zeros()**: cria um array NumPy preenchido com zeros. É possível especificar o número de elementos (para arrays multidimensionais passar uma tupla com o tamanho da array em cada dimensão) e o tipo de dados:

```
zeros_array = np.zeros((3, 4), dtype=int)
```

Nesse exemplo, é criado um array bidimensional de 3 linhas e 4 colunas, preenchido com zeros inteiros.

- **numpy.ones()**: cria um array NumPy preenchido com uns. É possível especificar o número de elementos e o tipo de dados:

```
ones_array = np.ones(10, dtype=float)
```

Nesse exemplo, é criado um array unidimensional de 10 elementos, preenchido com uns do tipo float.

- **numpy.arange()**: cria um array NumPy com valores sequenciais em um intervalo

especificado. É possível definir o valor inicial, o valor final e o incremento, de maneira similar a função `range()`:

```
arange_array = np.arange(0, 10, 2)
```

Nesse exemplo, é criado um array unidimensional com valores de 0 a 8 (exclusivo), com incremento de 2: `[0, 2, 4, 6, 8]`.

- `numpy.linspace()`: cria um array NumPy com valores igualmente espaçados em um intervalo especificado. É possível definir o valor inicial, o valor final e o número de elementos desejados:

```
linspace_array = np.linspace(0, 1, 5)
```

Nesse exemplo, é criado um array unidimensional com 5 elementos, espaçados igualmente entre 0 e 1: `[0. , 0.25, 0.5 , 0.75, 1.]`.

Esses são apenas alguns dos métodos disponíveis para criar arrays NumPy. Cada método possui seus próprios parâmetros e opções, permitindo a criação flexível de arrays para atender às necessidades específicas de cada aplicação.

## §12.4 Indexação e fatiamento

A indexação em arrays NumPy funciona de forma semelhante à indexação em listas Python. Os índices são baseados em zero, o que significa que o primeiro elemento do array tem o índice 0. Para acessar um elemento específico, basta utilizar o índice ou fatiamento desejado entre colchetes. Em arrays multidimensionais a indexação segue o mesmo princípio, utilizando colchetes e separando os índices/fatiamentos por vírgulas.

### Exemplo 12.4.1. Indexando e fatiando arrays.

```
1 import numpy as np
2
3 # Criação dos arrays
4 array1d = np.array([1, 2, 3])
5 array2d = np.array([[3, 5, 2], [4, 2, 1], [0, 1, 3]])
6
7 # Acessando elementos específicos
8 print(array1d[1]) # Deve imprimir: 2
9 print(array2d[1, 0]) # Deve imprimir: 4
10 print(array2d[2, 0]) # Deve imprimir: 0
11
12 # Fatiamento dos arrays
13 print(array1d[0:2]) # Deve imprimir: [1, 2]
14 print(array2d[0:2, 1:3]) # Deve imprimir: [[5, 2], [2, 1]]
15
16 # Acessando uma coluna inteira de um array bidimensional
17 print(array2d[:, 1]) # Deve imprimir: [5, 2, 1]
18
19 # Acessando uma linha inteira de um array bidimensional
```

```
20 | print(array2d[0, :]) # Deve imprimir: [3, 5, 2]
```

Neste exemplo, são criados dois arrays numpy: **array1d**, que é unidimensional, e **array2d**, que é bidimensional. Em seguida, são demonstradas diferentes formas de acessar elementos ou subconjuntos desses arrays.

Primeiramente, são utilizados os índices para acessar elementos específicos dos arrays, sendo impressos os valores **2**, **1** e **0**.

Em seguida, são realizados fatiamentos nos arrays, imprimindo diferentes subconjuntos de elementos.

Por fim, é demonstrado como acessar uma coluna inteira de um array bidimensional, resultando em **[5, 2, 3]**, e como acessar uma linha inteira de um array bidimensional, resultando em **[3, 5, 2]**. Esses exemplos ilustram a flexibilidade do numpy para acessar e manipular elementos em arrays multidimensionais.

### §12.4.2 Indexação booleana

Os operadores relacionais também podem ser utilizados para comparar valores em arrays NumPy. Neste caso o operador é aplicado elemento a elemento, como nos operadores aritméticos e retornam um array booleano, onde cada elemento indica se a condição é verdadeira ou falsa para o respectivo elemento dos arrays originais.

#### Exemplo 12.4.3. Operadores relacionais aplicados em arrays

```
1 | import numpy as np
2 |
3 | # Criando arrays para serem comparadas
4 | array1 = np.array([1, 2, 3, 7, 8])
5 | array2 = np.array([2, 3, 4, 5, 6])
6 |
7 | # Comparando os elementos dos arrays
8 | resultado = array1 > array2
9 | print(resultado) # Saída: [False False False True True]
```

Nesse exemplo, o operador **>** é aplicado elemento a elemento nos arrays **array1** e **array2**. Como nenhum elemento de **array1** é maior do que o elemento correspondente de **array2**, o resultado é um array booleano com os valores: **[False False False True True]**.

Os arrays booleanos gerados pela aplicação de operadores relacionais com arrays podem ser utilizados para indexar os mesmos arrays. Esta é uma técnica para acessar e manipular elementos com base em uma condição booleana. Nesse tipo de indexação, criamos uma máscara booleana, que é a array booleana resultante de um operador relacional, e que especifica quais elementos do array original queremos acessar ou modificar.

#### Exemplo 12.4.4. Usando indexação booleana

```
1 | import numpy as np
2 |
3 | # Criando um array NumPy
4 | array = np.array([1, 2, 3, 4, 5])
5 |
6 | # Criando uma máscara booleana para selecionar os elementos
```



```
    maiores que 3
7  mask = array > 3
8
9  # Aplicando a máscara para obter apenas os elementos que atendem à
    condição
10 filtered_array = array[mask]
11
12 # Imprimindo o resultado
13 print(filtered_array) # Saída: [4, 5]
14
15 # Modificando os valores dos elementos maiores que 3 para 0
16 array[array > 3] = 0
17
18 # Imprimindo o array modificado
19 print(array) # Saída: [1, 2, 3, 0, 0]
```

Neste exemplo, criamos um array NumPy com os valores de 1 a 5. Em seguida, criamos uma máscara booleana utilizando a expressão `array > 3`, que retorna um array de valores booleanos indicando quais elementos são maiores que 3.

A máscara resultante é `[False, False, False, True, True]`, indicando que apenas os dois últimos elementos do array atendem à condição. Em seguida, aplicamos essa máscara para selecionar apenas os elementos que são maiores que 3, utilizando `array[mask]`. O resultado é o array `[4, 5]`.

Em seguida, utilizamos a indexação booleana para modificar os valores dos elementos maiores que 3 para 0, utilizando `array[array > 3] = 0` (observe que nesse caso a máscara booleana foi criada no momento do uso). Após essa operação, o array resultante é `[1, 2, 3, 0, 0]`.

## §12.5 Transposição e Reformatação

A biblioteca NumPy oferece funcionalidades para transposição e reformatação de arrays, permitindo manipular a forma e a organização dos dados.

A transposição de um array é obtida utilizando o método `transpose()` ou o atributo `T`. Esse processo troca as linhas pelas colunas, de modo que os elementos que antes estavam nas linhas agora estão nas colunas e vice-versa.

### Exemplo 12.5.1. Transposição de um array

```
1  import numpy as np
2
3  # Criando um array bidimensional
4  array2d = np.array([[1, 2, 3], [4, 5, 6]])
5
6  # Aplicando a transposição ao array
7  array_transposed = array2d.transpose()
8  # Ou equivalente:
9  array_transposed = array2d.T
10
11 # Imprimindo o array original
12 print("Array original:")
```

```
13 print(array2d)
14
15 # Imprimindo o array transposto
16 print("rray transposto:")
17 print(array_transposed)
```

Nesse exemplo, a transposição do array bidimensional `array2d` é realizada, resultando em um novo array onde as linhas e colunas foram invertidas.

A reformatação de um array é obtida utilizando o método `reshape()`. Esse método permite alterar a forma do array, redimensionando-o de acordo com a quantidade de elementos desejada. É importante que o número de elementos do array seja compatível com a nova forma especificada.

#### Exemplo 12.5.2. Reformatação de um array

```
1 import numpy as np
2
3 # Criando um array unidimensional
4 array1d = np.array([1, 2, 3, 4, 5, 6])
5
6 # Reformatando o array para uma matriz de 2 linhas e 3 colunas
7 array_reshaped = array1d.reshape(2, 3)
8
9 # Imprimindo o array original
10 print("Array original:")
11 print(array1d)
12 # [1 2 3 4 5 6]
13
14 # Imprimindo o array reformatado
15 print("\nArray reformatado:")
16 print(array_reshaped)
17 # [[1 2 3]
18 #   [4 5 6]]
```

Nesse exemplo, a reformatação do array `array1d` é realizada, resultando em um novo array unidimensional com 2 linhas e 3 colunas. Os elementos do array original são distribuídos na nova estrutura de acordo com a ordem de indexação.

A transposição e a reformatação de arrays são operações úteis para ajustar a organização dos dados de acordo com as necessidades de análise e manipulação.

# 13 | NumPy, parte 2

## §13.1 Funções Universais

As funções universais (ufuncs) no NumPy são funções que operam em ndarrays de uma maneira element-wise, o que significa que elas são aplicadas de forma independente a cada elemento de um array. Essas funções são fundamentais para o NumPy, pois proporcionam uma abstração de alto nível para operações eficientes em arrays de dados.

As funções universais são tipicamente muito mais eficientes do que as operações equivalentes em Python puro, pois elas são implementadas em C, que é muito mais rápido do que Python para operações de baixo nível.

Algumas das funções universais mais úteis no NumPy incluem:

- **Funções matemáticas:** `add`, `subtract`, `multiply`, `divide`, `log`, `exp`, `sqrt`, `sin`, `cos`, `tan`, etc. Essas funções realizam operações matemáticas básicas e funções elementares.
- **Funções de agregação:** `sum`, `prod`, `cumsum`, `cumprod`, `mean`, `std`, `var`, etc. Essas funções fornecem estatísticas básicas sobre o conteúdo de um array.
- **Funções de comparação:** `greater`, `less`, `equal`, `not_equal`, `greater_equal`, `less_equal`. Essas funções permitem comparar elementos de dois arrays e retornar um array de booleans.
- **Funções lógicas:** `logical_and`, `logical_or`, `logical_not`, etc. Essas funções realizam operações lógicas em arrays booleanos.
- **Funções de binárias:** `bitwise_and`, `bitwise_or`, `bitwise_xor`, `invert`, etc. Essas funções permitem manipular a representação binária de inteiros.
- **Funções de arredondamento:** `around`, `floor`, `ceil`, etc. Essas funções permitem arredondar os valores em um array para o número inteiro mais próximo, para o inteiro mais próximo abaixo ou para o inteiro mais próximo acima, respectivamente.

Essas são apenas algumas das muitas funções universais disponíveis no NumPy. Para obter a lista completa, você pode consultar a documentação oficial do NumPy.

<https://numpy.org/doc/stable/reference/ufuncs.html>.

### Exemplo 13.1.1. Utilizando funções universais

```
1  import numpy as np
2
3  # Criando um array 1D
4  arr = np.array([1, 2, 3, 4, 5])
5
6  # Usando a função matemática sqrt (raiz quadrada)
7  sqrt_arr = np.sqrt(arr)
8  print(f"Raiz quadrada do array: {sqrt_arr}")
9  # Raiz quadrada do array: [1.          1.41421356  1.73205081  2.
10                                2.23606798]
11
12 # Usando a função de agregação sum (soma)
```

```
12 sum_arr = np.sum(arr)
13 print(f"Soma dos elementos do array: {sum_arr}")
14 # Soma dos elementos do array: 15
15
16 # Usando a função de comparação less (menor que)
17 less_arr = np.less(arr, 4)
18 print(f"Elementos do array menores que 4: {less_arr}")
19 # Elementos do array menores que 4: [ True  True  True False
    False]
20
21 # Usando a função lógica logical_not (negação lógica)
22 not_arr = np.logical_not(less_arr)
23 print(f"Negação lógica do array anterior: {not_arr}")
24 # Negação lógica do array anterior: [False False False  True
    True]
```

Esse exemplo começa criando um array 1D com 5 elementos. Em seguida, ele utiliza a função `np.sqrt` para calcular a raiz quadrada de cada elemento do array. A função `np.sum` é usada para calcular a soma de todos os elementos do array. A função `np.less` é usada para criar um novo array booleano, onde cada elemento é `True` se o elemento correspondente no array original é menor que 4 e `False` caso contrário. Por fim, a função `np.logical_not` é usada para inverter os valores booleanos do array criado pela função `np.less`.

## §13.2 Funções de Álgebra Linear

As funções de álgebra linear fornecem a capacidade de realizar uma variedade de operações de álgebra linear eficientemente. Essas funções estão localizadas no submódulo `numpy.linalg` e incluem várias funções que são essenciais para a realização de cálculos matemáticos complexos.

Aqui estão algumas das funções de álgebra linear mais comumente usadas no NumPy:

- `dot(a, b)`: Calcula o produto de dois arrays. Se os arrays `a` e `b` forem ambos matrizes 1-D, a função `'dot'` irá realizar uma multiplicação de matrizes.
- `vdot(a, b)`: Esta função retorna o produto escalar de dois vetores. É a soma dos produtos dos elementos correspondentes dos dois vetores.
- `inner(a, b)`: Esta função retorna o produto interno de dois arrays.
- `matmul(a, b)`: Esta função retorna o produto matricial de dois arrays.
- `trace(a)`: Retorna a soma dos elementos da diagonal principal de um array 2-D (uma matriz).
- `det(a)`: Calcula o determinante de uma matriz.
- `eig(a)`: Calcula os autovalores e autovetores de uma matriz.
- `inv(a)`: Calcula a inversa de uma matriz.
- `solve(a, b)`: Resolve o sistema linear de equações  $ax = b$  para  $x$ , onde  $a$  é uma matriz e  $b$  é um vetor.
- `lstsq(a, b)`: Retorna a solução de mínimos quadrados para um sistema linear.

Estas funções podem ser extremamente úteis para realizar cálculos complexos de forma eficiente. Por exemplo, a função **solve** pode ser usada para resolver sistemas de equações lineares, enquanto a função **eig** pode ser usada para calcular os autovalores e autovetores de uma matriz, o que é útil em uma variedade de aplicações, incluindo a solução de problemas de valor próprio.

### §13.3 Funções de Operações de Conjunto

As funções de operações de conjunto são úteis para manipular e analisar conjuntos de dados. O NumPy fornece um conjunto de funções para realizar operações de conjunto em arrays. Essas funções são particularmente úteis para trabalhar com arrays de dados únicos e ordenados.

Aqui estão algumas das funções de operações de conjunto mais comumente usadas no NumPy:

- **np.unique(a)**: Esta função retorna os elementos únicos de um array, ordenados.
- **np.in1d(a, b)**: Esta função retorna um array booleano que indica se cada elemento do array 'a' está contido no array 'b'.
- **np.intersect1d(a, b)**: Esta função retorna a interseção ordenada de dois arrays, ou seja, os elementos comuns a ambos os arrays.
- **np.setdiff1d(a, b)**: Esta função retorna a diferença de conjunto de dois arrays, ou seja, os elementos em 'a' que não estão em 'b'.
- **np.union1d(a, b)**: Esta função retorna a união de dois arrays, ou seja, todos os elementos únicos em ambos os arrays.
- **np.setxor1d(a, b)**: Esta função retorna a diferença simétrica de dois arrays, ou seja, os elementos que estão em um dos arrays, mas não em ambos.

Estas funções podem ser extremamente úteis para analisar conjuntos de dados e encontrar interseções, diferenças e similaridades entre diferentes conjuntos de dados.

#### Exemplo 13.3.1. Utilizando funções de operações de conjunto

```
1 import numpy as np
2
3 # Definindo dois arrays
4 array1 = np.array([1, 2, 3, 4, 5])
5 array2 = np.array([4, 5, 6, 7, 8])
6
7 # Encontrando os valores únicos do primeiro array
8 unique_values = np.unique(array1)
9 print("Valores únicos do primeiro array:", unique_values)
10
11 # Verificando quais elementos do primeiro array estão presentes
    no segundo array
12 in_second = np.in1d(array1, array2)
13 print("Elementos do primeiro array presentes no segundo:",
    in_second)
14
15 # Encontrando elementos comuns aos dois arrays
16 common_elements = np.intersect1d(array1, array2)
17 print("Elementos comuns aos dois arrays:", common_elements)
```

```
18
19 # Encontrando elementos no primeiro array que não estão no
    segundo array
20 diff_elements = np.setdiff1d(array1, array2)
21 print("Elementos do primeiro array que não estão no segundo:",
    diff_elements)
22
23 # Combinando os elementos de ambos os arrays
24 union_elements = np.union1d(array1, array2)
25 print("Combinação de ambos os arrays:", union_elements)
26
27 # Encontrando elementos que estão em um dos arrays, mas não em
    ambos
28 xor_elements = np.setxor1d(array1, array2)
29 print("Elementos que estão em um dos arrays, mas não em ambos:",
    xor_elements)
```

Neste exemplo, dois arrays são definidos inicialmente. A função `np.unique` é usada para encontrar os valores únicos no primeiro array. A função `np.in1d` é usada para verificar quais elementos do primeiro array estão presentes no segundo. A função `np.intersect1d` é usada para encontrar os elementos comuns aos dois arrays. A função `np.setdiff1d` é usada para encontrar os elementos presentes no primeiro array e que não estão no segundo. A função `np.union1d` é usada para combinar os elementos de ambos os arrays, mantendo apenas uma ocorrência de cada elemento único. Por fim, a função `np.setxor1d` é usada para encontrar os elementos que estão presentes em um dos arrays, mas não em ambos.

## §13.4 Geração de números pseudoaleatórios

Números aleatórios são um recurso fundamental em muitas áreas da ciência da computação, desde a simulação e modelagem, passando por machine learning, até a criptografia. No entanto, verdadeiramente números aleatórios são difíceis de gerar em um computador, que é fundamentalmente uma máquina determinística. Por isso, usamos números pseudoaleatórios, que são gerados por algoritmos determinísticos, mas parecem estatisticamente aleatórios.

O módulo NumPy fornece várias funções para gerar números pseudoaleatórios, através do submódulo `numpy.random`. Alguns dos métodos mais comuns incluem:

- `np.random.rand()`: Gera números aleatórios uniformemente distribuídos no intervalo  $[0, 1)$ .
- `np.random.randn()`: Gera números aleatórios a partir de uma distribuição normal padrão (média 0, desvio padrão 1).
- `np.random.randint(low, high=None, size=None)`: Gera números inteiros aleatórios a partir de um intervalo especificado.
- `np.random.choice(a, size=None, replace=True, p=None)`: Gera uma amostra aleatória de um array 1-D especificado. Você pode especificar se a amostragem é com ou sem reposição e as probabilidades associadas a cada entrada.
- `np.random.seed()`: Inicializa o gerador de números aleatórios. Se você definir a semente, pode obter uma sequência reprodutível de números pseudoaleatórios.

**Exemplo 13.4.1.** Gerando números pseudoaleatórios

```
1 import numpy as np
2
3 # Configurando a semente do gerador de números aleatórios
4 np.random.seed(0)
5
6 # Criando um array de 5 números aleatórios uniformemente
   distribuídos entre 0 e 1
7 random_array1 = np.random.rand(5)
8 print("Array de 5 números aleatórios uniformemente distribuídos
   entre 0 e 1:", random_array1)
9
10 # Criando um array 2x2 de números aleatórios a partir de uma
   distribuição normal padrão
11 random_array2 = np.random.randn(2, 2)
12 print("Array 2x2 de números aleatórios a partir de uma
   distribuição normal padrão:", random_array2)
13
14 # Gerando um único número inteiro aleatório entre 0 e 10
15 random_int = np.random.randint(0, 10)
16 print("Número inteiro aleatório entre 0 e 10:", random_int)
17
18 # Selecionando três elementos aleatoriamente de um dado array
19 array = np.array([10, 20, 30, 40, 50])
20 random_choice = np.random.choice(array, 3)
21 print("Três elementos selecionados aleatoriamente do array:",
   random_choice)
```

Este exemplo começa configurando a semente do gerador de números aleatórios para 0 com `np.random.seed(0)`. Isso garante que a sequência de números pseudoaleatórios gerados será a mesma sempre que o código for executado. Em seguida, cria um array de 5 números aleatórios uniformemente distribuídos entre 0 e 1 com `np.random.rand(5)`. Depois, cria um array 2x2 de números aleatórios a partir de uma distribuição normal padrão com `np.random.randn(2, 2)`. O código então gera um único número inteiro aleatório entre 0 e 10 com `np.random.randint(0, 10)` e seleciona três elementos aleatoriamente de um dado array com `np.random.choice`.

Lembre-se de que a geração de números aleatórios pode ter um impacto significativo nos resultados de seu código, especialmente em algoritmos de machine learning, que podem depender fortemente de inicializações aleatórias. Portanto, é importante entender como usar corretamente o módulo de números aleatórios do NumPy.

# 14

## Captura de erros, manipulação de arquivos e simulações

### §14.1 Captura de erros

Capturar erros é uma prática essencial em programação para lidar com exceções ou erros que possam ocorrer durante a execução do código. Isso é especialmente útil quando lidamos com entradas do usuário ou quando executamos operações que podem falhar sob certas condições.

Python fornece a estrutura **try/except** para capturar e lidar com erros. A ideia básica é que o código que pode causar um erro é colocado dentro de um bloco **try**, e se ocorrer um erro, o fluxo do programa é imediatamente transferido para o bloco **except**. A sintaxe básica dessa estrutura é:

```
1 try:
2     # Código que pode causar algum erro
3 except Exception: # Ou 'except:' para não especificar uma exceção
4     # Código para ser executado caso ocorra um erro
```

#### Exemplo 14.1.1. Capturando divisão por zero

```
1 def divide_numbers(a, b):
2     try:
3         result = a / b
4         return result
5     except ZeroDivisionError:
6         print("Erro: Divisão por zero não é permitida.")
7         return None
8
9 # Exemplo de uso da função
10 print(divide_numbers(10, 2)) # Saída: 5.0
11 print(divide_numbers(10, 0)) # Saída: Erro: Divisão por zero não
    é permitida. None
```

Neste exemplo, uma função é definida para dividir dois números. A operação de divisão é colocada dentro de um bloco **try**, pois ela pode lançar um **ZeroDivisionError** se o divisor for zero. Se esse erro ocorrer, uma mensagem de erro personalizada é impressa e a função retorna **None**.

Além de capturar exceções específicas, é possível capturar múltiplas exceções em um único bloco **except**, fornecendo as exceções como uma tupla:

```
except (ZeroDivisionError, TypeError):
```

Ou criando múltiplos blocos **except** cada um com um bloco de código para ser executado no caso de cada exceção.



**Exemplo 14.1.2.** Capturando múltiplas exceções

```
1 def calculate_inverse(number):
2     try:
3         inverse = 1 / number
4         return inverse
5     except ZeroDivisionError:
6         print("Erro: Divisão por zero não é permitida.")
7         return None
8     except TypeError:
9         print("Erro: O número fornecido não pode ser invertido.")
10        return None
11
12 # Exemplo de uso da função
13 print(calculate_inverse(2))      # Saída: 0.5
14 print(calculate_inverse(0))      # Saída: Erro: Divisão por zero
15                                # não é permitida. None
16 print(calculate_inverse("abc")) # Saída: Erro: O número fornecido
17                                # não pode ser invertido. None
```

Neste exemplo, uma função é definida para calcular o inverso de um número. A operação para calcular o inverso é colocada dentro de um bloco **try**, pois ela pode lançar um **ZeroDivisionError** se o número for zero ou um **TypeError** se o número não for um tipo que possa ser invertido. Se qualquer uma dessas exceções ocorrer, uma mensagem de erro personalizada é impressa e a função retorna **None**.

A estrutura **try/except** pode ser estendida com os blocos **else** e/ou **finally**. O bloco **else** é executado se nenhuma exceção ocorrer no bloco **try**. O bloco **finally** é sempre executado, independentemente de uma exceção ser lançada ou não.

```
1 try:
2     # Código que pode causar algum erro
3 except: # Ou except Exception:
4     # Código para ser executado em caso de excessões
5 else:
6     # Código para ser executado caso não ocorra excessões
7 finally:
8     # Código que sempre será executado
```

**Exemplo 14.1.3.** Utilizando **else** e **finally**

```
1 def divide_numbers(a, b):
2     try:
3         result = a / b
4     except ZeroDivisionError:
5         print("Erro: Divisão por zero não é permitida.")
6     else:
7         print(f"Divisão: {a} / {b} = {result}")
8     finally:
```

```
9         print("Operação finalizada.")
10
11     # Exemplo de uso da função
12     divide_numbers(10, 2)    # Saída: Divisão: 10 / 2 = 5.0, Operação
                             # finalizada.
13     divide_numbers(10, 0)    # Saída: Erro: Divisão por zero não é
                             # permitida., Operação finalizada.
```

Neste exemplo, uma função é definida para dividir dois números. Se nenhuma exceção for lançada durante a operação de divisão, a divisão será impressa graças ao bloco **else**. Independente de uma exceção ter sido lançada ou não, o bloco **finally** é executado e imprime uma mensagem.

#### §14.1.4 Identificando exceções

Identificar exceções é uma parte crucial da captura de erros. No Python, as exceções são erros detectados durante a execução que não são fatais necessariamente. Python tem várias exceções embutidas que são lançadas quando ocorrem erros comuns.

Algumas das exceções embutidas mais comuns no Python incluem:

- **ImportError**: É lançada quando um módulo ou um nome não pode ser importado.
- **NameError**: É lançada quando um nome de variável ou função não é encontrado no escopo atual.
- **IndexError**: É lançada quando um índice de sequência está fora do intervalo.
- **KeyError**: É lançada quando uma chave de dicionário não existe.
- **TypeError**: É lançada quando uma operação ou função é aplicada a um objeto de tipo inadequado.
- **ValueError**: É lançada quando uma operação ou função recebe um argumento com o tipo correto, mas valor inadequado.
- **ZeroDivisionError**: É lançada quando o segundo operador na operação de divisão ou módulo é zero.

Para identificar as exceções corretas a serem capturadas, geralmente é útil entender o tipo de erro que pode ocorrer. Se você não tem certeza de quais exceções um certo bloco de código pode lançar, você pode capturar todas as exceções. Isso é feito usando a classe base **Exception**.

##### Exemplo 14.1.5. Capturando todas as exceções.

```
1     try:
2         # Código que pode gerar exceções
3         x = 10 / 0
4         print(x)
5     except Exception as e:
6         # Captura todas as exceções e imprime a mensagem
7         print("Ocorreu uma exceção:", str(e))
```

Neste exemplo, qualquer exceção que ocorra dentro do bloco **try** será capturada e sua

mensagem será impressa contendo a exceção que foi levantada.

No entanto, tenha em mente que capturar todas as exceções não é uma prática recomendada. É sempre melhor especificar as exceções exatas que você espera e sabe como lidar. Capturar todas as exceções pode mascarar erros que você não previu e que podem precisar de atenção.

## §14.2 Leitura e escrita de arquivos

A capacidade de ler e escrever arquivos é fundamental para muitos programas. No Python, você pode usar a função nativa **open** para abrir um arquivo. Essa função retorna um objeto de arquivo que fornece métodos e atributos que podem ser usados para ler, escrever e manipular o arquivo.

Para abrir um arquivo, você precisa especificar o caminho do arquivo e o modo de abertura. Os modos mais comuns incluem **r** para leitura, **w** para escrita (apaga qualquer conteúdo existente), **a** para anexar (adiciona ao final do arquivo) e **x** para criar um novo arquivo (lança um erro se o arquivo já existir).

### Exemplo 14.2.1. Abertura de um arquivo para leitura

```
1 | file = open('file.txt', 'r')
2 | content = file.read()
3 | print(content)
4 | file.close()
```

Neste exemplo, o arquivo **file.txt** é aberto para leitura e seu conteúdo é impresso. A função **read** é usada para ler todo o conteúdo do arquivo.

Depois de abrir um arquivo, sempre lembre de fechá-lo quando terminar. Isso libera os recursos do sistema que foram usados durante a manipulação do arquivo. Você pode usar o método **close** para fechar um arquivo.

### §14.2.2 Estrutura with

Uma maneira mais segura de trabalhar com arquivos é usando a estrutura **with**. Esta estrutura garante que o arquivo será fechado corretamente mesmo se ocorrer uma exceção durante a manipulação do arquivo.

### Exemplo 14.2.3. Usando a estrutura with para trabalhar com arquivos

```
1 | with open('file.txt', 'r') as file:
2 |     content = file.read()
3 |     print(content)
```

Neste exemplo, o arquivo **file.txt** é aberto para leitura dentro de um bloco **with**. O conteúdo do arquivo é impresso e o arquivo é automaticamente fechado no final do bloco **with**.

Além da leitura, você também pode escrever em um arquivo. Se você abrir um arquivo no modo de escrita **w**, qualquer conteúdo existente será apagado. Se você quiser adicionar ao conteúdo existente, use o modo de anexação **a**.

**Exemplo 14.2.4.** Escrevendo em um arquivo

```
1 with open('file.txt', 'a') as file:
2     file.write('\nNova linha')
```

Neste exemplo, uma nova linha é adicionada ao final do arquivo `file.txt`. A função `write` é usada para escrever no arquivo.

## §14.3 Simulações de probabilidades

Simulações de probabilidades são uma ferramenta que permitem modelar e analisar o comportamento de sistemas aleatórios. Com a capacidade do Python de realizar cálculos complexos e a biblioteca NumPy para eficiência numérica, podemos realizar simulações de probabilidades de maneira eficaz.

### §14.3.1 Simulando lançamento de moeda

Podemos simular o lançamento de uma moeda usando a função `np.random.choice`. Esta função permite que você escolha aleatoriamente a partir de uma lista de opções.

**Exemplo 14.3.2.** Simulando lançamento de moeda

```
1 import numpy as np
2
3 # Definir as possibilidades
4 choices = ['Heads', 'Tails']
5
6 # Inicializar o dicionário que vai armazenar a contagem dos
  resultados
7 results = {'Heads': 0, 'Tails': 0}
8
9 # Realizar o lançamento da moeda 1000 vezes
10 for _ in range(1000):
11     flip = np.random.choice(choices)
12     results[flip] += 1
13
14 print(results)
```

Neste exemplo, a função `np.random.choice` é usada para escolher aleatoriamente entre `'Heads'` e `'Tails'`. Isso é feito 1000 vezes, simbolizando 1000 lançamentos de uma moeda. A contagem de quantas vezes cada resultado foi obtido é armazenado no dicionário `results`.

### §14.3.3 Simulando lançamento de dados

Da mesma forma, podemos simular o lançamento de um dado usando a função `np.random.randint`. Esta função retorna um número inteiro aleatório dentro do intervalo especificado.

**Exemplo 14.3.4.** Simulando lançamento de dados

```

1  import numpy as np
2
3  # Definir as possibilidades do dado
4  possibilities = [1, 2, 3, 4, 5, 6]
5
6  # Inicializar o dicionário que vai armazenar a contagem dos
   resultados
7  results = {num: 0 for num in possibilities}
8
9  # Realizar o lançamento do dado 10 vezes
10 for _ in range(1000):
11     roll = np.random.randint(1, 7)
12     results[roll] += 1
13
14 print(results)

```

Neste exemplo, a função `np.random.randint` é usada para gerar um número aleatório entre 1 e 6, simbolizando o lançamento de um dado.

### §14.3.5 Problema de Monty Hall

O Problema de Monty Hall é um célebre problema probabilístico que foi ao ar no programa de game show americano *Let's Make a Deal*, nomeado após seu apresentador original, Monty Hall. O problema pode ser descrito da seguinte forma:

1. Você está em um game show e há três portas. Atrás de uma das portas há um carro (o prêmio principal), enquanto as outras duas portas têm cabras atrás delas (representando prêmios indesejáveis).
2. Você escolhe uma porta, digamos a Porta 1. A sua escolha é inicialmente mantida fechada.
3. Em seguida, Monty, que sabe o que há atrás de cada porta, abre uma das outras duas portas, digamos a Porta 3, revelando uma cabra. Ele não abre a porta que você escolheu (Porta 1), nem a porta com o carro (a menos que você já tenha escolhido a porta do carro inicialmente).
4. Agora, você tem uma escolha a fazer: ficar com a porta que escolheu inicialmente (Porta 1) ou mudar para a outra porta que ainda está fechada (Porta 2).

O problema de Monty Hall incita à discussão pois, intuitivamente, pode parecer que a escolha de ficar ou mudar não faria diferença, já que agora há duas portas e um carro, dando a impressão de que as chances são de 50/50. No entanto, a probabilidade favorece a troca de porta, pois ao mudar a escolha a probabilidade de ganhar o carro é de  $2/3$ , enquanto a probabilidade de ganhar ao ficar com a escolha inicial é de apenas  $1/3$ .

#### Exemplo 14.3.6. Simulando o problema de Monty Hall

```

1  import numpy as np
2
3  np.random.seed(0)
4
5  def monty_hall():

```

```
6     # Inicializar as portas
7     # 0 representa uma porta sem prêmio, 1 representa a porta com
      prêmio
8     doors = np.array([0, 0, 1])
9     np.random.shuffle(doors)
10
11     # Escolher uma porta aleatoriamente
12     choice = np.random.choice([0, 1, 2])
13
14     # Apresentador revela uma das portas restantes sem prêmio
15     revealed_door = np.random.choice([i for i in range(3) if i !=
      choice and doors[i] == 0])
16
17     # Mudar a escolha para a porta não revelada
18     new_choice = next(i for i in range(3) if i != choice and i !=
      revealed_door)
19
20     # Verificar se a nova escolha é uma porta vencedora
21     if doors[new_choice] == 1:
22         return 1
23     else:
24         return 0
25
26     N = 1000
27     # Inicializar o contador de vitórias
28     wins = 0
29     # Roda a simulação N vezes
30     for _ in range(N):
31         wins += monty_hall()
32     # Calcula a porcentagem de vitórias
33     win_percentage = (wins / N) * 100
34
35     print(f"Após {N} simulações, a porcentagem de vitórias ao mudar
      de porta é de {win_percentage}%.")
```

Neste exemplo, a simulação do problema de Monty Hall é realizada 1000 vezes. Primeiro, as portas são inicializadas e uma é escolhida como a porta vencedora. Em seguida, uma porta é escolhida aleatoriamente. O apresentador então revela uma das portas restantes que não contém o prêmio. O concorrente então decide se deve ficar com a sua escolha original ou mudar para a outra porta não revelada. A simulação mostra que mudar a porta aumenta as chances de ganhar.

# 15 | Pandas, parte 1

O pandas é uma biblioteca Python de código aberto que fornece estruturas de dados de alto desempenho e ferramentas de análise de dados fáceis de usar. Ele introduz duas estruturas de dados principais: **Series** e **DataFrame**.

## §15.1 Series

Um objeto de Series é como um array unidimensional, uma lista homogênea de valores. Cada valor recebe um rótulo, que é conhecido como índice. A maneira mais simples de criar uma Series é a partir de uma lista.

### Exemplo 15.1.1. Criação de uma Series

```
1  import pandas as pd
2
3  # Exemplo 1: Criando uma Series com índice automático
4  data1 = [1, 2, 3, 4, 5]
5  series1 = pd.Series(data1)
6  print(series1)
7  # Saída:
8  # 0      1
9  # 1      2
10 # 2      3
11 # 3      4
12 # 4      5
13 # dtype: int64
14
15 # Exemplo 2: Criando uma Series com índices nomeados
16 data2 = [10, 20, 30, 40, 50]
17 index2 = ['A', 'B', 'C', 'D', 'E']
18 series2 = pd.Series(data2, index=index2)
19 print(series2)
20 # Saída:
21 # A      10
22 # B      20
23 # C      30
24 # D      40
25 # E      50
26 # dtype: int64
```

Nesse exemplo, uma Series é criada a partir de uma lista de valores. A série resultante tem um índice atribuído automaticamente que começa em 0 e incrementa de um em um. Podemos também atribuir um índice específico a uma Series no momento da criação, como demonstrado no segundo exemplo. Lá, criamos uma Series com índices nomeados usando caracteres.

### §15.1.2 Indexando Series

A indexação em uma Series é semelhante a uma matriz NumPy, com a diferença importante de que uma Series pode ter um índice explícito definido. Para acessarmos valores na série pela posição no lugar do índice podemos utilizar o atributo `iloc`.

#### Exemplo 15.1.3. Indexando uma Series

```
1 import pandas as pd
2
3 # Criando uma Series com índices nomeados
4 data = [10, 20, 30, 40, 50]
5 index = ['a', 'b', 'c', 'd', 'e']
6 series = pd.Series(data, index=index)
7 print(series)
8 # Saída:
9 # a      10
10 # b      20
11 # c      30
12 # d      40
13 # e      50
14 # dtype: int64
15
16 # Acessando elementos da Series por índice numérico
17 print(series.iloc[0]) # Saída: 10
18
19 # Acessando elementos da Series por índice nomeado
20 print(series['b']) # Saída: 20
```

Neste exemplo, criamos uma Series com um índice nomeado. A indexação pode ser realizada tanto por meio do índice numérico (com `iloc`), como por meio do índice nomeado. O valor na posição 0 é acessado e também o valor no índice nomeado 'b'.

As Series também suportam vários tipos de operações de indexação, como fatiamento, indexação booleana, entre outras.

## §15.2 DataFrame

Um **DataFrame** é uma estrutura bidimensional de dados, como uma planilha ou uma tabela SQL. É semelhante a uma matriz NumPy, mas com índices de linhas e colunas nomeados. Cada coluna em um **DataFrame** pode conter diferentes tipos de dados (numéricos, booleanos, strings, etc.), tornando-o uma estrutura de dados muito flexível para o manejo de conjuntos de dados.

#### Exemplo 15.2.1. Criação de um DataFrame

```
1 import pandas as pd
2
3 # Criando um DataFrame a partir de um dicionário de listas
4 data = {'Nome': ['João', 'Maria', 'Pedro', 'Ana'],
5         'Idade': [25, 30, 35, 28],
```



```

6         'Cidade': ['São Paulo', 'Rio de Janeiro', 'Curitiba',
7                   'Belo Horizonte']}]
8 df = pd.DataFrame(data)
9 print(df)
10 # Saída:
11 #   Nome  Idade      Cidade
12 # 0  João   25   São Paulo
13 # 1  Maria  30  Rio de Janeiro
14 # 2  Pedro  35   Curitiba
15 # 3   Ana   28  Belo Horizonte
16
17 # Definindo índices explícitos ao criar o DataFrame
18 index = ['a', 'b', 'c', 'd']
19 df = pd.DataFrame(data, index=index)
20 print(df)
21 # Saída:
22 #   Nome  Idade      Cidade
23 # a  João   25   São Paulo
24 # b  Maria  30  Rio de Janeiro
25 # c  Pedro  35   Curitiba
26 # d   Ana   28  Belo Horizonte

```

Nesse exemplo, um DataFrame é criado a partir de um dicionário de listas. As chaves do dicionário se tornam os rótulos das colunas e as listas se tornam as linhas de dados. O DataFrame resultante tem rótulos de linha atribuídos automaticamente, começando de 0 e incrementando de um em um. Também é possível definir índices explícitos ao criar um DataFrame, similarmente à criação de uma Series.

Um **DataFrame** pode ser criado a partir de:

- A partir de **um dicionário de listas**: cada lista representa uma coluna do **DataFrame**, e as chaves do dicionário se tornam os rótulos das colunas.
- A partir de uma **lista de dicionários**: cada dicionário representa uma linha do **DataFrame**, e as chaves dos dicionários se tornam os rótulos das colunas.
- A partir de uma **lista de listas**: cada lista interna representa uma linha do **DataFrame**.
- A partir de um **NumPy array**: os elementos do array são usados para preencher o **DataFrame**.
- A partir de **outro DataFrame**: pode-se criar um novo **DataFrame** a partir de uma seleção de colunas de outro **DataFrame**.
- A partir de **um arquivo**: é possível ler dados de um arquivo (por exemplo, CSV, Excel) e criar um **DataFrame** a partir desses dados.

### §15.2.2 Acessando DataFrames

Existem várias maneiras de acessar os dados em um **DataFrame**, incluindo o acesso por rótulo de coluna e acesso por localização numérica ou acesso por rótulo de linha isso se dá pelos atributos **iloc** e **loc** respectivamente.

**Exemplo 15.2.3.** Acessando um DataFrame

```
1 import pandas as pd
2
3 # Criando um DataFrame de exemplo
4 data = {'Nome': ['João', 'Maria', 'Carlos', 'Ana'],
5         'Idade': [25, 30, 42, 28],
6         'Cidade': ['São Paulo', 'Rio de Janeiro', 'Belo
7                   Horizonte', 'Salvador']}
8
9 df = pd.DataFrame(data)
10
11 # Acessando uma coluna específica pelo nome da coluna
12 nome_coluna = df['Nome']
13 print(nome_coluna)
14
15 # Acessando uma fatia de linhas usando a sintaxe de fatiamento
16 fatia_linhas = df.loc[1:3]
17 print(fatia_linhas)
18
19 # Acessando uma fatia de linhas e de colunas usando a sintaxe de
20 # fatiamento
21 fatia = df.loc[2:4, "Idade":"Cidade"]
22 print(fatia)
23
24 # Acessando uma célula específica usando rótulo de coluna e
25 # rótulo de linha
26 celula_especifica = df.loc[2, 'Cidade']
27 print(celula_especifica)
28
29 # Acessando uma célula específica pela posição da linha e da
30 # coluna
31 celula_especifica = df.iloc[2, 2]
32 print(celula_especifica)
```

Nesse exemplo, os dados de uma coluna específica são acessados pelo nome da coluna. Em seguida, uma fatia de linhas é acessada usando a sintaxe de fatiamento padrão do Python. Finalmente, uma célula específica é acessada usando uma combinação de rótulo de coluna e rótulo de linha.

## §15.3 Filtrando DataFrames

A filtragem de dados é uma parte essencial do trabalho com **DataFrame**. O pandas fornece uma maneira eficiente e intuitiva de filtrar dados usando expressões booleanas. Essas expressões são aplicadas a uma ou mais colunas do **DataFrame** e o resultado é um novo **DataFrame** contendo apenas as linhas que satisfazem a condição especificada.

**Exemplo 15.3.1.** Filtrando um DataFrame

```
1 import pandas as pd
2
3 # Criando um DataFrame de exemplo
4 data = {'A': [1, -2, 3, -4, 5],
5         'B': [10, 20, 30, 40, 50]}
6
7 df = pd.DataFrame(data)
8
9 # Criando um filtro booleano
10 filtro = df['A'] > 0
11
12 # Aplicando o filtro ao DataFrame original
13 df_filtrado = df[filtro]
14
15 print(df_filtrado)
```

Neste exemplo, é criado um filtro booleano, que é uma Series cujos valores são **True** onde a condição especificada é atendida e **False** onde não é. Nesse caso, a condição é "todos os valores na coluna 'A' que são maiores que 0". Em seguida, o filtro é aplicado ao **DataFrame** original usando a sintaxe de colchetes, resultando em um novo **DataFrame** que contém apenas as linhas onde o valor na coluna 'A' é maior que 0.

Além de operações simples de comparação, é possível utilizar operadores lógicos como **&** (e), **|** (ou), e **~** (não) para criar condições de filtragem mais complexas.

**Exemplo 15.3.2.** Filtrando com múltiplas condições

```
1 import pandas as pd
2
3 # Criando um DataFrame de exemplo
4 data = {'A': [1, -2, 3, -4, 5],
5         'B': [10, -20, 30, -40, 50]}
6
7 df = pd.DataFrame(data)
8
9 # Aplicando os filtros combinados ao DataFrame original
10 df_filtrado = df[(df['A'] > 0) & (df['B'] < 0)]
11
12 print(df_filtrado)
```

Neste segundo exemplo, duas condições são especificadas: os valores na coluna 'A' devem ser maiores que 0, e os valores na coluna 'B' devem ser menores que 0. Ambas as condições devem ser verdadeiras para que uma linha seja incluída no **DataFrame** resultante. As condições são combinadas usando o operador lógico **&**.

Note que cada condição deve ser encapsulada entre parênteses. Isso ocorre porque o Python avalia os operadores lógicos antes dos operadores de comparação, então sem os parênteses, as expressões não seriam avaliadas na ordem correta.

## §15.4 Métodos de Series (ou colunas)

Os objetos `Series` em `pandas` têm uma variedade de métodos úteis para manipulação e análise de dados. Uma vez que cada coluna de um `DataFrame` é essencialmente um objeto `Series`, esses métodos também podem ser usados diretamente em colunas de `DataFrame`.

Aqui estão alguns dos métodos mais úteis:

- `head()` e `tail()`: Esses métodos retornam as primeiras ou últimas  $n$  linhas de uma `Series` ou `DataFrame`, respectivamente.
- `describe()`: Este método gera estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição de um conjunto de dados.
- `unique()`: Este método retorna uma array com os valores únicos da `Series`.
- `value_counts()`: Este método retorna uma `Series` contendo contagens de valores únicos em ordem decrescente, de modo que o elemento mais frequente apareça primeiro.
- `apply()`: Este método aplica uma função ao longo de um eixo do `DataFrame` ou em valores de `Series`.
- `sort_values()`: Este método ordena uma `Series` em ordem ascendente ou decendente. As colunas para serem utilizadas na ordenação podem ser especificadas com o parâmetro `by`.
- `idxmax()/idxmin()`: Esses métodos retornam o rótulo do índice do máximo e do mínimo, respectivamente.

### Exemplo 15.4.1. Usando métodos de Series

```
1 import pandas as pd
2 import numpy as np
3
4 # Criando um DataFrame de exemplo
5 data = {'A': [1, 2, 2, 3, 3, 3]}
6
7 df = pd.DataFrame(data)
8
9 # Exibindo estatísticas resumidas da coluna 'A'
10 descricao = df['A'].describe()
11 print("Estatísticas resumidas da coluna 'A':")
12 print(descricao)
13 print()
14
15 # Exibindo os valores únicos da coluna 'A'
16 valores_unicos = df['A'].unique()
17 print("Valores únicos da coluna 'A':")
18 print(valores_unicos)
19 print()
20
21 # Contando a frequência de cada valor único na coluna 'A'
22 frequencia = df['A'].value_counts()
23 print("Frequência dos valores na coluna 'A':")
```

```
24 print(frequencia)
25 print()
26
27 # Aplicando uma função a cada valor na coluna 'A'
28 quadrados = df['A'].apply(np.square)
29 print("Quadrados dos valores na coluna 'A':")
30 print(quadrados)
```

Este exemplo cria um DataFrame de exemplo e realiza várias operações em uma coluna (Series) desse DataFrame. Primeiro, o método `describe()` é usado para exibir estatísticas resumidas da coluna 'A'. Em seguida, o método `unique()` é usado para exibir os valores únicos presentes na coluna 'A'. Depois, o método `value_counts()` é usado para contar a frequência de cada valor único na coluna 'A'. Por fim, o método `apply()` é usado para aplicar uma função (neste caso, a função `np.square`, que calcula o quadrado de um número) a cada valor na coluna 'A'.

## §15.5 Iterando em DataFrames

Embora geralmente seja mais eficiente usar métodos integrados do pandas para realizar operações de dados sempre que possível, às vezes pode ser necessário iterar sobre as linhas de um **DataFrame**. Pandas oferece vários métodos para iterar sobre um **DataFrame**, cada um com suas próprias vantagens e desvantagens.

- `df.iterrows()`: Este método retorna um gerador que produz índice e linha, onde a linha é uma **Series**. Este método é lento e não é recomendado para DataFrames grandes.
- `df.itertuples()`: Este método retorna um objeto iterador que produz namedtuples, que são tuplas, mas com nomes de campo acessíveis como propriedades. Este método é mais rápido que `iterrows()`, e as tuplas resultantes são mais eficientes em termos de memória do que as Series.
- `df.apply()`: Embora não seja estritamente um método de iteração, o método `apply()` pode ser usado para aplicar uma função a cada linha ou coluna de um **DataFrame** ou a cada item de uma **Series**, o que pode ser útil em muitos casos onde você pode considerar a iteração.

### Exemplo 15.5.1. Iterando sobre um DataFrame

```
1 import pandas as pd
2 import numpy as np
3
4 # Criando um DataFrame de exemplo
5 data = {'A': [1, 2, 3], 'B': [4, 5, 6]}
6
7 df = pd.DataFrame(data)
8
9 # Iterando sobre as linhas do DataFrame usando iterrows()
10 print("Iterando sobre as linhas usando iterrows():")
11 for index, row in df.iterrows():
12     print(f"Índice: {index}, Linha: {row}")
13 print()
```

```
14
15 # Iterando sobre as linhas do DataFrame usando itertuples()
16 print("Iterando sobre as linhas usando itertuples():")
17 for row in df.itertuples():
18     print(f"Índice: {row.Index}, Linha: {row}")
19 print()
20
21 # Aplicando uma função a cada linha do DataFrame usando apply()
22 soma_linhas = df.apply(np.sum, axis=1)
23 print("Soma das linhas:")
24 print(soma_linhas)
```

Este exemplo mostra diferentes maneiras de iterar sobre um DataFrame. Primeiro, usa-se o método `iterrows()` para iterar sobre as linhas do DataFrame e imprimir o índice e a linha. Em seguida, o método `itertuples()` é usado para fazer a mesma coisa, mas retornando `namedtuples` em vez de `Series`. Finalmente, o método `apply()` é usado para aplicar uma função (neste caso, a função `np.sum` que soma os valores) a cada linha do DataFrame.

## §15.6 Aplicação de funções

A aplicação de funções é um componente central da análise de dados. Pandas fornece uma variedade de maneiras de aplicar funções tanto a elementos individuais quanto a grupos inteiros de dados de um DataFrame.

- `map()`: Este método é usado para substituir cada valor em uma `Series` por outro valor. Esses outros valores podem ser derivados de uma função, um dicionário ou uma `Series`.
- `apply()`: Este método é usado para aplicar uma função a cada elemento de uma `Series` ou a cada linha ou coluna de um DataFrame.
- `applymap()`: Este método é usado para aplicar uma função a cada elemento de um DataFrame. Isso é semelhante ao método `df.apply()`, mas `df.apply()` opera em linhas ou colunas inteiras, enquanto `df.applymap()` opera em cada elemento individualmente.

### Exemplo 15.6.1. Aplicação de funções a um DataFrame

```
1 import pandas as pd
2 import numpy as np
3
4 # Criando um DataFrame de exemplo
5 data = {'A': [1, 2, 3], 'B': [4, 5, 6]}
6
7 df = pd.DataFrame(data)
8
9 # Aplicando o método map() para substituir valores em uma Series
10 df['A'] = df['A'].map({1: 'One', 2: 'Two', 3: 'Three'})
11 print("Substituição de valores usando map():")
12 print(df)
13 print()
14
```

```
15 # Aplicando a função np.sum a cada coluna do DataFrame usando
    apply()
16 soma_colunas = df.apply(np.sum)
17 print("Soma das colunas usando apply():")
18 print(soma_colunas)
19 print()
20
21 # Aplicando a função np.square a cada elemento do DataFrame
    usando applymap()
22 quadrado_df = df.applymap(np.square)
23 print("Quadrado dos elementos do DataFrame usando applymap():")
24 print(quadrado_df)
```

Este exemplo mostra como aplicar funções a um DataFrame. Primeiro, o método `map()` é usado para substituir valores em uma Series. Em seguida, o método `apply()` é usado para aplicar a função `np.sum` a cada coluna do DataFrame. Finalmente, o método `applymap()` é usado para aplicar a função `np.square` a cada elemento do DataFrame.

# 16 | Pandas, parte 2

## §16.1 Leitura e escrita de DataFrames

Trabalhar com dados significa que você frequentemente precisará ler dados de fontes externas e também escrever seus resultados de volta para um formato que outros podem usar. Pandas oferece uma variedade de métodos para ler e escrever dados em muitos formatos diferentes.

- `pd.read_csv()` / `df.to_csv()`: Estes são os métodos mais comumente usados para ler e escrever dados em formato CSV (Comma Separated Values). Eles possuem uma série de opções para lidar com diferentes peculiaridades que podem existir nos seus dados CSV.
- `pd.read_excel()` / `df.to_excel()`: Estes métodos são usados para ler e escrever dados em formato Excel. Eles suportam tanto o formato antigo (.xls) quanto o novo (.xlsx).
- `pd.read_sql()` / `df.to_sql()`: Estes métodos são usados para ler e escrever dados de e para um banco de dados SQL. Eles requerem uma conexão de banco de dados estabelecida como um de seus argumentos.
- `pd.read_json()` / `df.to_json()`: Estes métodos são usados para ler e escrever dados em formato JSON.
- `pd.read_pickle()` / `df.to_pickle()`: Estes métodos são usados para ler e escrever dados em formato pickle, que é um formato binário para serializar e desserializar objetos Python.
- `df.to_html()` e `df.to_latex()`: São métodos que geram tabelas em formato html ou latex a partir do **DataFrame**.

### Exemplo 16.1.1. Leitura e escrita de DataFrames

```
1  import pandas as pd
2
3  # Lendo um DataFrame a partir de um arquivo CSV
4  df = pd.read_csv('input.csv')
5
6  # Realizando algumas operações no DataFrame
7  # ... (o código das operações depende do que você está tentando
   realizar)
8
9  # Escrevendo o DataFrame manipulado em um novo arquivo CSV
10 df.to_csv('output.csv', index=False)
```

Neste exemplo, dados são lidos de um arquivo CSV, manipulados e, em seguida, salvos em um novo arquivo CSV. Os métodos `read_csv()` e `to_csv()` são usados para realizar essas operações. Note que o método `to_csv()` é chamado em um **DataFrame**, o que significa que o **DataFrame** será escrito em um arquivo CSV.



### Exportando índices

No pandas, cada DataFrame tem um índice, que é uma etiqueta para cada linha. Por padrão, quando você salva um DataFrame para um arquivo CSV usando o método `to_csv()`, o pandas incluirá esse índice como a primeira coluna do arquivo CSV.

No entanto, em muitos casos, o índice do DataFrame não contém informações úteis e pode ser apenas uma sequência numérica automática. Em tais situações, salvar o índice para o arquivo CSV pode ser desnecessário e até confuso, pois adicionará uma coluna extra sem informações significativas.

Ao passar o argumento `index=False` para o método `to_csv()`, você está instruindo o pandas a não salvar o índice do DataFrame no arquivo CSV. Isso pode tornar o arquivo CSV mais limpo e mais fácil de entender, especialmente se o índice não for relevante para seus dados. E evita o surgimento de colunas `Unnamed: 0` quando o CSV for carregado novamente no Pandas.

## §16.2 Plotagem

Uma parte importante da análise de dados é a capacidade de visualizar dados e resultados, e a biblioteca Pandas fornece funcionalidades integradas para a plotagem de gráficos que se baseiam na biblioteca Matplotlib (ou outras bibliotecas de plotagem substituindo o backend). Para plots mais complexos ou personalizados, é recomendável usar diretamente a Matplotlib ou outras bibliotecas de plotagem, como Seaborn ou plotly, mas para plots simples e rápidos, a funcionalidade integrada do Pandas pode ser suficiente e mais conveniente. A sintaxe básica dessa funcionalidade é:

```
df.plot(x="coluna_x", y="coluna_y", kind="line")
```

O método `plot` do pandas aceita um argumento `kind` que pode assumir vários valores para especificar o tipo de gráfico a ser produzido. Aqui estão alguns desses valores possíveis e suas descrições:

- `'line'`: Produz um gráfico de linhas. Este é o tipo padrão de gráfico.
- `'bar'`: Produz um gráfico de barras verticais.
- `'barh'`: Produz um gráfico de barras horizontais.
- `'hist'`: Produz um histograma, que pode ser usado para visualizar a distribuição de frequência de uma única variável.
- `'box'`: Produz um gráfico de caixa (box plot), que é útil para visualizar a mediana, quartis, e possíveis outliers de uma variável.
- `'kde'` ou `'density'`: Produz uma estimativa de densidade de kernel, que é uma maneira suave de visualizar a distribuição de uma variável.
- `'area'`: Produz um gráfico de área, que é essencialmente um gráfico de linhas preenchido.
- `'scatter'`: Produz um gráfico de dispersão, útil para visualizar a relação entre duas variáveis numéricas.
- `'hexbin'`: Produz um gráfico de hexágonos bin, que é útil para visualizar a relação entre duas variáveis numéricas em grandes conjuntos de dados.
- `'pie'`: Produz um gráfico de pizza.

**Exemplo 16.2.1.** Plotagem com Pandas

```
1  import pandas as pd
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  # Gera um DataFrame com números aleatórios
6  np.random.seed(0)
7  df = pd.DataFrame(np.random.rand(10, 4), columns=['a', 'b', 'c',
8            'd'])
9
10 # Cria um gráfico de linha
11 df.plot(kind='line')
12 plt.title('Line plot')
13 plt.show()
14
15 # Cria um gráfico de barras
16 df.plot(kind='bar')
17 plt.title('Stacked Bar plot')
18 plt.show()
19
20 # Cria um gráfico de barras empilhadas
21 df.plot(kind='bar', stacked=True)
22 plt.title('Stacked Bar plot')
23 plt.show()
24
25 # Cria um gráfico de área
26 df.plot(kind='area', stacked=False)
27 plt.title('Area plot')
28 plt.show()
29
30 # Cria um gráfico de dispersão
31 df.plot(kind='scatter', x='a', y='b')
32 plt.title('Scatter plot')
33 plt.show()
34
35 # Cria um histograma
36 df['a'].plot(kind='hist')
37 plt.title('Histogram plot')
38 plt.show()
39
40 # Cria um gráfico de caixa
41 df.plot(kind='box')
42 plt.title('Box plot')
43 plt.show()
44
45 # Cria um gráfico de densidade (KDE)
46 df.plot(kind='density')
47 plt.title('Density plot')
```

```
47 | plt.show()
```

Neste exemplo, um DataFrame é criado com dados aleatórios e depois são gerados diversos tipos de gráficos. Observe pode utilizar a biblioteca Matplotlib pode ser utilizada para personalizar os gráficos.

## §16.3 Outros métodos de manipulação

Além dos métodos já discutidos, o Pandas oferece um conjunto de métodos poderosos para manipulação de dados. Estes permitem alterar a forma dos seus dados, agregá-los de diferentes maneiras e transformá-los para análises mais complexas. Alguns deles são:

- **df.groupby()**: O método **groupby** permite dividir seus dados em grupos com base em algum critério, aplicar uma função a cada grupo, e então combinar os resultados em uma estrutura de dados.
- **df.pivot\_table()**: O método **pivot\_table** permite reorganizar e resumir dados selecionados. Similar às tabelas dinâmicas no Excel, ela recebe argumentos simples para agrupar os dados em um DataFrame de duas dimensões e aplicar resumos estatísticos.
- **df.melt()**: A função **melt** é útil para remodelar um DataFrame, onde é possível transformar colunas em linhas, tornando os dados longos em vez de largos.
- **df.stack()**: O método **stack** “comprime” um nível nas colunas do DataFrame para produzir uma Series, um DataFrame ou um MultiIndex.

### Exemplo 16.3.1. Manipulação e plotagem de dados com Pandas

```
1 | import pandas as pd
2 |
3 | # Lê os primeiros 10 registros do arquivo csv
4 | df_csv = pd.read_csv('countries-table.csv', nrows=10)
5 |
6 | # Usa a função 'melt' para transformar cada coluna de população
   # em uma linha única
7 | df_pop = df_csv.melt(
8 |     # Mantém a coluna 'country' como identificador
9 |     id_vars=['country'],
10 |    # Seleciona as colunas que contêm 'pop' no título
11 |    value_vars=df_csv.columns[df_csv.columns.str.contains('pop')],
12 |    # Nomeia a coluna de valores como 'population'
13 |    value_name="population",
14 |    # Nomeia a coluna de variáveis como 'year'
15 |    var_name="year",
16 | )
17 |
18 | # Remove 'pop' do valor da coluna 'year' e converte para int
19 | df_pop['year'] = df_pop['year'].str.replace("pop", "").astype(int)
20 |
21 | # Agrupa o DataFrame por 'country' e plota um gráfico para cada
   # país
```

```
22 for country, sub_df in df_pop.groupby(by=['country']):  
23     sub_df.plot(kind='scatter', x='year', y='population',  
                title=country)
```

Nesse exemplo, um DataFrame é lido a partir de um arquivo CSV, e o método **melt** é usado para remodelar o DataFrame de modo que cada linha corresponda a uma observação de população para um país em um determinado ano, que anteriormente estavam separadas em várias colunas para cada ano. Em seguida, o método **groupby** é usado para dividir os dados por país, e um gráfico de dispersão da população ao longo do tempo é plotado para cada país.

O arquivo CSV utilizado neste exemplo pode ser baixado em:

<https://links.nngn.net/countries-table>

# II

## Combinatória e Probabilidade

# 17

## Permutações com e sem repetições; permutações circulares

Vamos começar com dois princípios muito simples, mas que serão úteis em todo o curso. São eles o princípio aditivo e o princípio multiplicativo.

**Princípio Aditivo:** Sejam  $A$  e  $B$  conjuntos disjuntos com  $n$  e  $m$  elementos respectivamente. Então  $A \cup B$  tem  $n + m$  elementos.

**Exemplo 17.0.1.** Ache o número de pares ordenados  $(x, y)$  de inteiros tais que  $x^2 + y^2 \leq 25$ .

*Solução:* separamos em 6 casos disjuntos :  $x^2 + y^2 = i$ , onde  $i = 1, 2, 3, 4, 5$ . Defina o conjunto  $S_i = \{(x, y) | x, y \in \mathbb{Z}, x^2 + y^2 = i\}$ . Note que  $S_0 = \{(0, 0)\}$ ;

$S_1 = \{(1, 0), (-1, 0), (0, 1), (0, -1)\}$ ;

$S_2 = \{(1, 1), (1, -1), (-1, 1), (-1, -1)\}$ ;

$S_3 = \emptyset$ ;

$S_4 = \{(0, 2), (0, -2), (2, 0), (-2, 0)\}$ ;

$S_5 = \{(1, 2), (1, -2), (2, 1), (2, -1), (-1, 2), (-1, -2), (-2, 1), (-2, -1)\}$ .

Pelo princípio aditivo temos que o número de pares é  $1 + 4 + 4 + 0 + 4 + 8 = 21$ .

Seja  $A$  um conjunto com  $m$  elementos e  $B$  um conjunto com  $n$  elementos. O produto cartesiano  $A \times B$  é o conjunto de todos os pares ordenados  $(a, b)$ , onde  $a \in A$  e  $b \in B$ .

**Princípio Multiplicativo:** O número de elementos em  $A \times B$  é  $m \times n$ , e isso pode ser interpretado como o número total de maneiras de escolher um elemento de  $A$  e um elemento de  $B$ .

**Exemplo 17.0.2.** Quantos números de três algarismos distintos existem ?

*Solução:* o algarismo das centenas pode escolhido de nove maneiras (só não pode escolher o zero), o algarismo das dezenas pode ser escolhido de nove maneiras (agora o zero pode ser escolhido, mas não o algarismo usado nas centenas). Por fim o algarismo das unidades pode ser escolhido de oito maneiras. Assim pelo princípio multiplicativo há  $9 \cdot 9 \cdot 8 = 648$  números distintos.

**Exemplo 17.0.3.** O Teorema Fundamental da Aritmética nos afirma que cada número natural  $n$  pode ser escrito como

$$n = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdots p_k^{\alpha_k}$$

onde os  $p_i$  são primos distintos e  $\alpha_i$  são inteiros não-negativos . Chame de  $d(n)$  a quantidade de divisores positivos de  $n$ .

(a) Ache  $d(24)$ .

(b) Prove que  $d(n) = (\alpha_1 + 1) \cdot (\alpha_2 + 1) \cdots (\alpha_k + 1)$ .

*Solução:* (a) Note que a fatora  o de 24    $24 = 2^3 \cdot 3$ . Portanto um divisor de 24   um

número da forma  $2^a \cdot 3^b$ . Note que  $a \in \{0, 1, 2, 3\}$  e  $b \in \{0, 1\}$ , assim há  $4 \cdot 2 = 8$  divisores possíveis para o 24.

(b) Um divisor de  $n$  é um número  $d$  da forma  $d = p_1^{\beta_1} \cdot p_2^{\beta_2} \cdots p_k^{\beta_k}$  onde  $0 \leq \beta_i \leq \alpha_i$ . Há  $\alpha_i + 1$  maneiras de escolher cada expoente. Portanto o princípio multiplicativo nos dá  $d(n) = (\alpha_1 + 1) \cdot (\alpha_2 + 1) \cdots (\alpha_k + 1)$ .

## §17.1 Permutações

Dada uma lista de objetos distintos  $a_1, \dots, a_n$ . De quantas maneiras é possível ordená-los? Começemos com um exemplo.

**Exemplo 17.1.1.** De quantas formas é possível ordenar  $\{a, b, c\}$  ?.

*Solução:* Podemos simplesmente listar todas as possibilidades:  $abc, acb, bac, bca, cab, cba$ . Há então seis formas .

Poderíamos usar o princípio multiplicativo. Para a primeira posição podemos colocar cada uma das três letras. Para a segunda posição há duas possibilidades. Para a última posição há uma única possibilidade. Portanto há  $3 \cdot 2 \cdot 1 = 6$  maneiras.

**Definição 17.1.** Defina  $n!$  (Lê-se  $n$  fatorial) como sendo  $n! = n \cdot (n - 1) \cdots 1$ .

Por exemplo,  $3! = 3 \cdot 2 \cdot 1 = 6$ .

**Observação 17.1.I.** Vamos convencionar que  $0! = 1$ .

**Proposição 17.1.2** – O número de maneiras de ordenar  $n$  objetos distintos é  $n!$ .

*Demonstração.* A prova deste fato é repetir o que fizemos no exemplo anterior.

Há  $n$  possibilidades para a primeira posição,  
Há  $n - 1$  possibilidades para a segunda posição,  
 $\vdots$   
Há 1 possibilidade para a  $n$ -ésima posição.

Pelo princípio multiplicativo há  $n \cdot (n - 1) \cdots 1 = n!$

□

Poderíamos generalizar um pouco a discussão anterior e nos perguntar de quantos modos podemos escolher  $k$  objetos distintos dentre  $n$  objetos distintos dados. Vamos denotar esse número por  $(n)_k$ . Por exemplo,

**Exemplo 17.1.3.** De quantos modos podemos escolher uma lista com 3 letras dentre  $\{a, b, c, d\}$ ?

*Solução:* Há quatro formas de escolher a primeira letra, três formas de escolher a segunda letra e duas formas de escolher a última letra, portanto pelo princípio multiplicativo temos  $(4)_3 = 4 \cdot 3 \cdot 2 = 24$  formas.

**Proposição 17.1.4** – O número de maneiras de escolher  $k$  objetos dentre  $n$  dados é

$$(n)_k = n \cdot (n - 1) \cdots (n - k + 1)$$

*Demonstração.* A prova deste fato é usando o princípio multiplicativo igual fizemos antes e deixaremos como exercício.  $\square$

Se usarmos a notação de fatorial, então podemos escrever de forma mais sucinta:

$$(n)_k = \frac{n!}{(n - k)!}.$$

O que aconteceria com o número de listas ordenadas se tivéssemos objetos repetidos ? Começemos com um exemplo.

**Exemplo 17.1.5.** Quantos são os anagramas da palavra MANGA ? ( um anagrama é simplesmente uma reordenação das letras.)

*Solução.* Perceba que há duas letras A repetidas. Considere por um momento que elas são diferentes: M  $A_1$  N G  $A_2$  então há  $5! = 120$  anagramas. Perceba que essa não é a resposta que gostaríamos pois se trocarmos as letras A de posição temos a mesma ordenação. Então basta perceber de quantas formas eu posso permutar os A de posição. Isso é  $2!$ . Logo o número de anagramas da palavra MANGA é  $\frac{5!}{2!} = 60$ .

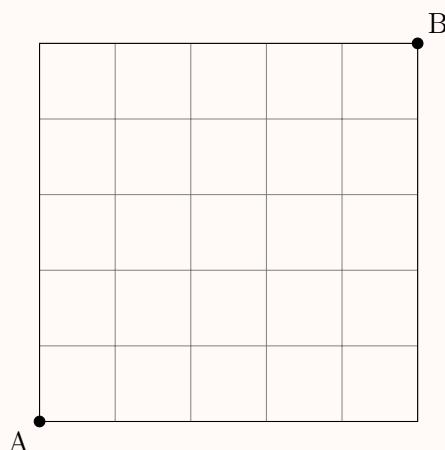
Podemos generalizar esse exemplo da seguinte maneira :

**Proposição 17.1.6** – Considere  $n$  objetos, dos quais  $k_1$  são do tipo 1,  $k_2$  são do tipo 2,  $\dots$ ,  $k_j$  são do tipo  $j$  com  $k_1 + \dots + k_j = n$  então o número de permutações destes  $n$  objetos é

$$\frac{n!}{k_1! \cdots k_j!}.$$

**Exemplo 17.1.7.** Uma formiga vai de A até B andando sempre sobre as arestas do quadriculado  $5 \times 5$  abaixo sempre um passo de cada vez para cima ou para a direita. De quantas formas ela pode fazer esse percurso ?

*Solução:*



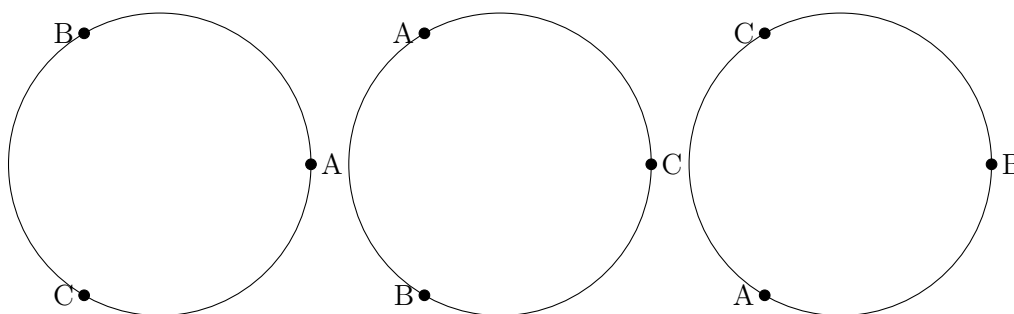


Um caminho pode ser representado por uma lista  $DDDDDDCCCCC$  em alguma ordem onde  $D$  significa para direita e  $C$  significa para cima. A resposta é então o número de maneiras de permutar esses objetos. Isso é

$$\frac{10!}{5! \cdot 5!}.$$

## §17.2 Permutações circulares

De quantos modos podemos colocar três pessoas em uma mesa circular ?



Perceba que se nós não numerarmos os lugares então cada uma das posições acima são iguais. Ou seja, rotações não mudam uma posição. Como no caso acima há três rotações possíveis do círculo então o número de maneiras de colocar três pessoas nessa mesa circular é  $\frac{3!}{3} = 2$ .

De modo geral, se tivermos  $n$  pessoas pra colocar em uma mesa circular então isso pode ser feito de  $\frac{n!}{n} = (n-1)!$  pois cada rotação gera  $n$  distribuições iguais.

**Exemplo 17.2.1.** De quantas formas podemos colocar 5 rapazes e 3 moças em uma mesa circular em cada um dos seguintes casos:

- i. Não há restrição alguma.
- ii. Maria não senta ao lado de João.
- iii. Não há 2 meninas que estão juntas.

*Solução:* (i). A resposta é  $(8-1)! = 7! = 5040$ . (ii) ( 1ª solução ) Deixemos João de fora por um momento, então há sete pessoas que podem ser colocadas na mesa circular de  $(7-1)! = 6!$  maneiras. Agora João só não pode sentar nos lados adjacentes a Maria, há cinco formas dele sentar, portanto a resposta é  $6! \cdot 5 = 3600$ . (ii)

(2ª solução) Considere João, Maria juntos ( formando um único bloco ). Então há sete "pessoas" no total de modo que há  $(7-1)! = 6!$  maneiras de dispor elas na mesa. Mas João, Maria podem trocar de lugares. Portanto há então  $6! \cdot 2 = 1440$  com João e Maria juntos. Como o total sem restrições é 5040 então o que queremos é  $5040 - 1440 = 3600$ .

(iii). Coloquemos primeiramente os rapazes. Isso pode ser feito de  $(5-1)! = 4!$  maneiras. Coloque as moças entre os rapazes. Há cinco espaços para três meninas. Isso é  $(5)_3 = 5 \cdot 4 \cdot 3$ . Portanto a resposta é  $4! \cdot 5 \cdot 4 \cdot 3 = 1440$ .

**Exemplo 17.2.2.** Ache o número de maneiras de dispor  $n$  casais em uma mesa circular nos seguintes casos :

- i. Homens e mulheres se alternam
- ii. Cada mulher está ao lado do seu marido.

*Solução:* (i). Coloque primeiros os homens. Há  $(n-1)!$  maneiras de fazer isso. Há  $n$  espaços entre os homens pra colocar as mulheres. Isso pode ser feito de  $n!$  maneiras. Portanto há  $(n-1)! \cdot n!$  maneiras.

(ii). Pense em cada casal como um único bloco. Há então  $n$  blocos que podem ser colocados de  $(n-1)!$  maneiras. Mas cada casal pode trocar as posições. Há então 2 formas para cada casal. Portanto a resposta é  $(n-1)! \cdot 2^n$ .

**Exemplo 17.2.3.** Prove usando combinatória que  $\frac{(2n)!}{2^n}$  é um número inteiro.

*Solução:* Considere  $n$  objetos. Agora pegue duas cópias de cada um desses objetos, ficamos então com  $2n$  objetos. O número de permutações destes objetos é  $\frac{(2n)!}{2!2! \cdots 2!} = \frac{(2n)!}{2^n}$ . Como o número de permutações é inteiro, segue que esse número é inteiro.

## §17.3 Exercícios

- O código Morse usa "palavras" contendo de 1 a 4 "letras", sendo que cada "letra" é um ponto ou um traço. Quantas palavras existem no código Morse?
- Ache o número ínteros ímpares entre 3000 e 8000 que não tem algarismos repetidos.
- Um campeonato é disputado por 12 clubes em rodadas de 6 jogos cada. De quantos modos é possível selecionar os jogos da primeira rodada?
- Quantas diagonais possui um polígono de  $n$  lados?
- Quantos são os anagramas da palavra CARAGUATATUBA? Quantas começam por vogal?
- Há 12 estudantes em uma festa incluindo João e Maria. 5 são meninas. De quantas maneiras podemos colocar os estudantes em fila em cada caso a seguir
  - Não há restrição alguma
  - As 5 meninas ficam juntas (formando um bloco)
  - Entre João e Maria não há meninos mas há exatamente 3 meninas.
- 6 rapazes e 5 moças sentarão em uma mesa circular. Ache o número de maneiras de fazer isso em cada caso a seguir
  - Não há restrição alguma.
  - Não há 2 moças adjacentes.
  - Todas as garotas estão juntas (formando um único bloco)
  - Maria (uma das moças) está entre João e Felipe.
- Prove que um produto de  $n$  inteiros consecutivos é divisível por  $n!$ .
- Prove usando um argumento combinatório que são os inteiros os números abaixo:
  - $\frac{(3n)!}{2^n \cdot 3^n}$
  - $\frac{(n)!}{(n!)^{(n-1)}}$

# 18

## Combinações a argumentos combinatórios; combinações com repetição; contagem Dupla

### §18.1 Combinações

Na aula anterior nós vimos como escolher  $k$  objetos distintos de um conjunto com  $n$  elementos, ali a ordem era importante. Na aula de hoje nós iremos aprender como escolher objetos sem que seja importante a ordem de escolha. Começemos com um exemplo.

**Exemplo 18.1.1.** De quantas maneiras podemos escolher uma comissão de 3 pessoas num conjunto de 5 pessoas?

*Solução:* Podemos inicialmente supor que a ordem é importante. Neste caso, já vimos que há  $\frac{5!}{(5-3)!} = 60$  comissões. Porém cada comissão abc foi contada  $3! = 6$  vezes. Portanto o número de comissões é  $\frac{5!}{(5-3)!3!} = 10$  comissões.

Repetindo o argumento da solução do exemplo anterior nós temos :

**Proposição 18.1.2** – O número de maneiras de escolher  $k$  objetos dentre  $n$  objetos dados é

$$\binom{n}{k} = \frac{n!}{(n-k)! \cdot k!}.$$

*Demonstração.* basta repetir a solução do problema anterior trocando o 5 por  $n$  e o 3 por  $k$ .  $\square$

$\binom{n}{k}$  lê-se " $n$  escolhe  $k$ ".

**Exemplo 18.1.3.** De quantos modos podemos escolher 6 pessoas incluindo pelo menos 2 mulheres , em um grupo de 7 homens e 4 mulheres ?

*Solução:* As possibilidades são

4 homens e 2 mulheres,  
3 homens e 3 mulheres,  
2 homens e 4 mulheres .

A Resposta é portanto

$$\binom{7}{4} \cdot \binom{4}{2} + \binom{7}{3} \cdot \binom{4}{3} + \binom{7}{2} \cdot \binom{4}{4} = 35 \cdot 6 + 35 \cdot 4 + 21 \cdot 1 = 371$$

Vamos agora a um relação importante envolvendo os números binomiais.

**Proposição 18.1.4** – (Relação de Stifel)

Sejam  $n \geq k \geq 1$  números naturais. Então

$$\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \quad (18.1)$$

*Demonstração.* (Primeira prova)

$$\begin{aligned} \binom{n}{k} + \binom{n}{k-1} &= \frac{n!}{(n-k)! \cdot k!} + \frac{n!}{(n-k+1)! \cdot (k-1)!} \\ &= \frac{n! \cdot (n-k+1)}{(n-k+1)! \cdot k!} + \frac{n! \cdot k}{(n-k+1)! \cdot k!} \\ &= \frac{n!(n+1)}{(n-k+1)!k!} \\ &= \binom{n+1}{k}. \end{aligned} \quad (18.2)$$

(Segunda prova) Vamos contar de quantas maneiras podemos formar uma comissão de  $k$  pessoas dentre  $n+1$  pessoas disponíveis. Uma resposta é  $\binom{n+1}{k}$ . Vamos contar isso de outra maneira. Fixe uma pessoa, digamos Maria. Pelo princípio aditivo, o número total de comissões é igual ao número de comissões nas quais Maria está mais o número de comissões nas quais Maria não está. Contemos cada uma delas. As que Maria está é  $\binom{n}{k-1}$  pois basta escolher  $k-1$  pessoas dentre  $n$ . As que Maria não está é  $\binom{n}{k}$ . Logo,

$$\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1}.$$

□

**Observação 18.1.I.** Na Segunda prova acima utilizamos duas técnicas importante em combinatória. A primeira delas foi de usar um argumento combinatório para provar identidades. A segunda foi o que chamamos de contagem dupla que é basicamente contar uma mesma coisa de duas maneiras distintas.

## §18.2 Combinações com repetições

**Exemplo 18.2.1.** De quantos modos é possível comprar 4 sorvetes em uma loja que os oferece em 7 sabores ?

*Solução:* Poderíamos pensar que a resposta é  $\binom{7}{4} = 35$ , mas não é,  $\binom{7}{4}$  é o número de maneiras de escolher 4 sabores distintos dentre 7. Para resolver o problema então, façamos o seguinte :  $x_i$  será a quantidade do sabor 1,  $x_2$  a do sabor 2, assim por diante até  $x_7$  do sabor 7. Note que  $x_i$  é inteiro não-negativo e

$$x_1 + x_2 + \cdots + x_7 = 4.$$

Portanto comprar 4 sorvetes nessa loja equivale a achar uma solução da equação acima. Vamos então achar o número de soluções inteiras e não-negativas da equação

$$x_1 + x_2 + \cdots + x_7 = 4.$$

Vamos usar pra isso um esquema de traço-bola. Os traços vão representar os sinais de + que servem para separar as quantidades de cada incógnita.

$$\bullet | \bullet | \quad | \bullet | \bullet | \quad |$$

Há então 6 traços e 7 espaços. As bolas serão a quantidade que queremos preencher. Neste caso há 4 bolas pra serem colocadas entre os traços. Portanto achar a quantidade de soluções da equação acima equivale a achar a quantidade de maneiras de dispor 4 bolas e 6 traços em alguma ordem. Já sabemos que isso é

$$\frac{10!}{6! \cdot 4!} = \binom{10}{4}.$$

De modo geral, usando um argumento igual ao do exemplo anterior vale a seguinte proposição:

**Proposição 18.2.2** – O número de soluções inteiras não negativas de

$$x_1 + x_2 + \cdots + x_n = k \quad \text{é} \quad \frac{(n+k-1)!}{(n-1)! \cdot k!} = \binom{n+k-1}{k}.$$

**Exemplo 18.2.3.** Quantas são as soluções inteiras e não-negativas da inequação  $x+y+z \leq 5$ ?

*Primeira Solução:* as soluções da inequação equivalem as seguintes equações  $x+y+z = 0, 1, 2, 3, 4, 5$ . Pela proposição anterior, a resposta é

$$\binom{7}{5} + \binom{6}{4} + \binom{5}{3} + \binom{4}{2} + \binom{3}{1} + \binom{2}{0} = 56.$$

*Segunda solução:* chame  $w = 5 - x - y - z$ , perceba que neste caso  $w \geq 0$ , portanto as soluções da inequação equivalem a solução da equação  $x+y+z+w = 5$  que é  $\binom{8}{5} = 56$ .

## §18.3 Exercícios Propostos

- Quantas são as maneiras de escolher 2 bolas vermelhas e 3 bolas azuis de uma urna que contém 4 bolas vermelhas e 5 bolas azuis?
- Tem-se 5 pontos sobre uma reta R e 8 pontos sobre uma reta R' paralela a R. Quantos quadriláteros convexos com vértices em 4 desses 13 pontos existem?
- Quantos são os subconjuntos com  $p$  elementos do conjunto  $\{1, 2, \dots, n\}$  para os quais :
  - 1 está presente;
  - 1 não está presente;
  - 1 e 2 estão presente;
  - Pelo menos um dos números 1 e 2 está presente;
  - Exatamente um dos números 1, 2 está presente.
- De quantos modos é possível dividir 20 pessoas :
  - em dois grupos de 10?

- (b) em quatro grupos de 5?
  - (c) em um grupo de 12 e um de 8?
  - (d) em três grupos de 6 e um de 2 ?
5. Calcule o valor de  $\binom{n}{n-2}$  em função de  $n$ .
6. (IME-adaptado) Mostre que

$$\binom{2016}{5} + \binom{2017}{5} + \binom{2018}{5} + \binom{2019}{5} + \binom{2020}{5} + \binom{2016}{6} = \binom{2021}{6}.$$

7. Marcela é diretora de um banco e coordena uma equipe de 11 pessoas. Ela recebeu uma solicitação da matriz do banco para formar duas comissões com os membros da sua equipe, sendo uma com 6 integrantes e a outra com 5 integrantes. Como João e Maria são irmãos, Marcela optou por não deixá-los na mesma equipe. Quantas comissões Marcela consegue formar ?
8. Prove que

$$\binom{n}{k} = \binom{n}{n-k}$$

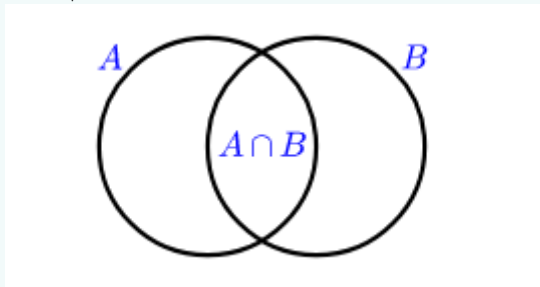
de duas maneiras distintas: usando a fórmula de combinação e via um argumento combinatório.

9. Quantas são as soluções inteiras da equação  $x + y + z = 20$  com  $x \geq 2$ ,  $y \geq 2$ ,  $z \geq 2$ ?
10. De quantas maneiras diferentes podemos distribuir 10 bolas em 4 urnas, sabendo que nenhuma urna pode ficar vazia?

# 19 | Princípio de Inclusão-Exclusão

Na aula 1 vimos que se  $A$  e  $B$  são conjuntos disjuntos então  $|A \cup B| = |A| + |B|$ , onde  $|Z|$  denota o número de elementos do conjunto  $Z$ . O que acontece se os conjuntos  $A$  e  $B$  não forem necessariamente disjuntos?

**Proposição 19.0.1** – Sejam  $A$  e  $B$  conjuntos finitos, não necessariamente disjuntos então  $|A \cup B| = |A| + |B| - |A \cap B|$ .



A ideia é separarmos  $A \cup B$  em conjuntos disjuntos. Primeiramente  $A \cup B = A \cup (B \setminus A)$  e portanto  $|A \cup B| = |A| + |(B \setminus A)|$ . Mais uma vez  $B = (A \cap B) \cup (B \setminus A)$ . Sendo assim,  $|B| = |(A \cap B)| + |(B \setminus A)|$ . Assim  $|A \cup B| = |A| + |B| - |A \cap B|$ .

**Exemplo 19.0.2.** Quantos inteiros entre 1 e 1000 são divisíveis por 3 ou por 7 ?

*Solução:* Denote por :

$A$  = Conjunto dos inteiros entre 1 e 1000 que são divisíveis por 3.

$B$  = Conjunto dos inteiros entre 1 e 1000 que são divisíveis por 7.

O número de inteiros entre 1 e 1000 que são divisíveis por 3 é igual a  $\lfloor \frac{1000}{3} \rfloor = 333$ . Ou seja  $|A| = 333$ .

O número de inteiros entre 1 e 1000 que são divisíveis por 7 é igual a  $\lfloor \frac{1000}{7} \rfloor = 142$ . Ou seja  $|B| = 142$ .

No entanto, alguns inteiros são divisíveis por ambos 3 e 7, ou seja, por 21.

O número de inteiros entre 1 e 1000 que são divisíveis por 21 é igual a  $\lfloor \frac{1000}{21} \rfloor = 47$ . Ou seja  $|A \cap B| = 47$ .

Portanto pelo Princípio de Inclusão-Exclusão temos que o número de inteiro entre 1 e 1000 que são divisíveis por 3 ou por 7 é  $333 + 142 - 47 = 428$ .

O que acontece se tivermos 3 conjuntos  $A, B, C$  não necessariamente disjuntos. Então usando o caso anterior e um pouco de propriedades de conjuntos chegamos a seguinte proposição :

**Proposição 19.0.3** – Se  $A, B, C$  são 3 conjuntos finitos não necessariamente disjuntos então  $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$ .

*Demonstração.* A prova será deixada como exercício. □

Essas duas proposições são casos particulares da forma mais geral do Princípio de Inclusão-Exclusão.

**Proposição 19.0.4** – (Princípio de Inclusão-Exclusão) Sejam  $A_1, A_2, \dots, A_n$  conjuntos finitos, Então

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{i=1}^n |A_i| - \sum_{1 \leq i < j \leq n} |A_i \cap A_j| + \sum_{1 \leq i < j < k \leq n} |A_i \cap A_j \cap A_k| - \dots + (-1)^{n-1} |A_1 \cap A_2 \cap \dots \cap A_n|.$$

Vamos dá algumas aplicações desse princípio.

**Definição 19.1.** Uma permutação dos números  $(1, 2, \dots, n)$  é dita ser *caótica* se estão fora de suas posições originais. Por exemplo, para  $n = 4$  a permutação  $(2, 4, 1, 3)$  é caótica mas  $(2, 4, 3, 1)$  não é caótica pois o 3 está na terceira posição.



**Proposição 19.0.5** – O número de permutações caóticas dos números  $(1, 2, \dots, n)$  é dado por

$$D_n = n! \left( \frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \dots + (-1)^n \frac{1}{n!} \right). \quad (19.1)$$

*Demonstração.* Defina  $A_i$  como o conjunto de todas as permutações em que o elemento  $i$  aparece em sua posição natural (ou seja, a posição  $i$ ). Então, o número de permutações em que o elemento  $i$  aparece em sua posição natural é  $(n-1)!$  (pois agora temos apenas  $n-1$  elementos para permutar). Portanto,  $|A_i| = (n-1)!$ .

Perceba que  $A_i \cap A_j$  é o conjunto de todas as permutações em que os elementos  $i$  e  $j$  aparecem em suas posições naturais. O número de permutações em que os elementos  $i$  e  $j$  aparecem em suas posições naturais é  $(n-2)!$ , pois agora temos apenas  $n-2$  elementos para permutar. Portanto,  $|A_i \cap A_j| = (n-2)!$  e assim por diante. Daí os somatórios que aparecem na fórmula do Princípio de inclusão-exclusão são dados por

$$\sum_{i=1}^n |A_i| = n \cdot (n-1)! = n!.$$

$$\sum_{1 \leq i < j \leq n} |A_i \cap A_j| = \binom{n}{2} \cdot (n-2)! = \frac{n!}{2!},$$

e assim por diante até o último termo que é

$$|A_1 \cap A_2 \cap \dots \cap A_n| = 1 = \frac{n!}{n!}.$$

Pelo princípio da inclusão-exclusão, o número de permutações caóticas é dado por:

$$\begin{aligned} |\bigcup_{i=1}^n A_i| &= \sum_{i=1}^n |A_i| - \sum_{1 \leq i < j \leq n} |A_i \cap A_j| + \sum_{1 \leq i < j < k \leq n} |A_i \cap A_j \cap A_k| - \\ &\quad \dots + (-1)^{n-1} |A_1 \cap A_2 \cap \dots \cap A_n| \\ &= n! - \frac{n!}{2!} + \frac{n!}{3!} - \frac{n!}{4!} + \dots + (-1)^{n-1} \cdot \frac{n!}{n!}. \end{aligned}$$

Como o número total de permutações é  $n!$ , então o número de permutações caóticas é dado por

$$\begin{aligned} D_n &= n! - \left( n! - \frac{n!}{2!} + \frac{n!}{3!} - \frac{n!}{4!} + \dots + (-1)^{n-1} \cdot \frac{n!}{n!} \right) \\ &= n! \left( \frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \dots + (-1)^n \frac{1}{n!} \right). \end{aligned}$$

□

## §19.1 Exercícios propostos

1. Numa classe de 30 alunos, 14 falam inglês, 5 falam alemão e 3 falam inglês e alemão. Quantos alunos falam pelo menos uma língua, dentre inglês e alemão?
2. Prove a proposição (19.0.3).
3. Quantos são os números entre 1 e 1000 que são múltiplos de 2,5 ou 7?

4. Qual o número de permutações de  $(1,2,3,4,5,6)$  nas quais nem o 4 ocupa o 4º lugar nem o 6 ocupa o 6º lugar
5. Quantas são as permutações de  $(1,2,3,4,5,6,7,8,9,10)$  que têm exatamente 4 elementos no seu lugar primitivo?
6. Quantas são as permutações das letras da palavra BRASIL em que o B ocupa o primeiro lugar, ou o R em segundo lugar, ou o L em sexto lugar?
7. Prove que  $D_n = nD_{n-1} + (-1)^n$  para  $n \geq 2$ .
8. Prove que se  $n \geq 3$  então  $D_n = (n-1)(D_{n-1} + D_{n-2})$ .
9. Quantos inteiros entre 1 e 1000000 que não são nem quadrados perfeitos nem cubos perfeitos?
10. Quantas são as funções  $f : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\}$  tais que a equação  $f(k) = k$  tem solução?

# 20

## Lemas de Kaplansky

Nesta aula, nosso objetivo é responder a seguinte pergunta: de quantos modos é possível formar um subconjunto com  $p$  elementos ( $p$ -subconjuntos) de  $I_n = \{1, 2, \dots, n\}$  de modo que não haja dois números consecutivos?

Começemos analisando alguns casos. Se  $n = 4$ , temos  $I_4 = \{1, 2, 3, 4\}$ . Neste caso temos

- 1-subconjunto  $\{1\}, \{2\}, \{3\}, \{4\}$ ;
- 2-subconjuntos  $\{1, 3\}, \{1, 4\}, \{2, 4\}$ .

Se  $n = 5$ , temos  $I_5 = \{1, 2, 3, 4, 5\}$ . Neste caso temos

- 1-subconjunto  $\{1\}, \{2\}, \{3\}, \{4\}, \{5\}$ ;
- 2-subconjuntos  $\{1, 3\}, \{1, 4\}, \{1, 5\}, \{2, 4\}, \{2, 5\}, \{3, 5\}$ ;
- 3-subconjuntos  $\{1, 3, 5\}$ .

Vamos codificar os possíveis subconjuntos por sequências de zeros e uns. Colocamos um 1 se o elemento pertencer ao conjunto e 0 se o elemento não pertencer ao conjunto. Por exemplo, para  $n = 6$

$$\begin{aligned}\{1, 3, 4\} &\mapsto 101100, \\ \{1, 3, 5\} &\mapsto 101010.\end{aligned}$$

Formar um 3-subconjunto de um conjunto com 5 elementos sem elementos consecutivos se resume a saber quantas sequências de zeros e uns existem de modo que não haja uns consecutivos. Coloquemos primeiro os três zeros.

$$\_ \ 0 \_ \ 0 \_ \ 0 \_$$

Há 4 espaços para serem colocados 3 uns. Isso pode ser feito de  $\binom{4}{3} = 4$  maneiras. Isso é um caso particular do resultado abaixo.

**Proposição 20.0.1** – (Primeiro Lema de Kaplansky) O número de  $p$ -subconjuntos de  $\{1, 2, \dots, n\}$  sem elementos consecutivos é igual a  $f(n, p) := \binom{n-p+1}{p}$ .

*Demonstração.* É essencialmente repetir o argumento que demos acima. Coloque os  $n - p$  zeros primeiro. Isso dá origem há  $n - p + 1$  espaços para se colocar os  $p$  uns. Isso pode ser feito de  $\binom{n-p+1}{p}$  maneiras.  $\square$

**Proposição 20.0.2** – (Segundo Lema de Kaplansky) Considere o número 1 como consecutivo do número  $n$ . O número de  $p$ -subconjuntos de  $\{1, 2, \dots, n\}$  sem elementos consecutivos é igual a  $g(n, p) := \frac{n}{n-p} \binom{n-p}{p}$ .

*Demonstração.* Vamos separar os  $p$ -subconjuntos em dois tipos. Os  $p$ -subconjuntos que têm o elemento 1 e os  $p$ -subconjuntos que não tem o elemento 1. **1º caso:** como o número 1 está presente no  $p$ -subconjunto, nem o número 2, nem o número  $n$  podem estar presentes, pois são vizinhos deste. Logo, precisamos escolher outros  $p-1$  elementos do conjunto  $\{3, \dots, n-1\}$ . Pelo Primeiro Lema de Kaplansky, o número de maneiras de se fazer isso é

$$f(n-3, p-1) = \binom{(n-3) - (p-1) + 1}{p-1} = \binom{n-p-1}{p-1}.$$

**2º caso:** como o número 1 não está presente, precisamos escolher  $p$  elementos do conjunto  $\{2, \dots, n\}$ . Mais uma vez, usando o Primeiro Lema de Kaplansky, o número de maneiras de fazer isso é

$$f(n-1, p) = \binom{(n-1) - p + 1}{p} = \binom{n-p}{p}.$$

Daí, somando os resultados

$$\begin{aligned} \binom{n-p-1}{p-1} + \binom{n-p}{p} &= \frac{(n-p-1)!}{(p-1)! \cdot (n-2p)!} + \frac{(n-p)!}{p! \cdot (n-2p)!} \\ &= \frac{(n-p-1)! \cdot p + (n-p)!}{p! \cdot (n-2p)!} \\ &= \frac{(n-p-1)! \cdot [p + (n-p)]}{p! \cdot (n-2p)!} \\ &= \frac{(n-p-1)! \cdot n}{p! \cdot (n-2p)!} \\ &= \frac{n}{n-p} \cdot \binom{n-p}{p}. \end{aligned} \tag{20.1}$$

□

**Exemplo 20.0.3.** Uma fila tem 15 cadeiras nas quais devem sentar-se 5 homens, de modo que não fiquem dois homens sentados em cadeiras vizinhas. De quantos modos podemos fazer isso ?

*Solução:* devemos inicialmente escolher 5 cadeiras sem que haja duas cadeiras consecutivas. Isso pode ser feito de  $f(15, 5) = \binom{11}{5} = 462$ ; escolhidas as cadeiras, vamos dispor os homens nas cadeiras. Isso pode ser feito de  $5! = 120$  maneiras. Portanto a resposta é  $462 \cdot 120 = 55440$ .

**Exemplo 20.0.4.** Lucas quer se preparar para um curso que irá fazer e então deverá fazer aula de natação três vezes por semana, durante um semestre. Para não se cansar, ele não deseja ter aulas em dias consecutivos. De quantos modos Lucas poderá fazer isso ?

*Solução:* como há 7 dias na semana e queremos escolher 3 desses dias. Pelo Segundo Lema de Kaplansky isso pode ser feito de  $g(7, 3) = \frac{7}{7-3} \cdot \binom{7-3}{3} = 7$  maneiras.

**Exemplo 20.0.5.** (Generalização do 1º Lema de Kaplansky) De quantos modos é possível formar um  $p$ -subconjunto de  $\{1, 2, \dots, n\}$  de modo que entre cada dois números escolhidos para o subconjunto haja, no conjunto, pelo menos  $r$  elementos não escolhidos para o subconjunto?

*Solução:* coloquemos primeiramente os  $n - p$  zeros em sequência. Isso dá origem a  $p + 1$  espaços. Chame de  $x_1$  a quantidade de uns que vamos colocar no primeiro espaço,  $x_2$  a quantidade que vamos colocar no segundo espaço e assim por diante até  $x_{p+1}$  a quantidade que vamos colocar no espaço  $p + 1$ . O que queremos é achar o número de soluções da equação

$$x_1 + x_2 + \dots + x_p + x_{p+1} = n - p \quad (20.2)$$

com  $x_1 \geq 0, x_2 \geq r, x_3 \geq r, \dots, x_p \geq r, x_{p+1} \geq 0$ . Chame de  $y_1 = x_1, y_2 = x_2 - r, \dots, y_p = x_p - r, y_{p+1} = x_{p+1}$ . Daí a equação (20.2) é equivalente a

$$y_1 + y_2 + \dots + y_{p+1} = n - p - pr - r. \quad (20.3)$$

cuja quantidade de soluções inteiras não-negativas é  $\binom{n-(p-1)r}{p}$ .

## §20.1 Exercícios propostos

1. De quantos modos podemos colocar 40 meninas e 10 meninos em uma mesa circular se meninos não podem sentar juntos ?
2. Quantos são os triângulos cujos vértices são vértices não adjacentes de um decágono regular ?
3. Quantos são os anagramas da palavra *depreender* que não possuem duas letras e consecutivas?
4. (IME) Doze cavaleiros estão sentados em torno de uma mesa redonda. Cada um dos doze cavaleiros considera seus dois vizinhos como rivais. Deseja-se formar um grupo de cinco cavaleiros para libertar uma princesa, nesse grupo não poderá haver cavaleiros rivais. Determine de quantas maneiras é possível escolher esse grupo.
5. Prove uma versão generalizada do Segundo Lema de Kaplansky igual fizemos no exemplo (20.0.5).
6. Seis pessoas devem se sentar em vinte cadeiras colocadas em torno de uma mesa circular. De quantos modos isso pode ser feito se para cada cadeira ocupada devemos ter duas cadeiras livres de cada lado?
7. (OBM 2010) Diamantino gosta de jogar futebol, mas se jogar dois dias seguidos ele fica com dores musculares. De quantas maneiras Diamantino pode escolher em quais de dez dias seguidos ele vai jogar bola sem ter dores musculares? Uma maneira é não jogar futebol em nenhum dos dias.
8. De quantas maneiras podemos acomodar 3 casais em torno de uma mesa circular de modo que pessoas do mesmo sexo não se sentem juntas e que nenhum homem fique ao lado de sua esposa ?

# 21 | Números Binomiais

Relembremos aqui a relação de Stiefel.

**Proposição 21.0.1** – (Relação de Stiefel) Sejam  $n, k$  inteiros não-negativos com  $k \leq n$ . Então

$$\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1}.$$

Vimos em aulas anteriores o teorema das linhas e o teorema das diagonais.

**Proposição 21.0.2** – 1. (Teorema das linhas)  $\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n$ .

2. (Teorema das Diagonais)  $\binom{n}{0} + \binom{n+1}{1} + \cdots + \binom{n+p}{p} = \binom{n+p+1}{p}$ .

Vamos provar agora um teorema similar a esses.

**Proposição 21.0.3** – (Teorema das colunas)

$$\binom{k}{k} + \binom{k+1}{k} + \cdots + \binom{k+n}{k} = \binom{k+n+1}{k+1}.$$

*Demonstração.* A prova usa apenas a Relação de Stiefel. Vamos a ela. Usando a Relação de Stiefel várias vezes, temos as seguintes igualdades

$$\begin{aligned}\binom{k+1}{k+1} &= \binom{k}{k+1} + \binom{k}{k} \\ \binom{k+2}{k+1} &= \binom{k+1}{k+1} + \binom{k+1}{k} \\ \binom{k+3}{k+1} &= \binom{k+2}{k+1} + \binom{k+2}{k} \\ &\vdots \\ \binom{k+n+1}{k+1} &= \binom{k+n}{k+1} + \binom{k+n}{k}\end{aligned}$$

Somando dois lados da igualdade e cancelando os termos iguais segue o teorema.  $\square$

Vamos dá algumas aplicações desses teoremas.

**Exemplo 21.0.4.** Calcule  $1 \cdot 2 \cdot 3 + 2 \cdot 3 \cdot 4 + \cdots + 50 \cdot 51 \cdot 52$ .

*Solução.* Escrevendo em notação de somatório,

$$\begin{aligned}\sum_{k=1}^{50} k(k+1)(k+2) &= \sum_{k=1}^{50} 3! \cdot \frac{k(k+1)(k+2)}{3!} = \sum_{k=1}^{50} 6 \cdot \binom{k+2}{3} \\ &= 6 \sum_{k=1}^{50} \binom{k+2}{3} = 6 \cdot \binom{53}{4}.\end{aligned}$$

Onde na última igualdade usamos o teorema das colunas.

**Exemplo 21.0.5.** Mostre que

$$S = 1^2 + 2^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}.$$

*Solução.* Escrevendo em notação de somatório, temos  $S = \sum_{k=1}^n k^2$ . Gostaríamos de usar uma ideia parecida com a do problema anterior. Para tal, precisamos de produto de naturais consecutivos. Então, vamos tentar achar  $A, B, C$  de forma que

$$k^2 = Ak(k+1) + Bk + C.$$

Por igualdade de polinômios, obtemos  $A = 1, A + B = 0, C = 0$ . Portanto,  $A = 1, B = -1, C = 0$ . Assim,  $k^2 = k(k+1) - k$ . Portanto,

$$\begin{aligned}S &= \sum_{k=1}^n k^2 = \sum_{k=1}^n [k(k+1) - k] = \sum_{k=1}^n k(k+1) - \sum_{k=1}^n k \\ &= 2! \sum_{k=1}^n \binom{k+1}{2} - \frac{n(n+1)}{2} = 2 \cdot \binom{n+2}{3} - \frac{n(n+1)}{2} \\ &= 2 \frac{(n+2)(n+1)n}{3!} - \frac{n(n+1)}{2} = \frac{n(n+1)}{6} (2(n+2) - 3) = \frac{n(n+1)(2n+1)}{6}.\end{aligned}$$

**Exemplo 21.0.6.** (IMO-1981) Sejam  $1 \leq r \leq n$  e considere todos os subconjuntos com  $r$  elementos do conjunto  $\{1, 2, \dots, n\}$ . Cada um desses subconjuntos tem um menor elemento. Chame de  $F(n, r)$  a média aritmética desses menores elementos. Prove que

$$F(n, r) = \frac{n+1}{r+1}.$$

**Proposição 21.0.7** – (Binômio de Newton)

Sejam  $x, y$  números reais e  $n$  um número inteiro não-negativo. Então

$$(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k} \quad (21.1)$$

*Solução.* Note que  $(x + y)^n = (x + y) \cdots (x + y)$ . Quando fizermos a distributiva e juntarmos todos os termos semelhantes, nosso objetivo é achar o coeficiente de  $x^k y^{n-k}$ . Note que isso equivale a escolher  $k$  elementos iguais a  $x$  e  $n - k$  elementos iguais a  $y$  nos parênteses. Isso é o mesmo que saber de quantas formas podemos permutar  $x \cdots x y \cdots y$ . Já sabemos que isso pode ser feito de  $\frac{n!}{(n-k)!k!} = \binom{n}{k}$ .

**Exemplo 21.0.8.** Fazendo  $x = y = 1$  no Binômio de Newton temos

$$2^n = (1 + 1)^n = \sum_{k=0}^n \binom{n}{k}$$

Isso nos dá uma nova prova do teorema das linhas.

**Exemplo 21.0.9.** Fazendo  $x = 1, y = -1$  no Binômio de Newton temos

$$0 = (1 - 1)^n = \sum_{k=0}^n (-1)^k \binom{n}{k} = \binom{n}{0} - \binom{n}{1} + \cdots + (-1)^n \binom{n}{n} = 0.$$

Logo,

$$\binom{n}{0} + \binom{n}{2} + \cdots = \binom{n}{1} + \binom{n}{3} + \cdots$$

**Exemplo 21.0.10.** Qual o coeficiente de  $x^2$  no desenvolvimento de  $\left(x^3 - \frac{1}{x^2}\right)^9$ ?

*Solução.* Pelo Binômio de Newton um termo genérico desse desenvolvimento é

$$\binom{9}{k} (x^3)^k \left(\frac{-1}{x^2}\right)^{9-k} = (-1)^{9-k} \binom{9}{k} x^{5k-18}$$

Portanto  $5k - 18 = 2$  o que implica  $k = 4$ . Logo o coeficiente de  $x^2$  é  $(-1)^{9-4} \binom{9}{4} = -126$ .

Podemos estender a fórmula do Binômio de Newton para quando tivermos não só 2 termos.



**Proposição 21.0.11** – Para quaisquer  $x_1, x_2, \dots, x_k$  números reais e  $n$  um inteiro não negativo vale

$$(x_1 + x_2 + \dots + x_k)^n = \sum \binom{n}{j_1, j_2, \dots, j_k} x_1^{j_1} \dots x_k^{j_k}$$

onde a soma é tomada sobre todos os  $j_i$  com  $j_1 + j_2 + \dots + j_k = n$  e relembremos que

$$\binom{n}{j_1, j_2, \dots, j_k} = \frac{n!}{j_1! \cdot j_2! \cdot \dots \cdot j_k!}$$

## §21.1 Exercícios Propostos

1. Resolva em  $k$  as equações abaixo.

a)  $\binom{10}{2k} = \binom{10}{k-2}$

b)  $\binom{9}{k^2+k} = \binom{9}{k+1}$ .

2. Prove que

$$\binom{n}{0} + \binom{n}{2} + \dots = \binom{n}{1} + \binom{n}{3} + \dots = 2^{n-1}.$$

3. Sejam  $n, m, k$  inteiros não-negativos com  $m \leq n$ . Prove que

$$\binom{n}{k} \binom{k}{m} = \binom{n}{m} \binom{n-k}{k-m}.$$

4. Calcule  $S = 1^3 + 2^3 + \dots + n^3$

5. Calcule  $S = 1^2 \cdot 3 + 3^2 \cdot 4 + \dots + 99^2 \cdot 52$ .

6. Seja  $n$  um inteiro positivo. Prove que

$$\sum_{k=1}^n \frac{(-1)^{k-1}}{k} \binom{n}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$$

7. Seja  $n$  um inteiro não-negativo. Prove que

$$\sum_{k=0}^n k \binom{n}{k}^2 = n \binom{2n-1}{n-1}.$$

8. Prove que

a)  $\sum_{k=1}^n k^2 \binom{n}{k} = n(n+1)2^{n-2}$ .

b)  $\sum_{k=1}^n k^3 \binom{n}{k} = n^2(n+3)2^{n-3}$ .

9. Determine o coeficiente de

a)  $x^2$  em  $(x+2)^5$ .

b)  $x^5$  em  $(x^3 - \frac{1}{x^2})^{15}$ .

c) Determine o coeficiente de  $x^{n+1}$  em  $(x+2)^3(1+x)^n$ .

10. No desenvolvimento de  $(1+x^2+x^3)^9$  qual o valor do coeficiente de  $x^8$  ?

# 22

## Princípio das Gavetas

Uma pergunta bem simples motiva o estudo do conteúdo deste capítulo.

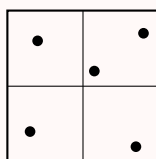
**Exemplo 22.0.1.** Qual o número mínimo de pessoas em uma sala de aula que precisamos ter para garantir que duas delas fazem aniversário no mesmo dia?

Intuitivamente sabemos que a resposta é 13, uma vez que se pensarmos no pior caso e tentarmos distribuir cada pessoa em um mês diferente, sobra uma pessoa (pois apenas cabem 12 pessoas, uma em cada mês do ano). Esta pessoa deve fazer aniversário em um dos 12 meses do ano, ou seja, ela faz aniversário no mês que alguma outra pessoa também faz por como organizamos o problema.

**Princípio das Gavetas.** Se  $n + 1$  meias forem guardadas em  $n$  gavetas, então com certeza haverá uma gaveta com duas meias.

**Exemplo 22.0.2.** Dados 5 pontos sobre um quadrado de lado, prove que existem dois deles com distância menor ou igual a  $\sqrt{2}$ .

Note que podemos dividir o quadrado de lado 2 em quatro quadrados menores de lado 1 como na figura abaixo. Como 5 pontos vão ser distribuídos no quadrado de lado 2, existe um quadrado menor que deve conter dois pontos (dentro ou em cima de seus lados).



Como a distância máxima num quadrado de lado 1 é sua diagonal que mede  $\sqrt{2}$ , os dois pontos dentro do mesmo quadrado têm distância menor ou igual a  $\sqrt{2}$ , como queríamos demonstrar.

**Exemplo 22.0.3.** Considere o conjunto  $A = \{1, 2, 3, \dots, 2022\}$ . Escolhendo aleatoriamente 1012 elementos de  $A$ , existem dois deles cujo mdc é 1.

Temos 1011 pares de números consecutivos, a saber os pares

$$\{1, 2\}, \{3, 4\}, \{5, 6\}, \dots, \{2021, 2022\}.$$

Como pegamos 1012 números, há dois deles que estão no mesmo conjunto acima. Ou seja, existem dois deles que são consecutivos, digamos  $i$  e  $i + 1$ . Utilizando o fato que  $i$  e  $i + 1$  sempre são primos entre si, concluímos que  $\text{mdc}(i, i + 1) = 1$ .

**Exemplo 22.0.4.** Dados 2023 números naturais, provar que existem dois deles cuja diferença é um múltiplo de 2022.

Seja  $A = \{a_1, a_2, \dots, a_{2023}\}$  o conjunto desses tais números. Pelo algoritmo da divisão

euclidiana, cada  $a_i$  pode ser escrito na forma

$$a_i = 2022 \cdot q_i + r_i, \quad (22.1)$$

onde  $0 \leq r_i < 2022$ . Ou seja, para cada  $i$ , temos 2022 possibilidades para o número  $r_i$ , a saber os valores  $0, 1, 2, \dots, 2021$ . Como temos 2023 números no conjunto  $A$  e apenas 2022 possibilidades para o resto de cada um deles na divisão por 2022, devem haver, pelo princípio das gavetas, dois deles, digamos  $a_m$  e  $a_n$ , que deixam mesmo resto, digamos  $r$ . Assim, temos que

$$a_m = 2022 \cdot q_m + r$$

$$a_n = 2022 \cdot q_n + r.$$

Tomando a diferença obtemos que

$$\begin{aligned} a_m - a_n &= (2022 \cdot q_1 + r) - (2022 \cdot q_2 + r) \\ &= 2022 \cdot (q_m - q_n), \end{aligned}$$

um múltiplo de 2022, o que demonstra o que queríamos.

**Exemplo 22.0.5.** Mostre que todo inteiro positivo  $n$  tem um múltiplo que só tem algarismos 0 e 1.

Para isto, imitaremos a demonstração do Exemplo 22.0.4 em certo sentido. Considere os  $n + 1$  números da forma

$$a_i = \overbrace{1111 \dots 1}^{i \text{ vezes}},$$

onde  $1 \leq i \leq n + 1$ . Então, por um argumento semelhante ao do exemplo anterior, existem  $n + 1$  números inteiros  $q_i$  e  $r_i$  tais que

$$a_i = n \cdot q_i + r_i,$$

onde  $0 \leq r_i < n$ . Daí, temos  $n + 1$  números  $a_i$  e apenas  $n$  opções de resto (os números  $0, 1, 2, \dots, n - 1$ ). Pelo princípio das gavetas, existe dois índices  $m$  e  $n$  tais que  $a_m$  e  $a_n$  têm mesmo resto na divisão por  $n$ . Suponha  $a_m > a_n$ .

Já sabemos que  $a_m - a_k$  é múltiplo de  $n$ , resta provar que é um número formado apenas por algarismos 0 e 1. De fato, temos que

$$\begin{array}{r} 1 \dots 11 \dots 1 \\ - \quad 1 \dots 1 \\ \hline 1 \dots 10 \dots 0 \end{array}$$

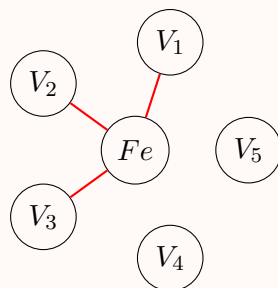
onde o primeiro número acima tem  $m$  algarismos 1 e o que está sendo subtraído tem  $k$  algarismos 1. O resultado  $(a_m - a_n)$  inicia com  $m - k$  algarismos 1 e termina com  $k$  algarismos 0.

No exemplo a seguir, consideraremos que duas pessoas  $A$  e  $B$  se conhecem mutuamente quando  $A$  conhece  $B$  e  $B$  conhece  $A$ . O mesmo vale para um grupo maior de pessoas, no sentido de que um grupo com  $n$  pessoas se conhece mutuamente se cada par de pessoas no grupo se conhece mutuamente.

**Exemplo 22.0.6 (Número de Ramsey).** Considere um grupo com 6 pessoas. Provar que sempre existem 3 pessoas que se conhecem mutuamente ou 3 pessoas que se desconhecem mutuamente.

Observe que podemos desenhar um grafo para a situação. Nele, cada uma das seis pessoas é um vértice e indicamos se duas pessoas se conhecem ligando os vértices que os representa por uma aresta azul; e ligamos os dois vértices por uma aresta vermelho se as duas pessoas não se conhecem.

No grafo, cada vértice tem 5 arestas que incidem sobre ele. Pelo princípio das gavetas, pelo menos 3 dessas arestas é azul ou vermelho. Suponha, sem perda, que há três arestas vermelhas incidindo sobre o vértice  $Fe$  (de Felipe). Abaixo representamos essa situação



Se os três vértices  $V_1, V_2, V_3$  (de Verônica, Verão e Vonildo) se ligam por arestas azuis, então temos 3 pessoas que se conhecem mutuamente e satisfazemos o que queremos. Caso contrário, há dois deles, digamos  $V_1$  e  $V_2$ , que se ligam por aresta vermelha. Assim  $V_1, V_2, Fe$  se ligam por arestas vermelhas e portanto se desconhecem mutuamente.

## §22.1 Exercícios Propostos

1. Suponha que você tenha 8 meias brancas e 7 meias pretas misturadas em uma gaveta escura. Você seleciona as meias aleatoriamente. Qual é o número mínimo de meias que você deve pegar para garantir que tenha pelo menos um par da mesma cor?
2. Num conjunto de 400 pessoas, quantas pessoas com CERTEZA, nasceram num mesmo mês?
3. Uma prova com 6 questões que tem 4 alternativas cada uma foi aplicada numa sala. Sabe-se que houve duas provas que tiveram exatamente as mesmas respostas. Quantas pessoas no mínimo tinham fazendo a prova?
4. Dê um exemplo mostrando que com 5 pessoas o exemplo (22.0.6) não é verdade.
5. Prove que existe uma potência de três que termina com os algarismos 001.
6. Uma matriz  $3 \times 3$  só tem algarismos  $-1, 0, 1$ . Prove que, entre as oito somas possíveis (linhas, colunas e diagonais), há duas que têm a mesma soma.
7. (IMO) Prove que, em um conjunto de números inteiros positivos de dois dígitos cada, há sempre dois subconjuntos disjuntos cujas somas de seus elementos são iguais.
8. (OBM 1991) Demonstre que existem infinitos múltiplos de 1991 que são da forma 1999...9991.

# 23

## Espaços de Probabilidade

A probabilidade vai tratar de experimentos aleatórios. Experimentos aleatórios é aquilo que não se pode prever. Por exemplo, lançar uma moeda e saber qual face ficará voltada para cima. Saber se irá chover amanhã. Tudo isso são experimentos aleatórios.

**Definição 23.1.** Um espaço amostral, que denotaremos pela letra grega ( $\Omega$ , lê-se ômega), é o conjunto de todos os resultados possíveis de um experimento aleatório.

**Definição 23.2.** Um evento é qualquer subconjunto  $A \subset \Omega$ .

**Exemplo 23.0.1.** Jogar uma moeda e observar a face. Neste caso,  $\Omega = \{\text{cara, coroa}\}$

**Exemplo 23.0.2.** Jogar um dado e observar o número da face voltada para cima. Neste caso,  $\Omega = \{1, 2, 3, 4, 5, 6\}$ . Ainda neste exemplo, poderíamos ter o evento  $A = \text{sair face par}$ . Neste caso,  $A = \{2, 4, 6\}$ .

**Exemplo 23.0.3.** Qual a probabilidade de ao jogarmos um dado, sair uma face par voltada para cima ?

*Solução.* A nossa intuição diz que isso deveria ser  $1/2$ . Pois temos 3 números pares num total de 6. Seja  $A = \{2, 4, 6\}$ . Denote por  $P$  a probabilidade. Temos então

$$P(A) = \frac{|A|}{|\Omega|} = \frac{3}{6} = \frac{1}{2}.$$

O exemplo acima é um protótipo do que conhecemos como probabilidade uniforme. Que é quando cada elemento do espaço amostral tem a mesma probabilidade. Mais precisamente, Seja  $\Omega = \{a_1, a_2, \dots, a_n\}$ .  $P(a_i) = \frac{1}{n}$ . Neste caso, podemos definir então a probabilidade de um evento  $A \subset \Omega$  como sendo

$$P(A) = \frac{|A|}{|\Omega|}$$

onde  $|X|$  denota o número de elementos do conjunto  $X$ . Note que dessas condições temos

- (i)  $P(\Omega) = 1$ ;
- (ii)  $0 \leq P(A) \leq 1$ ;
- (iii) Se  $A$  e  $B$  são eventos disjuntos, então  $P(A \cup B) = P(A) + P(B)$ .

*Demonstração.* (i)  $P(\Omega) = \frac{|\Omega|}{|\Omega|} = 1$ .

(ii) O número de elementos de  $A$  é sempre maior ou igual a zero e no máximo o número de elementos do espaço amostral  $\Omega$ .

(iii) Segue do princípio aditivo, pois se  $A$  e  $B$  são disjuntos então  $|A \cup B| = |A| + |B|$  e portanto

$$P(A \cup B) = \frac{|A \cup B|}{|\Omega|} = \frac{|A| + |B|}{|\Omega|} = \frac{|A|}{|\Omega|} + \frac{|B|}{|\Omega|} = P(A) + P(B).$$

□

**Exemplo 23.0.4.** Duas moedas são jogadas ao mesmo tempo. Qual a probabilidade de sair duas caras voltadas para cima ?

*Solução.* Denote Ca para representar Cara e Co para representar Coroa. O espaço amostral  $\Omega$  é  $\Omega = \{(Ca, Ca), (Ca, Co), (Co, Ca), (Co, Co)\}$ . O evento sair duas caras voltadas para cima é  $A = \{(Ca, Ca)\}$ . Logo a probabilidade pedida é

$$P(A) = \frac{|A|}{|\Omega|} = \frac{1}{4}.$$

A definição de probabilidade não precisa ser essa uniforme de casos favoráveis sobre casos possíveis. Vamos imitar o que fizemos acima para probabilidade uniforme e definir uma probabilidade um pouco mais geral.

**Definição 23.3.** Dado um espaço amostral  $\Omega$ , definimos uma probabilidade  $P$  como sendo uma função  $P$  que satisfaz

- (i)  $0 \leq P(A) \leq 1$ ;
- (ii)  $P(\Omega) = 1$ ;
- (iii) Se  $A$  e  $B$  são eventos disjuntos, então  $P(A \cup B) = P(A) + P(B)$ .

**Proposição 23.0.5 –** (1)  $P(\Omega \setminus A) = 1 - P(A)$ .

(2) Se  $A \subset B$  então  $P(B \setminus A) = P(B) - P(A)$ .

(3) Se  $A \subset B$  então  $P(A) \leq P(B)$ .

(4) Se  $A$  e  $B$  são eventos não necessariamente disjuntos então  $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ .

*Demonstração.* (1) Note que  $1 = P(\Omega) = P(A \cup (\Omega \setminus A)) = P(A) + P(\Omega \setminus A)$  e logo segue o resultado.

(2). Para 2, veja que vale a seguinte decomposição disjunta do conjunto  $B = (B \setminus A) \cup A$ . Portanto,

$$P(B) = P((B \setminus A) \cup A) = P(B \setminus A) + P(A)$$

segue então que

$$P(B \setminus A) = P(B) - P(A).$$

(3). Como  $P(C) \geq 0$  qualquer que seja o conjunto  $C \subset \Omega$ , em particular,  $P(B \setminus A) \geq 0$ , pelo item (2),  $P(B) - P(A) = P(B \setminus A) \geq 0$  e portanto  $P(A) \leq P(B)$ .

(4). Para (4), faça a seguinte decomposição disjunta do conjunto  $A \cup B$ .  $A \cup B = A \cup (B \setminus A)$ . Assim,

$$P(A \cup B) = P(A \cup (B \setminus A)) = P(A) + P(B \setminus A)$$

Por outro lado,  $B = (B \setminus A) \cup (A \cap B)$ , daí  $P(B) = P(B \setminus A) + P(A \cap B)$ . o que implicq que  $P(B \setminus A) = P(B) - P(A \cap B)$ . Concluimos que

$$P(A \cup B) = P(A \cup (B \setminus A)) = P(A) + P(B) - P(A \cap B).$$

**Exemplo 23.0.6.** É escolhido aleatoriamente um número entre 1 e 300. Qual a probabilidade dele ser divisível por 3 ou por 5 ?

*Solução.* Seja  $A$  o conjunto dos inteiros entre 1 e 300 que são divisíveis por 3. Seja  $B$  o conjunto dos inteiros positivos entre 1 e 300 que são divisíveis por 5. Neste caso  $A \cap B$  é o conjunto dos inteiros positivos entre 1 e 300 que são divisíveis por 15. Veja que o queremos calcular é  $P(A \cup B)$ . Note que  $|A| = 100, |B| = 60$  e  $|A \cap B| = 20$ . Logo

$$P(A \cup B) = P(A) + P(B) - P(A \cap B) = \frac{100}{300} + \frac{60}{300} - \frac{20}{300} = \frac{7}{15}.$$

**Exemplo 23.0.7.** Em um grupo de  $n$  pessoas, qual a probabilidade de haver pelo menos duas que faz aniversário no mesmo dia ?

*Solução.* Chamemos de  $P(n)$  essa probabilidade. Vamos calcular primeiramente a probabilidade do complementar. Ou seja, de todas as  $n$  pessoas fazerem aniversário em dias diferentes.

A primeira pessoa pode escolher qualquer dia do ano (365 possibilidades), a segunda pessoa pode escolher um dia diferente (364 possibilidades), a terceira pessoa pode escolher outro dia diferente (363 possibilidades) e assim por diante, até a última pessoa escolher um dia diferente ( $365 - (n - 1)$  possibilidades). Portanto, a probabilidade de que todas as  $n$  pessoas tenham aniversários diferentes é:

$$\frac{365 \cdot 364 \cdot 363 \cdots (365 - (n - 1))}{365^n}$$

Logo,

$$P(n) = 1 - \frac{365 \cdot 364 \cdot 363 \cdots (365 - (n - 1))}{365^n}.$$

Por exemplo,  $P(25) = 57\%$ ,  $P(50) = 97\%$  e  $P(60) = 99.5\%$ .

**Exemplo 23.0.8.**  $D = \{1, 2, \dots, 365\}$ . Escolhendo um subconjunto de 2 elementos de  $D$  ao acaso, qual a probabilidade de a soma de seus elementos ser 183?

*Solução.* O número de subconjuntos de 2 elementos é  $\binom{365}{2}$ . Dentre esses, os que dão soma 183 são  $\{1, 182\}, \{2, 181\} \cdots \{91, 92\}$  que são num total de 91 conjuntos. Logo a probabilidade pedida é

$$\frac{91}{\binom{365}{2}} = \frac{182}{365 \cdot 364} = \frac{1}{730}. \quad (23.1)$$

## §23.1 Exercícios Propostos

1. Em um baralho de 52 cartas, qual a probabilidade de escolher um Ás?
2. Uma urna contém 8 bolas vermelhas e 4 bolas azuis. Se duas bolas forem retiradas sem reposição, qual a probabilidade de que ambas sejam vermelhas?
3. Uma moeda justa é lançada 3 vezes. Qual a probabilidade de obter cara em todas as três vezes?
4. Um dado de seis faces é lançado duas vezes. Qual a probabilidade de obter um número par no primeiro lançamento e um número ímpar no segundo lançamento?

5. Um pacote de cartas contém 10 cartas numeradas de 1 a 10. Se duas cartas forem selecionadas ao acaso, com reposição, qual a probabilidade de que ambas tenham números pares?


6. Um armário tem 8 repartições, em 4 níveis, como mostra a figura abaixo. Ocupando-se metade das repartições, qual a probabilidade de que se tenha uma repartição ocupada em cada nível ?
7. Para determinada prova, são sorteados 3 questões dentre um conjunto de 5 questões. É aprovado o aluno que acertar pelo menos duas das três questões. Um aluno sabe 3 das 5 questões , qual a probabilidade dele passar na prova ?
8. Se 20 pessoas querem se sentar em fila, entre os quais estão Felipe e João. Qual a probabilidade de haver exatamente 5 pessoas entre Felipe e João?
9. Numa caixa tem  $n$  bolinhas numeradas de 1 a  $n$ . Três bolinha são sorteadas ( sem reposição). Calcule a probabilidade das 3 bolinhas terem números consecutivos.
10. Considere uma população com 10000 homens e 10000 mulheres, em que sejam daltônicos 5% dos homens e 025% das mulheres. Calcule a probabilidade de que seja mulher uma pessoa daltônica selecionada dessa população.
11. (ITA) Em um espaço amostral com uma probabilidade  $P$ , são dados os eventos  $A, B, C$  tais que  $P(A) = P(B) = \frac{1}{2}$ , com  $A$  e  $B$  independentes.  $P(A \cap B \cap C) = \frac{1}{16}$  e sabe-se que  $P((A \cap B) \cup (A \cap C)) = \frac{3}{10}$ . Calcule as probabilidade condicionais  $P(C|A \cap B)$  e  $P(C|A \cap B^c)$ .
12. (OBMU) Jogamos 10 dados comuns (com 6 faces equiprováveis numeradas de 1 a 6). Calcule a probabilidade de que a soma dos 10 resultados seja igual a 20
13. (OBMU) Joãozinho joga repetidamente uma moeda comum e honesta. Quando a moeda dá cara ele ganha 1 ponto, quando dá coroa ele ganha 2 pontos. Encontre a probabilidade (em função de  $n$ ) de que Joãozinho em algum momento tenha exatamente  $n$  pontos.



# 24

## Probabilidade Condicional

No exemplo abaixo,  $\Omega$  é o conjunto dos possíveis resultados em um dado de 6 faces não-viciado, ou seja,  $\Omega = \{1, 2, 3, 4, 5, 6\}$ .

**Exemplo 24.0.1.** Um dado não-viciado foi lançado. Responda os seguintes itens.

- Qual a probabilidade de o resultado ser menor ou igual a 3?
- Sabendo que o resultado é um número primo, qual a probabilidade dele ser menor ou igual a 3?

a. Estamos interessados no eventos  $A$  = “o resultado é um número menor ou igual a 3”. Assim, temos que

$$A = \{\text{resultado menor ou igual a 3}\} = \{1, 2, 3\},$$

e portanto

$$\mathbb{P}(A) = \frac{|A|}{|\Omega|} = \frac{3}{6} = \frac{1}{2}.$$

b. Agora considerando o evento  $B$  = “o resultado é um número primo”, temos que  $B = \{2, 3, 5\}$ , de onde segue que  $A \cap B = \{2, 3\}$  e portanto  $|B| = 3$  e  $|A \cap B| = 2$ . Assim, queremos calcular a probabilidade

$$\mathbb{P}(A|B) = \frac{|A \cap B|}{|B|} = \frac{2}{3}.$$

**Definição 24.1.** A probabilidade condicional de  $A$  dado que ocorreu  $B$  é

$$\mathbb{P}(A|B) := \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)},$$

onde  $A$  e  $B$  são eventos no espaço amostral  $\Omega$ .

**Exemplo 24.0.2.** Uma urna contém 4 bolas brancas e 6 bolas pretas. Retiram-se aleatoriamente duas bolas da urna sem reposição. Determine a probabilidade de ambas serem brancas.

Considere os eventos

$A$  = “a primeira bola é branca”

$B$  = “a segunda bola é branca”.

Queremos calcular  $\mathbb{P}(A \cap B)$ , que satisfaz

$$\mathbb{P}(A \cap B) = \mathbb{P}(B|A) \cdot \mathbb{P}(A).$$

Nos ocuparemos em calcular  $\mathbb{P}(B|A)$  e  $\mathbb{P}(A)$ . Para o evento  $A$ , temos 4 casos favoráveis (as 4 bolas brancas) dentro de 10 possíveis (as 10 bolas totais), de onde  $\mathbb{P}(A) = \frac{4}{10} = \frac{2}{5}$ . Agora dado

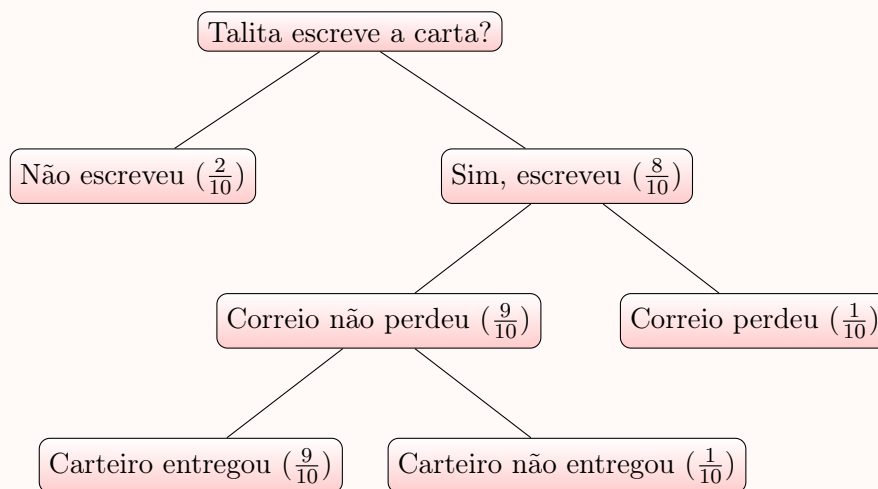
que  $A$  ocorreu, temos que a primeira bola retirada foi branca. Sabendo que não há reposição, sobram para a segunda retirada, 9 bolas, das quais apenas 3 são brancas. Daí,  $\mathbb{P}(B|A) = \frac{1}{3}$ .

Com os dois resultados acima, concluímos que

$$\mathbb{P}(A \cap B) = \frac{1}{3} \cdot \frac{2}{5} = \frac{2}{15}$$

**Exemplo 24.0.3.** Talita está pensando em escrever uma carta para Ana. A probabilidade de Talita escrever tal carta é  $\frac{8}{10}$ . O correio da cidade de Talita não perde a carta com probabilidade  $\frac{9}{10}$ . Dado que o correio não perdeu a carta, a probabilidade de o carteiro da cidade de Ana entregar é  $\frac{9}{10}$ . Dado que Ana não recebeu a carta, qual a probabilidade de Talita não ter escrito.

Temos o seguinte diagrama, com as probabilidades indicadas nos parenteses



Note que estamos interessados há três formas possíveis (e mutuamente exclusivas) de Ana não receber a carta:

1. Talita não escreveu a carta;
2. Talita escreveu, mas o correio perdeu a carta;
3. Talita escreveu, o correio não perdeu, mas o carteiro não entregou a carta.

Denote por  $A$  o evento “Talita não escreveu a carta” e  $B$  o evento “Ana não recebeu a carta”. Não é difícil ver que  $\mathbb{P}(A) = \frac{2}{10}$ . Além do mais, pelo diagrama acima e utilizando os itens 1., 2. e 3. acima, conseguimos ver que  $\mathbb{P}(B) = \frac{2}{10} + \frac{8}{10} \cdot \frac{1}{10} + \frac{8}{10} \cdot \frac{9}{10} \cdot \frac{1}{10} = \frac{352}{1000}$ . Por último, note que  $\mathbb{P}(A \cap B) = \mathbb{P}(A)$ , já que  $A$  está contido em  $B$ . Concluímos, portanto que a probabilidade de  $A$  dado  $B$  (o que queremos calcular) é dada por

$$\begin{aligned} \mathbb{P}(A|B) &= \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} = \frac{\mathbb{P}(A)}{\mathbb{P}(B)} \\ &= \frac{\frac{2}{10}}{\frac{352}{1000}} = \frac{25}{44}, \end{aligned}$$

a probabilidade pedida na questão.

Provaremos que dado um evento de probabilidade positiva  $A$  em  $\Omega$ , então a função definida

$$P_A : \Omega \rightarrow [0, 1]$$

$$B \mapsto P_A(B) = \mathbb{P}(B|A)$$

é uma probabilidade. Lembre-se que  $P : \Omega$  é uma probabilidade se para cada  $A, B \in \Omega$ , satisfaz

- i.  $P(\Omega) = 1$ ;
- ii.  $0 \leq P(A) \leq 1$ ;
- iii.  $P(A \cup B) = P(A) + P(B)$  se  $A \cap B = \emptyset$ .

Prossigamos então para a

**Proposição 24.0.4** – Seja  $A$  um evento de probabilidade positiva. Então a função  $P_A$  é uma probabilidade.

*Demonstração.* Sejam  $B, C \in \Omega$  com  $B \cap C = \emptyset$ . Então

- i. Por definição de probabilidade condicional temos que

$$P_A(\Omega) = \frac{\mathbb{P}(\Omega \cap A)}{\mathbb{P}(A)} = \frac{\mathbb{P}(A)}{\mathbb{P}(A)} = 1;$$

- ii. Para ver que  $P_A(B) \geq 0$  observe que

$$P_A(B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(A)} \geq 0,$$

pois  $\mathbb{P}$  é uma probabilidade. Além do mais, temos que  $A \cap B \subset A$ , de onde  $\mathbb{P}(A \cap B) \leq \mathbb{P}(A)$  e portanto

$$P_A(B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(A)} \leq 1.$$

- iii. Por fim, observe que

$$P_A(B \cup C) = \frac{\mathbb{P}((B \cup C) \cap A)}{\mathbb{P}(A)}.$$

Por outro lado, sabemos que  $(B \cup C) \cap A = (B \cap A) \cup (C \cap A)$ . Para concluir a demonstração, observe que como são disjuntos  $B$  e  $C$ , então são disjuntos  $B \cap A$  e  $C \cap A$  (estamos diminuindo conjuntos disjuntos, logo a interseção continua vazia). Daí, por definição de probabilidade temos que

$$\mathbb{P}((B \cup C) \cap A) = \mathbb{P}(B \cap A) + \mathbb{P}(C \cap A).$$

Substituindo a expressão acima em  $P_A(B \cup C)$ , obtemos o resultado desejado, ou seja, que  $P_A(B \cup C) = P_A(B) + P_A(C)$  sempre que  $B \cap C = \emptyset$ .  $\square$

## §24.1 Exercícios Propostos

- Escolhendo-se ao acaso um número entre 1 e 50. Se o número é primo, qual a probabilidade de que seja ímpar ?
- Uma moeda equilibrada é jogada duas vezes. Sejam  $A$  e  $B$  os eventos:

- A: cara na primeira jogada
- B: cara na segunda jogada.

Verifique que  $A$  e  $B$  são independentes.

3. Se  $A$  e  $B$  são eventos independentes tais que  $P(A) = \frac{1}{3}$  e  $P(B) = \frac{1}{2}$ . Calcule :
  - (a)  $P(A \cup B)$ ;
  - (b)  $P((\Omega \setminus A) \cup (\Omega \setminus B))$ .
4. Joguei um dado duas vezes. Calcule a probabilidade de obter 3 na primeira jogada, sabendo que a soma dos resultados foi 7.
5. Prove que se  $A$  e  $B$  são independentes então  $A$  e  $\Omega \setminus B$  também são.
6. Uma urna contém 10 bolas numeradas de 1 a 10. Sacam-se, com reposição, 4 bolas dessa urna. Sejam  $X$  e  $Y$  respectivamente o mínimo e o máximo dos números das bolas sacadas. Calcule
  - (a)  $P(X > 3)$
  - (b)  $P(X < 3)$
  - (c)  $P(X = 3)$
  - (d)  $P(X = 3|Y = 9)$
7. Felipe lança uma moeda não-viciada 4 vezes. João lança a mesma moeda 3 vezes. Calcule a probabilidade de Felipe obter mais caras do que João.
8. Quantas vezes, no mínimo, se deve lançar um dado não viciado para que a probabilidade de obter algum 6 seja superior a 0.9 ?
9. Um júri de 3 pessoas tem dois jurados que decidem corretamente ( cada um ) com probabilidade  $p$  e um terceiro jurado que decide por cara ou coroa. As decisões são tomadas por maioria. Outro júri tem probabilidade  $p$  de tomar uma decisão correta. Qual dos júris tem maior probabilidade de acerto ?
10. Um prisioneiro possui 50 bolas brancas, 50 bolas pretas e duas urnas iguais. O prisioneiro deve colocar do modo que preferir as bolas nas duas urnas ( nenhuma das urnas pode ficar vazia ). As urnas serão embaralhadas e o prisioneiro deverá, de olhos fechados, escolher uma urna e, nesta urna, uma bola. Se a bola for branca ele será libertado e, caso contrário, condenado. Como deve proceder o prisioneiro para maximizar a probabilidade de ser libertado ?
11. Considere uma população com 10000 homens e 10000 mulheres, em que sejam daltônicos 5% dos homens e 025% das mulheres. Calcule a probabilidade de que seja mulher uma pessoa daltônica selecionada dessa população.
12. (ITA) Em um espaço amostral com uma probabilidade  $P$ , são dados os eventos  $A, B, C$  tais que  $P(A) = P(B) = \frac{1}{2}$ , com  $A$  e  $B$  independentes.  $P(A \cap B \cap C) = \frac{1}{16}$  e sabe-se que  $P((A \cap B) \cup (A \cap C)) = \frac{3}{10}$ . Calcule as probabilidade condicionais  $P(C|A \cap B)$  e  $P(C|A \cap B^c)$ .

# 25 | Variáveis Aleatórias

Começaremos definindo o objeto central desta seção.

**Definição 25.1.** Uma variável aleatória é uma função

$$X : \Omega \rightarrow \mathbb{R}.$$

Uma variável aleatória associa para cada evento do espaço amostral um valor. É útil definir as variáveis aleatórias de forma esperta de acordo com o problema.

**Exemplo 25.0.1.** Imagine o experimento que consiste em lançar uma moeda  $n$  vezes. Podemos definir uma variável aleatória  $X$  que conta a quantidade de caras que aparece nos lançamentos. Normalmente representamos assim

$$X = \text{“número de caras nos } n \text{ lançamentos”}.$$

Para dois lançamentos, por exemplo, temos espaço amostral  $\Omega = \{CC, CK, KC, KK\}$ . Então  $X(CC) = 2$ ,  $X(CK) = X(KC) = 1$  e  $X(KK) = 0$ .

**Exemplo 25.0.2.** Lança-se um dado um total de 3 vezes. Defina a variável aleatória

$$X = \text{“máximo dentre os 3 resultados”}.$$

Por exemplo, se o primeiro dado deu 3; o segundo, 4; e o terceiro, 2, ou seja, o evento 342, temos que  $X(342) = 4$ . Por outro lado,  $X(166) = X(653) = 6$ .

Agora vamos calcular a probabilidade de uma variável aleatória assumir um valor desejado.

**Exemplo 25.0.3.** Lança-se um dado honesto um total de 2 vezes. Defina a variável aleatória

$$X = \text{“máximo dentre os 2 resultados”}.$$

Vamos calcular  $\mathbb{P}(X = 6)$ .

Comece observando que os resultados possíveis são um dos 36 pares ordenados

$$(1, 1), (1, 2), \dots, (6, 5), (6, 6),$$

todos com mesma probabilidade. Os resultados favoráveis são aqueles que têm 6 na primeira ou na segunda coordenada. Ou seja,

$$\{X = 6\} = \{(1, 6), (2, 6), (3, 6), (4, 6), (5, 6), (6, 6), \\ (6, 1), (6, 2), (6, 3), (6, 4), (6, 5)\},$$

de onde segue que  $|\{X = 6\}| = 11$ . Daí, temos que

$$\mathbb{P}(X = 6) = \frac{11}{36}.$$

**Definição 25.2.** Seja  $X$  uma variável aleatória. A esperança (ou valor esperado) de  $X$  é definida como sendo o valor

$$\mathbb{E}(X) = \sum x \cdot \mathbb{P}(X = x).$$

Note que na soma acima só não zeram os valores em que  $\mathbb{P}(X = x)$  é diferente de 0. Assim, só precisamos realizar a soma sobre os valores  $x$  tais que  $\mathbb{P}(X = x) \neq 0$ .

Continuando na saga das variáveis aleatórias, calcularemos agora o valor esperado de caras em um determinado experimento com lançamento de moedas utilizando uma variável aleatórias que conta as caras depois de realizado o experimento.

**Exemplo 25.0.4.** Lança-se uma moeda 3 vezes. Defina a variável aleatória

$$X = \text{“número de caras nos 3 lançamentos”}.$$

Iremos calcular o valor esperado de caras utilizando a variável aleatória  $X$ .

Note que a variável aleatória  $X$  pode assumir apenas algum dos quatro valores 0, 1, 2, 3. Por definição de valor esperado, temos que

$$\begin{aligned}\mathbb{E}(X) &= 0 \cdot \mathbb{P}(X = 0) + 1 \cdot \mathbb{P}(X = 1) + 2 \cdot \mathbb{P}(X = 2) + 3 \cdot \mathbb{P}(X = 3) \\ &= 0 \cdot \frac{1}{8} + 1 \cdot \frac{3}{8} + 2 \cdot \frac{3}{8} + 3 \cdot \frac{1}{8} = 3 \cdot \frac{1}{2}.\end{aligned}$$

Deixamos o resultado na forma  $3 \cdot \frac{1}{2}$  de maneira proposital. Veremos agora que para o tipo de experimento realizado no Exemplo 25.0.4, a esperança de  $X$  é  $np$ , onde  $n$  é o número de lançamentos e  $p$  é a probabilidade de sair o resultado que a variável aleatória  $X$  conta.

**Distribuição Uniforme.** Seja  $X$  uma variável aleatória que assume os valores  $x_1, x_2, \dots, x_n$ . Se  $\mathbb{P}(X = x_i) = \frac{1}{n}$  para todo  $1 \leq i \leq n$ , diremos que  $X$  tem distribuição uniforme

**Distribuição de Bernoulli.** Diremos que  $X$  é uma variável aleatória com distribuição de Bernoulli se  $X$  assume apenas os valores 0 e 1, com probabilidades  $P(X = 0) = 1 - p$  e  $P(X = 1) = p$ .

Costumamos escrever  $X \sim Be(p)$  para indicar que  $X$  tem distribuição de Bernoulli com probabilidade  $\mathbb{P}(X = 1) = p$ .

**Exemplo 25.0.5.** O experimento mais comum é o do lançamento de uma única moeda honesta, uma única vez, em que  $X$  indica se o resultado de cara. Temos a variável aleatória

$$X = \begin{cases} 0, & \text{se o resultado é coroa;} \\ 1, & \text{se o resultado é cara.} \end{cases}$$

Neste caso, associamos ao evento  $\{X = 1\}$  probabilidade  $\frac{1}{2}$  e ao evento  $\{X = 0\}$ , probabilidade  $\frac{1}{2}$ . Podemos, mais geralmente tratar experimentos aos resultados de um experimento os *status* de sucesso ou fracasso. Neste caso, temos a variável aleatória com distribuição de Bernoulli,  $X$  que assume o valor 0 em caso de fracasso (com probabilidade  $1 - p$ ) e assume o valor 1 em caso de sucesso (com probabilidade  $p$ ).

**Distribuição Binomial.** Diremos que  $X$  é uma variável com distribuição binomial se

# 26 | Esperança

A esperança tem as seguintes propriedades.

**Proposição 26.0.1** – Sejam  $X$  e  $Y$  variáveis aleatórias e  $c$  um número real então

1.  $E(X + Y) = E(X) + E(Y)$
2.  $E(X) = c$  se  $X$  for constante igual a  $c$ .
3.  $E(f(X)) = \sum_{x \in Im(X)} f(x)P(X = x)$ .

**Definição 26.1.** Seja  $X$  uma variável aleatória com  $E(X) = \mu$ . A variância de  $X$ , denotada por  $V(X)$  é

$$V(X) = E((X - \mu)^2).$$

A variância é uma medida de quanto a variável aleatória se afasta da Esperança.

**Exemplo 26.0.2.** Jogue um dado não-viciado. Seja  $X$  é o número da face voltada para cima. Já vimos que  $E(X) = \frac{7}{2}$ . Calcule  $V(X)$ .

*Solução.*

$$\begin{aligned} V(X) &= E((X - \frac{7}{2})^2) = \sum_{x \in Im(X)} \left(x - \frac{7}{2}\right)^2 P(X = x) \\ &= \left(1 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} + \left(2 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} + \left(3 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} + \left(4 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} + \left(5 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} + \left(6 - \frac{7}{2}\right)^2 \cdot \frac{1}{6} \\ &= \frac{1}{6} \left(\frac{25}{4} + \frac{9}{4} + \frac{1}{4} + \frac{1}{4} + \frac{9}{4} + \frac{25}{4}\right) = \frac{35}{12}. \end{aligned}$$

A proposição abaixo nos dá uma fórmula alternativa para a variância.

**Proposição 26.0.3** – Seja  $X$  uma variável aleatória, então

$$V(X) = E(X^2) - (E(X))^2.$$

*Demonstração.* Por simplicidade, chame  $E(X) = \mu$ .

$$\begin{aligned} V(X) &= E((X - \mu)^2) = E(X^2 - 2\mu X + \mu^2) \\ &= E(X^2) - 2\mu E(X) + E(\mu^2) = E(X^2) - 2\mu^2 + \mu^2 \\ &= E(X^2) - \mu^2. \end{aligned}$$

Vamos listar aqui algumas propriedades da variância.

**Proposição 26.0.4 –** 1.  $V(X + c) = V(X)$ , para qualquer  $c \in \mathbb{R}$ .

2.  $V(aX) = a^2V(X)$ .

3. Se  $X$  e  $Y$  são independentes, então  $V(X + Y) = V(X) + V(Y)$

Vamos provar aqui apenas a propriedade 3. 1 e 2 deixaremos como exercício.

$$\begin{aligned} V(X + Y) &= E((X + Y)^2) - (E(X + Y))^2 = E(X^2 + 2XY + Y^2) - (E(X) + E(Y))^2 \\ &= E(X^2) + 2E(XY) + E(Y^2) - [(E(X))^2 + 2E(X)E(Y) + (E(Y))^2] \\ &= V(X) + V(Y) \end{aligned}$$

□

Vamos calcular a variância de algumas variáveis aleatórias importantes que já vimos antes.

**Exemplo 26.0.5.** Seja  $X$  uniforme no conjunto  $\{1, \dots, n\}$ . Calcule  $V(X)$ . Primeiro, começamos calculando  $E(X^2)$ ,

$$E(X^2) = \sum_{k=1}^n k^2 P(X = k) = \frac{1^2 + 2^2 + \dots + n^2}{n}$$

Lembrando da fórmula

$$1^2 + 2^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

temos

$$E(X^2) = \frac{\frac{n(n+1)(2n+1)}{6}}{n}$$

Simplificando a expressão, temos:

$$E(X^2) = \frac{(n+1)(2n+1)}{6}$$

Já vimos que  $E(X) = \frac{n+1}{2}$ , portanto

$$\begin{aligned} V(X) &= E(X^2) - (E(X))^2 = \frac{(n+1)(2n+1)}{6} - \left(\frac{n+1}{2}\right)^2 \\ &= \frac{n^2 - 1}{12} \end{aligned} \tag{26.1}$$

## §26.1 Exercícios Propostos

*Demonstração 1.* Considere uma variável aleatória  $X$  com a seguinte função de massa de probabilidade:

$X$	0	1	2	3
$P(X)$	0.1	0.2	0.4	0.3

Calcule a variância de  $X$ .



2. Uma urna contém 5 bolas vermelhas e 7 bolas azuis. Duas bolas são retiradas sem reposição. Seja  $X$  o número de bolas vermelhas retiradas. Determine  $E(X)$  e  $V(X)$ .
3. Uma moeda viciada é lançada repetidamente até que ocorra o primeiro sucesso. A probabilidade de sucesso em um único lançamento é  $p$ . Seja  $X$  o número de lançamentos necessários. Calcule sua variância.
4. Suponha que  $X$  seja uma variável aleatória com função de massa de probabilidade dada por:

$X$	-2	0	2	4
$P(X)$	0.1	0.3	0.4	0.2

Calcule a variância de  $X$ .

5. Um jogo consiste em lançar dois dados justos e somar os resultados obtidos. Seja  $X$  a soma dos números obtidos nos lançamentos. Encontre a função de massa de probabilidade de  $X$  e calcule sua variância.
6. Considere uma variável aleatória  $X$  com a seguinte função de massa de probabilidade:

$X$	-1	0	1	2	3
$P(X)$	0.1	0.15	0.35	0.3	0.1

Calcule a variância de  $X$ .

7. Uma urna contém 8 bolas numeradas de 1 a 8. Três bolas são retiradas sem reposição. Seja  $X$  o maior número observado. Calcule sua variância.
8. Considere um experimento em que você joga um dado justo até obter um número par. Seja  $X$  o número de lançamentos necessários. Calcule sua variância.

# 27

## Enunciado das Leis dos grandes números

Imagine que você esteja jogando um dado justo de seis faces. Agora seja  $X$  a variável aleatória que anota a face que fica voltada para cima no dado.

Se você jogar o dado apenas algumas vezes, os resultados individuais podem variar consideravelmente. Por exemplo, em três lançamentos consecutivos, você pode obter os números 2, 4 e 6. A média desses três lançamentos é  $(2 + 4 + 6)/3 = 4$ .

No entanto, à medida que você continua jogando o dado um número cada vez maior de vezes, você notará que a média dos resultados se aproxima de 3,5, que é a média da variável  $X$ . Quanto mais vezes você jogar o dado, maior a probabilidade de a média dos resultados se aproximar de 3,5.

Esse é um exemplo clássico da Lei dos Grandes Números em ação. À medida que o número de lançamentos do dado aumenta, a média dos resultados se aproxima da média. Mais precisamente temos a seguinte proposição.

**Proposição 27.0.1** – (Lei fraca dos Grandes Números) Sejam  $X_1, X_2, \dots$ , variáveis aleatórias independentes todas elas com mesma média  $E[X_i] = \mu$ , então

$$\frac{X_1 + X_2 + \dots + X_n}{n} \rightarrow \mu$$

quando  $n$  tende ao infinito.

**Proposição 27.0.2** – (Lei forte dos Grandes Números) Sejam  $X_1, X_2, \dots$ , variáveis aleatórias independentes todas elas com mesma média  $E[X_i] = \mu$  e  $V(X_i) < \infty$ , então dado  $\varepsilon > 0$

$$P\left(\left|\frac{X_1 + X_2 + \dots + X_n}{n} - \mu\right| \geq \varepsilon\right) \rightarrow 0$$

Dito, de outra maneira,

$$P\left(\left|\frac{X_1 + X_2 + \dots + X_n}{n} - \mu\right| < \varepsilon\right) \rightarrow 1$$

A lei dos grandes nos afirma que à medida que aumentamos o número de experimento, a probabilidade dele ficar "longe" da média vai ficando cada vez mais perto de zero. Essa lei segue de uma desigualdade probabilística muito importante que é desigualdade de Chebyshev.

**Proposição 27.0.3** – (Desigualdade de Chebyshev) Seja  $X$  uma variável aleatória com  $E[X] = \mu$ , então para qualquer  $\varepsilon > 0$

$$P(|X - \mu| \geq \varepsilon) \leq \frac{V(X)}{\varepsilon^2}$$

Vamos dá uma ideia da demonstração da Lei forte dos grandes números como aplicação da Desigualdade de Chebyshev. Seja  $X = \frac{X_1 + \dots + X_n}{n}$ , então

$$E[X_i] = \mu \text{ e } V(X_i) = \sigma^2$$

, assim usando propriedades da variância, temos

$$V(X) = V\left(\frac{X_1 + \cdots + X_n}{n}\right) = \frac{1}{n^2}V(X_1 + \cdots + X_n) = \frac{\sigma^2}{n}.$$

Assim pela Desigualdade de Chebyshev, temos

$$P\left(\left|\frac{X_1 + X_2 + \cdots + X_n}{n} - \mu\right| \geq \varepsilon\right) \leq \frac{V(X)}{\varepsilon^2} = \frac{\sigma^2}{n\varepsilon^2} \rightarrow 0$$

## §27.1 Exercícios propostos

1. Faça uma simulação de jogar um dado algumas milhares de vezes e registre os resultados. Compare a média dos resultados com o valor esperado.
2. Considere uma variável aleatória  $X$  com média 10 e variância 25. Utilizando a Desigualdade de Chebyshev, determine o limite superior para a probabilidade de  $X$  estar no intervalo de 0 a 20.
3. Seja  $X$  uma variável aleatória com média 5 e variância 9. Utilize a Desigualdade de Chebyshev para obter uma estimativa superior para a probabilidade de  $X$  estar no intervalo de 1 a 9.
4. Jogue uma moeda 100 vezes. Ache uma cota superior para a probabilidade do número de caras ser pelo menos 40 e no máximo 60.
5. Seja  $X$  uma variável aleatória com  $E(X) = 0$  e  $V(X) = 1$ . Ache o maior inteiro  $k$  tal que  $P(|X| \geq k) \leq 0.1$ ?
6. Seja  $X$  uma variável aleatória cuja imagem é  $0, 1, 2, \dots, n$ . Além disso,  $E(X) = V(X) = 1$ . Mostre que para qualquer inteiro positivo  $k$ ,  $P(X \geq k+1) \leq \frac{1}{k^2}$ .
7. Seja  $X$  uma variável aleatória com  $E(X) = \mu$  e  $V(X) = \sigma^2$ . Use a desigualdade de Chebyshev para mostrar que

$$P(|X - \mu| < k\sigma) \geq 1 - \frac{1}{k^2}.$$

8. Seja  $X$  o número de caras em 100 lançamentos de uma moeda. Use o teorema central do limite para estimar as probabilidades abaixo.
  - a)  $P(X \leq 45)$
  - b)  $P(45 < X < 55)$
  - c)  $P(X > 63)$ .
9. Seja  $X$  o número de caras em 200 lançamentos de uma moeda. Estime  $P(X = 90)$ .
10. Considere uma dado especial com 10 faces numeradas de 1 a 10. Jogue esse dado 10000 vezes. Seja  $X$  o número de vezes que a face 3 ficou virada para cima. Faça o que se pede.
  - a) Calcule  $E(X)$
  - b) Calcule  $V(X)$ .
  - c) Use o teorema central do limite para estimar a probabilidade de sair no máximo 931 números 3.

11. Uma moeda justa é jogada 400 vezes. Seja  $X$  o número de caras no lançamentos. Determine o número  $xa$  tal que a probabilidade de  $X$  estar entre  $200 - a$  e  $200 + a$  é aproximadamente 80%.
12. Um dado é jogado 420 vezes. Qual a probabilidade da soma dos resultados obtidos estar entre 1400 e 1550?

# 28

## Enunciado do Teorema Central do Limite

Para falar sobre o Teorema Central do Limite precisamos introduzir a distribuição Normal de Gauss. Vimos até então exemplos de distribuições discretas. A Distribuição Normal de Gauss é um primeiro exemplo de Distribuição Contínua.

**Definição 28.1.** Uma variável aleatória  $X$  tem distribuição normal de Gauss com média 0 e variância 1, se

$$P(a \leq X \leq b) = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-x^2/2} dx. \quad (28.1)$$

A função  $f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$  é a função densidade da distribuição Normal.

**Proposição 28.0.1** – (Teorema Central do Limite) Sejam  $X_1, \dots, X_n, \dots$  uma sequência de variáveis aleatórias independentes todas com a mesma média  $E(X_i) = \mu$  e  $V(X_i) = \sigma^2$  então

$$\frac{X_1 + \dots + X_n}{n} - \frac{n\mu}{\sigma\sqrt{n}} \rightarrow N(0, 1) \quad (28.2)$$

quando  $n \rightarrow \infty$ .

**Exemplo 28.0.2.** Seja  $X$  o número de caras em 100 lançamentos de uma moeda não viciada. Estime  $P(X \leq 45)$ .

*Solução.* Se fôssemos fazer na força bruta isso seria  $\sum_{k=0}^{45} \frac{1}{2^{100}} \binom{100}{k}$ . Daí iríamos precisar estimar os binomiais. Mas usando o o teorema central do limite sabemos que  $X$  se aproxima de uma normal. Precisamos só padronizar ela. Calcular a esperança e a variância de  $X$ . Já vimos antes que

$$E(X) = \frac{100}{2} = 50 \text{ e } V(X) = 100 \cdot \frac{1}{2} \cdot \frac{1}{2} = 25$$

Portanto,

$$P(X < 45) = P\left(\frac{X - 50}{5} < \frac{45 - 50}{5}\right) = P(Z < -1)$$

onde  $Z = \frac{X - 50}{5}$  é uma normal com média 0 e variância 1. Usando uma tabela muito conhecida de uma gaussiana temos que  $P(Z < -1) = 0.1587$ . Ou seja,

$$P(X < 45) = 15.87\%.$$

### §28.1 Exercícios Propostos

1. A concentração de um poluente em água liberada por uma fábrica tem distribuição  $N(8; 1,5)$ . Qual a chance, de que num dado dia, a concentração do poluente exceda o limite regulatório de 10 ppm?

2. A altura
3. Seja  $X$  o número de caras em 100 lançamentos de uma moeda. Use o teorema central do limite para estimar as probabilidades abaixo.
  - a)  $P(X \leq 45)$
  - b)  $P(45 < X < 55)$
  - c)  $P(X > 63)$ .
4. Seja  $X$  o número de caras em 200 lançamentos de uma moeda. Estime  $P(X = 90)$ .
5. Considere uma dado especial com 10 faces numeradas de 1 a 10. Jogue esse dado 10000 vezes. Seja  $X$  o número de vezes que a face 3 ficou virada para cima. Faça o que se pede.
  - a) Calcule  $E(X)$
  - b) Calcule  $V(X)$ .
  - c) Use o teorema central do limite para estimar a probabilidade de sair no máximo 931 números 3.
6. Uma moeda justa é jogada 400 vezes. Seja  $X$  o número de caras no lançamentos. Determine o número  $xa$  tal que a probabilidade de  $X$  estar entre  $200 - a$  e  $200 + a$  é aproximadamente 80%.
7. Um dado é jogado 420 vezes. Qual a probabilidade da soma dos resultados obtidos estar entre 1400 e 1550?

# III

## Funções Elementares e Matrizes

### §29.1 Proposição

Assim como construir uma casa se faz pondo um bloquinho de tijolo por vez, o avanço da ciência se faz de modo extremamente similar, mas nesse caso, os “tijolos” normalmente são os trabalhos feitos por um grupo anterior de cientistas e os resultados obtidos por eles. É claro que queremos que nossa construção seja robusta e sem defeitos, assim, iremos criar ferramentas para verificar a qualidade da nossa matéria prima em questão.

Vamos agora introduzir o elemento mais básico do nosso estudo:

**Definição 29.1.** Uma **proposição** é qualquer sentença que podemos julgar como *verdadeira* ou *falsa* de maneira exclusiva, isto é, uma proposição não pode ser verdadeira e falsa ao mesmo tempo. Quando duas sentenças  $P$  e  $Q$  serão *equivalentes* quando possuem o mesmo valor lógico e, neste caso, escrevemos  $P \equiv Q$ .

**Exemplo 29.1.1.** Como exemplo de sentenças que são (ou não são) proposições mostramos os seguintes

SÃO	Francisco gosta de azul	Qual sua idade ?	
PROPOSIÇÕES	Se $\pi$ é par, então $2 + 2 = 4$	$8 \div (2 + 2)$	NÃO SÃO
	Neymar é cantor ou piloto	Eu gosto	PROPOSIÇÕES
	$x^2 + \sqrt{\pi} = -1$	Esta frase é falsa	

Tipicamente expressamos proposições por letras maiúsculas, como,  $P, Q$  ou  $R$ . Denotando proposições desse modo nos permite apenas considerar os valores lógicos delas, não levando em conta o seu significado.

### §29.2 Negação

Como cada proposição possui apenas um valor lógico, podemos estar interessados em inverter-lo. Assim, definimos a **negação** de uma proposição e denotaremos a negação da proposição  $P$  por  $\sim P$ . Alguns autores preferem usar a notação  $\neg P$  para denotar negação de  $P$ .

**Exemplo 29.2.1.** Considerando as proposições  $P$  : “6 é par”,  $Q$  : “a bola é vermelha” e  $R$  :  $x^2 + \sqrt{\pi} = -1$  nós temos

$P$	6 é par	$\sim P$	6 não é par
$Q$	A bola é vermelha	$\sim Q$	A bola não é vermelha
$R$	$x^2 + \sqrt{\pi} = -1$	$\sim R$	$x^2 + \sqrt{\pi} \neq -1$

E ao considerarmos  $S$  : “Está chovendo”, sua negação é  $\sim S$  : “Não está chovendo” e a negação dessa proposição é  $\sim(\sim S)$  : “Não é verdade que está chovendo”, que é equivalente a “Está chovendo” e esta é precisamente a proposição  $S$ .

Pelo exemplo anterior, vemos que as proposições  $S$  e  $\sim(\sim S)$  possuem o mesmo valor lógico. Podemos ver que isso ocorre com qualquer proposição através da **tabela verdade**

$P$	$\sim P$	$\sim(\sim P)$
V	F	V
F	V	F



Assim, podemos escrever simbolicamente que  $P \equiv \sim(\sim P)$ .

## §29.3 Conectivos

Imitando a nossa comunicação diária, podemos juntar duas proposições para criar uma situação mais complexa, por exemplo:

“Estou comendo e você tem 15 anos”

“Sua mãe é alta ou baixa”

Nas duas sentenças anteriores usamos os **conectivos** “e/ou”. Normalmente o conectivo “e” é usado para representar sentenças que ocorrem simultaneamente e este é representado simbolicamente por  $\wedge$  (esse operador é chamado de *conjunção*). Já o conectivo “ou” é usado para representar que as sentenças não precisam necessariamente ocorrer simultaneamente e este é representado simbolicamente por  $\vee$  (esse operador é chamado de *disjunção*).

**Exemplo 29.3.1.** Considere as seguintes afirmações:

- $P$  : 3 é ímpar
- $Q$  : 4 é par
- $R$  : 5 é par
- $S$  : 6 é ímpar

Ora, é claro que  $P$  e  $Q$  são verdadeiros e  $R$  e  $S$  são falsos. Usando os conectivos podemos criar as proposições

- $P \wedge Q$  : 3 é ímpar e 4 é par
- $R \vee S$  : 5 é par ou 6 é ímpar
- $\sim R \wedge P$  : 5 é ímpar e 3 é ímpar
- $P \wedge R$  : 3 é ímpar e 5 é par
- $P \vee S$  : 3 é ímpar ou 6 é ímpar
- $R \vee \sim P$  : 5 é par ou 3 é par

Usando os valores lógicos de  $P, Q, R$  e  $S$  concluímos que  $P \wedge Q, P \vee S$  e  $\sim R \wedge P$  são proposições verdadeiras e as outras demais são falsas.

Como vimos, os conectivos são responsáveis para combinar uma ou mais proposições, então podemos separar proposições em duas categorias distintas. Chamamos de **proposição simples** aquelas que são compostas apenas por uma proposição e **proposição composta** são aquelas que contêm duas ou mais proposições em sua composição.

Podemos resumir a natureza dos conectivos lógicos  $\wedge$  e  $\vee$  pelas tabelas verdades :

$P$	$Q$	$P \wedge Q$
$V$	$V$	$V$
$F$	$V$	$F$
$V$	$F$	$F$
$F$	$F$	$F$

$P$	$Q$	$P \vee Q$
$V$	$V$	$V$
$F$	$V$	$V$
$V$	$F$	$V$
$F$	$F$	$F$

**Observação 29.3.I.** A conjunção  $P \wedge Q$  é verdadeira apenas quando  $P$  e  $Q$  são *ambas* verdadeiras e a disjunção  $P \vee Q$  é verdadeira quando *peelo menos* uma das proposições  $P$  e  $Q$ .

A proposição a seguir resume propriedades de extrema importância no cálculo sentencial.

**Proposição 29.3.2** – Se  $P, Q$  e  $R$  são proposições, então valem

- a) Comutatividade
  - (i)  $P \vee Q \equiv Q \vee P$ ;
  - (ii)  $P \wedge Q \equiv Q \wedge P$ ;
- b) Associatividade
  - (i)  $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$ ;
  - (ii)  $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$ ;
- c) Distributividade
  - (i)  $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$ ;
  - (ii)  $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$ ;
- d) Leis de De Morgan
  - (i)  $\sim(P \wedge Q) \equiv \sim P \vee \sim Q$ ;
  - (ii)  $\sim(P \vee Q) \equiv \sim P \wedge \sim Q$ .

Uma demonstração simples destes resultados pode ser realizada através de tabelas verdade. Por exemplo, para verificar  $\sim(P \wedge Q) \equiv \sim P \vee \sim Q$  podemos considerar a tabela

$P$	$Q$	$\sim P$	$\sim Q$	$P \wedge Q$	$\sim(P \wedge Q)$	$\sim P \vee \sim Q$
$V$	$V$	$F$	$F$	$V$	$F$	$F$
$V$	$F$	$F$	$V$	$F$	$V$	$V$
$F$	$V$	$V$	$F$	$F$	$V$	$V$
$F$	$F$	$V$	$V$	$F$	$V$	$V$

E como as duas últimas colunas coincidem em todas as linhas, então fica demonstrado que  $\sim(P \wedge Q) \equiv \sim P \vee \sim Q$ .

## §29.4 Condicionais

Há ainda proposições que transmitem a ideia de causa e consequência, isso naturalmente ocorre no nosso dia a dia:

“Se eu estiver com dinheiro, então irei ao cinema.”  
 “Se amanhã é segunda-feira, então hoje é domingo.”  
 “Comemorarei se meu time ganhar.”

Na verdade, a matemática utiliza muito este tipo de sentença, por exemplo

**Teorema** (Teorema de Pitágoras) – Se  $\triangle ABC$  é triângulo retângulo de hipotenusa  $AC$ , então  $AC^2 = AB^2 + BC^2$ .

Estas são **proposições condicionais** e a proposição condicional composta por  $P$  e  $Q$  será representada por  $P \rightarrow Q$ , onde lê-se “Se  $P$ , então  $Q$ ”, “ $P$  é condição necessária para  $Q$ ” ou “ $Q$  é condição suficiente para  $P$ ”.

**Exemplo 29.4.1.** Para entendermos o funcionamento desse novo operador lógico vamos a um exemplo do cotidiano: “**Se nasci em Maceió, então sou alagoano**”.

Esta proposição é a condicional gerada pelas proposições  $P$  : “Nasci em maceió” e  $Q$  : “Sou alagoano”. Vamos procurar quando que esta sentença possui valor lógico falso. Faremos algumas observações simples:

- (i) Ora, se  $P$  for verdade, então é claro que  $Q$  também será.
- (ii) De modo similar, se  $Q$  for falso, então  $P$  não pode ser verdade.
- (iii) Se  $P$  for falso, então o valor lógico da nossa sentença muda de acordo com o valor de  $Q$ . Basta que o nascimento seja em qualquer outro município alagoano e daí  $P \rightarrow Q$  é verdadeiro. E o outro caso nos dá uma contradição, onde  $P \rightarrow Q$  é falsa.

Desse modo, o único caso que faz a proposição condicional  $P \rightarrow Q$  ser falsa é quando  $P$  é verdade e  $Q$  é falsa.

Embora o nosso último exemplo tenha a particularidade de  $P$  e  $Q$  possuírem uma relação geográfica entre si, ela nos induz a definir o condicional através da tabela verdade

$P$	$Q$	$P \rightarrow Q$
$V$	$V$	$V$
$V$	$F$	$F$
$F$	$V$	$V$
$F$	$F$	$V$

Note que  $P \rightarrow Q$  tem valor lógico falso apenas quando  $P$  é verdadeiro e  $Q$  é falsa. Além disso, observe que de qualquer afirmação falsa segue-se qualquer outra afirmação, independente se ela for verdadeira ou não. Por exemplo, a proposição

“Se 5 é par, então eu sei voar”

é logicamente verdadeira.

**Observação 29.4.I.** Vale ressaltar que o operador condicional não é uma criação abstrata de regras e sim na verdade uma composição de outras já tratadas. Para dar sustentação a afirmação anterior mostraremos que

$$P \rightarrow Q \equiv \sim P \vee Q$$

De fato, basta considerar a tabela verdade de ambas as proposições mencionadas.

$P$	$Q$	$\sim P$	$P \rightarrow Q$	$\sim P \vee Q$
$V$	$V$	$F$	$V$	$V$
$V$	$F$	$F$	$F$	$F$
$F$	$V$	$V$	$V$	$V$
$F$	$F$	$V$	$V$	$V$

Há outro condicional, conhecido como **proposição bicondicional** e representada por  $P \leftrightarrow Q$  onde lê-se “ $P$  se, e somente se,  $Q$ ” ou “ $P$  é condição necessária e suficiente para  $Q$ ”. Este operador lógico será verdade quando  $P$  e  $Q$  possuírem o mesmo valor lógico. Assim, o seu comportamento pode ser descrito pela seguinte tabela verdade

$P$	$Q$	$P \leftrightarrow Q$
$V$	$V$	$V$
$V$	$F$	$F$
$F$	$F$	$F$
$F$	$V$	$V$

**Exemplo 29.4.2.** Como a nomenclatura e o símbolo sugerem, podemos verificar que

$$P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$$

via tabela verdade.

$P$	$Q$	$P \rightarrow Q$	$Q \rightarrow P$	$P \leftrightarrow Q$	$(P \rightarrow Q) \wedge (Q \rightarrow P)$
$V$	$V$	$V$	$V$	$V$	$V$
$V$	$F$	$F$	$V$	$F$	$F$
$F$	$V$	$V$	$F$	$F$	$F$
$F$	$F$	$V$	$V$	$V$	$V$

Como as duas últimas colunas coincidem em todos os casos possíveis, então nossa afirmação está provada.

## §29.5 Tautologias

Dizemos que uma sentença é uma **tautologia** se esta é uma proposição composta por outras ligadas por conectivos e condicionais tal que seu valor lógico é verdadeiro, independente dos valores lógicos das sentenças que a compõe.

**Exemplo 29.5.1.** Há exemplos simples de tautologias, como  $P \vee \sim P$ ,  $P \rightarrow P$  ou  $P \leftrightarrow P$ . Um exemplo não tão simples é conhecido por *Modus Ponens*

$$[(P \rightarrow Q) \wedge P] \rightarrow Q.$$

Pela natureza do condicional “ $\rightarrow$ ”, sabemos que esta proposição seria falsa apenas se  $(P \rightarrow Q) \wedge P$  fosse verdadeira e  $Q$  falsa. Ora, se  $(P \rightarrow Q) \wedge P$  é verdadeira, então  $P$  e  $P \rightarrow Q$  são proposições verdadeiras e novamente pelo comportamento do operador “ $\rightarrow$ ” segue que  $Q$  deve ser verdadeira. Mas como  $Q$  não pode admitir dois valores lógicos simultaneamente, então a proposição Modus Ponens nunca é falsa, ou seja, é uma tautologia.

**Exemplo 29.5.2.** Poderíamos também utilizar a [Observação 29.4.I](#) para demonstrar que

$$[(P \rightarrow Q) \wedge \sim Q] \rightarrow \sim P$$

é tautologia do seguinte modo: Utilizando as leis de De Morgan e a distributividade dos conectivos nos vem que

$$\begin{aligned} [(P \rightarrow Q) \wedge \sim Q] \rightarrow \sim P &\equiv \sim[(P \rightarrow Q) \wedge \sim Q] \vee \sim P \equiv [\sim(P \rightarrow Q) \vee Q] \vee \sim P \\ &\equiv \sim(P \rightarrow Q) \vee (\sim P \vee Q) \equiv (P \rightarrow Q) \vee \sim(P \rightarrow Q) \end{aligned}$$

Como a última expressão é uma tautologia, segue que a nossa expressão inicial também o é.

Vale ressaltar que esses métodos se estendem para outras proposições. Desse modo, se desafie a encontrar novas tautologias.

## §29.6 Proposições logicamente falsas

Dizemos que uma sentença é uma **proposição logicamente falsa** (ou *contradição*) se esta é uma proposição composta por outras ligadas por conectivos e condicionais tal que seu valor lógico é falso, independente dos valores lógicos das sentenças que a compõe.

**Exemplo 29.6.1.** É notável que o exemplo

“3 é ímpar e 3 não é ímpar”

nos soa estranho, podemos observar que esta sentença é da forma  $P \wedge \sim P$ . Como  $P$  e  $\sim P$  possuem valores lógicos opostos, então esta é um exemplo de proposição logicamente falsa.

Outro modo para percebermos que esta é uma contradição é observar que  $P \vee \sim P$  é uma tautologia cuja a negação é  $P \wedge \sim P$ .

Ou seja, a negação de uma tautologia nos gera uma proposição logicamente falsa e vice-versa.

## §29.7 Relação de implicação

Quando a sentença  $P \rightarrow Q$  é uma tautologia, dizemos que “ $P$  implica  $Q$ ” e denotamos  $P \implies Q$ .

**Exemplo 29.7.1.** Ilustraremos essa propriedade com alguns exemplos

- a) “Se um polígono é um quadrado, então este é um retângulo”;
- b) “Se  $N$  é um número múltiplo de 22, então  $N$  é múltiplo de 2”;
- c) “Todo triângulo retângulo possui um ângulo interno de  $90^\circ$ ”;
- d) “Se  $x^2 = 4$  e  $x$  é positivo, então  $x = 2$ ”.

## §29.8 Relação de equivalência

Se  $P$  e  $Q$  são tais que  $P \leftrightarrow Q$  é uma tautologia, dizemos que  $P$  é equivalente a  $Q$  e denotamos  $P \iff Q$ .

Observe que nem todo teorema é desta forma, por exemplo, a relação entre quadrados e retângulos já apresentada no [Exemplo 29.7.1](#) não vale a recíproca, isto é, nem todo retângulo é um quadrado.

Por outro lado, é possível mostrar que [Teorema de Pitágoas](#) é uma relação de equivalência, ou seja, que se num triângulo  $\triangle ABC$  vale  $AC^2 = AB^2 + BC^2$ , então  $\triangle ABC$  é retângulo em  $B$ .

## §29.9 Sentenças abertas

Nem sempre podemos julgar a veracidade de uma proposição com facilidade pois as vezes este fato depende das variáveis envolvidas. Este é o caso da proposição “ $x^2 = 25$ ”, de fato, se  $x$  tomar qualquer outro valor diferente de  $\pm 5$  a proposição possui valor lógico falso e exatamente nesses valores a proposição é verdadeira.

Proposições que apenas podem ser julgadas verdadeira ou falsamente quando são atribuídas valores para suas variáveis são chamadas de **sentenças abertas**. Normalmente denotamos sentenças abertas explicitando a variável em questão, como por exemplo  $P(x)$  : “ $x$  é racional”.

**Exemplo 29.9.1.** Considere as sentenças abertas

$$P(n) : n^2 - n + 1 \text{ é primo} \quad \text{e} \quad Q(x) : n^3 - n + 1 \text{ é primo}$$

Iremos determinar o valor lógico do condicional  $P(n) \rightarrow Q(n)$  quando  $n = 1, 2$  ou  $3$ .

- $P(1) = 1$  é primo
- $P(2) = 3$  é primo
- $P(3) = 7$  é primo
- $Q(1) = 1$  é primo
- $Q(2) = 7$  é primo
- $Q(3) = 25$  é primo

É claro que  $P(2), Q(2)$  e  $P(3)$  são verdadeiros e os demais são falsos. Portanto, apenas os condicionais  $P(1) \rightarrow Q(1), P(2) \rightarrow Q(2)$  são verdadeiros.

## §29.10 Quantificadores

Em sua maioria, os exemplos anteriores trataram apenas de sentenças que não levassem em conta a ideia de quantidade, mas esta se faz presente na lógica e estes artifícios são conhecidos **quantificadores**. Dentre os quantificadores há dois tipos: **quantificador universal** e o **quantificador existencial**, que serão denotados por  $\forall$  e  $\exists$ , respectivamente.

Podemos ler o símbolo  $\forall$  como “para todo”, “para cada” ou “qualquer que seja”. Já o símbolo  $\exists$  pode ser lido como “existe”, “existe algum” ou “existe pelo menos um”.

**Exemplo 29.10.1.** a) Podemos considerar a proposição  $P(x, y) : “x^2 - y^2 = (x+y)(x-y)”$  e nos perguntar se é falsa ou não. Um cálculo rápido nos mostra que  $P(x, y)$  é verdadeira para quaisquer números  $x$  e  $y$

$$(x + y)(x - y) = x(x - y) + y(x - y) = x^2 - xy + yx - y^2 = x^2 - y^2$$

Desse modo, ao escrevemos  $(\forall x, y)P(x, y)$  obtemos uma proposição verdadeira.

b) Já para a proposição  $Q(p) : “Se  $p$  é primo, então  $2^p - 1$  é primo” podemos observar que  $Q(2), Q(3), Q(5)$  e  $Q(7)$  são verdadeiras. Porém, para  $p = 11$  nós temos$

$$2^{11} - 1 = 2047 = 23 \times 89,$$

logo  $Q(11)$  é falso e desse modo  $(\exists p) \sim Q(p)$  é outra proposição verdadeira.

c) Podemos misturar os dois quantificadores numa mesma proposição como a seguintes  $(\forall x \in \mathbb{R}, \exists y \in \mathbb{R})x^2 = y$  e esta é lida como “Para cada  $x$  real existe  $y$  real tal que  $x^2 = y$ ”.

## §29.11 Como negar proposições

Até agora somos capazes de negar proposições simples e usando as Leis de De Morgan podemos negar proposições com conectivos. Pela **Observação 29.4.1**, nós concluímos que  $P \rightarrow Q$  e  $\sim P \vee Q$  são equivalentes e pela **Exemplo 29.4.2** vimos que  $P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$ . Desse modo, podemos agora negar os condicionais da seguinte forma  $\sim(P \rightarrow Q) \equiv \sim(\sim P \vee Q) \equiv P \wedge \sim Q$ .

Já para o bicondicional temos que sua negação é dada por

$$\begin{aligned}\sim(P \leftrightarrow Q) &\equiv \sim[(P \rightarrow Q) \wedge (Q \rightarrow P)] \equiv \sim(P \rightarrow Q) \vee \sim(Q \rightarrow P) \equiv (P \wedge \sim Q) \vee (Q \wedge \sim P) \\ &\equiv [(P \wedge \sim Q) \vee Q] \wedge [(P \wedge \sim Q) \vee \sim P] \\ &\equiv [(P \vee Q) \wedge (\sim Q \vee Q)] \wedge [(P \vee \sim P) \wedge (\sim Q \vee \sim P)] \equiv (P \vee Q) \wedge (\sim P \vee \sim Q)\end{aligned}$$

Agora nos resta mostrar como que são as negações de proposições que possuem quantificadores. Para dar motivação vamos a uma exemplo do cotidiano

**Exemplo 29.11.1.** Suponha que Abinaldo possua um guarda-roupas com 2023 camisas, das quais uma é azul e as 2022 demais são vermelhas.



E então Bernardo, seu irmão, chega perto de Abinaldo e acontece o seguinte diálogo:

- Poxa Abinaldo, todas as suas camisas são vermelhas !
- Epa ! Você está errado, olha aqui essa camisa azul.

Então este é um exemplo extremamente simples, mas nos indica que a negação do quantificador “para todo” não é “não existe nenhuma” e sim “existe algum que falha a regra”. Simbolicamente, se  $P(x)$  : “ $x$  é vermelho”, então  $(\forall \text{ camisa})P(\text{camisa})$  : “Toda camisa é vermelha” e sua negação é  $(\exists \text{ camisa})\sim P(\text{camisa})$  : “Existe alguma camisa que não é vermelha”.

**Exemplo 29.11.2.** Suponha que Abinaldo possua um chaveiro com 2023 chaves e ele quer abrir um cadeado E, novamente seu irmão chega perto de Abinaldo e acontece o seguinte diálogo:

- Poxa Abinaldo, você já tentou aquela chave ali ?
- Já sim ! Não existe chave que abra o cadeado.
- Entendi. Todas as chaves do chaveiro não abrem o cadeado.

Então este é outro exemplo extremamente simples, mas nos indica que a negação do quantificador “existe algum” não é “não existe” e sim “para todos elementos a regra falha”. Simbolicamente, se  $P(x)$  : “ $x$  abre o cadeado”, então  $(\exists \text{ chave})P(\text{chave})$  : “Alguma chave abre o cadeado” e sua negação é  $(\forall \text{ chave})\sim P(\text{chave})$  : “Todas as chaves não abrem o cadeado”.

**Observação 29.11.I.** Generalizando os exemplos anteriores concluímos que

$$\sim(\forall x)(P(x)) \equiv (\exists x)\sim P(x) \quad \text{e} \quad \sim(\exists x)(P(x)) \equiv (\forall x)\sim P(x).$$

## §29.12 Princípio de Indução Finita

Em alguns problemas surgem padrões de fácil reconhecimento, embora que a passagem de uma conjectura para uma prova não seja da mais agradável possível. Agora mostraremos uma ferramenta do mais poderoso calibre matemático, isto é, embora que na sua essência possua simplicidade na argumentação ela se mostra capaz de atacar os mais diversos problemas.

A ferramenta que venho mencionando é conhecida por **Princípio de Indução Finita** ou simplesmente *PIF*. Na linguagem que construímos, este é um resultado que pode ser expresso na forma de um teorema

**Teorema 29.1** (Princípio de Indução Finita) – Seja  $P(n)$  é uma sentença aberta para todo  $n \in \mathbb{N}$ . Suponha que:

- (i)  $P(1)$  é verdadeira;
- (ii) Se  $k \in \mathbb{N}$  é tal que  $k \geq 1$  e  $P(k)$  é verdadeira e isto implica que  $P(k+1)$  é verdadeira.

Então,  $P(n)$  é verdade para todo  $n \in \mathbb{N}$ .

**Observação 29.12.I.** No teorema anterior, o item (i) é normalmente chamado de **Base de Indução** e no item (ii) a primeira parte é chamada de **Hipótese de Indução** e a passagem  $P(k) \implies P(k+1)$  é conhecida por **Passo Indutivo**.

De modo leigo, podemos pensar o PIF como uma fileira de dominós onde o primeiro deles cairá e para qualquer outro que caí, este derruba o seguinte, então, eventualmente todos os dominós cairão. Para estimular a importância desta nova ferramenta vamos expor alguns exemplos

**Exemplo 29.12.1.** Considere a sentença aberta

$$P(n) : 1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}.$$

Mostraremos que  $P(n)$  é verdadeira para todo natural  $n \in \mathbb{N}$ . Ora, o caso  $P(1)$  é verdadeira pois  $1 = \frac{1(1+1)}{2}$ . Suponha agora que para algum  $k \geq 1$  seja verdadeira a sentença  $P(k)$ , ou seja, vale  $1 + \cdots + k = \frac{k(k+1)}{2}$ . Desse modo, somando  $k+1$  em ambos os lados nos vem que

$$1 + 2 + \cdots + k + (k+1) = \frac{k(k+1)}{2} + (k+1) = (k+1) \left[ \frac{k}{2} + 1 \right] = \frac{(k+1)(k+2)}{2}$$

Dai  $P(k+1)$  é verdadeiro. Portanto, pelo Princípio de Indução Finita  $P(n)$  é verdadeira para todo  $n \in \mathbb{N}$ , ou seja, vale para todo  $n \in \mathbb{N}$

$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}.$$

**Exemplo 29.12.2.** Considere a sentença aberta

$$P(n) : 1 + 3 + 5 + 7 + \cdots + (2n-1) = n^2.$$



Mostraremos que  $P(n)$  é verdadeira para todo natural  $n \in \mathbb{N}$ . Ora, o caso  $P(1)$  é verdadeira pois  $1 = 1^2$ . Suponha agora que para algum  $k \geq 1$  seja verdadeira a sentença  $P(k)$ , ou seja, vale  $1 + 3 + 5 + \dots + (2k - 1) = k^2$ . Desse modo, somando  $2k + 1$  em ambos os lados nos vem que

$$1 + 3 + 5 + \dots + (2k - 1) + (2k + 1) = k^2 + (2k + 1) = (k + 1)^2$$

Dai  $P(k + 1)$  é verdadeiro. Portanto, pelo Princípio de Indução Finita  $P(n)$  é verdadeira para todo  $n \in \mathbb{N}$ , ou seja, vale para todo  $n \in \mathbb{N}$

$$1 + 3 + 5 + 7 + \dots + (2n - 1) = n^2.$$

**Exemplo 29.12.3** (Desigualdade de Bernoulli). Para qualquer número real  $x \geq -1$ , então para todo natural  $n > 0$  vale

$$(1 + x)^n \geq 1 + nx.$$

Considerando  $P(n)$  como a desigualdade acima, então imediatamente temos que  $P(1)$  é verdadeira. Supondo  $n \geq 1$  é tal que  $(1 + x)^n \geq 1 + nx$ , então multiplicando por  $1 + x \geq 0$  em ambos os lados nós obtemos

$$(1 + x)^{n+1} = (1 + x)(1 + x)^n \geq (1 + x)(1 + nx) = 1 + (n + 1)x + nx^2 \geq 1 + (n + 1)x$$

Portanto,  $P(n + 1)$  é verdadeiro, ou seja, vale para todo  $n \in \mathbb{N}$  que  $(1 + x)^n \geq 1 + nx$ .

Há outra versão do princípio da indução com uma pequena variação:

**Teorema 29.2** (Princípio da Indução Forte) – Seja  $P(n)$  é uma sentença aberta para todo  $n \in \mathbb{N}$ . Suponha que:

- (i)  $P(a)$  é verdadeira;
- (ii) Se para todo  $a \leq n \leq k$  e  $P(n)$  implica que  $P(k + 1)$  é verdadeira.

Então,  $P(n)$  é verdade para todo  $n \in \mathbb{N}$ .

**Exemplo 29.12.4** (Teorema Fundamental da Aritmética). O *Teorema Fundamental da Aritmética* afirma que qualquer inteiro  $n > 1$  pode ser escrito de forma única como produto de primos. Uma demonstração da existência dos primos não pode ser pela Indução usual pois a fatoração de dois números consecutivos não possuem relação nenhuma entre si. Desse modo, faremos uso da Indução Forte para provar a existência deste teorema.

Veja que 2 é o produto de um primo (ele mesmo), assim temos a base de indução verificada. Suponha que  $2, 3, \dots, n - 1, n$  pode ser escrito de forma única como produto de primos. Se  $n + 1$  for primo então nada temos a demonstrar. Porém se  $n + 1$  não for primo, então existem  $a, b \in \mathbb{N}$  tais que  $n + 1 = a \cdot b$ . Assim, como  $a$  e  $b$  são menores que  $n + 1$  então esses podem ser escritos como produtos de primos, logo,  $a = p_1 p_2 \dots p_s$  e  $b = q_1 q_2 \dots q_t$  e daí segue

$$n + 1 = a \cdot b = (p_1 p_2 \dots p_s)(q_1 q_2 \dots q_t)$$

**Exemplo 29.12.5** (Representação Binária). Todo número pode ser escrito de uma maneira única como soma de potências de 2 com expoentes inteiros não negativos dois-a-dois distintos.

Provaremos por indução forte que a representação existe para todo  $n \in \mathbb{N}$ . Para tanto, considere a maior potência de 2 menor ou igual a  $n$ , isto é,  $2^k \leq n < 2^{k+1}$ . Se  $n = 2^k$ , então nada temos a provar. Senão,  $1 \leq n - 2^k < n$ , e pela hipótese de indução existem inteiros não-negativos  $0 \leq a_1 < \dots < a_p$  tais que  $n - 2^k = 2^{a_1} + 2^{a_2} + \dots + 2^{a_p} < 2^k$ , logo, segue  $a_p < k$ . Portanto,  $n = 2^{a_1} + 2^{a_2} + \dots + 2^{a_p} + 2^k$  com  $0 \leq a_1 < a_2 < \dots < a_p < k$ .

**Exemplo 29.12.6** (Tome cuidado com PIF!). Usos errados do PIF pode causar resultados absurdamente falsos !

**Afirmção 29.12.7** (Todos os números naturais são pares.) — Assuma que os números  $1, 2, 3, \dots, n$  são pares. Assim, como a soma de números pares é ainda um número par, então pela hipótese de indução temos que  $n - 1$  e  $2$  são pares, logo  $n + 1 = (n - 1) + 2$  é ainda um número par. Portanto, todo número inteiro é par.  $\square$

O erro nessa “prova” está na base de indução, pois claramente ela não é verdadeira.

**Afirmção 29.12.8** (Todos os animais são da mesma raça.) — Por simplicidade se  $a$  e  $b$  são dois animais e escreveremos  $a = b$  quando eles possuírem a mesma raça. Além disso, consideramos a afirmação  $P(n)$  : “Qualquer conjunto de  $n$  animais, então todos possuem a mesma raça”. Ora,  $P(1)$  é verdadeira, pois apenas tem um único animal. Supondo que  $P(n)$  é verdade, ou seja, “Qualquer grupo de  $n$  animais é tal que todos possuem a mesma raça”. Então considere um grupo de  $n + 1$  animais, digamos,  $a_1, \dots, a_n, a_{n+1}$ . Então os animais  $a_1, \dots, a_n$  possuem a mesma raça e isto acontece com o grupo  $a_2, \dots, a_n, a_{n+1}$ . Desse modo,  $a_1 = a_2 = \dots = a_n = a_{n+1}$ , ou seja,  $P(n + 1)$  é verdadeira. E portanto todos os animais são da mesma raça.

Sabemos que a afirmação anterior possui um erro e deixamos ao leitor encontra-lo.

## §29.13 Exercícios

### Exercícios Introdutórios

**Exercício 29.1.** Para a sentença aberta  $P(n) : 2n - 4 > 2023$  onde  $n \in \mathbb{Z}$ , determine os valores  $n \in \mathbb{N}$  tais que  $P(n)$  é verdade e quais os valores  $n \in \mathbb{N}$  para que  $P(n)$  falsa.

**Exercício 29.2.** Considere as proposições :

$$P : \sqrt{2} \text{ é racional}, \quad Q : \frac{1}{2023} \text{ é racional}, \quad R : \sqrt{2023} \text{ é inteiro}.$$

Escreva as proposições a seguir e determine quais delas são verdades.

- |                                      |  |
|--------------------------------------|--|
| a) $(P \wedge Q) \rightarrow R$      | c) $(P \leftrightarrow R) \vee \sim Q$ |
| b) $(P \wedge Q) \rightarrow \sim R$ | d) $\sim(\sim P \rightarrow Q)$        |

**Exercício 29.3.** Use tabela verdade para provar as Leis de De Morgan

$$\sim(P \wedge Q) \equiv \sim P \vee \sim Q \quad \text{e} \quad \sim(P \vee Q) \equiv \sim P \wedge \sim Q.$$

**Exercício 29.4.** Considere o condicional  $P \rightarrow Q$ , então definimos a *contrapositiva*  $\sim Q \rightarrow \sim P$ , a *recíproca*  $Q \rightarrow P$  e a *inversa*  $\sim P \rightarrow \sim Q$ . Encontre quais dos novos condicionais são equivalentes a  $P \rightarrow Q$  e dê exemplos.

### Exercícios de Aprofundamento

**Exercício 29.5.** Para todo  $n \in \mathbb{N}$  valem:

- a)  $1^2 + 3^2 + 5^2 + \dots + (2n-1)^2 = \frac{n(4n^2-1)}{3};$   
 b)  $1^3 + 2^3 + 3^3 + \dots + n^3 = (1 + 2 + 3 + \dots + n)^2.$

**Exercício 29.6.** Decida quais das sentenças abaixo são tautologia:

- |  |   |
|--|---|
| a) $[(P \wedge Q) \rightarrow R] \rightarrow [P \rightarrow (Q \rightarrow R)];$   | c) $(P \rightarrow Q) \rightarrow [(P \wedge R) \rightarrow (Q \wedge R)];$ |
| b) $[P \rightarrow (Q \vee R)] \leftrightarrow [(P \wedge \sim Q) \rightarrow R];$ | d) $[(P \rightarrow Q) \wedge \sim Q] \rightarrow \sim P.$                  |

**Exercício 29.7.** Mostre que vale  $2^n \geq 1 + n$  para todo  $n \in \mathbb{Z}$  e conclua que  $2^{m+n} \geq 4mn$  para quaisquer  $m, n \in \mathbb{Z}$ .

**Exercício 29.8.** Mostre que para todo  $n \in \mathbb{N}$  o número  $2^{n+2} + 7^n$  é divisível por 5.

### Exercícios Avançados

**Exercício 29.9.** Um plano está dividido em regiões com  $n$  retas. Mostre que é possível colorir essas regiões com duas cores de modo que quaisquer duas regiões adjacentes tenham cores diferentes.

**Exercício 29.10.** Mostre que um tabuleiro de tamanho  $2^n \times 2^n$  que teve um quadrado removido pode ser coberto com peças *triminó*, onde triminó é uma peça onde foi subtraído uma das quinas de um quadrado  $2 \times 2$ .

**Exercício 29.11.** A *Sequência de Fibonacci* é definida como  $f_1 = f_2 = 1$  e  $f_{n+1} = f_n + f_{n-1}$ . O *Teorema de Zekendorf* afirma que todo número natural pode ser unicamente escrito como soma de números de Fibonacci de índices maiores que 1 e não consecutivos. Mostre, por indução, a existência desta representação para todo número natural. Embora a segunda parte também possa ser provada com indução esta é um pouco mais delicada, mesmo assim recomendo a tentativa !

### §30.1 Operações com conjuntos

Entendemos como **conjunto** uma coleção de objetos ou símbolos, que chamamos de **elementos**. Costumeiramente escrevemos letras maiúsculas para indicar conjuntos e letras minúsculas indicam seus elementos. Para indicar que  $x$  é um elemento de  $X$  nós escrevemos  $x \in X$ , isto pode ser lido como “ $x$  pertence a  $X$ ”. Já se  $y$  não for um elemento de  $X$ , escrevemos  $y \notin X$ . Por exemplo, no conjunto das vogais, que podemos indicar por  $V$ , existem os elementos  $a, e, i, o, u$ , mas a letra  $v$  não pertence a  $V$  e então  $v \notin V$ .

Um elemento de um conjunto pode ser de qualquer espécie, ou seja, pode ser um número, letras ou nomes e até mesmo outro conjunto. Como exemplo do cotidiano de conjuntos formados por conjuntos podemos considerar o conjunto dos cidadãos brasileiros e os estados onde cada um deles moram.

Iremos admitir a existência de um conjunto sem elementos conhecido por **conjunto vazio** e o denotaremos por  $\emptyset$ . Quando todos os elementos de um conjunto  $X$  também são elementos do conjunto  $Y$ , então dizemos “ $X$  **está contido em**  $Y$ ” (ou “ $Y$  **contém**  $X$ ”, ou “ $X$  **é um subconjunto de**  $Y$ ”) e denotamos  $X \subset Y$ . Quando os conjuntos  $X$  e  $Y$  são tais que  $X \subset Y$  e  $Y \subset X$ , escreveremos  $X = Y$ .

O conjunto vazio está contido em qualquer outro conjunto  $X$ . De fato, se assim não fosse, deveria existir pelo menos um elemento do conjunto vazio que não pertencesse a  $X$ , mas como  $\emptyset$  não contém elementos o argumento é sempre válido.

**Definição 30.1** (União de conjuntos). Dados dois conjuntos  $X$  e  $Y$  podemos considerar o conjunto da **união** (ou *reunião*) formado por eles, isto é, a coleção dos elementos que pertencem a  $a$ , pelo menos, um dos conjuntos  $X$  ou  $B$ . Representamos a união de  $X$  e  $Y$  por  $X \cup Y$ .

Como propriedades imediatas da definição de união de conjuntos obtemos a seguinte proposição:

**Proposição 30.1.1** – Para quaisquer conjuntos  $A, B$  e  $C$  temos:

- a)  $A \cup B = B \cup A$ ;
- b)  $B \subset A \implies A \cup B = A$ ;
- c)  $A \cup \emptyset = A$ ;
- d)  $(A \cup B) \cup C = A \cup B \cup C = A \cup (B \cup C)$

**Definição 30.2** (Interseção de conjuntos). Dados dois conjuntos  $X$  e  $Y$  podemos considerar o conjunto da **interseção** formado por eles, isto é, a coleção dos elementos que pertencem a ambos os conjuntos  $X$  e  $B$ . Representamos a interseção de  $X$  e  $Y$  por  $X \cap Y$ .

**Proposição 30.1.2** – Para quaisquer conjuntos  $A, B$  e  $C$  temos:

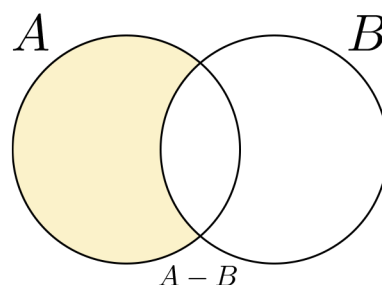
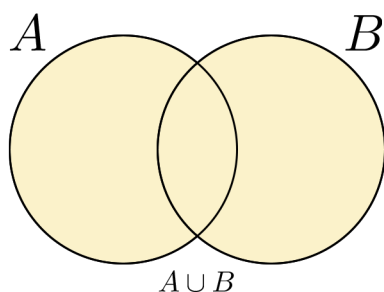
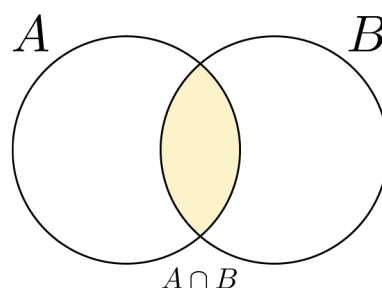
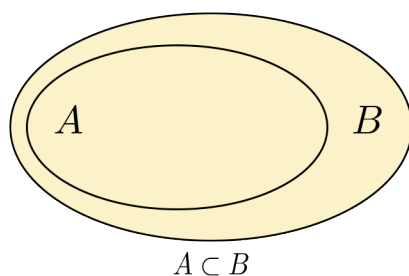
- a)  $A \cap B = B \cap A$ ;
- b)  $B \subset A \implies A \cap B = B$ ;
- c)  $A \cap \emptyset = \emptyset$ ;
- d)  $(A \cap B) \cap C = A \cap B \cap C = A \cap (B \cap C)$ .

**Definição 30.3** (Diferença de conjuntos). Dados dois conjuntos  $X$  e  $Y$  podemos considerar o conjunto da **diferença** formado por  $X$  e  $Y$ , nessa ordem, é a coleção dos elementos que pertencem  $X$  mas não a  $Y$ . Representamos a diferença de  $X$  e  $Y$  por  $X - Y$ .

**Proposição 30.1.3** – Para quaisquer conjuntos  $A$  e  $B$  temos:

- a)  $A - B \subset A$ ;
- b)  $\emptyset - A = \emptyset$ ;
- c)  $A - \emptyset = A$ ;
- d)  $A - (A \cap B) = A - B$ .

As operações com conjuntos podem ser representadas geometricamente através dos Diagramas de Venn:



## §30.2 Descrição de um conjunto

Podemos descrever conjuntos e seus elementos de algumas formas, as mais comuns são :

- Enumerando ou explicitando os seus elementos e os escrevendo entre chaves. Independentemente se o conjunto é finito ou não, o expressaremos seus elementos escrevendo entres chaves. Vale ressaltar que a adição de reticências é útil para assegurar a não finitude de um conjunto, embora que uma notação parecida também é empregada quando o conjunto é finita e nesse caso escrevemos os elementos iniciais, colocamos reticências e logo após vem os elementos finais. Como exemplos da nossa notação veja os casos a seguir :
  - i) O conjunto dos números pares pode ser descrito por  $\{0, 2, 4, 6, \dots\}$ .
  - ii) Os números inteiros de 1 até 100 pode ser expresso por  $\{1, 2, 3, 4, \dots, 98, 99, 100\}$ .
  - iii) Os países africanos com a Língua Portuguesa como idioma oficial podem ser identificados por  $\{\text{Angola, Cabo Verde, Guiné-Bissau, Moçambique, São Tomé e Príncipe}\}$ .

- Outra forma é cita-los por meio de propriedades e características. Digamos que queremos escrever os elementos que possuem a propriedade  $P$ , assim, denotaremos por  $\{x \mid x \text{ tem a propriedade } P\}$ . Como exemplos dessa notação veja os casos a seguir :
  - i) O conjunto dos números pares pode ser descrito por  $\{n \mid n \text{ é par}\}$  ou ainda  $\{2n \mid n \text{ é natural}\}$ .
  - ii) Os números inteiros de 1 até 100 pode ser expresso por  $\{n \mid n \text{ é natural e } 1 \leq n \leq 100\}$ .
  - iii) Os países africanos com a Língua Portuguesa como idioma oficial podem ser identificados por  $\{x \mid x \text{ é país africano} \wedge \text{a população de } x \text{ fala português}\}$ .

**Observação 30.2.I.** O mesmo conjunto pode ter várias representações, então como regra de prioridade utilizaremos qual for mais conveniente ao contexto. Por exemplo, podemos expressar o conjunto vazio como o conjunto dos elementos que satisfazem qualquer proposição falsa  $\{x \mid x \text{ é positivo e negativo}\} = \{x \mid x \neq x\} = \{x \mid x \text{ é inteiro e } 0 < x < 1\} = \emptyset$ .

**Exemplo 30.2.1.** Dado o conjunto  $X = \{1, 2, 3, 3, 4\}$ , podemos afirmar que :

- (a) O conjunto  $X$  possui 4 elementos, a saber, os números 1, 2 e 3 e o conjunto 3, 4.
- (b) Os números 1 e 2 pertencem a  $X$ , ou seja,  $1 \in X$  e  $2 \in X$ .
- (c) Pelo item anterior, todos os elementos de 1, 2 são também elementos de  $X$ , ou seja,  $1, 2 \subset X$ .
- (d) **Não** é verdade que  $\{3, 4\} \subset X$ , pois 4 não é elemento de  $X$ . Mas é verdade que  $\{\{3, 4\}\} \subset X$  pois  $\{3, 4\}$  é um elemento de  $X$ .

### §30.2.2 Conjuntos dos números

Há conjuntos que são fundamentais para nossos estudos são eles

- Os **números naturais** são denotados por  $\mathbb{N} = \{0, 1, 2, 3, 4, \dots\}$ . É comum denotar por  $\mathbb{N}^* = \mathbb{N} - \{0\}$  o conjunto dos naturais não nulos.

**Observação 30.2.II.** Definimos as operações de *adição* e *multiplicação* entre números naturais satisfazendo para quaisquer  $a, b, c \in \mathbb{N}$  os seguintes itens:

- (A.I). Associatividade da adição:  $(a + b) + c = a + (b + c)$ ;
- (A.II). Comutatividade da adição:  $a + b = b + a$ ;
- (A.III). Elemento *neutro* da adição:  $a + 0 = a = 0 + a$ ;
- (M.I). Associatividade da multiplicação:  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ ;
- (M.II). Comutatividade da multiplicação:  $a \cdot b = b \cdot a$ ;
- (M.III). Elemento *neutro* da multiplicação:  $a \cdot 1 = a = 1 \cdot a$ ;
- D. Distributividade da multiplicação com relação a adição:  $(a + b) \cdot c = a \cdot c + b \cdot c$ .

- Os **números inteiros** são denotados por  $\mathbb{Z} = \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ . Também denotamos  $\mathbb{Z}^* = \mathbb{Z} - \{0\}$  como o conjunto dos inteiros não nulos. Veja que se  $n$  é natural, então é imediato que  $n \in \mathbb{Z}$ , desse modo  $\mathbb{Z}$  contém  $\mathbb{N}$ .

**Observação 30.2.III.** As operações de *adição* e *multiplicação* entre números inteiros podem ser estendidas dos números naturais de modo que ainda satisfaçam as propriedades (A.I), (A.II), (A.III), (M.I), (M.II), (M.III) e D com a adição da seguinte propriedade: (A.IV) Dado  $a \in \mathbb{Z}$ , existe  $-a \in \mathbb{Z}$  tal que  $a + (-a) = 0$ .

- Os **números racionais** são denotados por  $\mathbb{Q} = \{p/q \mid p \in \mathbb{Z}, q \in \mathbb{Z}^*\}$ , isto é, os números racionais são aqueles que podem ser representados pelas frações  $\frac{p}{q}$ , onde  $p \in \mathbb{Z}$  e  $q \in \mathbb{Z}^*$ .

**Observação 30.2.IV.** Observe que para todo  $n \in \mathbb{Z}$  temos  $n = \frac{n}{1} \in \mathbb{Q}$ , logo o conjunto dos números inteiros está contido nos números racionais. Quando estamos considerando frações do tipo  $\frac{p}{q}$ , chamamos  $p$  de *numerador* e  $q$  de *denominador* e quando  $\text{mdc}(p, q) = 1$  dizemos que a fração é *irredutível*. Sobre os racionais iremos adotar como definições

- (i) *Igualdade* :  $\frac{a}{b} = \frac{c}{d} \iff ad = bc$ ;      (iii) *Multiplicação*  $\frac{a}{b} \cdot \frac{c}{d} = \frac{ac}{bd}$ .
- (ii) *Adição* :  $\frac{a}{b} + \frac{c}{d} = \frac{ad + bc}{bd}$ ;

Assim, as propriedades (A.I), (A.II), (A.III), (A.IV), (M.I), (M.II), (M.III) e D são similares e adicionamos mais uma propriedade da multiplicação: (M.IV) Dado  $\frac{p}{q} \in \mathbb{Q}^*$ , existe  $\frac{q}{p} \in \mathbb{Q}$  tal que  $\frac{p}{q} \cdot \frac{q}{p} = 1$ . Por conta de (M.IV) podemos estabelecer que a divisão de números racionais satisfaz  $\frac{a/b}{c/d} = \frac{a}{b} \cdot \frac{d}{c}$ .

Os números racionais possuem uma propriedade interessante quanto a sua representação. Existem dois casos para a representação de uma fração na base decimal:

- Sua representação tem uma quantidade finita de algarismos, diferentes de zero, isto é, uma decimal exata, como por exemplo  $2/1 = 2$  ou  $1/4 = 0,25$ .
- Sua representação tem uma quantidade infinita de algarismos que se repetem periodicamente, isto é, uma dízima periódica, como por exemplo  $1/3 = 0.333 \dots = 0, \overline{3}$ .

Quando os números não pertencem a essa categoria de representação são chamados de **números irracionais**. Para mostrarmos que existem números irracionais segue a seguinte afirmação

**Afirmação 30.2.3** — O número  $\sqrt{2}$  não é racional, isto é,  $\sqrt{2} \notin \mathbb{Q}$ .

*Prova:* Suponha que podemos escrever  $\sqrt{2}$  como uma fração  $p/q$ , onde  $\text{mdc}(p, q) = 1$ . Elevando ao quadrado obtemos  $p^2 = 2q^2$ , logo  $p^2$  é um número par e assim  $p$  é par. Então existe  $k \in \mathbb{Z}$  tal que  $p = 2k$  e substituindo em  $2q^2 = p^2$  obtemos  $2q^2 = (2k)^2 \implies q^2 = 2k^2$ , ou seja,  $q$  é par. Mas isto é um absurdo, pois tínhamos assumido que  $\text{mdc}(p, q) = 1$ . Portanto, não pode existir uma representação de  $\sqrt{2}$  por meio de uma fração.  $\square$

O argumento anterior pode ser adaptado para nos mostrar que  $\sqrt{p}$  não é um número racional sempre que  $p \in \mathbb{N}$  é primo. Em 300 A.C., Euclides de Alexandria provou em sua

obra “*Os Elementos*” que existem infinitos números primos. Então a nossa afirmação implica que existem infinitos números irracionais.

Agora podemos considerar o último conjunto numérico que trabalharemos :

- Chama-se conjunto dos **números reais**, representado por  $\mathbb{R}$ , aquele formado pela reunião do conjunto dos números racionais e dos números irracionais.

**Observação 30.2.V.** Novamente, as operações de soma e multiplicação possuem as mesmas propriedades das operações em  $\mathbb{Q}$ . Além disso, se  $\alpha$  é irracional e  $r \in \mathbb{Q}^*$ , então os números

$$\alpha + r, \quad \alpha \cdot r, \quad \frac{\alpha}{r} \quad \text{e} \quad \frac{r}{\alpha}$$

são irracionais.

Em geral, se  $X \subset \mathbb{R}$ , então denotaremos  $X_-$  o conjunto dos elementos negativos de  $X$ ,  $X_+$  o conjunto dos elementos positivos de  $X$  e  $X^*$  o conjunto dos elementos não-nulos de  $X$ .

**Exemplo 30.2.4.** Considerando os conjuntos

i)  $A = \{n \in \mathbb{N}^* \mid n \leq 9\}$

iii)  $C = \{x \in \mathbb{R} \mid x^2 - 2 = 0\}$

ii)  $B = \{x \in \mathbb{Q} \mid x \leq 9\}$

iv)  $D = \{x \in \mathbb{Q} \mid x^2 - 2 = 0\}$

pede-se:

- Descreva os elementos de  $A$  listando seus elementos;
- Dê um exemplo de três elementos que pertencem a  $B$  mas não pertencem a  $A$ ;
- Descreva o conjunto  $C$  listando seus elementos;
- Descreva o conjunto  $D$  de outra maneira.

- Podemos listar os elementos de  $A$  por  $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ ;
- Ora, como números inteiros são ainda números racionais e o conjunto  $B$  contém os inteiros negativos, então podemos considerar como exemplo os números  $-1, -2$  e  $-3$ ;
- Os números reais tais que  $x^2 - 2 = 0$ , então  $x^2 = 2$  e isto nos diz que  $x = \pm\sqrt{2}$ , logo  $C = \{-\sqrt{2}, \sqrt{2}\}$ ;
- Já demonstramos que  $\sqrt{2} \notin \mathbb{Q}$ , assim  $-\sqrt{2} \notin \mathbb{Q}$ . Portanto,  $D = \emptyset$

## §30.3 Descrição de intervalos

Os números reais são ordenados, isto é, podemos impor uma noção de grandeza entre os números. Definimos que  $a < b$  sempre que  $b - a$  é um número positivo e  $a > b$  sempre que  $a - b$  é um número positivo diremos que “ $a$  é menor que  $b$ ” e “ $a$  é maior que  $b$ ”, respectivamente.

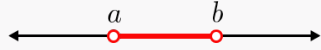
A notação  $a \leq b$  significa que  $b - a$  é positivo ou zero e de modo similar definimos  $a \geq b$ . Quando  $a < b$  e  $b < c$  escreveremos apenas  $a < b < c$ .

Chamaremos de **intervalo** um conjunto  $X \subset \mathbb{R}$  tal que se  $x, y \in X$  então qualquer  $t \in \mathbb{R}$  satisfazendo  $x < t < y$  implica que  $t \in X$ . Dados  $a, b \in \mathbb{R}$ , com  $a < b$ , podemos considerar o intervalo cujo os *extremos* são  $a$  e  $b$ , os únicos casos são:



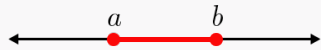
- (a) Intervalo aberto de extremos  $a$  e  $b$  é o conjunto:

$$(a, b) = \{x \in \mathbb{R} \mid a < x < b\};$$



- (b) Intervalo fechado de extremos  $a$  e  $b$  é o conjunto:

$$[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\};$$



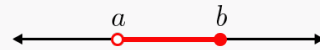
- (c) Intervalo fechado à esquerda (ou aberto à direita) de extremos  $a$  e  $b$  é o conjunto:

$$[a, b) = \{x \in \mathbb{R} \mid a \leq x < b\};$$



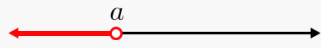
- (d) Intervalo fechado à direita (ou aberto à esquerda) de extremos  $a$  e  $b$  é o conjunto:

$$(a, b] = \{x \in \mathbb{R} \mid a < x \leq b\};$$

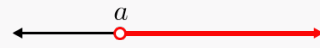


Também são intervalos reais os “intervalos infinitos” assim definidos:

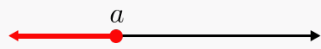
- (a)  $(-\infty, a) = \{x \in \mathbb{R} \mid x < a\};$



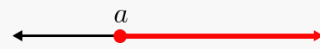
- (c)  $(a, +\infty) = \{x \in \mathbb{R} \mid a < x\};$



- (b)  $(-\infty, a] = \{x \in \mathbb{R} \mid x \leq a\};$



- (d)  $[a, +\infty) = \{x \in \mathbb{R} \mid a \leq x\};$



### §30.4 Produto cartesiano (par ordenado)

Embora a representação de um conjunto não dependa da ordem da qual seus elementos são listados (por exemplo, os conjuntos  $\{1, 2\}$  e  $\{2, 1\}$  são os mesmos), podemos impor uma noção de ordem para pares de elementos.

Esses são conhecidos por **pares ordenados** e representaremos o par ordenado formado pelos elementos  $x$  e  $y$ , nesta ordem, por  $(x, y)$ , de modo que tenhamos

$$(x, y) = (z, w) \iff x = z \text{ e } y = w$$

O **produto cartesiano** dos conjuntos  $A$  e  $B$  é definido como a união dos pares ordenados, onde a primeira entrada é um elemento de  $A$  e a segunda é um elemento de  $B$ , ou seja,

$$A \times B = \{(a, b) \mid a \in A, b \in B\}$$

**Exemplo 30.4.1.** Se  $A = \{1, 2, 3\}$  e  $B = \{x, y\}$ , então o produto cartesiano entre  $A$  e  $B$  é dado por

$$A \times B = \{(a, b) \mid a \in A, b \in B\} = \{(1, x), (1, y), (2, x), (2, y), (3, x), (3, y)\}$$

e o produto cartesiano entre  $B$  e  $A$  é dado por

$$B \times A = \{(b, a) \mid a \in A, b \in B\} = \{(x, 1), (x, 2), (x, 3), (y, 1), (y, 2), (y, 3)\}$$

Note que não há um elemento de  $A \times B$  que pertença a  $B \times A$  e vice-versa, então,  $A \times B \neq B \times A$ .

**Observação 30.4.I.** Convencionaremos que o produto cartesiano que envolva o conjunto vazio é ainda o conjunto vazio, isto é,  $A \times \emptyset = \emptyset$  e  $\emptyset \times A = \emptyset$  para qualquer que seja o conjunto  $A$ .

## §30.5 Noções básicas de funções

Podemos definir uma **relação entre conjuntos** como um subconjunto do produto cartesiano entre eles que satisfazem certa propriedade.

**Exemplo 30.5.1.** Por exemplo, considerando  $A = \{-1, 0, 1, 2\}$  e  $B = \{1, 2\}$  temos

$$A \times B = \{(-1, 1), (-1, 2), (0, 1), (0, 2), (1, 1), (1, 2), (2, 1), (2, 2)\}$$

e os seguintes subconjuntos de  $A \times B$

$$R = \{(x, y) \in A \times B \mid x = y\} = \{(1, 1), (2, 2)\}$$

e

$$\tilde{R} = \{(x, y) \in A \times B \mid y = x + 1\} = \{(0, 1), (1, 2)\}.$$

Para certos conjuntos pequenos podemos representar geometricamente relações com um *diagrama de flechas*

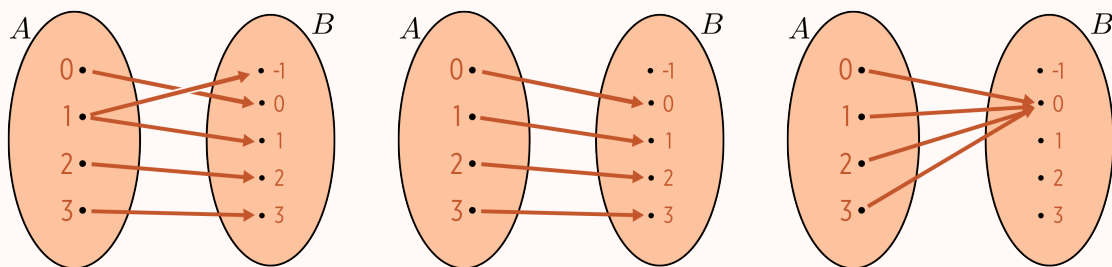
**Exemplo 30.5.2.** Considerando os conjuntos  $A = \{0, 1, 2, 3\}$  e  $B = \{-1, 0, 1, 2, 3\}$  e as relações entre esses conjuntos

$$P = \{(x, y) \in A \times B \mid y^2 - x^2 = 0\} = \{(0, 0), (1, 1), (1, -1), (2, 2), (3, 3)\}$$

$$Q = \{(x, y) \in A \times B \mid y = x\} = \{(0, 0), (1, 1), (2, 2), (3, 3)\}$$

$$R = \{(x, y) \in A \times B \mid y = 0\} = \{(0, 0), (1, 0), (2, 0), (3, 0)\}$$

Geometricamente, podemos representar essas relações pelos diagramas a seguir



As relações  $Q$  e  $R$  possuem uma peculiaridade: para cada  $x \in A$  existe um único  $y \in B$  tal que  $(x, y)$  é um elemento da relação. A relação  $P$  falha esta condição pois para o elemento  $1 \in A$  temos que os elementos  $(1, -1)$  e  $(1, 1)$  pertencem a relação.

**Definição 30.4.** Dados dois conjuntos  $A$  e  $B$  não vazios dizemos que uma relação  $f$  entre  $A$  e  $B$  é uma **função de  $A$  em  $B$**  quando para cada  $x \in A$  existe um *único* elemento  $y \in B$  tal que  $(x, y)$  é um elemento de  $A \times B$  que pertence a relação  $f$ . Neste caso, a função  $f$  será indicada por  $f : A \rightarrow B$ .

**Observação 30.5.I.** Observe que a definição de função entre dois conjuntos nos diz que o diagrama de flechas de uma função deve satisfazer as duas condições :

- (i) Todo elemento de  $A$  deve servir como ponto de partida de uma flecha;
- (ii) Cada elemento de  $A$  deve servir como ponto de partida de uma **única** flecha.

Isto nos dá uma forma prática e fácil de decidir se uma relação é uma função ou não

**Definição 30.5.** Se  $f : A \rightarrow B$  é uma função, então dizemos que  $A$  é o **domínio da função** e  $B$  é o **contradomínio da função**. Chamamos de **imagem da função** o conjunto dos elementos  $y \in B$  tais que existe  $x \in A$  tal que o par  $(x, y)$  pertence a relação. Os conjuntos domínio e imagem da função  $f$  são normalmente denotados por  $\text{Dom}(f)$  e  $\text{Im}(f)$ .

Geralmente, podemos definir uma função através da sentença aberta  $y = f(x)$ , isto é, dado  $x \in A$  determina-se  $y \in B$  tal que  $(x, y)$  é um elemento da relação. Esta sentença aberta é conhecida como **lei de correspondência** (ou *lei de formação*).

**Exemplo 30.5.3.** Considere a função  $f : \{0, 1, 2, 3\} \rightarrow \mathbb{R}$  definida por  $f(x) = x + 2$ . Assim, teríamos que o domínio de  $f$  é o conjunto  $\text{Dom}(f) = \{0, 1, 2, 3\}$  e o contradomínio da função é o conjunto  $\mathbb{R}$  dos números reais. Já a imagem de  $f$  deve ser o conjunto que contém os elementos  $f(0) = 0 + 2 = 2, f(1) = 1 + 2 = 3, f(2) = 2 + 2 = 4, f(3) = 3 + 2 = 5$ , assim,  $\text{Im}(f) = \{2, 3, 4, 5\}$ .

Desse modo, a função  $f$  de  $\{0, 1, 2, 3\}$  em  $\mathbb{R}$  definida por  $f(x) = x + 2$  é um exemplo de função tal que o contradomínio de  $f$  e sua imagem são conjuntos diferentes.

**Exemplo 30.5.4.** Para funções entre conjuntos numéricos, podemos procurar qual o “maior” conjunto de forma que os valores de  $x$  a expressão matemática  $y = f(x)$  está definida, isto é, quais valores podem ser atribuídos à variável  $x$  de modo a não violar as condições de existência da expressão matemática. Vejamos alguns casos que ilustram essa busca :

- i) Para a função  $f(x) = 2023x$  temos para todo  $x \in \mathbb{R}$  o produto com 2023 é ainda um número real, então podemos definir  $\text{Dom}(f) = \mathbb{R}$ .
- ii) Para a função  $g(x) = \frac{1}{2023x - 2022}$  não podemos definir seu domínio como  $\mathbb{R}$ , pois não podemos dividir um número real por zero. Neste caso, devemos remover de  $\mathbb{R}$  os valores de  $x$  tais que  $2023x - 2022 = 0$ , ou seja,  $x = 2022/2023$ . Desse modo,  $\text{Dom}(g) = \mathbb{R} - \left\{ \frac{2022}{2023} \right\}$ .
- iii) Um caso parecido do anterior acontece para a função definida por  $h(x) = \sqrt{2023 - x}$ . De fato, não podemos tomar raiz quadrada de números negativos, então o domínio  $h$  será os valores para os quais  $2023 - x \geq 0$ , logo,  $x \leq 2023$  e assim  $\text{Dom}(h) = (-\infty, 2023]$ .

**Exemplo 30.5.5.** Podemos encontrar a lei de formação de uma função se for dado algumas características sobre ela. Por exemplo, considere a função  $f : \mathbb{R} \rightarrow \mathbb{R}$  tal que

$$f(x + y) = f(x) \cdot f(y), \quad f(1) = 2 \quad \text{e} \quad f(\sqrt{2}) = 4$$

e pede-se para calcular  $f(3 + \sqrt{2})$ . Note que

$$f(2) = f(1 + 1) = f(1) \cdot f(1) = 2 \cdot 2 = 4$$

$$f(3) = f(1 + 2) = f(1) \cdot f(2) = 2 \cdot 4 = 8$$

$$f(3 + \sqrt{2}) = f(3) \cdot f(\sqrt{2}) = 8 \cdot 4 = 32$$

Com indução podemos mostrar que  $f(n + m\sqrt{2}) = 2^n \cdot 4^m = 2^{n+2m}$  para todos  $n, m \in \mathbb{N}$ .

**Definição 30.6.** Duas funções  $f : A \rightarrow B$  e  $g : C \rightarrow D$  são iguais quando possuem o mesmo domínio, o mesmo contradomínio e mesma imagem.

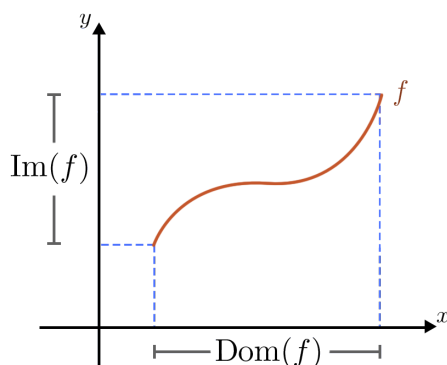
**Exemplo 30.5.6.** As funções  $f : \mathbb{N} \rightarrow \mathbb{R}$  e  $g : \mathbb{N} \rightarrow \mathbb{R}$  dadas por  $f(x) = x - 1$  e  $g(x) = \frac{x^2 - 1}{x + 1}$  são iguais pois para todo  $x \in \mathbb{N}$  temos  $x^2 - 1 = (x + 1)(x - 1)$ , logo,  $\frac{x^2 - 1}{x + 1} = x - 1$ .

Já as funções cuja as leis de formação são  $f(x) = \sqrt{\frac{x-1}{x+1}}$  e  $g(x) = \frac{\sqrt{x-1}}{\sqrt{x+1}}$  podem não ser iguais pois seus domínios não coincidem. Maiores detalhes serão deixados ao leitor cuidadoso.

## §30.6 Gráfico de uma função

**Definição 30.7.** O **gráfico de uma função**  $f : \mathbb{R} \rightarrow \mathbb{R}$  é o conjunto dos pontos  $(x, y)$  que pertencem ao plano cartesiano tais que  $y = f(x)$ .

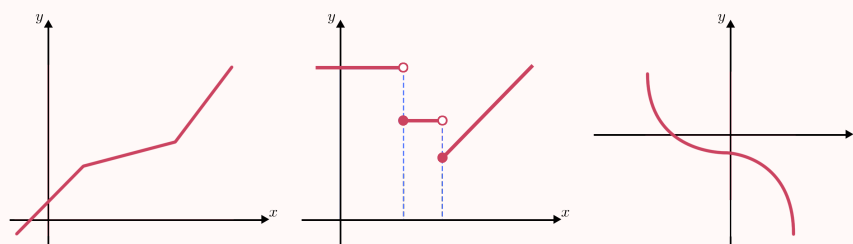
Em geral, uma vez que temos o gráfico de uma função é bem tranquilo identificar quais são o domínio e imagem dela. Além disso, o gráfico nos serve de guia para o comportamento de uma função em algum intervalo determinado.



**Observação 30.6.I.** Vale ainda observarmos que se  $f$  é uma função, então para cada  $x \in \text{Dom}(f)$  devemos ter um único  $y \in \mathbb{R}$  tal que  $y = f(x)$  isto pode ser traduzido geometricamente como: “Nenhuma reta paralela ao eixo- $y$  pode tocar o gráfico de  $f$  mais de uma vez.”

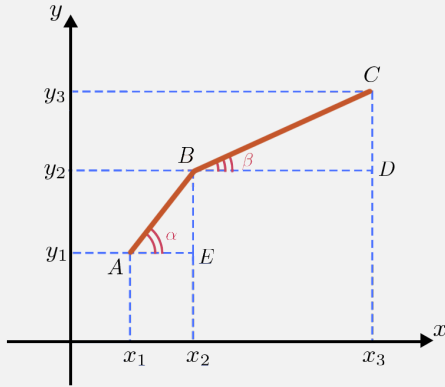
Então por exemplo, uma circunferência não pode ser o gráfico de uma função.

**Exemplo 30.6.1.** Vamos exibir alguns exemplos para gráficos de funções, mas sem se preocupar em qual o domínio ou imagem delas.



**Teorema 30.1** – O gráfico da função  $f(x) = ax + b$ , onde  $a, b \in \mathbb{R}$  é uma reta.

*Demonstração.* No caso  $a = 0$  nós temos que  $f(x) = b$ , ou seja,  $f$  é uma função constante, logo todos os pontos estão sobre a reta  $y = b$ . Já se  $a \neq 0$ , então considerando os pontos  $A(x_1, y_1)$ ,  $B(x_2, y_2)$  e  $C(x_3, y_3)$ . Observe que os triângulos  $\triangle ABE$  e  $\triangle BCD$  são semelhantes.



De fato, como  $A, B$  e  $C$  são pontos do gráfico de  $f$ , então temos  $y_1 = ax_1 + b$ ,  $y_2 = ax_2 + b$  e  $y_3 = ax_3 + b$ . Subtraindo membro a membro obtemos

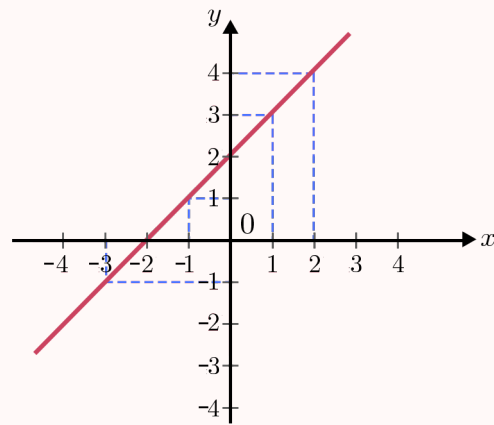
$$\begin{cases} y_3 - y_2 = a(x_3 - x_2) \\ y_2 - y_1 = a(x_2 - x_1) \end{cases}$$

donde temos  $\frac{y_3 - y_2}{x_3 - x_2} = a = \frac{y_2 - y_1}{x_2 - x_1}$ . Assim, os triângulos  $\triangle ABE$  e  $\triangle BCD$  tem lados proporcionais, logo são semelhantes.

Desse modo, os ângulos  $\widehat{BAE}$  e  $\widehat{CBD}$  são iguais e portanto  $A, B$  e  $C$  são pontos colineares.  $\square$

**Exemplo 30.6.2.** Com o Teorema 30.1 em mãos podemos tentar esboçar o gráfico de funções da forma  $f(x) = ax + b$ . Por exemplo, pondo  $a = 1$  e  $b = 2$  nós temos que  $f(x) = x + 2$ . Tomando alguns valores obtemos o esboço a seguir

$x$	$y = f(x)$
-3	-1
-2	0
-1	1
0	2
1	3
2	4



## §30.7 Função injetora, sobrejetora e bijetora

**Definição 30.8.** Se  $f : A \rightarrow B$  é uma função, então dizemos que  $f$  é uma **função injetora** quando  $f(x) = f(y)$  implica que  $x = y$ . Ou equivalentemente,  $x \neq y$  implica que  $f(x) \neq f(y)$ .

**Exemplo 30.7.1.** A função  $f(x) = ax + b$  é uma função injetora para quaisquer  $a, b \in \mathbb{R}$ , com  $a \neq 0$ , pois

$$f(x) = f(y) \implies ax + b = ay + b \implies ax = ay \implies x = y$$

Já a função  $f : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = x^2$  não é injetora pois  $f(1) = 1 = f(-1)$ . Mas se tomarmos  $g : \mathbb{R}_+ \rightarrow \mathbb{R}$  com a lei  $g(x) = x^2$  nós temos que  $g$  é injetiva.

**Observação 30.7.I.** Voltando ao diagrama de flechas, uma função injetora pode ser pensada como aquela que o diagrama não possui duas flechas chegando num mesmo elemento do contradomínio. Já o gráfico de uma função injetora pode ser caracterizado por: “toda reta paralela ao eixo- $x$  intersecta o gráfico de  $f$  no máximo em um ponto”.

**Definição 30.9.** Uma função  $f : A \rightarrow B$  é dita ser **função sobrejetora** quando para todo  $y \in B$  existe  $x \in A$  tal que  $f(x) = y$ . Ou equivalentemente, quando sua imagem coincide com o contradomínio.

**Exemplo 30.7.2.** As funções da forma  $f(x) = ax + b$ , onde  $a, b \in \mathbb{R}$  e  $a \neq 0$  são sobrejetoras. De fato, dado  $y \in \mathbb{R}$ , considerando  $x = (y - b)/a$  nós obtemos

$$f(x) = f\left(\frac{y-b}{a}\right) = a \cdot \frac{y-b}{a} + b = (y-b) + b = y.$$

**Observação 30.7.II.** No diagrama de flechas de uma função sobrejetoras, todo elemento do contradomínio deve ser o ponto de chegada de pelo menos uma flecha. No plano cartesiano, se todas as retas horizontais cortam o gráfico, então a função é sobrejetiva.

**Exemplo 30.7.3.** Suponha que  $f : [0, 1] \rightarrow [0, 1]$  é uma função sobrejetora tal que  $|f(x) - f(y)| \leq |x - y|$  para quaisquer  $x, y \in [0, 1]$ . Pela sobrejetividade de  $f$  existem  $a, b \in [0, 1]$  tais que  $f(a) = 0$  e  $f(b) = 1$ , assim

$$1 = |1 - 0| = |f(b) - f(a)| \leq |b - a| \leq 1$$

Logo,  $|b - a| = 1$  e isto nos implica que  $a = 0$  e  $b = 1$  ou  $a = 1$  e  $b = 0$ . Suponha que  $a = 0$  e  $b = 1$  e considere  $c \in (0, 1)$ . Assim, pela desigualdade triangular nós obtemos

$$1 = |f(1) - f(0)| \leq |f(1) - f(c)| + |f(c) - f(0)| \leq |1 - c| + |c - 0| = (1 - c) + c = 1$$

donde concluímos que  $|f(c) - f(0)| = |c - 0|$  e uma vez que  $c$  e  $f(c)$  são números não-negativos então  $f(c) = c$ . Como  $c \in (0, 1)$  foi arbitrário, segue que  $f(x) = x$  para todo  $x \in [0, 1]$ .

O caso  $a = 1$  e  $b = 0$  é tratado de modo similar e concluímos que  $f(x) = 1 - x$  para todo  $x \in [0, 1]$ .

**Definição 30.10.** Dizemos que  $f$  é uma **função bijetora** se esta é *injetora* e *sobrejetora*.

**Exemplo 30.7.4.** Juntando os argumentos dados no [Exemplo 30.7.1](#) e no [Exemplo 30.7.2](#) obtemos as funções da forma  $f(x) = ax + b$ , onde  $a, b \in \mathbb{R}$  e  $a \neq 0$ , como exemplo imediato de bijeção.

**Observação 30.7.III.** Há um princípio de contagem conhecido como *Princípio da Casas dos Pombos*, este afirma que se tivermos  $n + 1$  pombos para serem colocados em  $n$  casas, então pelo menos uma casa deverá conter dois ou mais pombos. A prova deste princípio é por contradição. Suponha que nenhum das casas possui mais de um pombo, assim a quantidade de pombos será no máximo  $n$  o que contradiz a hipótese inicial sobre a quantidade de pombos.

Usaremos esse princípio para estudar o “tamanho” dos domínios e contradomínios de funções injetoras e sobrejetoras.

A noção de tamanho de um conjunto finito pode ser tratada como a quantidade de elementos que ele possui e chamaremos isto de **cardinalidade de um conjunto**. É comum denotarmos a cardinalidade de um conjunto finito  $A$  por  $\text{Card}(A)$ ,  $\#A$  ou  $|A|$ .

**Proposição 30.7.5** – Seja  $f : A \rightarrow B$  uma função onde  $A$  e  $B$  são conjuntos finitos.

- a) Se  $|A| > |B|$ , então  $f$  não é injetiva;
- b) Se  $|A| < |B|$ , então  $f$  não é sobrejetora.

*Demonstração.* a) Considere cada elemento de  $A$  como um pombo e os elementos de  $B$  como uma casa. Dado um elemento  $a \in A$ , colocaremos este pombo na casa  $b \in B$  se  $f(a) = b$ . Como existem mais casas do que pombos, pelo Princípio da Casa dos Pombos há uma casa que possui pelo menos dois pombos, ou seja, existem  $a, \tilde{a} \in A$  tais que  $f(a) = f(\tilde{a})$ . Portanto,  $f$  não é injetiva.

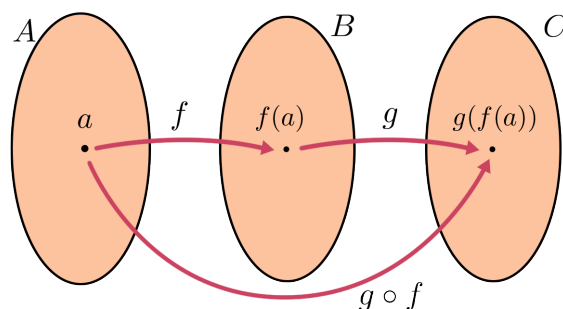
- b) Como  $f$  é uma função, então para cada  $a \in A$  deve existir um único  $b \in B$  a ser associado por  $f(a) = b$ . Assim, no máximo  $|A|$  elementos de  $B$  podem ser mapeados por  $f$ . Desse modo, se  $|A| < |B|$ , então deve existir algum elemento de  $B$  que não foi relacionado por  $f$ , ou seja,  $f$  não é sobrejetiva.

□

**Observação 30.7.IV.** A contrapositiva da [Proposição 30.7.5](#) é mais interessante pois ela nos diz que se  $f$  é injetiva, então  $|A| \leq |B|$  e se  $f$  é sobrejetiva, então  $|A| \geq |B|$ . Isto implica que para uma função ser bijeção devemos ter  $|A| = |B|$ . Essas afirmações também valem quando os conjuntos considerados são infinitos, mas este estudo foge do escopo deste texto.

## §30.8 Função composta

**Definição 30.11.** Sejam  $A, B$  e  $C$  conjuntos,  $f : A \rightarrow B$  e  $g : B \rightarrow C$ . Definimos a **função composta**  $g \circ f : A \rightarrow C$  por  $(g \circ f)(a) = g(f(a))$  para todo  $a \in A$ . Escreveremos apenas  $g \circ f$  para denotar  $(g \circ f)(a)$  para todo  $a$  no domínio de  $f$ .



**Exemplo 30.8.1.** Considere as funções  $f, g : \mathbb{R} \rightarrow \mathbb{R}$  dadas por  $f(x) = x^2$  e  $g(x) = x + 1$ , assim as composições  $f \circ g$  e  $g \circ f$  são dadas por

$$(f \circ g)(x) = f(g(x)) = f(x + 1) = (x + 1)^2 \quad \text{e} \quad (g \circ f)(x) = g(f(x)) = g(x^2) = x^2 + 1.$$

Este exemplo ilustra que em geral temos  $f \circ g \neq g \circ f$ , ou seja, a composição de funções não é uma operação comutativa. Pode ainda acontecer que apenas uma dessas composições estejam definidas.

**Exemplo 30.8.2.** Dizemos que uma função  $h$  é *ímpar* se vale  $h(-x) = -h(x)$  para todo  $x \in \mathbb{R}$ . Se  $g : \mathbb{R} \rightarrow \mathbb{R}$  é uma função ímpar tal que  $g(x) > 0$  para todo  $x > 0$ , então existe  $f : \mathbb{R} \rightarrow \mathbb{R}$  tal que  $g = f \circ f$ . De fato, basta considerar a função definida por partes

$$f(x) = \begin{cases} -x, & \text{se } x \geq 0, \\ -g(x), & \text{se } x < 0. \end{cases}$$

Note que para  $x \geq 0$  temos

$$(f \circ f)(x) = f(f(x)) = f(-x) = -g(-x) = g(x)$$

e para  $x < 0$  temos

$$(f \circ f)(x) = f(f(x)) = f(-g(x)) = f(g(-x)) = -g(-x) = g(x).$$

**Proposição 30.8.3** – Sejam  $f : A \rightarrow B$  e  $g : B \rightarrow C$ , então :

- a) Se  $g \circ f$  é injetora, então  $f$  é injetora;
- b) Se  $g \circ f$  é sobrejetora, então  $g$  é sobrejetora;
- c) Se  $f$  e  $g$  são injetoras, então  $f \circ g$  é injetora;
- d) Se  $f$  e  $g$  são sobrejetoras, então  $f \circ g$  é sobrejetora;
- e) Se  $f$  e  $g$  são bijetoras, então  $f \circ g$  é bijetora.

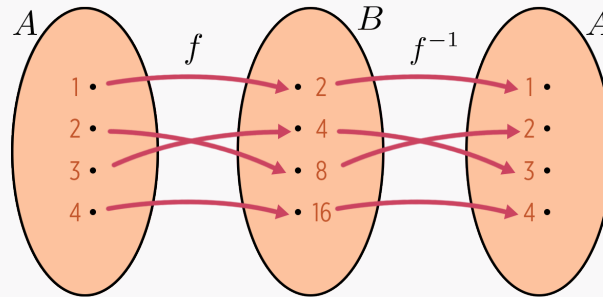
**Observação 30.8.I.** A prova da proposição anterior é bastante simples e deixamos ao cargo do leitor. Vale ressaltar que as recíprocas dos itens a) e b) não são verdadeiras pois basta considerar  $f(x) = x$  e  $g(x) = x^2$  no primeiro item e  $f(x) = x^2$  e  $g(x) = x$  no segundo.

## §30.9 Função inversa

**Definição 30.12.** Para um conjunto  $A$  definimos a *função identidade*  $\text{Id}_A : A \rightarrow A$  por  $\text{Id}_A(x) = x$  para todo  $x \in A$ . Dado uma função  $f : A \rightarrow B$  definimos a **função inversa** de  $f$  pela função  $f^{-1} : B \rightarrow A$  que satisfaz  $f^{-1} \circ f = \text{Id}_A$  e  $f \circ f^{-1} = \text{Id}_B$ .



**Observação 30.9.I.** Podemos ter uma noção de como a *função inversa* funciona através do diagrama de flechas. Intuitivamente podemos pensar que as flechas de  $f^{-1}$  fazem o caminho contrário das flechas de  $f$ , assim se tomarmos  $A = \{1, 2, 3, 4\}$ ,  $B = \{2, 4, 8, 16\}$  e a função  $f(x) = 2^x$  nós temos que os diagramas de flechas de  $f$  e  $f^{-1}$  são os seguintes

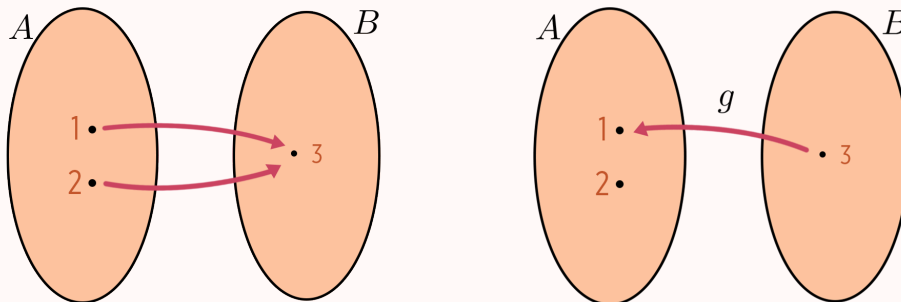


**Exemplo 30.9.1** (Nem toda função admite inversa !). Para mostrar que na definição Definição 30.12 não garante a existência de inversa para qualquer função iremos considerar os conjuntos  $A = \{1, 2\}$ ,  $B = \{3\}$  e definir a função  $f : A \rightarrow B$  por  $f(1) = 3$  e  $f(2) = 3$ . Agora nos perguntamos quem deveria ser a inversa de  $f$ ? Bom, pela definição deveria ser uma função  $f^{-1} : B \rightarrow A$  que satisfaz  $f^{-1} \circ f = \text{Id}_A$ , ou seja,

$$f^{-1}(3) = f^{-1}(f(1)) = (f^{-1} \circ f)(1) = 1 \quad \text{e} \quad f^{-1}(3) = f^{-1}(f(2)) = (f^{-1} \circ f)(2) = 2$$

Mas isto nos diz que  $f^{-1}$  não é uma função, pois no diagrama flechas de  $f^{-1}$  há duas flechas saindo de 3. Então mesmo que a função seja sobrejetora, não é garantido que exista função inversa.

Definindo  $g : B \rightarrow A$  por  $g(3) = 1$  nós temos uma função injetiva que também não possui inversa pois  $g^{-1}(x)$  não está definida para  $x = 2$ , logo  $g^{-1} : B \rightarrow A$  não é função.



Desse modo, as condições de injetividade e sobrejetividade por si só não garantem a existência de inversa e daí devemos impor mais condições para garantir a existência desta função.

**Teorema 30.2** – Seja uma função  $f : A \rightarrow B$ . A função inversa  $f^{-1} : B \rightarrow A$  existe se, e somente se,  $f$  for uma bijeção.

*Demonstração.* ( $\Rightarrow$ ) Se  $f^{-1} : B \rightarrow A$  é uma função, então para todo  $y \in B$  existe um único  $x \in A$  tal que  $f^{-1}(y) = x$ , logo,  $f(x) = f(f^{-1}(y)) = y$  e portanto  $f$  é sobrejetiva. Nos resta mostrar que  $f$  é injetiva. Desse modo, considere  $a_1, a_2 \in A$  tais que  $f(a_1) = f(a_2)$ ,

aplicando  $f^{-1}$  em ambos os lados segue

$$f(a_1) = f(a_2) \implies f^{-1}(f(a_1)) = f^{-1}(f(a_2)) \implies a_1 = a_2$$

e portanto  $f$  é uma função injetora. Donde concluímos que  $f$  é uma bijeção.

( $\Leftarrow$ ) Assuma que  $f$  é uma bijeção, assim, para todo  $b \in B$  existe um único  $a \in A$  tal que  $f(a) = b$ . Definiremos uma função  $g : B \rightarrow A$  por  $g(b) = a$  sempre que  $f(a) = b$ . Como  $f$  é sobrejetora, então para todo  $b \in B$  existe  $a \in A$  tal que  $(a, b) \in f$  e isto implica que  $(b, a) \in g$ . Agora nos resta mostrar que  $(b, a)$  é o único elemento de  $g$  cuja a primeira coordenada é  $b$ . Daí, se  $b \in B$  é tal que  $g(b) = a_1$  e  $g(b) = a_2$ , para algum  $a_1, a_2 \in A$ , pela definição de  $g$  temos  $f(a_1) = b$  e  $f(a_2) = b$ . Usando o fato de  $f$  ser injetora, então segue que  $a_1 = a_2$ . Desse modo, mostramos que para todo  $b \in B$  existe um único  $a \in A$  tal que  $(b, a) \in g$ , ou seja,  $g$  é uma função de  $B$  em  $A$ . Além do mais, temos para todo  $a \in A$  e  $b \in B$

$$(g \circ f)(a) = g(f(a)) = a \quad \text{e} \quad (f \circ g)(b) = f(g(b)) = b$$

Portanto,  $g$  é a inversa de  $f$ .

□

Como um corolário imediato da demonstração do teorema anterior temos

**Corolário 30.3** – Se  $f : A \rightarrow B$  é uma bijeção, então  $f^{-1} : B \rightarrow A$  é uma bijeção.

**Exemplo 30.9.2.** Pelo [Exemplo 30.7.2](#) sabemos que as funções da forma  $f(x) = ax + b$ , onde  $a, b \in \mathbb{R}$  e  $a \neq 0$ , são bijeções, logo podemos encontrar sua inversa. Desse modo, queremos encontrar a expressão de  $f^{-1} : B \rightarrow A$  que satisfaz  $(f \circ f^{-1})(x) = x$  para todo  $x \in \mathbb{R}$ , logo

$$(f \circ f^{-1})(x) = x \iff a \cdot f^{-1}(x) + b = x \iff f^{-1}(x) = \frac{x - b}{a}.$$

Note que desta expressão também temos  $(f^{-1} \circ f)(x) = f^{-1}(ax + b) = x$ . Portanto a inversa de  $f(x) = ax + b$  é dada por  $f^{-1}(x) = \frac{b - x}{a}$ .

Para encerrarmos o nosso estudo, convidamos ao leitor a provar e explorar os casos que não são igualdades no máximo de itens da proposição a seguir

**Proposição 30.9.3** – Sejam  $f : X \rightarrow Y$  e  $g : W \rightarrow X$  funções e suponhamos que  $R \subseteq W$ ,  $S, S' \subseteq X$ , e  $T, T' \subseteq Y$ . Então :

- |  |  |
|--|--|
| i) $T \supset f(f^{-1}(T))$ ;                                  | ix) $f(S \cap S') \subset f(S) \cap f(S')$ .           |
| ii) $T \subset T' \Rightarrow f^{-1}(T) \subset f^{-1}(T')$ .  | x) $f(S \setminus S') \supset f(S) \setminus f(S')$ .  |
| iii) $f^{-1}(T \cup T') = f^{-1}(T) \cup f^{-1}(T')$ .         | xi) $f(S) \cap T = f(S \cap f^{-1}(T))$ .              |
| iv) $f^{-1}(T \cap T') = f^{-1}(T) \cap f^{-1}(T')$ .          | xii) $f(S) \cup T \supset f(S \cup f^{-1}(T))$ .       |
| v) $f^{-1}(T \setminus T') = f^{-1}(T) \setminus f^{-1}(T')$ . | xiii) $S \cap f^{-1}(T) \subset f^{-1}(f(S) \cap T)$ . |
| vi) $S \subset f^{-1}(f(S))$ .                                 | xiv) $S \cup f^{-1}(T) \subset f^{-1}(f(S) \cup T)$ .  |
| vii) $S \subset S' \Rightarrow f(S) \subset f(S')$ .           | xv) $(f \circ g)^{-1}(T) = g^{-1}(f^{-1}(T))$ .        |
| viii) $f(S \cup S') = f(S) \cup f(S')$ .                       | xvi) $(f \circ g)(R) = f(g(R))$ .                      |

## §30.10 Exercícios

### Exercícios Introdutórios

**Exercício 30.1.** Dados os conjuntos  $A = \{n \in \mathbb{Z} \mid n^2 \leq 5\}$ ,  $B = \{2023, 2024, 2025\}$  e  $C = \mathbb{N}$ . Descreva o conjuntos  $A$  e  $B$  de outro modo. Além disso, determine  $A \cup B$ ,  $A - C$  e  $B \cap C$ .

**Exercício 30.2.** Seja  $f : \mathbb{R} - \{5\} \rightarrow \mathbb{R}$  dada por  $f(x) = \frac{2x+3}{x-5}$ . Qual elemento do domínio de  $f$  tal que sua imagem é 2023 ?

**Exercício 30.3.** Sejam  $f, g : \mathbb{R} \rightarrow \mathbb{R}$  funções tais que  $g(x) = 2x - 3$  e  $(f \circ g)(x) = 2x^2 - 4x + 1$ . Encontre a expressão que define a função  $f$ .

**Exercício 30.4.** Seja  $A = \mathbb{R} - \{2\}$  e considere  $f(x) = \frac{3x}{x-2}$ . Determine um conjunto  $B \subset \mathbb{R}$  para que  $f : A \rightarrow B$  seja uma bijeção. Para esse tal conjunto  $B$ , determine a inversa de  $f$ .

### Exercícios de Aprofundamento

**Exercício 30.5.** Encontre dois conjuntos  $X$  e  $Y$  tais que  $(X \times Y) \cap (Y \times X)$  possua exatamente 2 elementos. É possível que nesta interseção existam 3 elementos ?

**Exercício 30.6.** Mostre que:

- a)  $f : \mathbb{R} \rightarrow \mathbb{R}$  onde  $f(x) = x^2$  não é injetiva, sobrejetiva ou bijetiva;
- b)  $g : \mathbb{R}_+ \rightarrow \mathbb{R}$  onde  $g(x) = x^2$  é injetiva, mas não é sobrejetiva ou bijetiva;
- c)  $h : \mathbb{R} \rightarrow \mathbb{R}_+$  onde  $h(x) = x^2$  é sobrejetiva, mas não é injetiva ou bijetiva;
- d)  $p : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  onde  $p(x) = x^2$  é injetiva, sobrejetiva e bijetiva.

**Exercício 30.7.** Sejam  $A$  e  $B$  conjuntos tais que  $|A| = |B|$ . Mostre que  $f : A \rightarrow B$  é injetora se, e somente se, é sobrejetora.

### Exercícios Avançados

**Exercício 30.8.** Considere a função  $f : \mathbb{N} \rightarrow \mathbb{R}$  onde  $f(1) = 1$  e  $f(n+1) = \frac{f(n)}{1+2f(n)}$  para todo  $n \geq 1$ . Calcule  $f(n)$  para todo  $n \in \mathbb{N}$ .

**Exercício 30.9.** Seja  $f : \mathbb{Q} \rightarrow \mathbb{Q}$  uma função tal que  $f(x+y) = f(x) + f(y)$  para todos  $x, y \in \mathbb{Q}$ . Prove os seguintes itens:

- a)  $f(0) = 0$  e  $f(-x) = -f(x)$  para todo  $x \in \mathbb{Q}$ ;
- b)  $f(x-y) = f(x) - f(y)$  para todos  $x, y \in \mathbb{Q}$ ;
- c)  $f(kx) = kf(x)$  para todo  $x \in \mathbb{Q}$  e  $k \in \mathbb{Z}$ ;
- d)  $f(1/n) = f(1)/n$  para todo  $n \in \mathbb{Z}^*$ ;
- e)  $f(m/n) = m/n \cdot f(1)$  para todos  $m, n \in \mathbb{Z}, n \neq 0$ .

**Exercício 30.10.** Dê um exemplo de uma função sobrejetora  $f : \mathbb{N} \rightarrow \mathbb{N}$  tal que para todo  $n \in \mathbb{N}$  o conjunto  $\{k \in \mathbb{N} \mid f(k) = n\}$  seja infinito.

# 31

## Funções Reais e Funções Elementares

O gráfico mostra como a função varia em relação aos diferentes valores de entrada. Veremos que o gráfico de uma função pode ter diferentes formas, como uma linha reta, uma curva suave, ou até mesmo uma série de pontos desconectados.

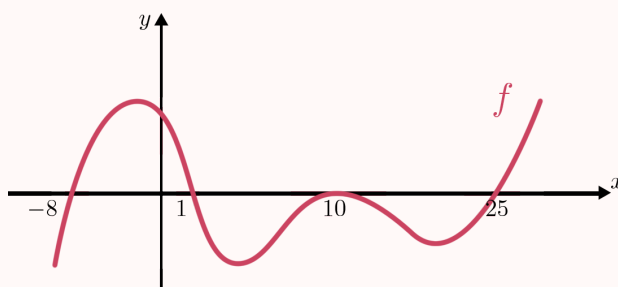
Estudar o gráfico de funções é importante para compreender as relações entre variáveis, então este estudo é importante se queremos resolver equações, desigualdades ou simplesmente comunicar informações de forma clara mostrando padrões, tendências e relações complexas de uma maneira que seja fácil de entender.

Portanto, o gráfico pode ser usado para visualizar a natureza da função, incluindo seu domínio, sua imagem e suas propriedades, como simetria, períodos e descontinuidades. Também pode ser usado para encontrar interseções, máximos e mínimos locais e globais, e para traçar gráficos de funções inversas e compostas.

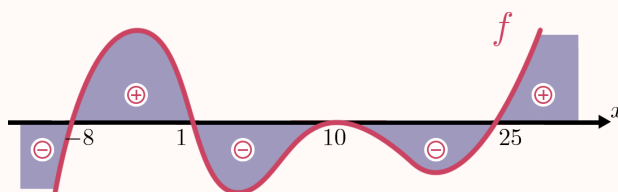
### §31.1 Sinais de uma função

Dada uma função  $f : A \rightarrow B$  vamos estudar o problema de encontrar os valores de  $x \in A$  tal que  $f(x) > 0$  ou  $f(x) < 0$ . No caso de funções reais, ou seja, funções da forma  $f : X \rightarrow \mathbb{R}$ , onde  $X \subset \mathbb{R}$ , podemos usar o gráfico de  $f$  para determinar onde as imagem tomam valores positivos ou negativo.

**Exemplo 31.1.1.** Vamos considerar uma função real que possui o seguinte gráfico



Então podemos representar as regiões positivas e negativas do gráfico como a seguir



ou de forma mais simplificada ainda



Desse modo, temos que a função considerada é positiva em  $(-8, 1) \cup (25, +\infty)$  e negativa em  $(-\infty, -8) \cup (1, 10) \cup (10, 25)$ .

**Exemplo 31.1.2.** Para considerarmos um caso mais complexo, tomemos as funções  $f(x) = x - 2$ ,  $g(x) = -3x + 1$  e  $h(x) = 7x + 4$ . Vejamos que

$$f(x) > 0 \iff x - 2 > 0 \iff x > 2 \quad \text{e} \quad f(x) < 0 \iff x - 2 < 0 \iff x < 2.$$

Podemos repetir o processo para as funções  $g$  e  $h$ , onde encontramos que  $g(x)$  é positiva para  $x < 1/3$  e negativa para  $x > 1/3$  e  $h(x)$  é positiva para  $x > -4/7$  e negativa para  $x < -4/7$ . Então para a função  $F(x) = (x - 2)(-3x + 1)(7x + 4) = -21x^3 + 37x^2 + 14x - 8$  temos o seguinte diagrama

		$-4/7$		$1/3$		$2$	
$f$	$\ominus$		$\ominus$		$\ominus$		$\oplus$
$g$	$\oplus$		$\oplus$		$\ominus$		$\ominus$
$h$	$\ominus$		$\oplus$		$\oplus$		$\oplus$
$F$	$\oplus$		$\ominus$		$\oplus$		$\ominus$

Assim, podemos afirmar que  $-21x^3 + 37x^2 + 14x - 8$  é negativa quando  $x \in (-4/7, 1/3) \cup (2, +\infty)$  e positiva quando  $x \in (-\infty, -4/7) \cup (1/3, 2)$ .

**Observação 31.1.I.** Apesar de ser uma regra simples, queremos dar destaque a seguinte propriedade: Vale  $-(-x) = x$  para todo  $x \in \mathbb{R}$ .

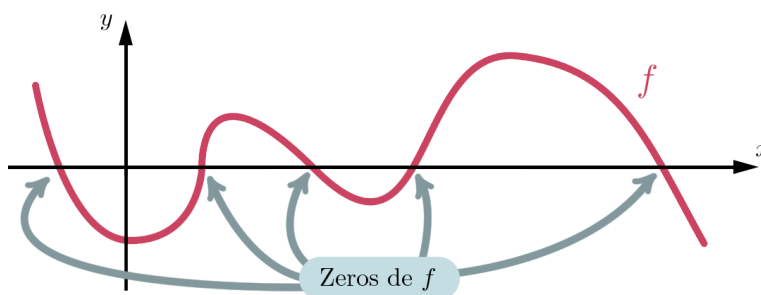
*Demonstração.* Considere  $x \in \mathbb{R}$ , então devemos ter que  $-(-x) + (-x) = 0$ . Somando  $x$  em ambos os lados e observando que  $(-x) + x = 0$ , nos vem que

$$\begin{aligned} -(-x) + (-x) = 0 &\iff -(-x) + (-x) + x = 0 + x \\ &\iff -(-x) + (-x + x) = x \\ &\iff -(-x) + 0 = x \\ &\iff -(-x) = x. \end{aligned}$$

Portanto, temos  $-(-x) = x$  para qualquer  $x \in \mathbb{R}$ . □

## §31.2 Zeros de uma função

Já que expomos ideias para estudar o sinal de uma função  $f : X \subset \mathbb{R} \rightarrow \mathbb{R}$ , nos resta estudarmos os **zeros de funções reais**, isto é, os números  $x \in \mathbb{R}$  tais que  $f(x) = 0$ . Pensando no gráfico da função, os seus zeros, se existirem, determinam onde em quais pontos a função cruza o eixo horizontal (também chamado de eixo  $x$ ) e também podem ajudar a entender o comportamento geral da função.



**Exemplo 31.2.1.** Vamos mostrar alguns casos onde os zeros podemos calcular zeros de funções elementares

a) Considere a função  $f(x) : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = ax + b$ , onde  $a$ . O zero de  $f$  existe e é único. Podemos determiná-lo do seguinte modo  $f(x) = 0 \iff ax + b = 0$ , daí  $x = -\frac{b}{a}$ .

b) Usando o método de *completamento de quadrados* podemos encontrar os zeros da função  $g(x) = ax^2 + bx + c$ , onde  $a \neq 0$ . De fato, podemos reescrever  $ax^2 + bx + c = 0$  como  $ax^2 + bx = -c$ . Pondo  $a$  em evidência e somando e retirando o termo  $\frac{b^2}{4a^2}$  nos vem que

$$\begin{aligned} ax^2 + bx + c = 0 &\iff a \left( x^2 + \frac{b}{a}x + \frac{b^2}{4a^2} - \frac{b^2}{4a^2} \right) = -c \iff \left( x + \frac{b}{2a} \right)^2 - \frac{b^2}{4a^2} = -\frac{c}{a} \\ &\iff \left( x + \frac{b}{2a} \right)^2 = \frac{b^2 - 4ac}{4a^2} \iff x + \frac{b}{2a} = \pm \frac{\sqrt{b^2 - 4ac}}{2a} \\ &\iff x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \end{aligned}$$

c) Já para a função  $h(x) = x^3 + ax^2 + bx + c$  o trabalho de procurar zeros agora se torna bem mais complicado. O processo para tal se inicia definindo  $y = \sqrt[3]{x_1} + \sqrt[3]{x_2}$ , onde  $x_1$  e  $x_2$  são soluções da equação  $x^2 - (x_1 + x_2)x + x_1x_2 = 0$ . Por simplicidade, definiremos  $S = x_1 + x_2$  e  $P = x_1x_2$ . Elevando  $y$  ao cubo nós obtemos,

$$\begin{aligned} y^3 &= (\sqrt[3]{x_1} + \sqrt[3]{x_2})^3 = x_1 + 3\sqrt[3]{x_1^2x_2} + 3\sqrt[3]{x_1x_2^2} + x_2 \\ &= x_1 + x_2 + 3\sqrt[3]{x_1x_2}(\sqrt[3]{x_1} + \sqrt[3]{x_2}) = S + 3\sqrt[3]{P}y \end{aligned}$$

Ou seja, sabemos encontrar os zeros de funções  $\bar{h}(y) = y^3 - 3\sqrt[3]{P}y - S$  e observe que essa expressão falta o termo quadrático que aparece na função  $h$  que estamos considerando. Assim, para acabarmos com o termo quadrático de  $h$  faremos a substituição  $x = y - a/3$  em  $h(x)$  e assim a equação que nos resta é

$$\begin{aligned} h(y - a/3) &= \left( y - \frac{a}{3} \right)^3 + a \left( y - \frac{a}{3} \right)^2 + b \left( y - \frac{a}{3} \right) + c \\ &= y^3 + \left( b - \frac{a^2}{3} \right) y + \left( \frac{2a^3}{27} - \frac{ab}{3} + c \right) = y^3 + py + q, \end{aligned}$$

onde  $p = b - \frac{a^2}{3}$  e  $q = \frac{2a^3}{27} - \frac{ab}{3} + c$ . Comparando os termos de  $h(y - a/3)$  com os de  $\bar{h}(y)$  obtemos as relações  $p = -3\sqrt[3]{P}$  e  $q = -S$ , ou seja,  $P = -p/27$  e  $S = -q$ . Vamos agora retornar para as expressões  $S = x_1 + x_2$  e  $P = x_1x_2$  vemos  $x_2 = -(x_1 + p/27)$  e substituindo isso em  $P$  nós ganhamos pelo item anterior

$$-x_1(x_1 + q) = -\frac{p^3}{27} \Rightarrow x_1^2 + qx_1 - \frac{p^3}{27} = 0 \Rightarrow x_1 = \frac{-q + \sqrt{q^2 + 4\frac{p^3}{27}}}{2} = -\frac{q}{2} + \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}$$

logo,

$$x_2 = -1 - x_1 = -\frac{q}{2} - \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}$$

Voltando para a expressão  $x + a/3 = y = \sqrt[3]{x_1} + \sqrt[3]{x_2}$  obtemos uma fórmula para os zeros de  $h(x) = x^3 + ax^2 + bx + c$  que pode ser escrita como

$$x = -\frac{a}{3} + \sqrt[3]{-\frac{q}{2} + \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}}$$

**Observação 31.2.I.** Toda a ideia do método exposto no item c) do Exemplo 31.2.1 é atribuída aos matemáticos *Scipione Del Ferro* (1465-1526), *Girolamo Cardano* (1501-1576) e *Tartaglia* (1500-1557) que, de certo modo, se envolveram numa batalha matemática que rendeu na publicação do tratado de álgebra de Cardano intitulado de *Ars Magna*. Em resumo, por volta de 1515, Scipione conseguiu encontrar uma expressão para as soluções de  $x^3 + bx + c = 0$  mas não publicou sua descoberta. Em torno de 1535, Tartaglia divulgou que tinha desenvolvido obter uma soluções de  $x^3 + ax^2 + c = 0$  e Cardano foi capaz de reduzir a equação geral ao caso de Tartaglia. Também podemos citar que a demonstração feita no item c) é inspirada no artigo “Uma solução das equações do 3º e do 4º graus” escrito por *Carlos Gustavo T. de A. Moreira* na Revista Professor de Matemática.

**Observação 31.2.II.** Para a função  $f(x) = x^3 + px + q$  podemos definir o discriminante  $\delta = 27q^2 + 4p^2$ . Este é dispositivo é capaz de determinar a quantidade de zeros que  $f$  possui: Se  $\delta > 0$ , então  $f$  possui apenas um único zero real; Se  $\delta = 0$ , então  $f$  possui dois zeros reais; Se  $\delta < 0$ , então  $f$  possui 3 zeros reais. A demonstração deste resultado não é tão difícil mas foge do escopo do texto.

**Exemplo 31.2.2.** Existem funções que não possuem zeros e vamos expor alguns exemplos.

- Por exemplo a função  $f : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = 1$  para todo  $x \in \mathbb{R}$  não possui zeros.
- Outra função que é mais interessante seria  $g : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $g(x) = x^2 + 1$ . Para explicar o motivação para considerar esta função é o fato que  $x^2 \geq 0$  para todo  $x \in \mathbb{R}$ , logo

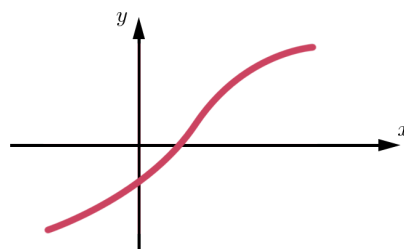
$$g(x) = x^2 + 1 \geq 0 + 1 = 1 > 0.$$

- Há casos que podem se parecer com o item a) do Exemplo 31.2.1, como a função  $h : \mathbb{Z} \rightarrow \mathbb{R}$  dada por  $h(x) = 2x - 1$ , mas  $h$  não admite zeros. De fato, se existisse teríamos que o zero de  $h$  seria dado por  $x = 1/2$ , mas esse não é um elemento de  $\mathbb{Z}$ .

### §31.3 Função crescente e função decrescente

**Definição 31.1.** Dizemos que  $f : A \rightarrow B$  é uma **função crescente em  $X \subset A$**  se para todo  $x, y \in X$  onde  $x < y$  implica que  $f(x) < f(y)$ . Equivalentemente,  $f$  é crescente em  $X \subset A$  se  $x \neq y$  implique que  $\frac{f(x) - f(y)}{x - y} > 0$  para quaisquer  $x, y \in X$ .

Voltando para a ideia gráfica, dizer que uma função é crescente significa que o gráfico da função sobe quando variamos argumento a partir da esquerda para a direita. Este movimento crescente ocorre sem pontos de inflexão, isto é, a curva nunca começa a descer.



**Exemplo 31.3.1.** Casos simples como as funções identidade  $\text{Id}_A : A \subset \mathbb{R} \rightarrow \mathbb{R}$  dada por  $\text{Id}_A(x) = x$  é uma função crescente. Outros casos não tão imediatos são as seguintes

- a) A função  $f : \mathbb{R}_+ \rightarrow \mathbb{R}$  dada por  $f(x) = x^3 + 2x$  é crescente. De fato, para números reais quaisquer  $0 < a < b$ , temos

$$\begin{aligned} f(b) - f(a) &= (b^3 + 2b) - (a^3 + 2a) = (b^3 - a^3) + 2(b - a) \\ &= (a - b)(a^2 + ab + b^2) + 2(b - a) = (b - a)(b^2 + ab + a^2 + 2) \end{aligned}$$

Nos resta mostrar que  $a^2 + ab + b^2 + 2 > 0$ . Mas usando a desigualdade entre as médias  $\sqrt{xy} \leq (x + y)/2$ , para  $x, y > 0$ , temos

$$a^2 + b^2 + ab + 2 \geq 2ab + ab + 2 = 3ab + 2 > 0$$

- b) A função  $g : \mathbb{R}_+ \rightarrow \mathbb{R}$  dada por  $g(x) = x/(1 + x)$  é crescente. De fato, para  $0 < a < b$  temos

$$\begin{aligned} a < b &\iff a + ab < b + ab \iff a(1 + b) < b(1 + a) \\ &\iff f(a) = \frac{a}{1 + a} < \frac{b}{1 + b} = f(b) \end{aligned}$$

- c) Se  $a > 0$ , então a função  $h(x) = ax^2 + bx + c$  é crescente no conjunto  $[-b/2a, +\infty)$ . De fato, se  $y > x \geq -b/2a$ , temos  $x + y \geq -\frac{b}{a}$  daí segue

$$\begin{aligned} h(y) - h(x) &= (ay^2 + by + c) - (ax^2 + bx + c) = a(y^2 - x^2) + b(y - x) \\ &= a(y - x)(y + x) + b(y - x) = (y - x)[a(y + x) + b] \\ &= a(y - x) \left( x + y + \frac{b}{a} \right) \geq 0 \end{aligned}$$

Portanto,  $h(x) = ax^2 + bx + c$ , onde  $a > 0$ , é crescente em  $[-b/2a, +\infty)$ .

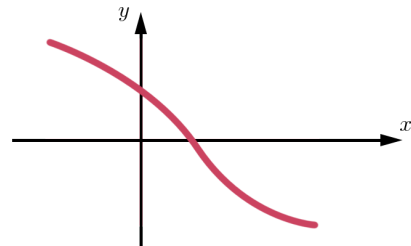
No item b) mostramos que a função  $g : \mathbb{R}_+ \rightarrow \mathbb{R}$  dada por  $g(x) = x/(1 + x)$  é crescente, então temos para  $0 < a < b + c$  que  $g(a) < g(b + c)$ , ou seja,

$$\frac{a}{1 + a} < \frac{b + c}{1 + b + c} = \frac{b}{1 + b + c} + \frac{c}{1 + b + c} \leq \frac{b}{1 + b} + \frac{c}{1 + c}$$

Portanto, podemos usar funções crescentes para demonstrar certas desigualdades.

**Definição 31.2.** Dizemos que  $f : A \rightarrow B$  é uma **função decrescente em  $X \subset A$**  se para todo  $x, y \in X$  onde  $x < y$  implica que  $f(x) > f(y)$ . Equivalentemente,  $f$  é decrescente em  $X \subset A$  se  $x \neq y$  implique que  $\frac{f(x) - f(y)}{x - y} < 0$  para quaisquer  $x, y \in X$ .

Voltando para a ideia gráfica, dizer que uma função é decrescente significa que o gráfico da função desce quando variamos o argumento a partir da esquerda para a direita. Este movimento decrescente ocorre sem pontos de inflexão, isto é, a curva nunca começa a subir.





**Exemplo 31.3.2.** Seja  $f : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = ax^2 + bx + c$ , onde  $a > 0$ . A função  $h(x) = ax^2 + bx + c$  é decrescente no conjunto  $(-\infty, -b/2a]$ . De fato, se  $x < y \leq -b/2a$ , temos  $x + y \leq -\frac{b}{a}$  daí segue

$$\begin{aligned} f(y) - f(x) &= (ay^2 + by + c) - (ax^2 + bx + c) = a(y^2 - x^2) + b(y - x) \\ &= a(y - x)(y + x) + b(y - x) = (y - x)[a(y + x) + b] \\ &= a(y - x) \left( x + y + \frac{b}{a} \right) \leq 0 \end{aligned}$$

Portanto,  $f(x) = ax^2 + bx + c$ , onde  $a > 0$ , é decrescente em  $(-\infty, -b/2a]$ .

**Exemplo 31.3.3.** Mostraremos que a função  $a : \mathbb{N} \rightarrow \mathbb{R}$  dada por  $a(n) = \sqrt[n]{n!}$  é crescente e depois provaremos que  $b(n) = \frac{a(n+1) - a(n)}{a(n)} = \frac{a(n+1)}{a(n)} - 1$  é decrescente. Por simplicidade denotaremos  $a_n$  e  $b_n$  para as funções anteriores. Note que  $a_{n+1}^{n+1} = (n+1)! = (n+1)a_n^n$ , logo  $\left(\frac{a_{n+1}}{a_n}\right)^n = \frac{n+1}{a_{n+1}}$ . Ora, cada um dos termos envolvidos no produto  $(n+1)!$  são menores que  $n+1$ , logo  $a_{n+1}^{n+1} = (n+1)! < (n+1)^{n+1}$  o que implica que  $a_{n+1} < n+1$ . Daí concluímos

$$\left(\frac{a_{n+1}}{a_n}\right)^n = \frac{n+1}{a_{n+1}} > 1 \iff a_{n+1}^n > a_n^n \iff a_{n+1} > a_n$$

o que conclui a demonstração da primeira parte da nossa afirmação. Para a segunda parte, devemos mostrar que  $b_{n+1} < b_n$ , ou seja,  $\frac{a_{n+2}}{a_{n+1}} < \frac{a_{n+1}}{a_n}$  vale para todo  $n \in \mathbb{N}$ , ou equivalentemente,  $a_{n+2}a_n < a_{n+1}^2$  e elevando a  $n+2$  nós obtemos  $a_{n+2}^{n+2}a_n^{n+2} < a_{n+1}^{2n+4}$ . Pela definição de  $a_n$  temos

$$\begin{aligned} a_{n+2}^{n+2}a_n^{n+2} < a_{n+1}^{2n+4} &\iff (n+2)!n!a_n^2 < [(n+1)!]^2a_{n+1}^2 \\ &\iff \left(\frac{a_n}{a_{n+1}}\right)^2 < \frac{[(n+1)!]^2}{n!(n+2)!} = \frac{n+1}{n+2}. \end{aligned}$$

Iremos provar a última desigualdade por indução em  $n$ . Para  $n = 1$  temos  $(a_1/a_2)^2 = 1/2 < 2/3$ . Suponha que vale para algum  $k \in \mathbb{N}$ , então para  $k+1$  temos

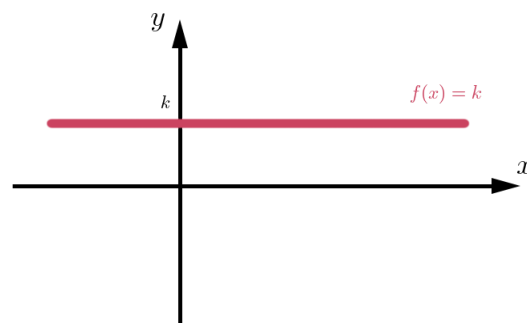
$$\begin{aligned} \left[\left(\frac{a_{k+1}}{a_{k+2}}\right)^2\right]^{k+2} &= \left[\frac{(k+1)!a_{k+1}}{(k+2)!}\right]^2 = \left[\frac{(k+1)!a_{k+1}}{(k+2)!}\right]^2 = \left[\frac{(k+1)k!a_{k+1}}{(k+2)(k+1)!}\right]^2 \\ &= \left[\frac{(k+1)a_k^k a_{k+1}}{(k+2)a_{k+1}^{k+1}}\right]^2 = \left(\frac{a_k}{a_{k+1}}\right)^k \left(\frac{k+1}{k+2}\right)^2 < \left(\frac{k+1}{k+2}\right)^{k+2} < \left(\frac{k+2}{k+3}\right)^{k+2} \end{aligned}$$

Tomando raízes completamos a nossa prova por indução. Portanto  $b_n$  é decrescente.

## §31.4 Função constante

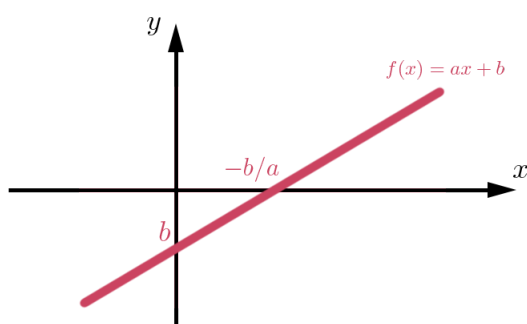
Uma **função constante** é uma função matemática que sempre retorna o mesmo valor, independentemente do valor de entrada. Matematicamente, dizemos que a função  $f : A \subset \mathbb{R} \rightarrow \mathbb{R}$  é constante se para todo  $x$  real temos  $f(x) = k$  para alguma constante  $k \in \mathbb{R}$ .

No Exemplo 31.2.2 comentamos que a função  $f(x) = 1$  não possui zeros e este argumento pode ser generalizado para qualquer função constante  $g : X \subset \mathbb{R} \rightarrow \mathbb{R}$  dada por  $g(x) = k$ , onde  $k \neq 0$ . Para o caso  $k = 0$ , dizemos que a  $g$  é a **função nula** em  $X$ . Além disso, a função constante possui sinal constante e depende do valor de  $k$ . Note ainda que função constante não é injetiva ou sobrejetiva. E o gráfico da função constante é uma reta paralela ao eixo- $x$ .



### §31.5 Função afim

**Definição 31.3.** Uma função  $f : \mathbb{R} \rightarrow \mathbb{R}$  recebe o nome de **função afim** quando é dada por  $f(x) = ax + b$ , onde  $a \neq 0$ . Para o caso  $b = 0$ , chamamos a função  $g(x) = ax$  de **função linear**. É comum chamarmos a constante  $a$  de **coeficiente angular** e  $b$  de **coeficiente linear**.



Vimos no Exemplo 31.2.1 zeros da função afim  $f(x) = ax + b$ , é dado por  $x = -\frac{b}{a}$ . Além disso, já comentamos também em exemplos anteriores que uma função afim é injetiva e sobrejetiva, logo, uma bijeção. Relembre também que o gráfico de uma função afim é uma reta que não é paralela ao eixo- $x$ .

**Proposição 31.5.1** – Seja  $f : \mathbb{R} \rightarrow \mathbb{R}$  uma função afim, dada por  $f(x) = ax + b$ . Então :

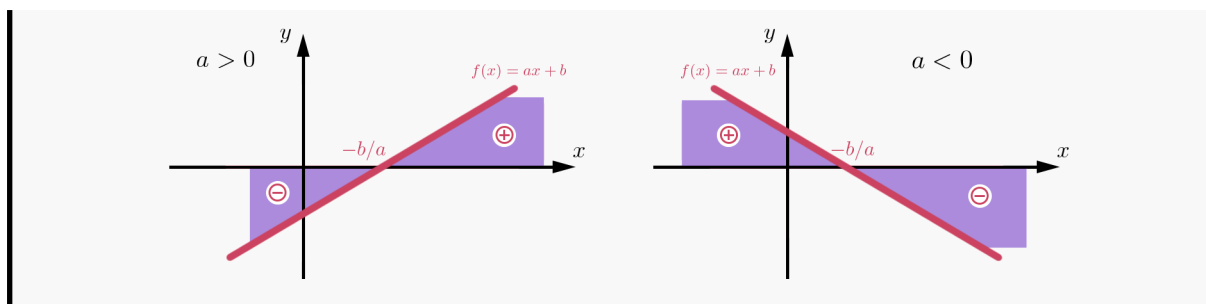
- a) A função  $f$  é crescente se, e somente se,  $a > 0$ .
- b) A função  $f$  é decrescente se, e somente se,  $a < 0$ .

*Demonstração.* Para o item a), vejamos que

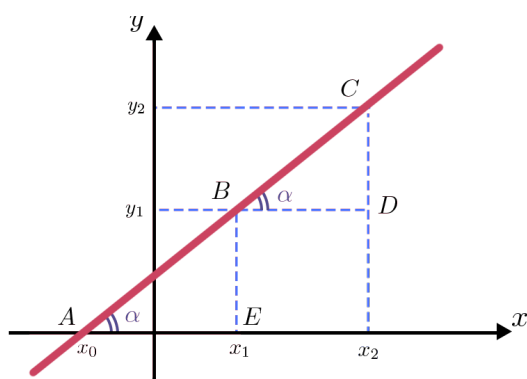
$$\begin{aligned} f(x) = ax + b \text{ é crescente} &\iff \frac{f(y) - f(x)}{y - x} > 0, \forall x \neq y \\ &\iff \frac{(ay + b) - (ax + b)}{y - x} > 0, \forall x \neq y; \\ &\iff \frac{a(y - x)}{y - x} = a > 0, \forall x \neq y. \end{aligned}$$

O item b) é tratado de forma semelhante. □

**Observação 31.5.I.** Note que a proposição anterior nos diz que a condição de crescimento, ou decrescimento, da função afim não depende de um conjunto específico ou outra propriedade sobre a função. Desse modo, sabendo que o único zero de uma função afim  $f(x) = ax + b$  ocorre em  $-b/a$ , podemos facilmente determinar o sinal desta função. Por exemplo, se  $f$  é crescente então para qualquer  $x > -b/a$  temos  $f(x) > f(-b/a) = 0$  e se  $x < -b/a$  temos  $f(x) < f(-b/a) = 0$ .

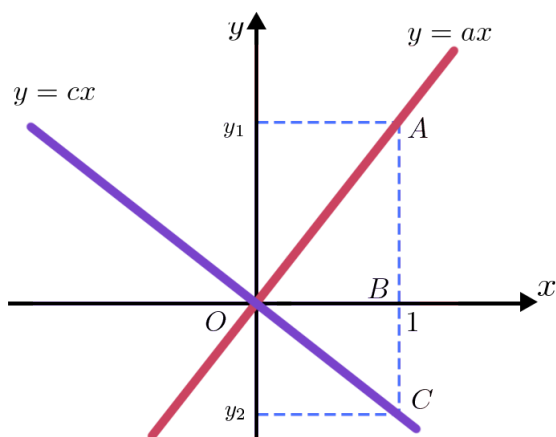


Vamos explicar o motivo para o coeficiente da variável  $x$  da função afim ser chamado de coeficiente angular. Relembre que a razão dos catetos de um triângulo retângulo é chamada de tangente e este está relacionado com ângulo que é feito entre a hipotenusa e um dos catetos.



Pelo [Teorema 30.1](#), sabemos que o gráfico da função afim  $f(x) = ax + b$  é uma reta, logo, na figura ao lado, os triângulos  $\triangle ABE$  e  $\triangle BCD$  são retângulos e semelhantes. Assim, como os pontos  $A, B$  e  $C$  são pontos do gráfico de  $f$ , temos  $y_1 = ax_1 + b$  e  $y_2 = ax_2 + b$ . Daí, a razão dos catetos do triângulo  $\triangle BCD$  é  $\frac{y_2 - y_1}{x_2 - x_1} = a$  e pelo comentário anterior concluímos  $\tan \alpha = a$ . Portanto, podemos considerar a equação o gráfico de  $f$  faz um ângulo constante  $\alpha$  com as paralelas do eixo- $x$ .

Explorando este resultado com um pouco mais de trigonometria, podemos por exemplo considerar a reta que é perpendicular ao gráfico de  $y = ax + b$ . Assuma que a equação da reta procurada é da forma  $y = cx + d$ .



modo que a interseção entre as retas seja o ponto  $O(0,0)$ , assim as equações das retas são  $y = ax$  e  $y = cx$ . Considerando os pontos  $A(1, y_1)$ ,  $B(1, 0)$  e  $C(1, y_2)$ , onde  $y_1 = a$  e  $y_2 = c$ . Por Pitágoras, temos que  $\overline{OA} = \sqrt{1 + a^2}$ ,  $\overline{OC} = \sqrt{1 + c^2}$  e  $\overline{AC} = \sqrt{(1-1)^2 + (a-c)^2} = a - c$ . Já no triângulo  $\triangle OAC$ , temos  $\overline{OA}^2 + \overline{OC}^2 = \overline{AC}^2$

$$\begin{aligned} \left(\sqrt{1+a^2}\right)^2 + \left(\sqrt{1+c^2}\right)^2 &= (a-c)^2 \\ (1+a^2) + (1+c^2) &= a^2 - 2ac + c^2 \end{aligned}$$

Por simplicidade, transladamos a figura de

$$2 = -2ac \iff ac = -1 \iff c = -1/a$$

Reciprocamente, se  $c = -1/a$ , então vale a relação de pitágoras no triângulo  $\triangle OAC$  então este é retângulo em  $O$ , ou seja, as retas são perpendiculares. Portanto, acabamos de demonstrar a seguinte proposição

**Proposição 31.5.2** – Para que as retas  $y = ax + b$  e  $y = cx + d$  sejam perpendiculares é necessário e suficiente que tenhamos  $ac = -1$ .

## §31.6 Exercícios

### Exercícios Introdutórios

**Exercício 31.1.** Se  $\alpha$  e  $\beta$  são os zeros da função  $f(x) = x^2 + x - 1$ , encontre uma função da forma  $g(x) = ax^2 + bx + c$  onde os zeros são exatamente  $\alpha^3$  e  $\beta^3$ .

**Exercício 31.2.** Determine os valores de  $m \in \mathbb{R}$  tais que  $f(x) = (m^2 + 1 - 10/m)x + 2023$  é crescente.

**Exercício 31.3.** Encontre os zeros da função  $f : \mathbb{R}_+ \rightarrow \mathbb{R}$  dada por  $f(x) = \sqrt{2 + \sqrt{x}} - 8/x$ .

**Exercício 31.4.** Sejam  $\alpha$  e  $\beta$  reais os zeros da função  $f(x) = x^2 - 13x + 9$  e  $\alpha^2$  e  $\beta^2$  são os zeros da função  $g(x) = x^2 + ax + b$ , onde  $a, b \in \mathbb{R}$ . Calcule  $a + b$ .

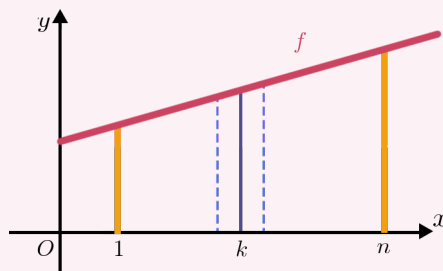
**Exercício 31.5.** Sejam  $a, b \in \mathbb{R}$  quaisquer e considere o intervalo  $[a, b] \subset \mathbb{R}$ . Mostre que se  $f : [a, b] \rightarrow \mathbb{R}$  é uma função crescente, então  $f$  é injetiva

### Exercícios de Aprofundamento

**Exercício 31.6.** Definimos uma *Progressão Aritmética* (ou simplesmente P.A.) de razão  $r \in \mathbb{R}$  como uma sequência  $(a_k)_{k \geq 1}$  de números que satisfazem  $a_{n+1} = a_n + r$  para todo  $n \in \mathbb{N}$ . Para reais dados  $a$  e  $b$ , com  $a \neq 0$ , defina  $f(x) = ax + b$ . Mostre que se  $(a_k)_{k \geq 1}$  é uma P.A. de razão  $r$ , então a sequência  $(b_k)_{k \geq 1}$  onde  $b_k = f(a_k)$  é uma P.A. e encontre sua razão.

**Exercício 31.7.** Os valores de uma sequência  $(a_k)_{k \geq 1}$  é uma P.A. onde  $a_1, a_2, a_3, \dots$  são os valores  $f(1), f(2), f(3), \dots$  onde  $f$  é uma função afim. Mostre que:

- a) Mostre que cada  $a_k$  é igual à área de um trapézio delimitado pelo gráfico de  $f$ , pelo eixo- $x$  e pelas retas verticais de equações  $x = k - 1/2$  e  $x = k + 1/2$ .



- b) Mostre que a soma  $S = a_1 + a_2 + \dots + a_n$  é igual à área do trapézio delimitado pelo gráfico de  $f$ , pelo eixo- $x$  e pelas retas verticais  $x = 1/2$  e  $x = n + 1/2$ .

- c) Conclua que  $S = \frac{(a_1 + a_n)n}{2}$ .

**Exercício 31.8.** Sejam  $(a_k)_{k \geq 1}$  e  $(b_k)_{k \geq 1}$  duas progressões aritméticas. Mostre que existe uma, e somente uma, função afim tal que  $f(a_n) = b_n$  para todo  $n \in \mathbb{N}$ .

### Exercícios Avançados

**Exercício 31.9.** Encontre todas as funções crescentes  $f : \mathbb{N} \rightarrow \mathbb{N}$  tais que tenhamos  $f(n + f(n)) = 2f(n)$  para todo  $n \in \mathbb{N}$ .

**Exercício 31.10.** Seja  $f : \mathbb{R} \rightarrow \mathbb{R}$  uma função crescente. Mostre que as seguintes afirmações são equivalentes:

- (i)  $f(nx) = nf(x)$  para todo  $n \in \mathbb{Z}$  e todo  $x \in \mathbb{R}$ .
- (ii) Pondo  $a = f(1)$ , tem-se  $f(x) = ax$  para todo  $x \in \mathbb{R}$ .
- (iii)  $f(x + y) = f(x) + f(y)$  para quaisquer  $x, y \in \mathbb{R}$ .

**Exercício 31.11.** Prove por indução que  $\left(\frac{n+1}{n}\right)^n \leq n$  para todo  $n \geq 3$  e conclua que a sequência  $1, \sqrt{2}, \sqrt[3]{3}, \sqrt[4]{4}, \dots$  é decrescente a partir do terceiro termo.

## §32.1 Definições e Preliminares

**Definição 32.1.** Uma função  $f : \mathbb{R} \rightarrow \mathbb{R}$  chama-se **função quadrática** quando existem números reais  $a, b$  e  $c$ , com  $a \neq 0$ , tais que  $f(x) = ax^2 + bx + c$  para todo  $x \in \mathbb{R}$ . O número  $a$  é conhecido por coeficiente do termo quadrático,  $b$  é o coeficiente do termo linear e  $c$  é o coeficiente constante. Os zeros de uma função quadrática também são chamados de *raízes*.

Já discutimos os zeros de uma função quadrática na Aula 3, mais precisamente no [Exemplo 31.2.1](#). Neste momento, mergulharemos mais a fundo nesse tópico para explorar outras propriedades que serão de extrema importância para o nosso estudo futuro.

Relembre que a técnica de somar e subtrair certo termo a uma expressão dada a fim de *completar o quadrado* é bastante poderosa. Dados  $a, b, c \in \mathbb{R}$ , com  $a \neq 0$ , consideremos o trinômio  $ax^2 + bx + c$  e notemos que

$$\begin{aligned} ax^2 + bx + c &= a \left( x^2 + \frac{b}{a}x + \frac{c}{a} \right) \\ &= a \left( x^2 + 2 \cdot \frac{b}{2a} \cdot x + \frac{b^2}{4a^2} - \frac{b^2}{4a^2} + \frac{c}{a} \cdot \frac{4a}{4a} \right) \\ &= a \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{b^2 - 4ac}{4a^2} \right] \end{aligned}$$

Aplicando o método de completar quadrados, acabamos de mostrar outra forma de expressar o trinômio  $ax^2 + bx + c$  e esta forma é conhecida como *forma canônica do trinômio*. É comum ainda chamarmos o termo  $\Delta = b^2 - 4ac$  de *discriminante*, esta é uma quantidade que depende apenas dos coeficientes do trinômio e nos permite deduzir algumas propriedades das raízes sem calculá-las. De fato, a partir da forma canônica imediatamente nós temos a seguinte proposição:

**Proposição 32.1.1** – Sejam  $a, b$  e  $c$  reais, com  $a \neq 0$ . A função quadrática  $f(x) = ax^2 + bx + c$  possui zeros se, e somente se,  $\Delta \geq 0$ . Neste caso, as soluções são dadas por  $x = \frac{-b \pm \sqrt{\Delta}}{2a}$ .

*Demonstração.* Se queremos que  $x \in \mathbb{R}$  seja zero de  $f$ , então pela forma canônica nós temos

$$ax^2 + bx + c = 0 \iff a \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] = 0 \iff \left( x + \frac{b}{2a} \right)^2 = \frac{\Delta}{4a^2} \quad (32.1)$$

$$\iff x + \frac{b}{2a} = \pm \sqrt{\frac{\Delta}{4a^2}} \iff x = \frac{-b \pm \sqrt{\Delta}}{2a} \quad (32.2)$$

Veja que  $(x + b/2a)^2 \geq 0$  para todo  $x$  real, então a passagem da primeira para a segunda linha só faz sentido se tivermos  $\Delta \geq 0$  □

**Observação 32.1.I.** Note que se  $\Delta = 0$ , a função quadrática  $f(x) = ax^2 + bx + c$  ainda possui dois zeros, mas eles não são distintos. Nesse caso, dizemos que a equação  $ax^2 + bx + c = 0$

possui uma raiz dupla.

**Observação 32.1.II.** O caso  $\Delta < 0$  gera problemas pois queremos calcular a raiz quadrada deste número e valores negativos não está no domínio desta função. Isto deu motivação para matemáticos para definir  $i$  como a solução da equação  $x^2 + 1 = 0$  e assim nasce o conjunto  $\mathbb{C}$  dos *números complexos*, onde são os números que são da forma  $a + bi$ , onde  $a, b \in \mathbb{R}$ .

Veja que se  $\Delta \geq 0$ , então a soma e o produto dos zeros de uma função quadrática podem ser expressos em fator de seus coeficientes. Com efeito, para verificar a afirmação sobre a soma basta observar que

$$\frac{-b + \sqrt{\Delta}}{2a} + \frac{-b - \sqrt{\Delta}}{2a} = \frac{(-b + \sqrt{\Delta}) + (-b - \sqrt{\Delta})}{2a} = \frac{-2b}{2a} = -\frac{b}{a}$$

e para o produto dos zeros temos

$$\frac{-b + \sqrt{\Delta}}{2a} \cdot \frac{-b - \sqrt{\Delta}}{2a} = \frac{(-b + \sqrt{\Delta}) \cdot (-b - \sqrt{\Delta})}{4a^2} = \frac{(-b)^2 - (\sqrt{\Delta})^2}{4a^2} = \frac{b^2 - (b^2 - 4ac)}{4a^2} = \frac{c}{a}$$

Essa observação feita com uma linguagem bem mais moderna, na verdade tem sua origem em textos cuneiformes escritos pelos babilônios. O método descrito por eles remonta uma técnica muito boa para encontrar dois números conhecendo sua soma  $S$  e seu produto  $P$ . No nosso contexto, isto é equivalente a encontrar soluções da equação  $x^2 - Sx + P = 0$ . Vamos expor um pouco mais sobre isso no próximo exemplo.

**Exemplo 32.1.2 (Um método babilônico!).** Veja primeiramente que se  $a$  e  $b$  são dois números reais então

$$(x - a)(x - b) = x^2 - (a + b)x + ab$$

ou seja, o problema de encontrar dois números reais conhecendo a sua soma  $S$  e seu produto  $P$  é equivalente a encontrar os zeros de  $f(x) = x^2 - Sx + P$ . Suponha que  $\alpha$  e  $\beta$  são os números procurados e  $\alpha \leq \beta$ . Assim, sua média aritmética é  $S/2 = \frac{\alpha + \beta}{2}$ . Conhecendo a diferença  $d = \beta - S/2 = S/2 - \alpha$ , podemos expressar os números procurados por  $\alpha = S/2 - d$  e  $\beta = S/2 + d$ . Curiosamente, é mais simples encontrar qual o valor de  $d$  pois

$$P = \alpha\beta = \left(\frac{S}{2} - d\right) \left(\frac{S}{2} + d\right) = \left(\frac{S}{2}\right)^2 - d^2.$$

Reorganizando os termos nos vem que

$$d^2 = \left(\frac{S}{2}\right)^2 - P \iff d = \sqrt{\left(\frac{S}{2}\right)^2 - P}$$

e portanto, os números procurados são dados por

$$\alpha = \frac{S}{2} - \sqrt{\left(\frac{S}{2}\right)^2 - P} \quad \text{e} \quad \beta = \frac{S}{2} + \sqrt{\left(\frac{S}{2}\right)^2 - P}$$

**Observação 32.1.III.** Já sabemos que  $f(x) = ax^2 + bx + c$  possui zeros se, e só se, seu discriminante é maior ou igual a zero, então no [Exemplo 32.1.2](#) estamos assumindo que o problema admite solução, ou seja, implicitamente assumimos que o discriminante da função

$f(x) = x^2 - Sx + P$  possui essa propriedade. Desse modo, estamos considerando que

$$(-S)^2 - 4P \geq 0 \iff \left(\frac{S}{2}\right)^2 - P \geq 0.$$

Pelos textos babilônicos é possível notar que eles não se preocuparam com esse fato, o que provavelmente causou algum tipo de mistério quando tentaram encontrar os números cujo a soma e o produto são iguais a 2. Nesses casos eles simplesmente diziam que os números procurados não existem.

**Exemplo 32.1.3.** Observe que tudo o que foi aprendido até agora não se restringe apenas a encontrar zeros de funções quadráticas. Por exemplo, podemos encontrar as soluções de  $x^4 + 5x^2 - 7 = 0$ . Fazendo a substituição  $y = x^2$ , a equação se torna  $y^2 + 5y - 7 = 0$ , onde seu discriminante é  $\Delta = 5^2 - 4(-7) = 53 \geq 0$  e assim podemos aplicar qualquer método para determinarmos que as soluções são  $y = \frac{-5 \pm \sqrt{53}}{2}$ . Note que  $-\frac{5 + \sqrt{53}}{2} < 0$ , logo não faz parte da solução da equação original. Portanto,  $x^2 = \frac{-5 + \sqrt{53}}{2}$ , ou seja,  $x = \pm \sqrt{\frac{-5 + \sqrt{53}}{2}}$ .

Outras substituições também são interessantes. Por exemplo, se queremos encontrar os zeros da função  $f(x) = 2x^4 + 5x^3 + 6x^2 + 5x + 2$ , podemos dividir ambos os lados da equação  $2x^4 + 5x^3 + 6x^2 + 5x + 2 = 0$  por  $x^2$  e assim encontramos

$$2x^4 + 5x^3 + 6x^2 + 5x + 2 = 0 \iff 2\left(x^2 + \frac{1}{x^2}\right) + 5\left(x + \frac{1}{x}\right) + 6 = 0.$$

Fazendo uso agora da substituição  $y = x + 1/x$  temos  $y^2 = x^2 + 1/x^2 + 2$ , logo podemos reescrever a equação como  $2(y^2 - 2) + 5y + 6 = 0$  e novamente podemos prosseguir com todas as ferramentas do nosso estudo para encontrar as soluções.

Como último exemplo, vamos encontrar as soluções reais da equação

$$x^2 + x + 1 = \frac{156}{x^2 + x}.$$

Observe que o termo  $x^2 + x$  aparece duas vezes na equação, então isso nos induz considerar a substituição  $y = x^2 + x$ , donde a expressão se torna

$$x^2 + x + 1 = \frac{156}{x^2 + x} \iff y + 1 = \frac{156}{y} \iff y^2 + y = 156.$$

Note que o discriminante desta equação é  $\Delta = 1^2 - 4(-156) = 625 = 25^2 > 0$ , então os possíveis valores para  $y$  são 13 ou 12. Logo, obtemos duas equações  $x^2 + x = -13$  ou  $x^2 + x = 12$ . Por conta do discriminante, vemos que a primeira equação não possui solução, já na segunda temos que as soluções são  $x = -4$  ou  $x = 3$ .

## §32.2 A imagem da função quadrática

Vamos agora estudar a imagem da função quadrática  $f(x) = ax^2 + bx + c$ . Para realizarmos esse estudo basta encontrarmos  $y \in \mathbb{R}$  tais que  $ax^2 + bx + c = y$ , mas pela discussão anterior, desse modo, queremos que a equação  $ax^2 + bx + (c - y) = 0$  possua solução. Pela [Proposição 32.1.1](#) sabemos que esta condição é equivalente a pedirmos que o discriminante dessa equação seja maior

ou igual a zero. Daí,

$$b^2 - 4a(c - y) \geq 0 \iff b^2 - 4ac + 4ay \geq 0 \iff 4ay \geq -\Delta$$

No caso em que  $a > 0$ , temos  $y \geq -\frac{\Delta}{4a}$  e se  $a < 0$ , temos  $y \leq -\frac{\Delta}{4a}$ . Portanto, demonstramos a seguinte proposição

**Proposição 32.2.1** – Considerando a função quadrática  $f(x) = ax^2 + bx + c$ , então temos :

- i) Se  $a > 0$ , a imagem de  $f$  é o intervalo  $\text{Im}(f) = \left[-\frac{\Delta}{4a}, +\infty\right)$ ;
- ii) Se  $a < 0$ , a imagem de  $f$  é o intervalo  $\text{Im}(f) = \left(-\infty, -\frac{\Delta}{4a}\right]$ .

**Exemplo 32.2.2.** i) A imagem da função  $f(x) = 6x^2 - 5x + 8$  pode ser determinada pela [Proposição 32.2.1](#). De fato, note que temos nesse caso  $a = 6, b = -5$  e  $c = 8$ , então segue que  $\Delta = (-5)^2 - 4 \cdot 6 \cdot 8 = -167$ . Logo, a imagem de  $f$  é o intervalo  $[167/24, +\infty)$ .

ii) Já a função  $g(x) = -2x^2 - 3x + 4$  possui discriminante  $\Delta = (-3)^2 - 4 \cdot (-2) \cdot 4 = 41$ , então pela mesma proposição podemos afirmar que sua imagem é o intervalo  $(-\infty, 41/8]$ .

Podemos ainda nos perguntar quando a função é crescente ou decrescente. Desse modo, para o primeiro caso, vamos procurar os valores para  $x \neq y$  tal que  $\frac{f(y)-f(x)}{y-x} > 0$ . Assim,

$$\begin{aligned} \frac{f(y) - f(x)}{y - x} > 0 &\iff 0 < \frac{(ay^2 + by + c) - (ax^2 + bx + c)}{y - x} = \frac{a(y^2 - x^2) + b(y - x)}{y - x} \\ &\iff 0 < \frac{a(y - x)(y + x) + b(y - x)}{y - x} = a(y + x) + b = a\left(x + y + \frac{b}{a}\right). \end{aligned}$$

Assim, se  $a > 0$  então podemos considerar  $x$  e  $y$  pertencentes ao intervalo  $[-b/2a, +\infty)$ , pois assim teremos  $x + y \geq -\frac{b}{2a} - \frac{b}{2a} = -\frac{b}{a}$ , ou equivalentemente,  $x + y + \frac{b}{a} \geq 0$ . Já se  $a < 0$ , podemos considerar  $x, y \in (-\infty, -b/2a]$ . Uma conta totalmente análoga nos mostra o caso em que  $f$  é decrescente. Podemos, então, resumir esses fatos na seguinte proposição:

**Proposição 32.2.3** – Considerando a função quadrática  $f(x) = ax^2 + bx + c$ , então :

- a) Se  $a > 0$ , então  $f$  é crescente em  $\left[-\frac{b}{2a}, +\infty\right)$  e decrescente em  $\left(-\infty, -\frac{b}{2a}\right]$ ;
- b) Se  $a < 0$ , então  $f$  é crescente em  $\left(-\infty, -\frac{b}{2a}\right]$  e decrescente em  $\left[-\frac{b}{2a}, +\infty\right)$ .

### §32.2.4 Sinais de uma função quadrática

O caso  $\Delta \leq 0$  é bem especial pois assim  $f$  tem sinal constante. De fato, suponha primeiramente que  $\Delta < 0$  e lembre que a forma canônica de  $f$  é dada por  $f(x) = a\left[\left(x + \frac{b}{2a}\right)^2 - \frac{\Delta}{4a^2}\right]$ . Note



que pela nossa hipótese temos que  $-\Delta/4a$  é positivo. Assim, para todo  $x \in \mathbb{R}$

$$a \cdot f(x) = a^2 \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] \implies a \cdot f(x) > 0$$

Já se  $\Delta = 0$ , então pela forma canônica temos para  $x \in \mathbb{R} - \{-b/2a\}$

$$a \cdot f(x) = a^2 \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] = a^2 \left( x + \frac{b}{2a} \right)^2 > 0$$

Isto pode ser resumido na seguinte proposição.

**Proposição 32.2.5** – Se  $f(x) = ax^2 + bx + c$  é tal que seu discriminante é não-positivo, então:

- i) Se  $\Delta < 0$ , então  $a$  e  $f(x)$  possuem o mesmo sinal para todo  $x \in \mathbb{R}$ ;
- ii) Se  $\Delta = 0$ , então  $a$  e  $f(x)$  possuem o mesmo sinal para todo  $x \in \mathbb{R} - \{-b/2a\}$ .

**Exemplo 32.2.6.** i) A função  $f(x) = x^2 - 2x + 4$  é positiva para todo  $x \in \mathbb{R}$ , pois seu discriminante é  $\Delta = 4 - 4 \cdot 4 = -12 < 0$  e o coeficiente do termo quadrático é positivo.

ii) A função  $g(x) = -x^2 + x - 1$  é negativa para todo  $x \in \mathbb{R}$ , pois seu discriminante é  $\Delta = 1^2 - 4(-1)(-1) = -3 < 0$  e o termo quadrático possui coeficiente positivo.

iii) Já a função  $h(x) = -2x^2 + 8x - 8$  possui discriminante  $\Delta = 8^2 - 4(-2)(-8) = 0$  então  $h$  possui uma raiz dupla em  $-b/2a = 2$  e o coeficiente do termo quadrático é negativo, então concluímos que  $h(x) < 0$  para todo  $x \in \mathbb{R} - \{2\}$ .

Nos falta considerarmos o caso  $\Delta > 0$ , onde a função quadrática admite duas raízes distintas, para estudarmos o seu sinal. Primeiramente, note que se  $\Delta \geq 0$ , então pela forma canônica nós temos

$$\begin{aligned} f(x) &= a \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] = a \left[ \left( x + \frac{b}{2a} \right)^2 - \left( \frac{\sqrt{\Delta}}{2a} \right)^2 \right] \\ &= a \left[ \left( x + \frac{b}{2a} + \frac{\sqrt{\Delta}}{2a} \right) \left( x + \frac{b}{2a} - \frac{\sqrt{\Delta}}{2a} \right) \right] = a(x - x_1)(x - x_2) \end{aligned}$$

onde  $x_1 = \frac{-b - \sqrt{\Delta}}{2a}$  e  $x_2 = \frac{-b + \sqrt{\Delta}}{2a}$ . Pela [Proposição 32.1.1](#), sabemos que  $x_1$  e  $x_2$  são os zeros de  $f$ . Assim, toda função quadrática pode ser escrita como produto de funções afim, onde os zeros desses fatores são  $x_1$  e  $x_2$ . Vamos considerar um exemplo para mostrar que tipo de informação podemos extrair deste comentário.

**Exemplo 32.2.7.** Veja que a função  $f(x) = 5x^2 - 15x + 10$  possui discriminante igual a  $\Delta = (-15)^2 - 4 \cdot 5 \cdot 10 = 25 > 0$ , logo essa possui duas raízes distintas. Pela [Proposição 32.1.1](#), sabemos que os zeros são  $x = 1$  e  $x = 2$ . Desse modo, a função pode ser expressa por  $f(x) = 5(x - 1)(x - 2)$ . Assim, podemos analisar o sinal de cada função afim junto do coeficiente do termo quadrático para determinar o sinal de  $f$ .

	1	2	
$x - 1$	$\ominus$	$\oplus$	$\oplus$
$x - 2$	$\ominus$	$\ominus$	$\oplus$
$f$	$\oplus$	$\ominus$	$\oplus$

Portanto,  $f$  é negativa no intervalo  $(1, 2)$  e positiva no seu complementar.

O sinal do coeficiente quadrático tem que ser levado em conta para o estudo do sinal, pois a função  $g(x) = -5x^2 + 15x - 10$  também possui raízes em  $x = 1$  e  $x = 2$ , mas em  $x = 3/2$ , que pertence ao intervalo  $(1, 2)$ , temos que  $g(3/2) = -5(3/2)^2 + 15 \cdot 3/2 - 10 = 5/4 > 0$ .

**Proposição 32.2.8** – Se uma função quadrática  $f(x) = ax^2 + bx + c$  possui discriminante  $\Delta > 0$ , então a função quadrática  $f$  pode ser escrita como  $f(x) = a(x - x_1)(x - x_2)$ , onde  $x_1$  e  $x_2$  são as raízes de  $f$ . Assumindo que  $x_1 < x_2$  podemos afirmar que:

- i) Se  $a > 0$ , então  $f$  é negativa sobre o intervalo aberto  $(x_1, x_2)$  e positiva em seu complementar.
- ii) Se  $a < 0$ , então  $f$  é positiva sobre o intervalo aberto  $(x_1, x_2)$  e negativa em seu complementar.

A [Proposição 32.2.5](#) pode ainda gerar uma aplicação bastante útil para o estudo de desigualdades. Vamos mostrar agora um resultado bastante importante e básico dessa teoria.

**Exemplo 32.2.9 (Desigualdade de Cauchy-Schwarz).** Considere  $n > 1$  e números reais  $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n$  não-negativos. Defina a função  $f(x) = (a_1x - b_1)^2 + (a_2x - b_2)^2 + (a_3x - b_3)^2 + \dots + (a_nx - b_n)^2$ , por simplicidade escreveremos  $f(x) = Ax^2 - 2Bx + C$ , onde  $A = a_1^2 + a_2^2 + \dots + a_n^2$ ,  $B = a_1b_1 + a_2b_2 + \dots + a_nb_n$  e  $C = b_1^2 + b_2^2 + \dots + b_n^2$ . Como  $f(x)$  é a soma de quadrados, então devemos ter  $f(x) \geq 0$  para todo  $x \in \mathbb{R}$ , assim  $f$  possui sinal constante. Por outro lado, como  $A > 0$  temos que seu discriminante  $\Delta = (-2B)^2 - 4AC = 4(B^2 - AC) \leq 0$ , ou seja,  $B^2 \leq AC$ . Substituindo os valores de  $A, B$  e  $C$  obtemos a expressão

$$(a_1b_1 + a_2b_2 + \dots + a_nb_n)^2 \leq (a_1^2 + a_2^2 + \dots + a_n^2) (b_1^2 + b_2^2 + \dots + b_n^2).$$

Esta é conhecida como *Desigualdade de Cauchy-Schwarz*. Deixamos ao leitor analisar quando ocorre a igualdade na Desigualdade de Cauchy-Schwarz.

## §32.3 Gráfico da função quadrática

Depois da discussão da seção anterior sobre a imagem, os intervalos de crescimento e decrescimento, assim como o sinais de uma função quadrática podemos esboçar o gráfico desta família de funções. Vamos juntar mais alguns resultados sobre o seu gráfico que usaremos mais tarde.

Em geometria, o **eixo de simetria** é uma linha que divide uma figura em duas partes simétricas, isto é, como se fossem o objeto e a sua imagem refletida em um espelho. Vejamos que o gráfico de uma função quadrática possui este eixo.

**Proposição 32.3.1** – O gráfico de uma função quadrática possui um eixo de simetria sobre uma reta perpendicular ao eixo- $x$ .

*Demonstração.* Considere a função quadrática  $f(x) = ax^2 + bx + c$ , onde  $a \neq 0$ . Sabemos que sua forma canônica é  $f(x) = a \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right]$ . Assim, para  $x = p - \frac{b}{2a}$  temos

$$\begin{aligned} f\left(p - \frac{b}{2a}\right) &= a \left[ \left( p - \frac{b}{2a} + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] = a \left[ p^2 - \frac{\Delta}{4a^2} \right] = a \left[ (-p)^2 - \frac{\Delta}{4a^2} \right] \\ &= a \left[ \left( -p - \frac{b}{2a} + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right] = f\left(-p - \frac{b}{2a}\right) \end{aligned}$$

Assim, os pontos  $\left(p - \frac{b}{2a}, f\left(p - \frac{b}{2a}\right)\right)$  e  $\left(-p - \frac{b}{2a}, f\left(-p - \frac{b}{2a}\right)\right)$  estão sobre uma reta paralela ao eixo  $-x$  e eles possuem a mesma distância da reta vertical  $x = -b/2a$ . Como  $p \in \mathbb{R}$  é arbitrário, então a reta  $x = -b/2a$  é de fato o eixo de simetria do gráfico de  $f$ .  $\square$

**Definição 32.2.** Dizemos que o número  $y_M \in \text{Im}(f)$  é o valor máximo da função  $y = f(x)$  se, e somente se,  $y_M \geq y$  para qualquer  $y \in \text{Im}(f)$ . O número  $x_M \in \text{Dom}(f)$  tal que  $y_M = f(x_M)$  é chamado ponto de máximo da função.

**Definição 32.3.** Dizemos que o número  $y_m \in \text{Im}(f)$  é o valor mínimo da função  $y = f(x)$  se, e somente se,  $y_m \leq y$  para qualquer  $y \in \text{Im}(f)$ . O número  $x_m \in \text{Dom}(f)$  tal que  $y_m = f(x_m)$  é chamado ponto de mínimo da função.

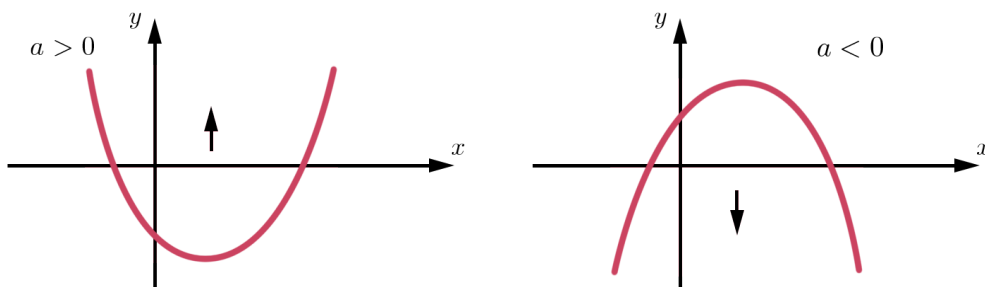
Relembre que para uma função quadrática  $f(x) = ax^2 + bx + c$ , onde  $a \neq 0$  temos  $f(-b/2a) = \Delta/4a$ . Então o ponto  $(-b/2a, f(-b/2a)) = (-b/2a, -\Delta/4a)$  pertence tanto ao eixo de simetria quanto ao gráfico da função quadrática. Portanto, chamamos o ponto

$$V = \left(-\frac{b}{2a}, -\frac{\Delta}{4a}\right)$$

de **vértice do gráfico de  $f$** .

Note que a primeira coordenada de  $V$  está sobre a reta vertical que contém o ponto médio das raízes de  $f$ .

Observe que a maioria dos resultados dependem do sinal do coeficiente do termo quadrático da função  $f(x) = ax^2 + bx + c$ . Quando  $a > 0$ , a **concavidade de  $f$  está voltada para cima**. Já se  $a < 0$ , a **concavidade de  $f$  está voltada para baixo**.

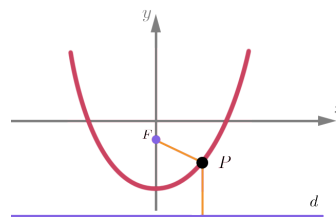


Pela [Proposição 32.2.1](#) vemos que se a função quadrática possui concavidade para cima, então a função possui valor mínimo e se possui concavidade para baixo, a função possui valor máximo.

Veja que a forma canônica  $f(x) = a \left[ \left( x + \frac{b}{2a} \right)^2 - \frac{\Delta}{4a^2} \right]$  nos mostra com facilidade quem é o valor máximo ou mínimo de  $f$ . Suponhamos que  $a > 0$ , então o menor valor de  $f$  ocorre quando o primeiro termo, aquele que depende de  $x$ , é igual a zero, ou seja, quando  $x = -b/2a$ . E a mesma análise pode ser feita quando  $a < 0$ . Deste modo, segue a seguinte proposição

**Proposição 32.3.2** – O valor de máximo ou de mínimo de uma função quadrática  $y = ax^2 + bx + c$  ocorre sobre o vértice de seu gráfico. Ou seja, quando  $x = -b/2a$  e  $y = -\Delta/4a$ .

**Definição 32.4.** Dado um ponto  $F$  e uma reta  $d$  tal que  $F \notin d$  pertencentes ao plano cartesiano  $\mathbb{R}^2$ . Definimos a **parábola** de foco  $F$  e reta diretriz como o conjunto dos pontos que possuem a mesma distância da reta e do ponto  $F$ .



**Proposição 32.3.3** – O gráfico de uma função quadrática é uma parábola.

Assumiremos esse resultado sem mostrar a ideia da prova, mas o leitor cuidadoso não deverá ter muitas dificuldades em demonstrar essa proposição.

Podemos finalmente esboçar o gráfico de  $f(x) = ax^2 + bx + c$ , onde  $a \neq 0$ . Vamos sempre procurar algumas informações para nos guiar de que modo o gráfico de  $f$  está situado no plano, que são:

- 1º) O gráfico de  $f$  é uma parábola, cujo eixo de simetria é a reta  $x = -b/2a$ .
- 2º) Se  $a > 0$ , a parábola possui concavidade virada para cima. Se  $a < 0$ , a parábola possui concavidade virada para baixo.
- 3º) i) Se o discriminante  $\Delta$  for positivo, então os zeros de  $f$  são os pontos

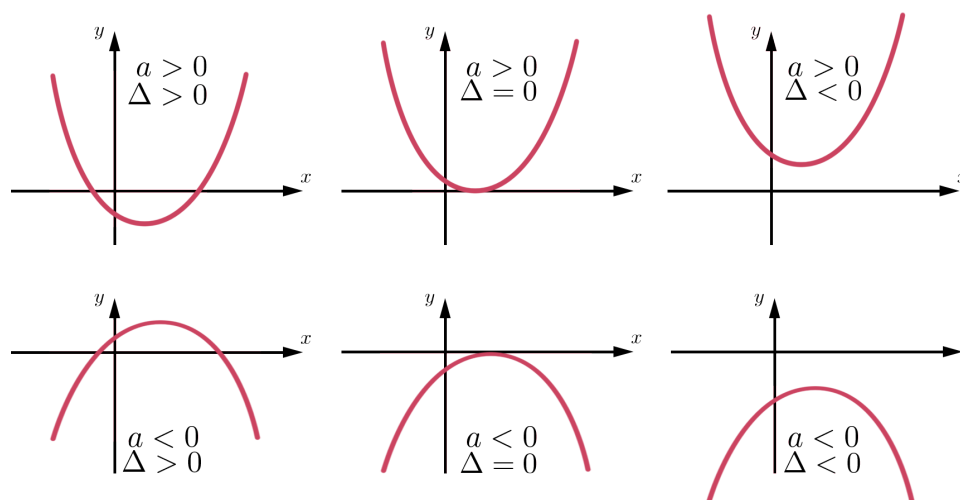
$$P_1 \left( \frac{-b - \sqrt{\Delta}}{2a}, 0 \right) \quad \text{e} \quad P_2 \left( \frac{-b + \sqrt{\Delta}}{2a}, 0 \right)$$

ii) Se  $\Delta = 0$ , então  $f$  encontra o eixo- $x$  apenas no ponto  $P(-b/2a, 0)$ .

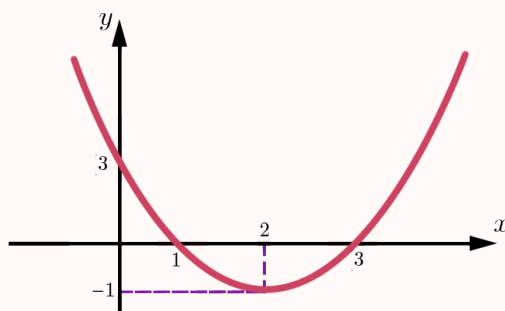
iii) Já se  $\Delta < 0$ , então  $f$  não intersecta o eixo- $x$ .

- 4º) o vértice  $V \left( -\frac{b}{2a}, -\frac{\Delta}{4a^2} \right)$  é o ponto de máximo de  $f$  se  $a < 0$  e de mínimo se  $a > 0$ .

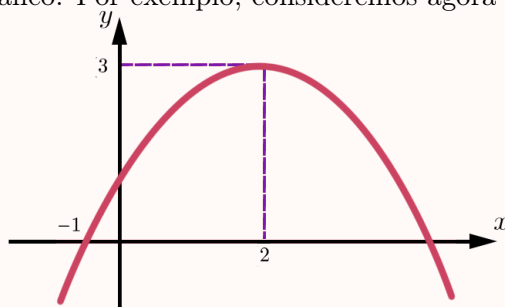
Os únicos tipos de gráficos podem ser os seguintes



**Exemplo 32.3.4.** Façamos o esboço do gráfico da função  $y = x^2 - 4x + 3$ . Vejamos primeiro que esta deverá ser uma parábola com a concavidade virada para cima, pois seu coeficiente quadrático é igual a 1. Calculando o seu discriminante temos  $\Delta = (-4)^2 - 4 \cdot 1 \cdot 3 = 4 > 0$ , então a função dada possui duas raízes distintas e estas são dadas por  $x = \frac{4 - \sqrt{4}}{2} = 1$  e  $x = \frac{4 + \sqrt{4}}{2} = 3$ . O vértice de  $y = x^2 - 4x + 3$  é o ponto cuja as coordenadas são  $x_m = -\frac{-b}{2a} = 2$  e  $y_m = -\frac{\Delta}{4a} = -1$ . Portanto, obtemos o seguinte esboço



**Exemplo 32.3.5.** Podemos também encontrar a função quadrática se temos algumas informações sobre seu gráfico. Por exemplo, consideremos agora o seguinte gráfico



Veja que agora o coeficiente do termo quadrático deve ser negativo, pois sua concavidade está virada para baixo. Além disso, sabemos que as coordenadas do seu vértice são  $x_M = 2$  e  $y_M = 3$ . Daí devemos ter

$$-\frac{b}{2a} = 2 \implies b = 4a \implies b^2 = 16a^2$$

e substituindo isto em  $y_M$  nós obtemos

$$-\frac{b^2 - 4ac}{4a} = 3 \implies b^2 - 4ac = -12a \implies 16a^2 - 4ac = -12a.$$

Dividindo ambos os lados por  $4a$ , nós obtemos  $4a - c = -3$ . Relembre que a soma das raízes  $x_1$  e  $x_2$  é igual a  $-b/a = 4$ , logo como  $x_1 = -1$  temos  $x_1 + x_2 = -b/a \implies -1 + x_2 = 4$ , ou seja,  $x_2 = 5$ . E o produto das raízes é igual a  $c/a$  daí  $x_1 \cdot x_2 = c/a \implies -5 = c/a \implies 5a + c = 0$ . Somando esta com a equação  $4a - c = -3$  nós obtemos  $9a = -3$ , donde temos  $a = -1/3$ . Concluimos assim que  $b = 4/3$  e  $c = 5/3$ , portanto a função que possui o gráfico dado é

$$f(x) = -\frac{1}{3}x^2 - \frac{4}{3}x + \frac{5}{3}.$$

## §32.4 Problemas de Otimização

Um problema de otimização é um tipo de problema matemático que envolve encontrar a melhor solução possível entre várias alternativas possíveis, de acordo com um critério ou objetivo específico. Essa solução é conhecida como “ótima” ou “ideal” e é geralmente determinada por meio da maximização ou minimização de uma função objetivo.

Por exemplo, em um problema de otimização de produção, o objetivo pode ser maximizar o lucro ou minimizar o custo. Em um problema de otimização de transporte, o objetivo pode ser minimizar a distância percorrida ou o tempo de entrega.

Na maioria das vezes, em um problema de otimização, a tarefa é descobrir a mistura perfeita das variáveis disponíveis que nos leve à melhor solução, mas sem quebrar as regras impostas pelo problema. Ou seja, é como fazer um bolo delicioso seguindo a receita, mas ajustando os ingredientes até chegar ao ponto ideal.

Vamos utilizar as ideias das proposições anteriores para determinarmos a solução de problemas em que procuramos o máximo ou o mínimo de certas funções.

**Exemplo 32.4.1.** Suponha que  $y = -t^2 + 5t + 1$  descreve a altura, em metros, de uma pedra lançada após  $t > 0$  segundos. Vamos nos restringir apenas aos 6 primeiros segundos e determinaremos, neste intervalo, o maior ( $y_M$ ) e o menor ( $y_m$ ) valor que  $y$  assume. Note que a altura da pedra é descrita por uma função quadrática de coeficientes  $a = -1$ ,  $b = 5$  e  $c = 1$ , logo seu discriminante é  $\Delta = 5^2 - 4 \cdot (-1) \cdot 1 = 29 > 0$ . Pensando no problema físico, a condição  $\Delta > 0$  nos diz que a função considerada possui duas raízes distintas e estas são exatamente os dois momentos em que a pedra toca o chão. Calculando suas raízes, temos

$$t = \frac{5 + \sqrt{29}}{2} > 0 \quad \text{e} \quad t = \frac{5 - \sqrt{29}}{2} < 0,$$

vamos desconsiderar a segunda raiz pois estamos trabalhando apenas com valores positivos para segundos. Além disso, o seu vértice possui coordenadas  $x_M = -\frac{b}{2a} = \frac{5}{2}$  e  $y_M = -\frac{\Delta}{4a} = \frac{29}{4}$ . Portanto, como  $x_M < 6$ , a maior altura da pedra nos 6 primeiros segundos é  $y_M = 29/4$  metros e a altura mínima da pedra é de 0 metros, quando ela toca o chão.

**Exemplo 32.4.2.** Dentre todos os números reais de soma 8, pergunta-se aqueles cujo produto é máximo. Para responder esse problema chamemos  $A$  e  $B$  para esses números e seu produto denotaremos por  $P$ , logo,  $P = A \cdot B$  e  $A + B = 8$ . Veja que substituindo  $B = 8 - A$  na expressão do produto temos  $P = A(8 - A) = -A^2 + 8A$ . Desse modo, obtemos que  $P$  é uma função quadrática na variável  $A$  com concavidade virada para baixo, e assim,  $P$  possui um valor máximo. Veja ainda que o vértice desta função quadrática possui coordenadas

$$A = -\frac{8}{2 \cdot (-1)} = 4 \quad \text{e} \quad P = -\frac{8^2 - 4 \cdot (-1) \cdot 0}{4 \cdot (-1)} = 16.$$

Portanto, os números que somam 8 e possuem produto máximo são  $A = 4$  e  $B = 8 - A = 4$ .

## §32.5 Exercícios

### Exercícios Introdutórios

**Exercício 32.1.** Determine o valor de  $m \in \mathbb{R}$  na função real  $f(x) = 3x^2 - 2x + m$  para que o valor mínimo seja 5.

**Exercício 32.2.** Determine os valores de  $m \in \mathbb{R}$  tais que  $f(x) = x^2 + mx + 5$  possua raízes inteiras.

**Exercício 32.3.** Dadas as equações  $x^2 - 5x + k = 0$  e  $x^2 - 7x + 2k = 0$ , sabe-se que uma das raízes da segunda equação é o dobro de uma das raízes da primeira equação. Sendo  $k \neq 0$ , determine  $k$ .

**Exercício 32.4.** Determine os valores de  $m \in \mathbb{R}$  para que a função quadrática  $f(x) = mx^2 + (2m - 1)x + (m - 2)$  tenha dois zeros reais e distintos.

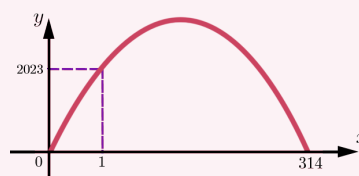
**Exercício 32.5.** Um avião de 100 lugares foi fretado para uma excursão. A companhia exigiu de cada passageiro R\$ 800,00 mais R\$ 10,00 por cada lugar vago. Para que número de passageiros a rentabilidade da empresa é máxima?

**Exercício 32.6.** O diretor de uma orquestra percebeu que, com o ingresso a R\$ 9,00, em média 300 pessoas assistem aos concertos e que, para cada redução de R\$ 1,00 no preço dos ingressos, o público aumenta 100 espectadores. Qual deve ser o preço do ingresso para que a receita seja máxima?

### Exercícios de Aprofundamento

**Exercício 32.7.** Encontre as soluções da equação  $x^4 - 7x^3 + 14x^2 - 7x + 1 = 0$ .

**Exercício 32.8.** Um objeto foi arremessado e tem sua trajetória representada pelo gráfico da função quadrática esboçado na figura a seguir. Qual é a altura máxima atingida por esse objeto?



**Exercício 32.9.** Determine o retângulo de maior área contido num triângulo equilátero de lado 4 cm, estando a base do retângulo num lado do triângulo.

**Exercício 32.10.** Mostre que, para todos  $a, b, c \in \mathbb{R}$ , sendo  $a \neq 0$ , a equação

$$\frac{1}{x-b} + \frac{1}{x-c} = \frac{1}{a^2}$$

possui exatamente duas raízes reais distintas.

**Exercício 32.11.** Resolva a equação  $x = \sqrt{x-1/x} + \sqrt{1-1/x}$  para  $x \in \mathbb{R}$ .

### Exercícios Avançados

**Exercício 32.12.** Prove que se  $a, b$  e  $c$  são inteiros ímpares, as raízes de  $y = ax^2 + bx + c$  não são racionais.

**Exercício 32.13.** Mostre que se  $(a_k)$  é uma P.A. e  $f$  é uma função quadrática qualquer, então a sequência definida por  $b_n = f(a_{n+1}) - f(a_n)$  é uma P.A.. Dê um exemplo para mostrar que a sequência  $c_n = f(a_n)$  não é uma P.A..

**Exercício 32.14.** Considere a função quadrática  $f(x) = ax^2 + bx + c$ , onde  $a \neq 0$ . Considere  $x_1, x_2 \in \mathbb{R}$  quaisquer e mostre que vale

$$f\left(\frac{x_1 + x_2}{2}\right) \leq \frac{f(x_1) + f(x_2)}{2}.$$

Mais ainda, mostre que para todo  $0 < t < 1$  temos  $f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$ .

# 33

## Função Modular

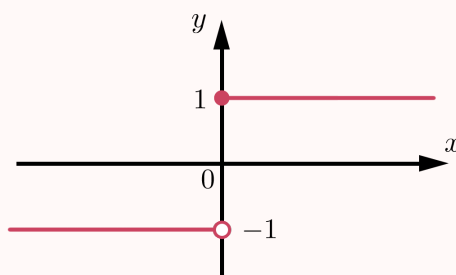
### §33.1 Definições Básicas

A motivação para definir funções por partes vem da necessidade de modelar situações em que uma única expressão não é suficiente para descrever todo o comportamento da função em seu domínio. Em muitos casos, uma função pode apresentar diferentes comportamentos em intervalos distintos de seu domínio, o que torna necessária a definição de expressões diferentes para cada intervalo.

Por exemplo, considere a função que descreve o custo de um serviço de entrega em uma empresa de transporte. O custo pode variar de acordo com a distância percorrida e o peso da carga. Nesse caso, é possível definir uma função por partes, em que cada expressão descreve o custo para um conjunto específico de condições. Essa abordagem permite que a função descreva corretamente o comportamento do custo em todas as situações possíveis.

**Exemplo 33.1.1.** Mostraremos alguns exemplos de funções definidas por partes.

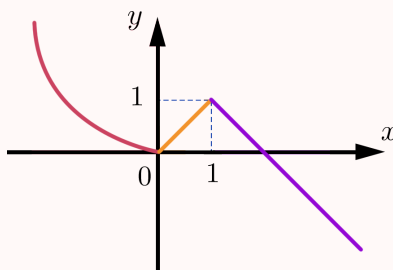
- i) Um exemplo simples de funções definido por partes é a função do tipo degrau, onde  $f(x) = 1$  se  $x \geq 0$  e  $f(x) = -1$ . Uma outra forma de expressar essa função é do seguinte modo  $f(x) = \begin{cases} 1, & \text{se } x \geq 0 \\ -1, & \text{se } x < 0 \end{cases}$ . O gráfico dessa função é dado por



- ii) Podemos usar quantas proposições quisermos para definir funções. Uma função definida pelas proposições abertas  $P(x) : x^2$ ,  $Q(x) : x$  e  $R(x) : 2 - x$  é a seguinte

$$f(x) = \begin{cases} x^2, & \text{se } x < 0 \\ x, & \text{se } 0 \leq x < 1 \\ 2 - x, & \text{se } x \geq 1 \end{cases}$$

e esta função possui o seguinte gráfico





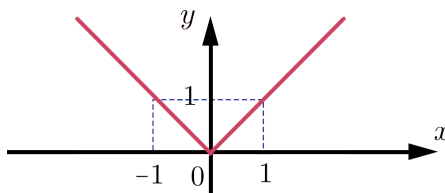
## §33.2 A função modular

**Definição 33.1.** O **valor absoluto**, ou **módulo**, de um número real  $x$  é definido por

$$|x| = \begin{cases} x, & \text{se } x \geq 0 \\ -x, & \text{se } x < 0 \end{cases}$$

Outra maneira de definirmos o módulo de  $x$  consiste em por  $|x| = \max\{x, -x\}$ , isto é, o valor de  $|x|$  é simplesmente o maior dos valores entre  $x$  e  $-x$ .

Assim como fizemos na seção anterior, podemos facilmente esboçar o gráfico da função modular  $|x|$ .



**Exemplo 33.2.1.** O estudo da função  $|x|$  não é tão interessante, por isso os casos mais interessantes são da forma  $|f(x)|$ , onde  $f(x)$  é uma função real qualquer. Já sabemos que a função modular é uma função por partes, então o mesmo deve acontecer com a função

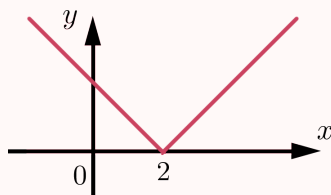
$$|f(x)| = \begin{cases} f(x), & \text{se } f(x) \geq 0 \\ -f(x), & \text{se } f(x) < 0 \end{cases},$$

cada expressão pode ser descrita se conhecermos o sinal da função  $f$  em todo o seu domínio.

Por exemplo, para a função linear  $f(x) = x - 2$  temos que  $f(x) > 0$  sempre que  $x > 2$  e  $f(x) < 0$  se  $x < 2$ . Então, seu módulo  $|f(x)| = |x - 2|$  pode ser expresso como a seguinte função por partes

$$|f(x)| = |x - 2| = \begin{cases} x - 2, & \text{se } x \geq 2 \\ 2 - x, & \text{se } x < 2 \end{cases}$$

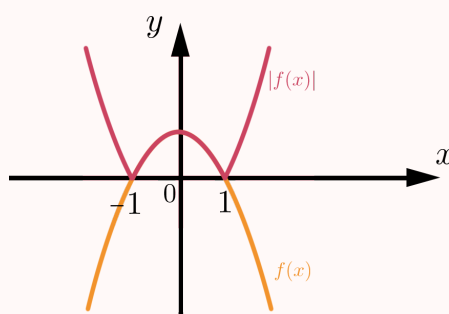
Assim, o seu gráfico é



**Exemplo 33.2.2.** Para explicarmos a relação entre o gráfico da função com seu módulo vamos considerar a função  $f(x) = 1 - x^2$ . Observe que nesse caso, o discriminante é igual a  $\Delta = b^2 - 4ac = 0^2 - 4 \cdot (-1) \cdot 1 = 4 > 0$ , logo  $f$  possui duas raízes e elas são  $-1$  e  $1$ . Veja ainda que o vértice dessa parábola é  $V(-b/2a, -\Delta/4a) = (0, 1)$ . Assim, sabemos que  $f(x) = -(x - 1)(x + 1)$  é positiva apenas no intervalo  $(-1, 1)$ . Desse modo, quando tomamos o módulo de  $f$  devemos obter

$$|f(x)| = \begin{cases} 1 - x^2, & \text{se } -1 \leq x \leq 1 \\ x^2 - 1, & \text{se } x < -1 \text{ ou } x > 1 \end{cases}$$

Assim, seu gráfico é simplesmente



Note que toda a parte negativa da  $f$  foi excluída e substituída pelo seu “reflexo” em torno do eixo- $x$ .

**Exemplo 33.2.3.** Suponha que queremos resolver a seguinte equação modular  $|x - 3| = 6$ . Diferente do que fazíamos antes, a função modular nos força a considerar separadamente os casos onde a função  $x - 3$  é positiva ou negativa. Mas como  $x - 3$  é positiva apenas no intervalo  $(3, +\infty)$ , então temos que

$$|x - 3| = \begin{cases} x - 3, & \text{se } x \geq 3 \\ 3 - x, & \text{se } x < 3 \end{cases}.$$

Daí, para  $x \in [3, +\infty)$  temos a equação  $x - 3 = 6$ , onde a solução é  $x = 9$  e para  $x \in (-\infty, 3)$  temos a equação  $3 - x = 6$ , onde  $x = -3$  é a solução. Portanto, as soluções de  $|x - 3| = 6$  são  $x = -3$  e  $x = 9$ .

**Observação 33.2.1.** Podemos mostrar uma solução geral para as equações da forma  $|x - a| = b$ , onde  $a, b \in \mathbb{R}$ . De fato, observe primeiramente que o módulo de qualquer número é sempre um número maior ou igual a zero, logo, se  $b < 0$ , então a equação modular considerada não possui solução. Já se  $b > 0$ , então após analisar o sinal de  $x - a$  temos que as soluções da equação dada são  $x = a + b$  ou  $x = a - b$ .

**Exemplo 33.2.4.** Consideremos agora a seguinte equação modular  $|x + 1| + |x - 2| = 7$ . Como fizemos anteriormente, podemos afirmar que

$$|x + 1| = \begin{cases} x + 1, & \text{se } x \geq -1 \\ -x - 1, & \text{se } x < -1 \end{cases}$$

$$|x - 2| = \begin{cases} x - 2, & \text{se } x \geq 2 \\ 2 - x, & \text{se } x < 2 \end{cases}$$

Deste modo, temos três intervalos a considerar  $(-\infty, -1)$ ,  $[-1, 2)$  e  $[2, +\infty)$ .

- i) No caso em que  $x \in (-\infty, -1)$  a expressão  $|x + 1| + |x - 2|$  se torna  $-x - 1 + 2 - x = 1 - 2x$ , assim, podemos afirmar que a solução de  $1 - 2x = |x + 1| + |x - 2| = 7$  é  $x = -3$ .
- ii) Se  $-1 < x < 2$ , então  $|x + 1| + |x - 2| = x + 1 + 2 - x = 3$ . Logo, neste intervalo não existe solução da equação  $|x + 1| + |x - 2| = 7$ .

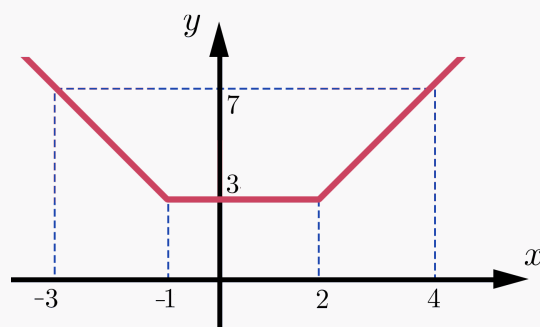
iii) Já para  $x \geq 2$  temos que  $|x+1| + |x-2| = x+1 + x-2 = 2x-1$  e disso temos que a solução da equação modular nesse intervalo é  $x = 4$ .

Portanto, as soluções da equação  $|x+1| + |x-2| = 7$  são  $x = -3$  e  $x = 4$ .

**Observação 33.2.II.** De certo modo, ao analisarmos os intervalos no [Exemplo 33.2.4](#) mostramos que

$$|x+1| + |x-2| = \begin{cases} 1-2x, & \text{se } x < -1 \\ 3, & \text{se } -1 \leq x < 2 \\ 2x-1, & \text{se } x \geq 2 \end{cases}.$$

Então resolver a equação  $|x+1| + |x-2| = 7$  nada mais é que encontrar os pontos em que o gráfico desta função toca a reta  $y = 7$ . Assim, a versão geométrica desse problema pode ser analisada através do seguinte esboço



**Exemplo 33.2.5.** Vamos encontrar os valores de  $x \in \mathbb{R}$  tais que  $x^2 - x|x| - 2 = 0$ . Como esta expressão envolve o valor absoluto  $|x|$  devemos, então, considerar dois casos:

(I) Suponha que  $x \geq 0$ . Assim,  $|x| = x$  e logo temos

$$x^2 - x|x| - 1 = x^2 - x^2 - 2 = -1$$

assim  $x^2 - x|x| - 2 = 0$  não possui solução em  $[0, +\infty)$ .

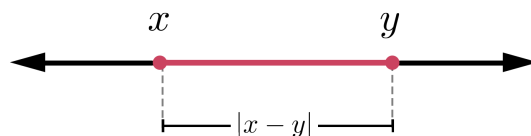
(II) Suponha agora que  $x < 0$ . Assim,  $|x| = -x$  e segue que

$$x^2 - x|x| - 1 = 0 \implies x^2 - x(-x) - 2 = 0 \implies 2x^2 - 2 = 0 \implies x^2 = 1 \implies x = \pm 1$$

Mas observe que  $x = 1$  não pertence ao intervalo que estamos considerando, logo,  $x = -1$  é a única solução.

### §33.3 Uma interpretação geométrica do módulo

Existe uma interpretação geométrica importante sobre o valor absoluto de um número. Usaremos como base a reta real e consideremos dois números  $x$  e  $y$  que foram marcados, assim a distância entre os números marcados é exatamente  $|x - y|$ .



Desse modo, o valor  $|x|$  mede a distância do ponto  $x$  até a origem 0, pois  $|x| = |x - 0|$ . Além disso, é claro que a distância entre  $x$  e  $y$  é o mesmo que a distância entre  $y$  e  $x$ , assim isso pode ser traduzido como  $|x - y| = |y - x|$ . Além disso, igualdades como  $|x - 1| = 3$  significa  $x$  assume os valores que distam 3 do número 1, ou seja,  $x = 1 + 3 = 4$  e  $x = 1 - 3 = -2$ . Resumimos algumas propriedades da função modular na próxima proposição.

**Proposição 33.3.1** – Para quaisquer  $x, y \in \mathbb{R}$  temos:

- a)  $x \leq |x|$ .
- b)  $|x| = 0$  se, e somente se,  $x = 0$ ;
- c)  $|x \cdot y| = |x| \cdot |y|$ ;
- d)  $|x|^2 = x^2$ ;
- e) (*Desigualdade Triangular*)  $|x + y| \leq |x| + |y|$ ;
- f)  $||x| - |y|| \leq |x - y|$ .

*Demonstração.* a) Relembre que  $|x|$  é o maior dos valores entre  $-x$  e  $x$ , assim,  $x \leq |x|$  para qualquer que seja  $x \in \mathbb{R}$ .

b) Se  $x = 0$ , então é claro que  $|x| = 0$ . Por outro lado, se  $|x| = 0$ , então o máximo entre  $x$  e  $-x$  é zero, em qualquer que seja o caso temos  $x = 0$ .

c) Deixamos a prova pro leitor.

d) Se  $x \geq 0$ , então  $|x| = x$ . Disso temos  $|x|^2 = |x| \cdot |x| = x \cdot x = x^2$ .

Se  $x < 0$ , então  $|x| = -x$ . Daí,  $|x|^2 = |x| \cdot |x| = (-x)(-x) = x^2$ .

e) Usando os itens a), c) e d) temos

$$|x + y|^2 = (x + y)^2 = x^2 + 2xy + y^2 \leq |x|^2 + 2|x||y| + |y|^2 = (|x| + |y|)^2$$

Ou seja,  $|x + y|^2 \leq (|x| + |y|)^2$  e como ambos os lados dessa desigualdade são quadrados de termos não-negativos podemos concluir que  $|x + y| \leq |x| + |y|$ .

f) Veja que usando a desigualdade triangular temos  $|x| = |(x - y) + y| \leq |x - y| + |y|$ , ou seja,  $|x| - |y| \leq |x - y|$ . De mesmo modo, podemos concluir  $|y| - |x| \leq |y - x|$ , ou ainda,  $-|x - y| \leq |x| - |y|$ . Assim,

$$-|x - y| \leq |x| - |y| \leq |x - y| \implies ||x| - |y|| \leq |x - y|.$$

□

**Observação 33.3.I.** Para motivar o nome *Desigualdade Triangular* para  $|x + y| \leq |x| + |y|$  substitua  $x = a - b$  e  $y = b - c$ , assim, a desigualdade se torna  $|a - c| \leq |a - b| + |b - c|$ . Pela nossa intuição geométrica, isto nos diz que a distância entre  $a$  e  $c$  é sempre menor do que caminhar sobre o trajeto de  $a$  até  $b$  e depois de  $b$  até  $c$ .

**Observação 33.3.II.** Usamos no fim da proposição anterior o fato que  $|x| \leq a \iff -a \leq x \leq a$ . A prova disto é bem simples. De fato,

$$|x| \leq a \iff x^2 = |x|^2 \leq a^2 \iff x^2 - a^2 = (x - a)(x + a) \geq 0 \iff -a \leq x \leq a.$$

Podemos mostrar, com uma ideia similar, que  $|x| \geq a \iff x \leq -a$  ou  $x \geq a$ . Com efeito,

$$|x| \geq a \iff x^2 = |x|^2 \geq a^2 \iff x^2 - a^2 = (x - a)(x + a) \geq 0 \iff x \leq -a \text{ ou } x \geq a$$

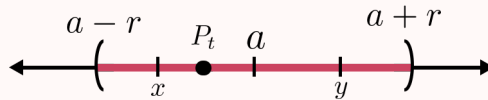
Podemos obter uma visão geométrica da [Observação 33.3.II](#) interpretando que a desigualdade  $|x - a| < b$ , onde  $a$  e  $b$  são números reais fixos, significa os números  $x$  tais que a distância até o número  $a$  é no máximo  $b$ . Então por um pensamento parecido com o anterior, podemos concluir que  $|x - a| > b$  são os números  $x$  tais que a distância até  $a$  é pelo menos  $b$ .



**Exemplo 33.3.2.** Considere o intervalo centrado em  $a$  de raio  $r > 0$ , isto é, o intervalo  $I = (a - r, a + r)$ . Para qualquer par  $x, y \in I$  de elementos de  $I$  e  $t \in [0, 1]$  o número  $P_t = tx + (1 - t)y$  ainda está em  $I$ . De fato, note que  $x, y \in I$  implica  $|x - a| < r$  e  $|y - a| < r$ . Assim, tomando  $t \in [0, 1]$  nós obtemos pela Desigualdade Triangular

$$\begin{aligned} |[tx + (1 - t)y] - a| &= |t(x - a) + (1 - t)(y - a)| \\ &\leq |t(x - a)| + |(1 - t)(y - a)| \\ &= t|x - a| + (1 - t)|y - a| \\ &< tr + (1 - t)r = r \end{aligned}$$

Podemos ver esse resultado geometricamente através da seguinte imagem



**Exemplo 33.3.3 (A razão áurea).** Dados dois números reais  $a < b$ , considere o intervalo de extremos  $a$  e  $b$ . Assim, pelo [Exemplo 33.3.2](#), sabemos que o ponto  $P_t = ta + (1 - t)b$  está contido em  $(a, b)$ , sempre que  $t \in [0, 1]$ . Além disso, para este ponto  $P_t$ , nós temos  $\frac{P_t - a}{b - a} = t$  e esta é a razão em que foi dividido o intervalo  $(a, b)$ . Em particular, temos que para  $t = 1/2$  temos  $P_{1/2} = a \cdot 1/2 + (1 - 1/2)b = (a + b)/2$  ou seja, o *Ponto Médio* do segmento de extremos  $a$  e  $b$ .

Diz-se que o ponto  $x \in (a, b)$  divide o segmento de extremos  $a$  e  $b$  de acordo com a *divisão áurea* quando se tem  $\frac{x - a}{b - a} = \frac{b - x}{x - a}$ . Observe que isso se transforma em uma equação do

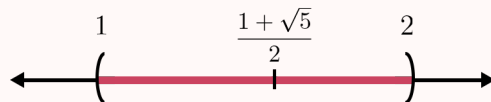
segundo grau em que podemos determinar as soluções

$$\begin{aligned}\frac{x-a}{b-a} = \frac{b-x}{x-a} &\implies (x-a)^2 = (b-x)(b-a) \\ &\implies x^2 - 2ax + a^2 = b^2 - ab - x(b-a) \\ &\implies x^2 + (b-3a)x + [a^2 - b^2 + ab] = 0 \\ &\implies x = \frac{(3a-b) \pm \sqrt{(b-3a)^2 - 4(a^2 - b^2 + ab)}}{2}\end{aligned}$$

Veja que

$$\begin{aligned}(b-3a)^2 - 4(a^2 - b^2 + ab) &= b^2 - 6ab + 9a^2 - 4a^2 + 4b^2 - 4ab \\ &= 5(b^2 - 2ab + a^2) = 5(b-a)^2.\end{aligned}$$

Disso podemos concluir que

$$x = \frac{(3a-b) \pm (b-a)\sqrt{5}}{2}.$$


Então se tomarmos  $a = 1$  e  $b = 2$  temos um segmento de comprimento 1 e que é dividido na razão áurea apenas quando  $x = \frac{1+\sqrt{5}}{2}$ , que é aproximadamente, 1,618.... Por este motivo o número  $\frac{1+\sqrt{5}}{2}$  é conhecido como *número de ouro*.

**Exemplo 33.3.4.** Uma aplicação interessante para a desigualdade triangular é encontrar estimativas para o valor de funções em um certo intervalo. Por exemplo, considere a função  $f(x) = \frac{x^2-3}{x-2}$ . Veja que se  $|x-1| < 2/3$ , ou seja, se  $x \in (1-2/3, 1+2/3) = (1/3, 5/3)$  temos

$$\left| \frac{x^2-3}{x-2} \right| = \left| \frac{(x-1)^2 + 2(x-2)}{x-2} \right| = \left| \frac{(x-1)^2}{x-2} + 2 \right| \leq \left| \frac{(x-1)^2}{x-2} \right| + 2$$

Mas como  $|x-1| < 2/3$ , então elevando ambos os lados ao quadrado temos  $(x-1)^2 < 4/9$  e disso segue

$$|x-2| = |(x-1) - 1| \geq \left| \frac{2}{3} - 1 \right| = \frac{1}{3} \implies \frac{1}{|x-2|} \leq 3.$$

Portanto, temos que

$$\left| \frac{x^2-3}{x-2} \right| < 3 \cdot \frac{4}{9} + 2 = \frac{10}{3}.$$

Com um pouco mais de trabalho da pra mostrar que esta função é positiva no intervalo  $(1/3, 5/3)$  e que o seu valor máximo é igual a 2 neste mesmo intervalo.

## §33.4 Exercícios

### Exercícios Introdutórios

**Exercício 33.1.** Durante o ano de 2023 uma empresa teve seu lucro diário  $L$  dado pela função  $L(x) = 50(|x - 100| + |x - 200|)$ . Para qual dia  $x > 0$  temos o lucro igual R\$ 10000?

**Exercício 33.2.** Esboce o gráfico das funções

$$(a) f(x) = \begin{cases} -2, & \text{se } x \leq -2 \\ x, & \text{se } -2 < x < 2 \\ 2, & \text{se } x \geq 2 \end{cases} \quad (b) f(x) = \begin{cases} -x^2 + 4x + 3, & \text{se } x < 0 \\ -x^2 - 4x + 3, & \text{se } x \geq 0 \end{cases}$$

**Exercício 33.3.** Determine os zeros da função  $f(x) = x^2 - 3|x| + 1$  e esboce seu gráfico.

**Exercício 33.4.** Determine os valores de  $x \in \mathbb{R}$  tais que:

$$\text{i) } |2x + 5| > 1; \quad \text{ii) } |3x + 1| < -4; \quad \text{iii) } |5x + 1| \leq 3; \quad \text{iv) } |7x + 1| \geq -9$$

### Exercícios de Aprofundamento

**Exercício 33.5.** Esboce o gráfico da função  $f(x) = ||2x - 2| - 4|$ .

**Exercício 33.6.** Quando a igualdade é atingida na desigualdade triangular? Isto é, dados  $x, y \in \mathbb{R}$  quando vale  $|x + y| = |x| + |y|$ ?

**Exercício 33.7.** Se as soluções de  $x^2 - |x| - 6 = 0$  são raízes da equação  $x^2 - \alpha x + \beta = 0$ , calcule o valor de  $\alpha$  e  $\beta$ .

**Exercício 33.8.** Mostre os seguintes itens:

- Usando a *Desigualdade Triangular*, mostre que  $|x - \ell| + |x - 50 - \ell| \geq 50$  vale para qualquer  $y \in \mathbb{R}$  e  $\ell \in \mathbb{N}$ .
- Conclua que  $|x - 1| + |x - 2| + \cdots + |x - 100| \geq 50^2$  para todo  $x \in \mathbb{R}$ .

### Exercícios Avançados

**Exercício 33.9.** Sejam  $\alpha$  e  $\beta$  números reais tais que a equação  $x^2 + \alpha|x| + \beta = 0$  possua raízes reais. Prove que a soma de tais raízes é igual a zero.

**Exercício 33.10.** Considere a sequência  $(f_n)$  dada pela recorrência  $f_{n+1} = f_n + f_{n-1}$  com  $f_1 = 1$  e  $f_2 = 2$ . Mostre que vale  $(f_n)^2 + f_{n-1}f_n \geq 2f_{n-1}f_n$  para todo  $n \in \mathbb{N}$ . Definindo  $x_n = \frac{f_n}{f_{n-1}}$  verifique para todo  $n \in \mathbb{N}$  a desigualdade

$$|x_{n+2} - x_{n+1}| \leq \frac{1}{2}|x_{n+1} - x_n|.$$

**Exercício 33.11.** Seja  $f : \mathbb{R} \rightarrow \mathbb{R}$  uma função tal que  $|f(y) - f(x)| = |y - x|$  para quaisquer  $x, y \in \mathbb{R}$ .

- Pondo  $f(0) = a$ , defina a função  $g : \mathbb{R} \rightarrow \mathbb{R}$  assim:  $g(x) = f(x) - a$ . Prove então que  $|g(x)| = |x|$  para todo  $x \in \mathbb{R}$ . Em particular,  $g(1) = 1$  ou  $g(1) = -1$ . Também  $(g(x))^2 = x^2$ .
- Use a identidade  $xy = \frac{1}{2}[x^2 + y^2 - (x - y)^2]$  para mostrar que  $xy = g(x) \cdot g(y)$ .
- Se  $g(1) = 1$ , mostre que  $g(x) = x$  para todo  $x \in \mathbb{R}$ . Se  $g(1) = -1$ , mostre que  $g(x) = -x$  para todo  $x \in \mathbb{R}$ .
- Conclua que  $f(x) = x + a$  para todo  $x \in \mathbb{R}$  ou então  $f(x) = -x + a$  para todo  $x \in \mathbb{R}$ .

Os polinômios são expressões matemáticas que consistem em uma soma de termos, cada um dos quais é um produto de uma constante e uma ou mais variáveis elevadas a potências inteiras. Eles têm uma ampla variedade de utilidades em diversas áreas da matemática, da física, da engenharia e de outras ciências.

Por exemplo, para análise de dados, os polinômios são frequentemente usados para ajustar modelos de regressão, que ajudam a descrever a relação entre duas variáveis. Nesta situação, é usado fortemente uma propriedade interessante de polinômios, onde podemos usar esses para aproximar qualquer tipo de função, onde normalmente essas são de grande complexidade computacional.

Essas são apenas algumas das muitas utilidades dos polinômios. Em geral, eles são uma ferramenta poderosa para descrever e resolver problemas matemáticos em uma variedade de campos.

### §34.1 Definições Básicas

**Definição 34.1.** Dada uma coleção de números reais  $a_0, a_1, a_2, \dots, a_{n-1}, a_n$  consideramos a função  $f : \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0$ , esta é conhecida por **função polinomial** ou **polinômio**. Chamamos cada termo da forma  $a_k x^k$  de **monômio** e o termo  $a_k$  são os **coeficientes**. Para o monômio de maior potência de  $x$  que possui coeficiente não nulo, definimos esta potência sendo o **grau do polinômio**, isto é, para a função definida anteriormente temos que seu grau é igual a  $n$  e denotamos  $\text{Grau}(f) = n$ .

Um bom observador irá perceber que já tratamos de polinômios em aulas anteriores. De fato, quando estudamos funções afim, na verdade estávamos estudando polinômios de grau 1 e no estudo das funções quadráticas foram tratados polinômios de grau 2. Até as funções constantes são polinômios de grau zero, a menos da função identicamente nula que apesar de ser polinômio não definimos o seu grau.

**Exemplo 34.1.1.** Veja que temos duas operações elementares para funções polinomiais que são a soma e a multiplicação. Por exemplo, sejam  $f(x) = x^3 + x^2 + 1$  e  $g(x) = 2x^2 + x$ , então

$$\begin{aligned}(f + g)(x) &= f(x) + g(x) = (x^3 + x^2 + 1) + (2x^2 + x) = x^3 + 3x^2 + x + 1 \\(f \cdot g)(x) &= f(x) \cdot g(x) = (x^3 + x^2 + 1) \cdot (2x^2 + x) = 2x^5 + 3x^4 + x^3 + 2x^2 + x\end{aligned}$$

Note que as funções  $(f + g)(x) = f(x) + g(x)$  e  $(f \cdot g)(x) = f(x) \cdot g(x)$ , são ainda polinômios para quaisquer que sejam os polinômios  $f(x)$  e  $g(x)$ . Assim, como vimos no caso anterior, a soma de polinômios nos dá um polinômio cujo o grau não pode passar nem do grau de  $f$  ou de  $g$ .

Para ver que o grau da soma de polinômios pode diminuir, basta consideramos os polinômios de grau 3 dados por  $p(x) = -3x^3 + 2x^2 + 1$  e  $q(x) = 3x^3 - x^2$ , onde temos

$$(p + q)(x) = p(x) + q(x) = (-3x^3 + 2x^2 + 1) + (3x^3 - x^2) = x^2 + 1$$

Por outro lado, o produto de polinômios gera um polinômio de grau igual a soma dos graus dos polinômios envolvidos no produto. Este é mais intuitivo de pensar por conta da propriedade distributiva que já conhecemos dos números reais.

Resumindo as observações do [Exemplo 34.1.1](#) nós temos a seguinte proposição que apresentamos sem demonstração



**Proposição 34.1.2** – Dados os polinômios não nulos  $f$  e  $g$ , temos:

- (i)  $\text{Grau}(f + g) \leq \max\{\text{Grau}(f), \text{Grau}(g)\}$ ;
- (ii)  $\text{Grau}(fg) = \text{Grau}(f) + \text{Grau}(g)$ .

Definimos a **igualdade entre polinômios**  $f(x) = a_n x^n + \dots + a_1 x + a_0$  e  $g(x) = b_k x^k + \dots + b_1 x + b_0$  quando ambos possuem o mesmo grau, ou seja, quando  $n = k$  e se  $a_i = b_i$  para todo  $1 \leq i \leq k$ , isto é, quando todos os coeficientes dos monômios de mesmo grau coincidem.

**Exemplo 34.1.3.** Suponhamos estamos buscando polinômios que satisfaçam a relação

$$f(x) + f(1 - x) = 1.$$

Para grau 0 a única solução é a função  $f(x) = 1/2$ . Já para  $\text{Grau}(f) = 1$ , temos que a função é da forma  $f(x) = ax + b$ , assim, a condição se traduz como

$$\begin{aligned} f(x) + f(1 - x) = 1 &\implies (ax + b) + (a(1 - x) + b) = 1 \\ &\implies a + 2b = 1 \implies b = \frac{1 - a}{2} \end{aligned}$$

Logo, todas as funções polinomiais de grau um que satisfazem  $f(x) + f(1 - x) = 1$  são da forma  $f(x) = ax + \frac{1 - a}{2}$ . Para um exemplo concreto, tome  $a = 1$  e assim obtemos  $f(x) = x$ .

Já para com grau três devemos encontrar uma função da forma  $f(x) = x^3 + ax^2 + bx + c$  que satisfaz a relação dada. Neste caso, como  $f(1 - x) = -x^3 + (a + 3)x^2 + (-2a - b - 3)x + (a + b + c + 1)$  temos,

$$\begin{aligned} 1 &= f(x) + f(1 - x) \\ &= [x^3 + ax^2 + bx + c] + [-x^3 + (a + 3)x^2 + (-2a - b - 3)x + (a + b + c + 1)] \\ &= (2a + 3)x^3 + (-2a - 3)x + (a + b + 2c + 1) \end{aligned}$$

Igualando os coeficientes nós obtemos  $2a + 3 = 0$  e  $a + b + 2c + 1 = 1$ , ou seja,  $a = -3/2$  e  $b = -2c + 3/2$ . Então todos os polinômios de grau 3 que satisfazem a relação  $f(x) + f(1 - x) = 1$  são da forma  $f(x) = x^3 - \frac{3}{2}x^2 + \left(\frac{3}{2} - 2c\right)x + c$ .

### §34.1.4 Relação entre coeficientes e raízes de polinômios

Vejamos uma aplicação bem legal para a igualdade entre polinômios. Sabemos que um polinômio de grau 2 é da forma  $ax^2 + bx + c$  e que também pode ser escrito como  $a(x - r_1)(x - r_2)$ . Expandindo este ultimo produto temos

$$a(x - r_1)(x - r_2) = a(x^2 - (r_1 + r_2)x + r_1 r_2) = ax^2 - a(r_1 + r_2)x + ar_1 r_2.$$

Mas, pela igualdade entre os polinômios  $ax^2 + bx + c$  e  $a(x - r_1)(x - r_2)$  devemos ter as seguintes igualdades

$$\begin{cases} b = -a(r_1 + r_2) \\ c = ar_1 r_2 \end{cases} \implies \begin{cases} r_1 + r_2 = -\frac{b}{a} \\ r_1 \cdot r_2 = \frac{c}{a} \end{cases}$$

Essas relações são bem interessantes pois conectam as raízes de um polinômio quadrático com seus coeficientes. Vejamos que o mesmo ocorre para polinômios de grau 3, ou seja, polinômios que são

da forma  $ax^3 + bx^2 + cx + d$  e que por outro lado podem ser expressos por  $a(x - r_1)(x - r_2)(x - r_3)$ . Assim, como anteriormente, vamos expandir o produto e encontramos

$$\begin{aligned} a(x - r_1)(x - r_2)(x - r_3) &= a[x^2 - (r_1 + r_2)x + r_1r_2](x - r_3) \\ &= a[x^3 - (r_1 + r_2 + r_3)x^2 + (r_1r_2 + r_1r_3 + r_2r_3)x - r_1r_2r_3] \\ &= ax^3 - a(r_1 + r_2 + r_3)x^2 + a(r_1r_2 + r_1r_3 + r_2r_3)x - ar_1r_2r_3 \end{aligned}$$

Pela igualdade de polinômios temos

$$\begin{cases} b = -a(r_1 + r_2 + r_3) \\ c = a(r_1r_2 + r_1r_3 + r_2r_3) \\ d = -a(r_1r_2r_3) \end{cases} \implies \begin{cases} r_1 + r_2 + r_3 = -\frac{b}{a} \\ r_1 \cdot r_2 + r_1 \cdot r_3 + r_2 \cdot r_3 = \frac{c}{a} \\ r_1 \cdot r_2 \cdot r_3 = -\frac{d}{a} \end{cases}.$$

Então já temos base para conjecturar que um polinômio da forma  $ax^4 + bx^3 + cx^2 + dx + e$  cuja as raízes são  $r_1, r_2, r_3$  e  $r_4$  deve satisfazer

$$\begin{cases} r_1 + r_2 + r_3 + r_4 = -\frac{b}{a} \\ r_1r_2 + r_1r_3 + r_1r_4 + r_2r_3 + r_2r_4 + r_3r_4 = \frac{c}{a} \\ r_1r_2r_3 + r_1r_2r_4 + r_1r_3r_4 + r_2r_3r_4 = -\frac{d}{a} \\ r_1r_2r_3r_4 = -\frac{e}{a} \end{cases}$$

De fato isso ocorre e a verificação é similar ao que já fizemos. Essas equações são conhecidas como **Equações de Girard**. Para polinômios de grau  $n$ , as equações podem ser enunciadas dizendo que o polinômio  $a_nx^n + a_{n-1}x^{n-1} + \dots + a_2x^2 + a_1x + a_0$  pode ser expresso por  $a_n(x^n - \sigma_1x^{n-1} + \sigma_2x^{n-2} + \dots + (-1)^n\sigma_n)$ , onde  $\sigma_1$  significa a soma das raízes desse polinômio,  $\sigma_2$  significa a soma dos produtos dois a dois das raízes,  $\sigma_3$  a soma dos produtos três a três das raízes e assim por diante.

**Exemplo 34.1.5.** Considere o polinômio  $x^3 + 2023x - 10$  e suponha que suas raízes são  $a, b$  e  $c$ . Vamos calcular quanto vale  $a^3 + b^3 + c^3$ . Vejamos que como  $a, b$  e  $c$  são raízes, então temos as três equações  $a^3 + 2023a - 10 = 0$ ,  $b^3 + 2023b - 10 = 0$  e  $c^3 + 2023c - 10 = 0$  a soma dessas igualdades nos dá

$$a^3 + b^3 + c^3 + 2023(a + b + c) - 30 = 0.$$

Mas pelas equações de Girard,  $a + b + c = 0$  pois este é o coeficiente quadrático do polinômio que estamos trabalhando. Assim,  $a^3 + b^3 + c^3 = 30$ .

**Exemplo 34.1.6 (ITA – 2018).** Seja  $f(x) = x^3 + ax^2 + bx$  um polinômio cuja raízes são não-negativas e estão em Progressão Aritmética. Sabendo que a soma dos seus coeficientes é igual a 10, podemos afirmar que a soma das raízes é igual a quanto ?

**Solução.** Pelas Equações de Girard, temos que o produto das raízes é igual a 0, pois o coeficiente constante não aparece na expressão de  $f(x) = x^3 + ax^2 + bx$ . Assim, uma das raízes é  $x = 0$  e como as outras estão em progressão aritmética, então vamos dizer que as raízes são  $0, r$  e  $2r$ . Como a soma dos coeficientes é igual a 10, então  $1 + a + b = 10$ , ou ainda,  $a + b = 9$ . Além disso, usando novamente Girard, temos que  $-a = 0 + r + 2r$  e

$b = 0 \cdot r + 0 \cdot 2r + r \cdot 2r$ , ou seja,  $a = -3r$  e  $b = 2r^2$ . Onde,

$$a + b = 9 \implies -3r + 2r^2 - 9 = 0 \implies r = -3/2 \text{ ou } r = 3$$

Veja que as raízes de  $f(x)$  são números não-negativos, então o único valor válido é  $r = 3$ , assim, a soma das raízes é igual a  $0 + r + 2r = 9$ .

**Observação 34.1.I.** Veja que não estamos nos restringindo a raízes reais, logo as Equações de Girard também no caso mais geral, onde admitimos raízes complexas.

Suponha que um polinômio  $a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$  possa ser escrito como  $a_n(x - r_1)(x - r_2) \dots (x - r_n)$ , onde  $r_1, \dots, r_n$  são as raízes deste polinômio. Daí vem

Uma investigação interessante que podemos fazer sobre polinômios nasce de um questionamento simples: Se conhecermos algumas informações sobre o seu comportamento, como seu grau e alguns valores numéricos, então será possível encontrar sua expressão? Essa questão ainda não será totalmente respondida, mas mostraremos casos onde podemos sim encontrar as expressões de polinômios.

**Exemplo 34.1.7.** Suponhamos que estamos à procura de um polinômio  $f(x)$  de grau 1 tal que  $f(4) = 4$  e  $f(16) = 10$ . Assim, procuramos funções da forma  $f(x) = ax + b$  que satisfaçam ao seguinte sistema de equações lineares

$$\begin{aligned} 4 &= f(4) = 4a + b \\ 10 &= f(16) = 16a + b \end{aligned}$$

Subtraindo a segunda da primeira linha nós obtemos

$$(16a + b) - (4a + b) = 10 - 4 \implies 12a = 6 \implies a = 1/2$$

Logo, substituindo este valor na primeira linha nos vem que

$$4a + b = 4 \implies 2 + b = 4 \implies b = 2$$

Portanto, o polinômio procurado é  $f(x) = \frac{1}{2}x + 2$ .

**Exemplo 34.1.8.** Suponhamos que estamos à procura de um polinômio  $f(x)$  de grau 2 tal que  $f(-1) = f(2) = 0$  e  $f(0) = 2$ . Assim, procuramos funções da forma  $f(x) = ax^2 + bx + c$  que satisfaçam ao seguinte sistema de equações lineares

$$\begin{cases} 0 = f(-1) = a \cdot (-1)^2 + b \cdot (-1) + c \\ 0 = f(2) = a \cdot 2^2 + b \cdot 2 + c \\ 2 = f(0) = a \cdot 0^2 + b \cdot 0 + c \end{cases} \implies \begin{cases} a - b + c = 0 \\ 4a + 2b + c = 0 \\ c = 2 \end{cases} \implies \begin{cases} a - b = 2 \\ 2a + b = -1 \\ c = 2 \end{cases}$$

Somando as duas primeiras linhas nós obtemos

$$(a - b) + (2a + b) = 0 + 0 \implies 3a = -3 \implies a = -1.$$

Daí, substituindo este valor em  $a - b = 2$  concluímos que  $b = 1$ . Portanto, o polinômio procurado é  $f(x) = -x^2 + x + 2$ .

**Observação 34.1.II.** Em geral, se conhecermos  $n + 1$  valores de um polinômio de grau  $n$  então, pelo método anterior, podemos determinar a sua expressão. Se mudarmos o grau e mantivermos os mesmos valores numéricos, o polinômio procurado deixa de ser único, por exemplo, os polinômios  $f(x) = 0$ ,  $g(x) = (x - 2)(x - 5)$  e  $h(x) = (x - 2)(x - 5)(x - 8)$  possuem graus distintos e são tais que se anulam para  $x = 2$  e  $x = 5$ .

O matemático *Joseph Louis Lagrange (1736-1813)* foi capaz de desenvolver um método conhecido por *Interpolação de Lagrange*, que é bastante simples para encontrar o polinômio de menor grau que satisfaz  $P(a_1) = b_1, P(a_2) = b_2, \dots, P(a_{n+1}) = b_{n+1}$ , onde  $a_1, \dots, a_{n+1}, b_1, \dots, b_{n+1}$  são números reais conhecidos. Naturalmente, todos os  $a_1, \dots, a_{n+1}$  são distintos. Este método parte do fato do polinômio

$$L_i(x) = \frac{(x - a_1)(x - a_2) \cdots (x - a_{i-1})(x - a_{i+1}) \cdots (x - a_{n+1})}{(a_i - a_1)(a_i - a_2) \cdots (a_i - a_{i-1})(a_i - a_{i+1}) \cdots (a_i - a_{n+1})}$$

possuir os números  $a_1, a_2, \dots, a_{i-1}, a_{i+1}, \dots, a_{n+1}$  como seus zeros e ainda  $L_i(a_i) = 1$ . Assim, o *Polinômio Interpolador de Lagrange* é dado por

$$P(x) = b_1 L_1(x) + b_2 L_2(x) + \cdots + b_{n+1} L_{n+1}(x).$$

**Exemplo 34.1.9.** Vejamos o Polinômio Interpolador de Lagrange que satisfaz  $P(-1) = 3, P(2) = 6$  e  $P(4) = -2$ . Assim,

$$\begin{aligned} L_1(x) &= \frac{(x - 2)(x - 4)}{[(-1) - 2][(-1) - 4]} = \frac{1}{15}(x - 2)(x - 4) \\ L_2(x) &= \frac{[x - (-1)](x - 4)}{[2 - (-1)](2 - 4)} = -\frac{1}{6}(x + 1)(x - 4) \\ L_3(x) &= \frac{[x - (-1)](x - 2)}{[4 - (-1)](4 - 2)} = \frac{1}{10}(x + 1)(x - 2) \end{aligned}$$

Donde temos que o polinômio interpolador é dado por

$$\begin{aligned} P(x) &= 3 \cdot L_1(x) + 6 \cdot L_2(x) + (-2) \cdot L_3(x) \\ &= \frac{3}{15}(x - 2)(x - 4) - \frac{6}{6}(x + 1)(x - 4) - \frac{2}{10}(x + 1)(x - 2) = -x^2 + 2x + 6 \end{aligned}$$

## §34.2 Divisão de polinômios

Vamos considerar dois polinômios  $A(x)$  e  $B(x)$ , onde  $B(x)$  é não nulo. Dividir o polinômio  $A(x)$  por  $B(x)$  é o mesmo que encontrar os polinômios  $Q(x)$  e  $R(x)$  que satisfazem  $A(x) = B(x)D(x) + R(x)$  e  $\text{Grau}(R) < \text{Grau}(Q)$  ou  $R(x)$  é o polinômio nulo. Assim como fazemos para números reais,  $A(x)$  é o polinômio dividendo,  $B(x)$  é o polinômio divisor,  $Q(x)$  é o polinômio quociente e  $R(x)$  é o polinômio resto. Quando temos que  $R(x)$  é o polinômio nulo, dizemos que  **$A(x)$  é um divisível por  $B(x)$** .

### §34.2.1 Método da Chave

Mostraremos o **Método da Chave** como uma forma prática de realizar a divisão de dois polinômios. Faremos a divisão entre os polinômios  $A(x) = 8x^3 + 4x^2 + 1$  e  $B(x) = 4x^2 + 1$ . Seguiremos as seguintes instruções:

- (1°) Escrevemos o polinômio dividendo e divisor seguindo a ordem decrescente das potências de  $x$ , completando ou deixando em branco os termos de coeficientes zero.

$$\begin{array}{r} 8x^3 + 4x^2 \quad + 1 \quad | \quad 4x^2 + 1 \\ \hline \end{array}$$

- (2°) Dividimos o termo de maior grau de  $A(x)$  pelo termo de maior grau de  $B(x)$ , obtendo assim,  $8x^3/4x^2 = 2x$  como o primeiro termo do polinômio  $Q(x)$

$$\begin{array}{r} 8x^3 + 4x^2 \quad + 1 \quad | \quad 4x^2 + 1 \\ \hline 2x \end{array}$$

- (3°) Multiplicando o termo encontrado,  $2x$ , pelo divisor e subtraímos do dividendo o resultado obtido,  $8x^3 + 2x$ , chegando ao resultado parcial  $4x^2 - 2x + 1$ .

$$\begin{array}{r} 8x^3 + 4x^2 \quad + 1 \quad | \quad 4x^2 + 1 \\ - 8x^3 \quad - 2x \quad | \quad 2x \quad \hline \end{array}$$

- (4°) Dividimos o termo de maior grau do resultado parcial pelo termo de maior grau do divisor, assim encontramos  $4x^2/4x^2 = 1$  como o próximo termo do quociente. Repetimos o mesmo processo anterior para obtermos um novo resultado parcial.

$$\begin{array}{r} 8x^3 + 4x^2 \quad + 1 \quad | \quad 4x^2 + 1 \\ - 8x^3 \quad - 2x \quad | \quad 2x + 1 \quad \hline 4x^2 - 2x + 1 \\ - 4x^2 \quad - 1 \quad \hline - 2x \end{array}$$

- (5°) A divisão termina quando o grau do resto é menor que o grau do divisor ou quando o resto é zero. Nesse caso, ela termina agora e podemos verificar que os polinômios encontrados  $Q(x) = 2x + 1$  e  $R(x) = -2x$  são tais que  $A(x) = B(x) \cdot Q(x) + R(x)$ , ou seja,

$$8x^3 + 4x^2 + 1 = (4x^2 + 1)(2x + 1) - 2x.$$

**Exemplo 34.2.2.** Vamos mostrar mais dois exemplos de divisão com chave e encorajamos o leitor cuidadoso a verificar todos os passos para que este método se torne mais natural.

$$\begin{array}{r} x^3 + 3x^2 + 5x + 6 \quad | \quad x + 2 \\ - x^3 - 2x^2 \quad | \quad x^2 + x + 3 \\ \hline x^2 + 5x \\ - x^2 - 2x \quad \hline 3x + 6 \\ - 3x - 6 \quad \hline 0 \end{array} \qquad \begin{array}{r} 6x^3 - 2x^2 + x + 3 \quad | \quad x^2 - x + 1 \\ - 6x^3 + 6x^2 - 6x \quad | \quad 6x + 4 \\ \hline 4x^2 - 5x + 3 \\ - 4x^2 + 4x - 4 \quad \hline - x - 1 \end{array}$$

Neste primeiro caso temos que  $x^3 + 3x^2 + 5x + 6$  é divisível por  $x + 2$ , ou seja, podemos fatorar o polinômio dividendo como  $(x + 2)(x^2 + x + 3)$ .

**Exemplo 34.2.3.** Determine os valores de  $a$  e  $b$  para que o polinômio  $-x^3 + 2x^2 - ax + 2b$  seja divisível por  $x^2 - x + 1$ .

Pelo Método das Chaves temos,

$$\begin{array}{r|l} \begin{array}{r} -x^3 + 2x^2 \\ x^3 - x^2 \end{array} & \begin{array}{r} -ax \\ +x \end{array} \\ \hline \begin{array}{r} x^2 + (1 - 1a)x \\ -x^2 \end{array} & \begin{array}{r} +2b \\ +x \\ -1 \end{array} \\ \hline (2 - 1a)x + (-1 + 2b) & \end{array} \quad \left| \begin{array}{l} x^2 - x + 1 \\ -x + 1 \end{array} \right.$$

Assim queremos que o polinômio  $(2 - a)x + (2b - 1)$  seja igual ao polinômio nulo, assim, pela igualdade de polinômios devemos ter  $2 - a = 0$  e  $2b - 1 = 0$ , ou seja,  $a = 2$  e  $b = 1/2$ .

Vamos agora nos restringir a divisão por polinômios da forma  $x - a$ . Fazendo uso do método da chave vemos que o resto divisão de  $P(x) = 2x^3 - 9x^2 + 3x + 6$  por  $x - 1$  é igual a 2.

$$\begin{array}{r|l} 2x^3 - 9x^2 + 3x + 6 & x - 1 \\ - 2x^3 + 2x^2 & \\ \hline & - 7x^2 + 3x \\ & 7x^2 - 7x \\ \hline & - 4x + 6 \\ & 4x - 4 \\ \hline & 2 \end{array}$$

Por outro lado,  $P(1) = 2 \cdot 1^2 - 9 \cdot 1^2 + 3 \cdot 1 + 6 = 2 - 9 + 3 + 6 = 2$ , que é o mesmo valor do resto encontrado! Talvez isso já fosse óbvio, pois na divisão de um polinômio  $P(x)$  por  $x - a$  o polinômio resto tem que ser um polinômio de grau zero ou o polinômio nulo, em qualquer caso o resto sempre será um número real que chamaremos de  $R$ . Pela igualdade  $P(x) = Q(x)(x - a) + R$  ao substituir  $x = a$  encontramos  $P(a) = Q(x)(a - a) + R = R$ . O que prova o seguinte teorema

**Teorema 34.1** (Teorema do Resto) – O resto da divisão de um polinômio  $P(x)$  por  $x - a$  é igual a  $P(a)$ .

E assim obtemos um corolário imediato desse teorema. De fato, se  $P(x)$  for divisível por  $x - a$  então o resto da divisão entre eles é zero, mas pelo teorema esse é o valor de  $P(a)$ , ou seja,  $P(a) = 0$ .

**Corolário 34.2** – O polinômio  $P(x)$  é divisível por  $x - a$  se, e somente se,  $a$  é raiz de  $P(x)$ .

**Exemplo 34.2.4.** Vamos calcular o valor de  $c$  para que o polinômio  $x^3 + 2x^2 - (9c)x + 2c$  seja divisível por  $x + 2$ . Pelo corolário, para que isso ocorra devemos ter  $-2$  como uma das raízes do polinômio dado, assim,

$$2^3 + 2 \cdot 2^2 - (9c) \cdot 2 + 2c = 0 \implies 8 + 8 - 18c + 2c = 0 \implies 16c = 16 \implies c = 1.$$

**Observação 34.2.I.** O Teorema do Resto e o seu corolário se estendem a encontrar o resto da divisão por polinômios de grau 1 dados por  $ax + b$ . Com efeito, basta notar que  $P(x) = Q(x)(ax + b) + R$  e aplicando  $x = -b/a$  temos  $R = P(-b/a)$ . Assim, o resto da divisão de um polinômio  $P(x)$  por  $ax + b$  é  $P(-b/a)$ .

### §34.2.5 Método de Briot-Ruffini

Já estudamos a divisão de um polinômio  $P(x)$  por  $x - a$  de dois modos diferentes. Agora vamos mostrar um método para encontrar o quociente e o resto de uma forma rápida e prática para esses casos de divisão. Faremos a divisão entre os polinômios  $A(x) = 2x^3 - 4x + 1$  e  $B(x) = x - 2$  seguindo as seguintes instruções:

- (1°) Na primeira entrada colocamos a raiz de  $x - 2$  e depois dispomos apenas os coeficientes do polinômio  $2x^3 - 4x + 1$  na ordem decrescente das potências de  $x$ .

$$\begin{array}{r|rrrr} 2 & 2 & 0 & -4 & 1 \\ \hline \end{array}$$

- (2°) Repetimos, na linha de baixo, o coeficiente de maior potência de  $x$ .

$$\begin{array}{r|rrrr} 2 & 2 & 0 & -4 & 1 \\ \hline & 2 & & & \end{array}$$

- (3°) Multiplicamos a raiz de  $x - 2$  por esse coeficiente e somamos com o próximo coeficiente do dividendo, colocando o resultado  $2 \cdot 2 + 0 = 4$  imediatamente abaixo

$$\begin{array}{r|rrrr} 2 & 2 & 0 & -4 & 1 \\ \hline & 2 & 4 & & \end{array}$$

- (4°) Multiplicamos o resultado que acabamos de obter pela raiz de  $x - 2$  e somamos o produto obtido com o próximo coeficiente do dividendo, colocando o resultado  $2 \cdot 4 + (-4) = 4$  imediatamente abaixo

$$\begin{array}{r|rrrr} 2 & 2 & 0 & -4 & 1 \\ \hline & 2 & 4 & 4 & \end{array}$$

- (5°) Repetimos esse processo até o último coeficiente de  $P(x)$ . O último resultado obtido é o resto da divisão e todos os coeficientes anteriores são do polinômio quociente.

$$\begin{array}{r|rrrr} 2 & 2 & 0 & -4 & 1 \\ \hline & 2 & 4 & 4 & 9 \end{array}$$

Desse modo, 9 é o resto da divisão entre  $2x^3 - 4x + 1$  e  $x - 2$  e o polinômio quociente é  $Q(x) = 2x^2 + 4x + 4$ .

**Exemplo 34.2.6.** Vamos mostrar mais dois exemplos de divisão com chave e encorajamos o leitor cuidadoso a verificar todos os passos para que este método se torne mais natural.

$$\begin{array}{r|rrrrrr} 5 & -3 & 2 & -1 & 2 & 0 & -1 \\ \hline & -3 & -13 & -66 & -328 & -1640 & -8201 \end{array} \qquad \begin{array}{r|rrrr} -3 & 3 & 2 & -4 \\ \hline & 3 & -7 & 17 \end{array}$$

$$\begin{array}{r|rrrr}
 8 & 1 & -1 & 1 & -2 \\
 & 1 & 7 & 57 & 454
 \end{array}
 \qquad
 \begin{array}{r|rrrrrrrr}
 -1 & 1 & 0 & 0 & 0 & 2 & 0 & -3 \\
 & 1 & -1 & 1 & -1 & 3 & -3 & 0
 \end{array}$$

**Observação 34.2.II.** Note que podemos usar Briot-Ruffini para expressões da forma  $ax + b$ , onde a raiz é  $-b/a$ , porém, temos que multiplicar o quociente obtido por  $1/a$ . Pois, se  $Q(x)$  e  $R$  são o quociente e o resto da divisão de  $P(x)$  por  $x + b/a$ , então vale

$$P(x) = Q(x)(x + b/a) + R = Q(x) \cdot \frac{1}{a} \cdot a(x + b/a) + R = \left[ \frac{1}{a} Q(x) \right] (ax + b) + R.$$

Podemos repetir o método de Briot-Ruffini para encontrarmos a divisão por polinômios da forma  $(x - a)(x - b)$  se o polinômio for divisível por  $x - a$ . Neste caso, aplicamos o método primeiramente para  $x - a$  e como o resto será zero, aplicamos novamente o método para  $x - b$ .

**Exemplo 34.2.7.** Por exemplo, vamos mostrar que  $x^3 - 3x^2 - 6x + 8$  é divisível  $(x + 2)(x - 4)$ .

Assim, aplicando o método primeiro para  $x + 2$  temos

$$\begin{array}{r|rrrr}
 -2 & 1 & -3 & -6 & 8 \\
 & 1 & -5 & 4 & 0
 \end{array}$$

Como o

resto foi zero, aplicamos ao quociente encontrado o método para  $x - 4$ , assim,

$$\begin{array}{r|rrrr}
 4 & 1 & -5 & 4 & \\
 & 1 & -1 & 0 & 
 \end{array}$$

Portanto, podemos afirmar que  $x^3 - 3x^2 - 6x + 8 = (x - 1)[(x + 2)(x - 4)]$ . Em particular, encontramos todas as raízes deste polinômio.

Neste material, na maior parte dele, estamos trabalhando apenas com números reais. Porém, existe o conjunto dos números complexos  $\mathbb{C} = \{a + bi \mid a, b \in \mathbb{R}\}$ , onde  $i$  é conhecida como *unidade imaginária* e é raiz de  $z^2 + 1 = 0$ , ou seja,  $i^2 = -1$ . Nesse novo mundo, existe um teorema fantástico que faz parte da nossa teoria e não podemos deixá-lo de fora.

**Teorema 34.3** (Teorema Fundamental da Álgebra) – Qualquer polinômio  $P(z)$  com coeficientes complexos e de grau não-nulo, possui alguma raiz complexa.

Toda a nossa teoria sobre divisão de polinômios funciona para números complexos, uma vez que não nos foi necessário o fato de estarmos restritos a números reais. Assim, pelo Teorema do Resto, sabemos que um polinômio  $P(z)$  de grau  $n$  pode ser expresso por  $(z - z_1)Q_1(z)$ , onde  $z_1$  é sua raiz e  $Q_1(z)$  é um polinômio de coeficientes complexos de grau  $n - 1$ . O mesmo argumento se aplica a  $Q_1(z)$ , logo  $P(z) = (z - z_1)(z - z_2)Q_2(z)$ . Repetindo o mesmo processo uma quantidade finita de vezes obtemos o seguinte resultado.

**Corolário 34.4** – Qualquer polinômio de coeficientes complexos e grau  $n$  admite  $n$  raízes e pode ser expresso por  $a(z - z_1)(z - z_2) \cdots (z - z_n)$ , onde  $z_1, z_2, \dots, z_n$  são suas raízes.

### §34.2.8 Multiplicidade de uma raiz

A partir do Teorema Fundamental da Álgebra, vemos que polinômios de grau  $n$  admitem  $n$  raízes complexas, mas não necessariamente todas são distintas. Assim, nasce uma quantidade que está relacionada com a raiz do polinômio e sua fatoração chamada **multiplicidade de uma raiz**. Desse modo, uma raiz  $\alpha$  do polinômio  $P(z)$  possui multiplicidade  $k$  se  $P(z) = (z - \alpha)^k Q(z)$  e



$Q(\alpha) \neq 0$ . Por exemplo, o polinômio  $(x-3)^5(x-2)^2(x+1)$  admite  $x=3$ ,  $x=2$  e  $x=-1$  como raízes de multiplicidade 5, 2 e 1, respectivamente.

**Exemplo 34.2.9.** Sabendo que o polinômio  $x^4 + x^3 - Ax^2 - Bx - C$  admite  $x = -1$  como raiz de multiplicidade 3, determine os valores  $A, B$  e  $C$ .

Aplicando Briot-Ruffini três vezes para  $x = -1$  temos

$$\begin{array}{r|rrrr} -1 & 1 & 1 & -A & -B & -C \\ -1 & 1 & 0 & -A & A-B & -A+B-C \\ -1 & 1 & -1 & 1-A & 2A-B-1 & \\ \hline & 1 & 2 & 3-A & & \end{array}$$

Logo, pelo Corolário do Teorema do Resto, devemos ter que  $P(x)$  é divisível por  $(x+1)^3$

$$\begin{cases} -A + B - C = 0 \\ 2A - B - 1 = 0 \\ 3 - A = 0 \end{cases} \implies \begin{cases} C = -A + B \\ B = 2A - 1 \\ A = 3 \end{cases} \implies \begin{cases} C = B - A \\ B = 5 \\ A = 3 \end{cases} \implies \begin{cases} C = 2 \\ B = 5 \\ A = 3 \end{cases}$$

### §34.2.10 Raízes Racionais

**Teorema 34.5** (Teorema das Raízes Racionais) – Seja  $a_n x^n + \dots + a_1 x + a_0$  um polinômio de coeficientes inteiros e  $p/q$  uma raiz racional, onde  $\text{mdc}(p, q) = 1$ , então  $p$  é divisor de  $a_0$  e  $q$  é divisor de  $a_n$ .

Este teorema é bem útil para encontrarmos fatorações de polinômios que são de difícil manipulação. Exibiremos no próximo exemplo este fato.

**Exemplo 34.2.11.** Resolver a equação  $x^3 + 2x^2 - 5x + 2 = 0$ .

Efetuada a pesquisa de raízes racionais, temos que os possíveis valores para  $p$  são  $\{-1, 1, -2, 2\}$  e os possíveis valores para  $q$  são  $\{-1, 1\}$ . Assim,  $p/q \in \{-1, 1, -2, 2\}$  e veja que substituindo  $x = 1$  no polinômio dado vemos que esta é uma raiz. Por Briot-Ruffini, temos

$$\begin{array}{r|rrrr} 1 & 1 & 2 & -5 & 2 \\ & 1 & 3 & -2 & 0 \end{array}$$

Assim,  $x^3 + 2x^2 - 5x + 2 = (x-1)x^2 + 3x - 2$  nos resta encontrar as raízes de  $x^2 + 3x - 2$ , mas essas são facilmente encontradas e concluímos que os valores de  $x \in \mathbb{R}$  que satisfazem  $x^3 + 2x^2 - 5x + 2 = 0$  são  $1, \frac{-3 - \sqrt{17}}{2}$  e  $\frac{-3 + \sqrt{17}}{2}$ .

### §34.3 Exercícios

#### Exercícios Introdutórios

**Exercício 34.1 (UFF - RJ).** As raízes de um polinômio  $P(x)$  de grau 3 são  $r, s$  e  $t$ . Então, as raízes do polinômio  $Q(x) = [P(x)]^2$  são:

- (a)  $r^2, s^2, t^2$       (b)  $2r, 2s, 2t$       (c)  $r, s, t$       (d)  $r/2, s/2, t/2$       (e)  $r\sqrt{2}, s\sqrt{2}, t\sqrt{2}$

**Exercício 34.2.** O polinômio  $P(x)$ , quando dividido por  $x^2 + x + 1$ , fornece o quociente  $x + 1$  e o resto  $x\sqrt{1}$ . O coeficiente do termo do primeiro grau no polinômio  $P(x)$  é igual a quanto ?

**Exercício 34.3.** Determine os valores de  $a, b$  e  $c$  para que tenhamos a igualdade entre os polinômios  $(2a + b)x^3 + (a - b)x^2 + (b + c)x + c/2 - 3$  e  $4x^3 - 10x^2 + 2x - 6$ .

**Exercício 34.4.** Dados os polinômios  $F(x) = -3x^3 + 2x^2 + 2$  e  $G(x) = x^4 - x^3 + x^2 - x + 1$ , determine:

- (a)  $F(x) + G(x)$       (b)  $F(x) - 3G(x)$       (c)  $F(x) \cdot G(x)$       (d)  $G(x) + 3x \cdot F(x)$

**Exercício 34.5.** Sejam  $Q(x)$  e  $R(x)$  o resto da divisão de  $2x^4 + x^3 - x^2 + 2x - 1$  por  $x^3 - x^2 + 1$ . Obtenha o resto da divisão  $R(x)$  por  $Q(x)$ .

#### Exercícios de Aprofundamento

**Exercício 34.6.** A equação  $x^5 - 3x^4 + cx^3 + (3a + 8b)x^2 + 3(ab + 6)x + a + 2b - 1 = 0$  admite  $x = 1$  como raiz, zero como raiz de multiplicidade 2 e duas outras duas raízes reais. Determine  $a, b$  e  $c$ .

**Exercício 34.7 (PUC - PR).** Se o polinômio  $x^4 + px^2 + q$  é divisível pelo polinômio  $x^2 - 6x + 5$ , então  $p + q$  vale:

- (a)  $-1$       (b)  $3$       (c)  $5$       (d)  $-4$       (e)  $10$

**Exercício 34.8.** A razão harmônica entre 3 números  $a, b$  e  $c$  é dada por  $H = \frac{3}{1/a + 1/b + 1/c}$ . Desse modo, a razão harmônica das três raízes reais do polinômio  $x^3 - 3x^2 - 60x + 100$  é igual a quanto ?

**Exercício 34.9 (IME - 2016).** O polinômio  $x^3 + ax^2 + bx + c$  possui raízes reais  $r, -r$  e  $\frac{1}{r}$ . Portanto, o valor da soma  $b + c^2 + ac + \frac{b}{c^2}$  é igual a quanto ?

**Exercício 34.10 (IME - 2020).** Um polinômio  $P(x)$  de grau maior que 3 quando dividido por  $x - 2, x - 3$  e  $x - 5$  deixa restos 2, 3 e 5, respectivamente. O resto da divisão de  $P(x)$  por  $(x - 2)(x - 3)(x - 5)$  é:

- (a)  $1$       (b)  $x$       (c)  $30$       (d)  $x - 1$       (e)  $x - 30$

**Exercício 34.11 (ITA - 2015).** Considere o polinômio  $p$  dado por  $p(x) = 2x^3 + ax^2 + bx - 16$ , com  $a, b \in \mathbb{R}$ . Sabendo-se que  $p$  admite raiz dupla e que 2 é uma raiz de  $p$ , então o valor de  $b - a$  é igual a:

- (a)  $-36$       (b)  $-12$       (c)  $6$       (d)  $12$       (e)  $24$

#### Exercícios Avançados

**Exercício 34.12.** Considere o polinômio  $f(x) = x^3 + px^2 + qx + r$ , onde  $p, q, r \in \mathbb{R}$ . Suponha que  $f(x)$  possui 3 raízes reais.

- (i) Mostre que se as raízes estão em *Progressão Aritmética*, então vale  $2p^3 + 27r = 9pq$ .  
(ii) Mostre que se as raízes estão em *Progressão Geométrica*, então vale  $rp^3 = q^3$ .

**Exercício 34.13 (Canadá).** Seja  $f(x)$  um polinômio de grau  $n$ , tal que  $f(k) = k/(k + 1)$ , para todo inteiro  $0 \leq n$ . Calcule  $f(n + 1)$  para todo inteiro positivo  $n$ .

**Exercício 34.14 (Moldávia - 2000).** Os números inteiros  $a, b$  e  $c$  satisfazem à relação  $a + b + c = 0$ . Mostre que o número  $2(a^4 + b^4 + c^4)$  é um quadrado perfeito.

**Exercício 34.15.** Seja  $p(x) = x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0$  um polinômio de coeficientes reais satisfazendo  $a_{n-2}^2 < 2a_{n-2}$ . Mostre que  $p$  tem pelo menos duas raízes que são complexas não reais.

## §35.1 Definições de Potenciação e Radiciação

dri

**Definição 35.1.** Considerando um número real  $a$  e um número inteiro  $n \geq 1$  chamamos de potência de *base*  $a$  e *expoente*  $n$  ao número  $a^n = \underbrace{a \cdot a \cdots a}_{n \text{ fatores}}$ . Ou seja, podemos definir a potência

$a^n$  a partir da recorrência  $a^n = a \cdot a^{n-1}$  para todo inteiro  $n \geq 2$ .

Veja que dessa definição temos que se  $n$  e  $m$  são inteiros maiores que 1 e  $a, b \in \mathbb{R}$ , então

$$\begin{aligned} a^{n+m} &= \underbrace{a \cdot a \cdots a}_{n+m \text{ fatores}} = \underbrace{a \cdot a \cdots a}_{n \text{ fatores}} \cdot \underbrace{a \cdot a \cdots a}_{m \text{ fatores}} = a^n \cdot a^m \\ (a \cdot b)^n &= \underbrace{(a \cdot b) \cdots (a \cdot b)}_{n \text{ fatores}} = \underbrace{a \cdots a}_{n \text{ fatores}} \underbrace{b \cdots b}_{n \text{ fatores}} = a^n \cdot b^n \\ (a^n)^m &= \underbrace{a^n \cdots a^n}_{m \text{ fatores}} = \underbrace{\underbrace{a \cdots a}_{n \text{ fatores}} \cdots \underbrace{a \cdots a}_{n \text{ fatores}}}_{m \cdot n \text{ fatores}} = a^{n \cdot m} \end{aligned}$$

e se  $n \geq m$  e  $a \neq 0$ , então  $\frac{a^n}{a^m} = \frac{a^{m+(n-m)}}{a^m} = \frac{a^m \cdot a^{n-m}}{a^m} = a^{n-m}$ . Vamos resumir essas propriedades na seguinte proposição.

**Proposição 35.1.1** – Se  $a$  e  $b$  são números reais não nulos e  $n, m$  são inteiros positivos então valem:

(PI)  $a^{n+m} = a^n \cdot a^m$ ,

(PIII)  $(a^n)^m = a^{n \cdot m}$ ,

(PII)  $(a \cdot b)^n = a^n \cdot b^n$ ,

(PIV) Se  $n \geq m$ , então  $\frac{a^n}{a^m} = a^{n-m}$ .

**Observação 35.1.I.** Estas propriedades devem ser demonstradas por indução, mas esse nível de rigor não será necessário para o nosso estudo e as contas feitas anteriormente nos dão uma boa noção das propriedades de potências inteiras positivas.

**Exemplo 35.1.2.** Por exemplo,

$$(-2)^3 = (-2) \cdot (-2) \cdot (-2) = -8$$

$$\left(\frac{3}{2}\right)^5 = \left(3 \cdot \frac{1}{2}\right)^5 = 3^5 \cdot \frac{1}{2^5} = \frac{243}{32}$$

E se  $a$  e  $b$  são reais não nulos, então vale

$$(a^4 \cdot b^3)^3 \cdot (a^2 \cdot b)^2 = (a^4)^3 \cdot b^9 \cdot (a^2)^2 \cdot b^2 = a^{12} \cdot b^9 \cdot a^4 \cdot b^2 = a^{16} \cdot b^{11}$$

$$\left(\frac{a^4 \cdot b^3}{a^2 \cdot b}\right)^5 = \left(\frac{a^2 \cdot a^2 \cdot b^2 \cdot b}{a^2 \cdot b}\right)^5 = (a^2 \cdot b^2)^5 = (a^2)^5 \cdot (b^2)^5 = a^{10} \cdot b^{10}$$

**Definição 35.2.** Para todo número real  $a$  e inteiro positivo  $n$  definimos  $a^{-n}$  como o inverso multiplicativo de  $a^n$ , isto é,  $a^{-n} = \frac{1}{a^n}$ .

Veja que  $a^{-n} = a^{(-1) \cdot n} = (a^{-1})^n$ , assim, essa definição carrega as mesmas propriedades da proposição [Proposição 35.1.1](#), ou seja, as propriedades enunciadas lá são válidas para potências inteiras negativas.

**Exemplo 35.1.3.** Por exemplo,

$$(-2)^{-3} = \frac{1}{2^3} = \frac{1}{8} \quad \text{e} \quad \left(-\frac{3}{2}\right)^{-2} = \frac{1}{\left(-\frac{3}{2}\right)^2} = \frac{1}{\frac{9}{4}} = \frac{4}{9}.$$

Além disso, se  $a$  e  $b$  são não nulos,

$$\frac{(a^3 \cdot b^{-2})^{-2}}{(a^{-4} \cdot b^3)^3} = \frac{a^{3 \cdot (-2)} \cdot b^{(-2) \cdot (-2)}}{a^{(-4) \cdot 3} \cdot b^{3 \cdot 3}} = \frac{a^{-6} \cdot b^4}{a^{-12} \cdot b^9} = a^{(-6) - (-12)} \cdot b^{4-9} = a^6 \cdot b^{-5} = \frac{a^6}{b^5}.$$

**Observação 35.1.II.** Veja que para qualquer real  $a$  não nulo vale

$$a^0 = a^{n-n} = a^n \cdot a^{-n} = \frac{a^n}{a^n} = 1.$$

O que mostra que para qualquer real não nulo temos que  $a^0 = 1$ . Note ainda que não podemos assumir  $a = 0$ , uma vez que não podemos dividir por zero. Desse modo,  $0^0$  não é um número definido.

**Definição 35.3.** Definimos a raiz  $n$ -ésima de  $a \geq 0$  pelo número  $b > 0$  tal que  $b^n = a$  e indicaremos a raiz  $n$ -ésima de  $a$  por  $\sqrt[n]{a}$ . Normalmente, o número  $a$  é chamado radicando e  $n$  é o índice. Para  $n = 2$ , vamos denotar  $\sqrt{a}$  ao invés de  $\sqrt[2]{a}$ .

Veja que segue da definição  $(\sqrt[n]{a})^n = a$  para todo  $a \geq 0$ . Como exemplo vamos expor  $\sqrt{4} = 2$ ,  $\sqrt[3]{8} = 2$ ,  $\sqrt[3]{0} = 0$  e  $\sqrt[5]{32} = 2$ .

**Observação 35.1.III.** Para não causar confusões ao leitor vamos deixar claro que  $\sqrt{36} = 6$  e não é verdade que  $\sqrt{36} = \pm 6$ . Porém, as sentenças  $-\sqrt[3]{8} = -2$ ,  $-\sqrt{4} = -2$  e  $\pm\sqrt{9} = \pm 3$  são verdadeiras. Para não confundirmos, vamos por  $\sqrt{a^2} = |a|$ , assim,  $\sqrt{(-4)^2} = |-4| = 4$ .

**Definição 35.4.** Para  $p/q \in \mathbb{Q}$ , vamos definir para todo  $a \geq 0$  a potência de expoente racional  $a^{p/q} = \sqrt[q]{a^p}$ .

Há algumas propriedades para raízes  $n$ -ésimas que vamos enunciar na proposição a seguir.

**Proposição 35.1.4** – Sejam reais  $a, b \geq 0$  e os inteiros  $n, m, p \geq 1$ , então valem:

$$(R-I) \quad \sqrt[n]{a^{mp}} = \sqrt[n]{a^m},$$

$$(R-III) \quad \sqrt[n]{a^m} = (\sqrt[n]{a})^m,$$

$$(R-II) \quad \sqrt[n]{a \cdot b} = \sqrt[n]{a} \cdot \sqrt[n]{b},$$

$$(R-IV) \quad \sqrt[n]{\sqrt[m]{a}} = \sqrt[n \cdot m]{a}.$$

*Demonstração.* Cada item é facilmente verificado. Para expor essas ideias vamos demonstrar os itens (R-I) e (R-II). Para o primeiro item veja que  $\sqrt[p^m]{a^{pn}} = a^{pn/pm} = a^{n/m} = \sqrt[m]{a^n}$ . Já para o segundo item, chamemos  $x = \sqrt[n]{a} \cdot \sqrt[n]{b}$ , assim,  $x^n = (\sqrt[n]{a} \cdot \sqrt[n]{b})^n = (\sqrt[n]{a})^n \cdot (\sqrt[n]{b})^n = a \cdot b$ . Logo, pela definição temos  $\sqrt[n]{a \cdot b} = x = \sqrt[n]{a} \cdot \sqrt[n]{b}$ . Outros itens são deixados para verificação pelo leitor.  $\square$

**Exemplo 35.1.5.** Por exemplo,

$$\begin{aligned}\sqrt{8} &= \sqrt{2^3} = \sqrt{2^2 \cdot 2} = \sqrt{2^2} \cdot \sqrt{2} = 2\sqrt{2}, \\ \sqrt[3]{27} &= \sqrt[3]{2^6 \cdot 2} = \sqrt[3]{2^6} \cdot \sqrt[3]{2} = 2^2 \cdot \sqrt[3]{2} = 4\sqrt[3]{2}, \\ \frac{\sqrt[3]{4}}{\sqrt[4]{2}} &= \frac{\sqrt[12]{4^4}}{\sqrt[12]{2^3}} = \sqrt[12]{\frac{3^4}{2^3}} = \sqrt[12]{\frac{2^8}{2^3}} = \sqrt[12]{2^{8-3}} = \sqrt[12]{2^5} = \sqrt[12]{32}\end{aligned}$$

Além disso, podemos agora simplificar algumas expressões algébricas como

$$\begin{aligned}\sqrt{7 + \sqrt{24}} \cdot \sqrt{7 - \sqrt{24}} &= \sqrt{(7 + \sqrt{24})(7 - \sqrt{24})} = \sqrt{7^2 - (\sqrt{24})^2} = \sqrt{49 - 24} = \sqrt{25} = 5 \\ \frac{11}{7 - \sqrt{5}} &= \frac{11}{7 - \sqrt{5}} \cdot \frac{7 + \sqrt{5}}{7 + \sqrt{5}} = \frac{11(7 + \sqrt{5})}{7^2 - (\sqrt{5})^2} = \frac{1}{4}(7 + \sqrt{5})\end{aligned}$$

**Exemplo 35.1.6 (Radicais Internos).** Mostre a igualdade  $\sqrt[4]{161 + 72\sqrt{5}} = 2 + \sqrt{5}$ .

**Solução.** Veja que se  $\sqrt{a + \sqrt{b}} = x + \sqrt{y}$ , então elevando ao quadrado nos vem que  $a + \sqrt{b} = (x^2 + y) + 2x\sqrt{y}$ . Assim, devemos ter  $\begin{cases} a = x^2 + y \\ \sqrt{b} = 2x\sqrt{y} \end{cases}$ . Elevando a segunda linha ao quadrado e substituindo a primeira segue que

$$b = 4x^2y = 4y(a - y) \iff 4y^2 - 4ay + b = 0 \iff y = \frac{a + \sqrt{a^2 - b}}{2}$$

Assim, substituindo em  $x^2 = a - y$  obtemos  $x^2 = a - y = \frac{a - \sqrt{a^2 - b}}{2} \iff x = \sqrt{\frac{a - \sqrt{a^2 - b}}{2}}$ . Portanto,  $\sqrt{a + \sqrt{b}} = \sqrt{\frac{a + \sqrt{a^2 - b}}{2}} + \sqrt{\frac{a - \sqrt{a^2 - b}}{2}}$ . Em particular, temos que

$$\sqrt{161 + 72\sqrt{5}} = \sqrt{\frac{161 + \sqrt{161^2 - 5 \cdot 72^2}}{2}} + \sqrt{\frac{161 - \sqrt{161^2 - 5 \cdot 72^2}}{2}} = \sqrt{81} + \sqrt{80} = 9 + 4\sqrt{5}$$

E ainda,  $\sqrt{9 + 4\sqrt{5}} = \sqrt{\frac{9 + \sqrt{9^2 - 5 \cdot 4^2}}{2}} + \sqrt{\frac{9 - \sqrt{9^2 - 5 \cdot 4^2}}{2}} = \sqrt{4} + \sqrt{5} = 2 + \sqrt{5}$ . Daí, podemos afirmar que

$$\sqrt[4]{161 + 72\sqrt{5}} = \sqrt{\sqrt{161 + 72\sqrt{5}}} = \sqrt{9 + 4\sqrt{5}} = 2 + \sqrt{5}$$

**Observação 35.1.IV.** O aclamado matemático indiano Srinivāsa Rāmānujan foi capaz de desenvolver várias identidades algébricas elegantes sobre radicais internos. Por exemplo, em um dos seus problemas propostos à revista *Journal of the Indian Mathematical Society* envolvia a seguinte identidade

$$\begin{aligned}
3 &= \sqrt{9} = \sqrt{1+8} = \sqrt{1+2 \cdot 4} = \sqrt{1+2\sqrt{16}} = \sqrt{1+2\sqrt{1+15}} = \sqrt{1+2\sqrt{1+3 \cdot 5}} \\
&= \sqrt{1+2\sqrt{1+3\sqrt{25}}} = \sqrt{1+2\sqrt{1+3\sqrt{1+24}}} = \sqrt{1+2\sqrt{1+3\sqrt{1+4 \cdot 6}}} = \dots
\end{aligned}$$

Podemos ainda definir potências de expoentes irracionais. Veja que podemos aproximar números reais por números racionais, por exemplo, sabemos que  $3, 3.1, 3.14, 3.141, 3.1415, 3.14159, \dots$  é uma sequência de números racionais que se aproxima de  $\pi \approx 3.14159265359 \dots$ . Assim, podemos definir  $2^\pi$  pelo número que é aproximado por

$$\begin{aligned}
2^3 &= 8 \\
2^{3.1} &= 8.57418770029 \dots \\
2^{3.14} &= 8.81524092701 \dots \\
2^{3.141} &= 8.82135330455 \dots \\
2^{3.1415} &= 8.82441108248 \dots \\
2^{3.14159} &= 8.82496159506 \dots \\
2^{3.141592} &= 8.82497382906 \dots \\
&\vdots \\
2^\pi &= 8.82497782708 \dots
\end{aligned}$$

Como já temos as propriedades garantidas para números racionais, podemos estendê-las para números reais, ou seja, vale a seguinte proposição.

**Proposição 35.1.7** – Considere os números reais  $a, b, c \in \mathbb{R}$  onde  $a > 0$ , então valem:

$$(P-I) \quad a^{b+c} = a^b \cdot a^c$$

$$(P-III) \quad (a \cdot b)^c = a^c \cdot b^c$$

$$(P-II) \quad \frac{a^b}{a^c} = a^{b-c}$$

$$(P-IV) \quad (a^b)^c = a^{b \cdot c}$$

**Exemplo 35.1.8.** Note que agora podemos manipular com números do tipo

$$3 \cdot 2^{\sqrt{3}} \cdot 2^{-\sqrt{3}} = 3 \cdot 2^{\sqrt{3}-\sqrt{3}} = 3 \cdot 2^0 = 3$$

$$(3^{\sqrt{2}-1})^{\sqrt{2}+1} = 3^{(\sqrt{2}-1)(\sqrt{2}+1)} = 3^{(\sqrt{2})^2-1^2} = 3^{2-1} = 3$$

$$\left( \frac{4^{\sqrt{5}}}{8^{\sqrt{20}}} \right)^{-1/\sqrt{5}} = \frac{(4^{\sqrt{5}})^{-1/\sqrt{5}}}{(8^{\sqrt{20}})^{-1/\sqrt{5}}} = \frac{4^{-1}}{8^{-\sqrt{4}}} = \frac{4^{-1}}{(2 \cdot 4)^{-\sqrt{4}}} = \frac{4^{-1}}{(2 \cdot 4)^{-2}} = \frac{1}{2^{-2}} \cdot 4^{-1-(-2)} = 2^2 \cdot 4 = 16$$

## §35.2 Funções Exponenciais

**Definição 35.5.** Dado um número real  $a > 0$  tal que  $a \neq 1$ , chamamos de função exponencial de base  $a$  a função  $f: \mathbb{R} \rightarrow \mathbb{R}$  dada por  $f(x) = a^x$ .

Usando a [Proposição 35.1.7](#) nós temos que para qualquer função  $f(x) = a^x$  satisfaz para quaisquer  $x, y \in \mathbb{R}$

$$f(x+y) = a^{x+y} = a^x \cdot a^y = f(x) \cdot f(y)$$

Disto segue que

$$f(x) = f\left(\frac{x}{2} + \frac{x}{2}\right) = f\left(\frac{x}{2}\right) \cdot f\left(\frac{x}{2}\right) = \left[f\left(\frac{x}{2}\right)\right]^2 \geq 0$$

e veja que nenhuma função exponencial possui zeros pois se  $f(x_0) = 0$ , então para qualquer  $x \in \mathbb{R}$  teremos

$$f(x) = f(x_0 + (x - x_0)) = f(x_0) \cdot f(x - x_0) = 0 \cdot f(x - x_0) = 0$$

ou seja, a função é constante igual a zero, o que não pode ocorrer. Desse modo, acabamos de a proposição a seguir

**Proposição 35.2.1** – Seja  $a > 0$  e considere a função  $f(x) = a^x$ . Assim,  $f$  é uma função exponencial que não admite zeros e ainda  $f(x)$  é positivo para todo  $x \in \mathbb{R}$ .

**Exemplo 35.2.2** (O comportamento de uma função exponencial). Para ter uma ideia do comportamento de uma função exponencial considere que você está na extremidade de uma rua que mede 1 km. A cada hora você anda metade da distância que falta até a outra extremidade. Assim, na primeira hora você anda 500 m, na segunda hora você anda 250 m, na terceira 125 m e assim por diante. Como a cada hora você fica mais próximo da outra extremidade, então você caminhará distâncias menores a cada vez que as horas aumentam. Veja que este cenário é a representação da função  $f(t) = 1000(1/2)^t$ , onde  $t$  representa as horas passadas e  $f(t)$  o quanto que você precisa caminhar. Isto nada mais é que dizer que esta é uma função decrescente !

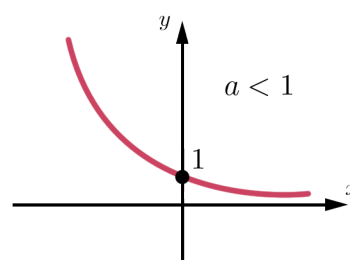
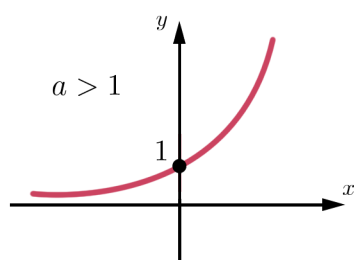
Suponha que você é um organizador de uma festa e convidou dois amigos dizendo que cada um deles podem levar outros dois amigos como convidados. Estes amigos convidaram 4 pessoas repetindo a mesma informação. Assim, Cada uma dessas 4 pessoas convidaram 2 outras pessoas e estas convidaram mais 2 e assim por diante. Ou seja, a quantidade de pessoas dobra a cada convite feito, pois no começo existia apenas seus 2 amigos, depois 4, depois 8, 16, 32 etc. Observe que isto pode ser representado matematicamente pela função  $f(n) = 2^n$ , onde  $n$  foi a quantidade de convites feitos e  $f(n)$  é a quantidade pessoas convidadas após  $n$  convites. Pelo nosso argumento, vemos que essa é uma função crescente !

Podemos generalizar o argumento feito no [Exemplo 35.2.2](#) para uma função  $f(x) = a^x$  qualquer dividindo em dois casos:  $a < 1$  ou  $a > 1$ .

**Proposição 35.2.3** – Considerando a função exponencial  $f(x) = a^x$ . Podemos afirmar que:

- a) Se  $a < 1$ , então  $f(x) = a^x$  é decrescente;
- b) Se  $a > 1$ , então  $f(x) = a^x$  é crescente.

Note que a natureza da função exponencial se traduz em velocidade, então podemos esboçar o gráfico de funções exponenciais facilmente.



Vale ressaltar que os gráficos de funções exponenciais são todos parecidos, o que muda é apenas a sua velocidade de crescimento ou decrescimento e isto está diretamente ligado com o número que aparece na base da exponencial.



### §35.2.4 Equações Exponenciais

Podemos mostrar que funções exponenciais são injetivas sobre  $\mathbb{R}$  e sobrejetivas em  $\mathbb{R}_+$  isso é bastante importante para resolvermos equações exponenciais. A ideia geral para resolvermos esses tipos de equações é sempre tentarmos expressar a equação sobre uma mesma base, pois assim teremos  $a^b = a^c \iff b = c$ .

**Exemplo 35.2.5.** Para resolvermos  $8^x = 1/32$  basta notar que  $1/32 = 32^{-1} = (2^5)^{-1} = 2^{-5}$  e  $8^x = (2^3)^x = 2^{3x}$ , assim, a equação se torna  $2^{3x} = 2^{-5}$ . Desse modo, temos que  $3x = -5$  e portanto a única solução da equação dada é  $x = -5/3$ .

De um modo parecido podemos resolver a equação  $\sqrt{5^{x-2}} \cdot \sqrt[3]{25^{2x-5}} - \sqrt[2]{5^{3x-2}} = 0$ . De fato, Note que

$$\begin{aligned} \sqrt{5^{x-2}} \cdot \sqrt[3]{25^{2x-5}} &= \sqrt[2]{5^{3x-2}} \iff 5^{\frac{x-2}{2}} \cdot (5^2)^{\frac{2x-5}{3}} = 5^{\frac{3x-2}{2}} \\ &\iff 5^{\frac{x-2}{2} + 2 \cdot \frac{2x-5}{3}} = 5^{\frac{3x-2}{2}} \\ &\iff \frac{x-2}{2} + 2 \cdot \frac{2x-5}{3} = \frac{3x-2}{2} \\ &\iff x^2 + 3x - 18 = 0 \end{aligned}$$

Desse modo, as soluções procuradas são  $x = 3$  e  $x = -6$ .

**Exemplo 35.2.6.** Após um longo estudo de 2023, foi descoberto que a população da cidade de Maçapiraca é dada pela função exponencial  $f(t) = 1000 \cdot 2^{0.2t}$ , onde  $t$  denota quantos anos se passaram desde o início do estudo.

- (i) Qual a quantidade da população em 2023 de Maçapiraca ?
- (ii) Haverá quantas pessoas nesta cidade após 10 anos ?
- (iii) Após quantos anos a população será de 64000 pessoas ?

**Solução :** (i) Ora, a população inicial desse estudo é dada quando decorrem 0 anos, logo, a quantidade procurada é  $f(0) = 1000 \cdot 2^{0.2 \cdot 0} = 1000 \cdot 2^0 = 1000$ .

- (ii) Após 10 anos teremos uma população de  $f(10) = 1000 \cdot 2^{0.2 \cdot 10} = 1000 \cdot 2^2 = 4000$  habitantes.
- (iii) Veja que basta encontrarmos a solução da equação  $f(t) = 64000$ , ou seja,

$$1000 \cdot 2^{0.2t} = 64000 \implies 2^{0.2t} = 2^6 \implies 0.2t = 6 \implies t = 30$$

**Exemplo 35.2.7.** Há outros truques algébricos que podemos usar para resolvermos equações exponenciais. Uma destas é a substituição. Vejamos algumas aplicações desta técnica.

**Aplicação 1** — A única solução da equação  $4^x + 4 \cdot 2^x = 5$  é  $x = 0$ .

Veja que  $4^x = (2^x)^2$ , assim a equação se transforma em  $(2^x)^2 + 4 \cdot 2^x = 5$  o que nos indica a considerarmos a substituição  $y = 2^x$ . Veja que agora temos  $y^2 + 4y = 5$ , onde as solução são  $y = 1$  ou  $y = -5$ . Mas lembre que não podemos ter imagens negativas em funções

exponenciais, então  $2^x = -5$  não possui soluções. Já de  $2^x = 1$  obtemos  $x = 0$ , uma vez que  $1 = 2^0$ .

**Aplicação 2** — A única solução da equação  $25^{\sqrt{x}} + 124 \cdot 5^{\sqrt{x}} = 125$  é  $x = 9$ .

Ora,  $25^{\sqrt{x}} = (5^{\sqrt{x}})^2$  e isto nos induz a considerar  $y = 5^{\sqrt{x}}$ . Desse modo, a equação inicial se transforma em  $y^2 - 124y = 125$ . Usando Bhaskara, podemos encontrar  $y = -1$  ou  $y = 125$ . Novamente, excluiremos a opção  $y = -1$ . Já de  $y = 125$  nós obtemos  $5^{\sqrt{x}} = 5^3$ , logo  $\sqrt{x} = 3$ , ou seja,  $x = 9$ .

**Exemplo 35.2.8.** Podemos ainda resolver sistemas de equações exponenciais. Considere o sistema dado por

$$\begin{cases} 2^x \cdot 4^y = \frac{1}{2} \\ 7^{x+y} = 1 \end{cases}$$

Na primeira linha temos  $2^x \cdot 4^y = 2^{-1}$ , ou ainda,  $2^{x+2y} = 2^{-1}$  e assim  $x + 2y = -1$ . Já na segunda linha temos  $7^{x+y} = 7^0$ , assim,  $x + y = 0$ . Juntando essas informações encontramos o seguinte sistema de equações

$$\begin{cases} x + 2y = -1 \\ x + y = 0 \end{cases}$$

Subtraindo a primeira da segunda linha nós temos  $y = -1$  e substituindo isto em  $x + y = 0$  concluímos que  $x = 1$ .

## §35.3 Exercícios

### Exercícios Introdutórios

**Exercício 35.1.** Se  $a = (1/4)^{-2} + (1/3)^{-2}$  e  $b = \frac{2 \cdot (1/3)^{-1} - 2^2}{(1/2)^{-2}}$ , determine o valor de  $a^b$ .

**Exercício 35.2.** Simplifique:

i)  $\frac{2^{n+4} - 2 \cdot 2^n}{2 \cdot 2^{n+3}}$

iii)  $\sqrt{\frac{2+\sqrt{3}}{2-\sqrt{3}}} + \sqrt{\frac{2-\sqrt{3}}{2+\sqrt{3}}}$

ii)  $\sqrt{7+\sqrt{24}} \cdot \sqrt{7-\sqrt{24}}$

iv)  $\frac{2+\sqrt{3}}{\sqrt{2}+\sqrt{2+\sqrt{3}}} + \frac{2-\sqrt{3}}{\sqrt{2}-\sqrt{2-\sqrt{3}}}$

**Exercício 35.3.** Os técnicos de um laboratório observaram que uma população de certo tipo de bactérias cresce segundo a função  $B(t) = 10^9 \cdot 4^{3t}$ , com  $t$  sendo medido em horas. Qual o tempo necessário para que ocorra uma reprodução de  $6.4 \cdot 10^{10}$  bactérias?

**Exercício 35.4 (FUVEST-SP-Modificada).** Seja  $f(x) = 2^{2x+1}$ . Se  $a$  e  $b$  são tais que  $f(a) = 4f(b)$ , pode-se afirmar que  $a - b$  é igual a quanto?

**Exercício 35.5.** Determine a quantidade de soluções da equação  $3^{2x^2-7x+5} = 1$ .

### Exercícios de Aprofundamento

**Exercício 35.6.** Resolva as seguintes equações exponenciais:

i)  $2^{3x-1} = 32$

vii)  $4^{x^2+4x} = 2^{24}$

ii)  $7^{4x+3} = 49$

viii)  $\sqrt{8^{x-1}} \cdot {}^{x+1}\sqrt{4^{2x-3}} = \sqrt[6]{2^{5x+3}}$

iii)  $11^{2x+5} = 1$

iv)  $7^{3x+4} = 49^{2x-3}$

ix)  ${}^{x-1}\sqrt[3]{2^{3x-1}} - {}^{3x-7}\sqrt{8^{x-3}}$

v)  $(\sqrt{2})^{3x-1} = (\sqrt[3]{16})^{2x-1}$

x)  $\frac{3^{x+2} \cdot 9^x}{243^{5x+1}} = \frac{81^{2x}}{27^{3-4x}}$

vi)  $8^{2x+1} = \sqrt[3]{4^{x-1}}$

**Exercício 35.7.** Resolva a equação  $4^x + 6^x = 2 \cdot 9^x$ .

**Exercício 35.8.** Determine a soma dos valores de  $x$  que satisfazem a equação  $25^x - 24 \cdot 5^x - 25 = 0$ .

**Exercício 35.9.** Sejam  $a, b$  e  $c$  números reais positivos tais que  $a^b = 343$ ,  $b^c = 10$  e  $a^c = 7$ . Quanto vale  $b^b$ ?

**Exercício 35.10.** Mostre que  $\sqrt[3]{9(\sqrt[3]{2}-1)} = 1 - \sqrt[3]{2} + \sqrt[3]{4}$ .

### Exercícios Avançados

**Exercício 35.11 (EsPCEX 2022).** Ao resolver a equação  $\frac{0.2^{x+0.5}}{5} = \sqrt[3]{5} \cdot 0.04^{x-2}$ , encontra-se um valor de  $x$  compreendido entre

(a) 1 e 2

(b) 2 e 3

(c) 3 e 4

(d) 4 e 5

(e) 5 e 6

**Exercício 35.12 (IME-2023).** Seja a equação  $\frac{144^x + 324^x}{64^x + 729^x} = \frac{6}{7}$ . A soma dos módulos das soluções reais desta equação é

(a) 1

(b) 2

(c) 3

(d) 8

(e) 9

**Exercício 35.13 (ITA-2013-Modificada).** A soma de todos os valores de  $x$  que satisfazem a equação  $8\sqrt{x+1} + 44(2\sqrt{x+1}) + 64 = 19(4\sqrt{x+1})$  é igual a quanto?

# 36

## Logaritmos

### §36.1 Definição de Logaritmo

**Definição 36.1.** Dados  $a \neq 1$  e  $b$  números reais positivos, chamamos de **logaritmo de  $b$  na base  $a$**  o número  $x = \log_a(b)$  tal que  $a^x = b$ .

Veja que com o nosso conhecimento sobre funções exponenciais éramos capazes de resolver equações como  $3^x = 9$  ou  $49^y = 7$  e o método para resolução era apenas fazer manipulações para deixarmos ambos os lados da igualdade com a mesma base. Agora, com o logaritmo em mãos somos capazes de dizer, por exemplo, que a solução de  $3^x = 2$  é  $x = \log_2(3)$ , mas infelizmente não conseguimos ainda dizer quanto esse número vale.

**Exemplo 36.1.1.** Vamos calcular alguns valores de logaritmos.

- $6^2 = 36 \iff \log_6(36) = 2$
- $3^{-2} = 1/9 \iff \log_3(1/9) = -2$
- $2023^0 = 1 \iff \log_{2023}(1) = 0$
- $(\sqrt[3]{12})^3 = 12 \iff \log_{\sqrt[3]{12}}(12) = 3$

Para exibir um exemplo um pouco mais complicado, podemos encontrar o valor de  $\log_{1/8}((\sqrt{2})^3)$  usando apenas a definição de logaritmo. Lembre que se  $x = \log_{1/8}((\sqrt{2})^3)$ , então temos

$$\left(\frac{1}{8}\right)^x = (\sqrt{2})^3 \implies \left(\frac{1}{2^3}\right)^x = 2^{3/2} \implies 2^{-3x} = 2^{3/2} \implies -3x = \frac{3}{2} \implies x = -\frac{1}{2}$$

Portanto, acabamos de mostrar que  $\log_{1/8}((\sqrt{2})^3) = -1/2$ .

**Exemplo 36.1.2** (Não se assuste com o logaritmo!). Vamos calcular quanto vale

$$\left(2^{\log_3(5)} - 5^{\log_3(2)}\right)^{2023}.$$

Chamemos de  $x = \log_3(5)$  e  $y = \log_3(2)$ , assim usando a definição nos vem que  $3^x = 5$  e  $3^y = 2$ . Veja que se elevarmos a primeira igualdade por  $y$  e a segunda por  $x$  obtemos a seguinte relação

$$5^{\log_3(2)} = 5^y = (3^x)^y = 3^{xy} = (3^y)^x = 2^x = 2^{\log_3(5)}.$$

Portanto, segue que  $\left(2^{\log_3(5)} - 5^{\log_3(2)}\right)^{2023} = 0$ .

### §36.2 Propriedades dos Logaritmos

Note que pela definição e nossos conhecimentos sobre funções exponenciais podemos deduzir algumas propriedades básicas sobre os logaritmos.

**Proposição 36.2.1** – Suponha que  $a \neq 1$ ,  $b$  e  $c$  são um números reais positivos. Então valem:

- (a)  $\log_a(1) = 0$ ; (b)  $\log_a(a) = 1$ ; (c)  $a^{\log_a b} = b$ ;  
 (d)  $\log_a(b \cdot c) = \log_a(b) + \log_a(c)$ ; (e)  $\log_a(b/c) = \log_a(b) - \log_a(c)$ ;  
 (f) Se  $k \in \mathbb{R}$ , então  $\log_a(b^k) = k \cdot \log_a(b)$ .  
 (g) Se  $k \in \mathbb{R} - \{0\}$ , então  $\log_{(a^k)}(b) = \frac{1}{k} \log_a(b)$ .

*Demonstração.* (a) Observe que  $x = \log_a 1$  é o mesmo que  $a^x = 1 = a^0$ , assim,  $x = 0$ .

(b) Se  $y = \log_a(a)$ , então  $a^y = a = a^1$ , ou seja,  $y = 1$ .

(c) Isto é imediato da definição pois  $z = \log_a(b)$  é justamente o número que é solução da equação  $a^z = b$ .

(d) Escrevendo  $x = \log_a(b)$  e  $y = \log_a(c)$  temos que  $a^x = b$  e  $a^y = c$ , então  $bc = a^x \cdot a^y = a^{x+y}$ . Por outro lado, a solução da equação  $a^{x+y} = bc$  é o logaritmo  $x + y = \log_a(bc)$ . Portanto, acabamos de mostrar que

$$\log_a(bc) = x + y = \log_a(b) + \log_a(c).$$

(e) Pelo item anterior veja que temos

$$\log_a(b) = \log_a((b/c)c) = \log_a(b/c) + \log_a(c) \implies \log_a(b/c) = \log_a(b) - \log_a(c).$$

(f) Sejam  $x = \log_a(b)$  e  $y = \log_a(b^k)$ . Assim, por definição,  $a^x = b$  e  $a^y = b^k$ . Elevando a primeira das igualdades por  $k$  temos que  $a^{xk} = b^k = a^y$ , o que implica que  $xk = y$ . Portanto,  $\log_a(b^k) = k \cdot \log_a(b)$ .

(g) Deixamos a prova deste item para o leitor cuidadoso. □

**Observação 36.2.1.** As propriedades citadas na [Proposição 36.2.1](#) são a alma dos logaritmos. Essas propriedades foram o impulso para a definição de logaritmo feita pelo inglês *John Napier* e o suíço *Joost Bürgi* no início do século XVII. Elas nos permitem simplificar e transformar cálculos exponenciais em operações mais simples, tornando-se ferramentas essenciais em problemas complexos. Isso é extremamente útil em situações onde precisamos lidar com números muito grandes ou muito pequenos, como em cálculos envolvendo crescimento exponencial, taxas de decaimento ou magnitude de grandezas físicas.

**Exemplo 36.2.2.** A tabela de logaritmos é uma ferramenta histórica que foi amplamente utilizada antes do advento das calculadoras e computadores para facilitar cálculos envolvendo logaritmos. Ela consiste em uma tabela que relaciona os valores dos logaritmos de diferentes números em uma determinada base.

Suponhamos que em um tábuá encontramos que  $\log_1 0(2,023) = 0,3059$ . Assim, para determinarmos  $\log_{10}(2023)$  vejamos que

$$\begin{aligned} \log_{10}(2023) &= \log_{10}(2,023 \cdot 1000) = \log_{10}(2,023) + \log_{10}(1000) \\ &= \log_{10}(2,023) + \log_{10}(10^3) = 0,3059 + 3 \log_{10}(10) = 3,3059 \end{aligned}$$

**Exemplo 36.2.3** (Não se assuste com o logaritmo! - Parte 2). Calcule

$$2019^{\log_{2019}(2020) \cdot \log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023)}.$$

Chamemos a expressão a ser calculada de  $N$  e note que pela [Proposição 36.2.1](#) temos que  $2019^{\log_{2019}(2020)} = 2020$ ,  $2020^{\log_{2020}(2021)} = 2021$ ,  $2021^{\log_{2021}(2022)} = 2022$  e  $2022^{\log_{2022}(2023)} = 2023$ . Assim,

$$\begin{aligned} N &= 2019^{\log_{2019}(2020) \cdot \log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023)} \\ &= \left(2019^{\log_{2019}(2020)}\right)^{\log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023)} \\ &= 2020^{\log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023)} \\ &= \left(2020^{\log_{2020}(2021)}\right)^{\log_{2021}(2022) \cdot \log_{2022}(2023)} \\ &= 2021^{\log_{2021}(2022) \cdot \log_{2022}(2023)} = \left(2021^{\log_{2021}(2022)}\right)^{\log_{2022}(2023)} \\ &= 2022^{\log_{2022}(2023)} = 2023 \end{aligned}$$

**Proposição 36.2.4 (Mudança de Base)** – Sejam  $a, b$  e  $c$  números reais positivos tais que  $a$  e  $c$  são diferentes de 1. Então vale

$$\log_a(b) = \frac{\log_c(b)}{\log_c(a)}.$$

*Demonstração.* Como sempre, escrevemos  $x = \log_a(b)$ ,  $y = \log_c(b)$  e  $z = \log_c(a)$ . Pela definição de logaritmo temos as igualdades  $a^x = b$ ,  $c^y = b$  e  $c^z = a$ . Por sua vez, as duas primeiras nos mostra que  $a^x = c^y$  e assim

$$c^y = a^x = (a)^x = (c^z)^x = c^{xz} \implies y = xz \implies \log_a(b) = \frac{\log_c(b)}{\log_c(a)}.$$

□

**Observação 36.2.II.** A *mudança de base* nos dá uma demonstração do último item da [Proposição 36.2.1](#). De fato,

$$\log_{a^k} b = \frac{\log_a b}{\log_a a^k} = \frac{\log_a b}{k \log_a a} = \frac{1}{k} \log_a b.$$

**Exemplo 36.2.5.** Vamos voltar a expressão calculada no [Exemplo 36.2.3](#). Veja que  $M = \log_{2019}(2020) \cdot \log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023)$  pode ser simplificada usando a mudança de base já que

$$\begin{aligned} M &= \log_{2019}(2020) \cdot \log_{2020}(2021) \cdot \log_{2021}(2022) \cdot \log_{2022}(2023) \\ &= \log_{2019}(2020) \cdot \frac{\log_{2019}(2021)}{\log_{2019}(2020)} \cdot \log_{2021}(2022) \cdot \log_{2022}(2023) \end{aligned}$$

$$\begin{aligned}
 M &= \log_{2019}(2021) \cdot \frac{\log_{2019} 2022}{\log_{2019}(2021)} \cdot \log_{2022}(2023) \\
 &= \log_{2019}(2022) \cdot \frac{\log_{2019}(2023)}{\log_{2019}(2022)} = \log_{2019}(2023)
 \end{aligned}$$

Portanto, podemos afirmar que  $2019^M = 2019^{\log_{2019}(2023)} = 2023$ .

### §36.3 Equações Logarítmicas

**Exemplo 36.3.1.** Podemos nos perguntar para quais valores de  $x \in \mathbb{R}$  temos  $\log_4(x+1) = 2$ . Pela definição temos que  $x+1 = 4^2$ , ou seja,  $x = 15$ . De modo similar, podemos encontrar a solução de  $\log_y 64 = 2$ , pois pela definição devemos ter  $y > 0$  e ainda  $64 = y^2$ , logo  $y = 8$ .

**Exemplo 36.3.2.** Encontre o valor de  $x \in \mathbb{R}$  tal que  $(\log_{10} x)^2 - 5 \log_{10} x + 4 = 0$ .

Podemos fazer a substituição  $y = \log_{10} x$  para nossa equação se tornar  $y^2 - 5y + 4 = 0$ . Desse modo, as únicas soluções são  $y = 1$  e  $y = 4$ , ou seja,  $\log_{10} x = 1$  e  $\log_{10} x = 4$ . Usando a definição de logaritmo vemos que as soluções para esse problema são  $x = 10$  e  $x = 10000$ .

**Exemplo 36.3.3.** Podemos ainda resolver equações exponenciais como  $2^{2x} - 7 \cdot 2^x + 12 = 0$ . Pois a substituição  $y = 2^x$  nos leva a equação  $y^2 - 7y + 12 = 0$  cuja as soluções são  $y = 3$  e  $y = 4$ . Mas isso nos diz que  $2^x = 3$  e  $2^x = 4$ , ou seja, as soluções são  $x = \log_2 3$  e  $x = \log 24 = 2$ .

## §36.4 Exercícios

### Exercícios Introdutórios

**Exercício 36.1.** Determine o valor dos logaritmos abaixo:

- |                  |  |   |
|------------------|--|---|
| a) $\log_2(32)$  | c) $\log_2(1/2)$                       | e) $\log_2(0,5) + \log_{10}(25)$              |
| b) $\log_5(125)$ | d) $\log_{10}(\log_3(\log_{10} 1000))$ | f) $\log_{1/5} 25 + \log_7(49^{25}/\sqrt{7})$ |

**Exercício 36.2.** Se  $A = \log_7 7$ ,  $B = \log_{76} 1$ ,  $C = \log_{0,5} 8$  e  $D = \log_8 8^{-2}$ , determine  $B^A + CD$ .

**Exercício 36.3.** Sabendo que  $\log_{10} 2 \approx 0,3$  e  $\log_{10} 5 \approx 0,7$ , determine o valor aproximado de:

- |                     |                    |                      |
|---------------------|--------------------|----------------------|
| a) $\log_{10}(5/2)$ | b) $\log_{10}(20)$ | c) $\log_2 \sqrt{5}$ |
|---------------------|--------------------|----------------------|

**Exercício 36.4.** Seja  $a = \log_{10} 8$ . Mostre que vale  $\log 5 = 1 - a/3$ . [DICA: NOTE QUE  $10/2 = 5$ .]

**Exercício 36.5.** Se  $A = \log_{1/5}(16) \cdot \log_{16}(1/5)$  e  $B = 1/\log_{25}(5)$ , determine:

- |            |          |          |                 |
|------------|----------|----------|-----------------|
| a) $A + B$ | b) $A/B$ | c) $B^A$ | d) $(B - A)/33$ |
|------------|----------|----------|-----------------|

**Exercício 36.6.** Determine o valor da expressão  $\log_b x \cdot \log_x y \cdot \log_y k$ . Para  $k = 128$  e  $b = 2$  qual sera este resultado ?

### Exercícios de Aprofundamento

**Exercício 36.7.** Sendo que  $\log_3 4 = a$  e  $\log_2 5 = b$ , determine, em função de  $a$  e  $b$ , o valor de  $\log_3 5$ .

**Exercício 36.8.** Sejam  $a > 0$  e  $b = \frac{\log_{10}(\log_{10} a)}{\log_{10}(a)}$ . Simplifique a expressão  $a^b$ .

**Exercício 36.9.** Determine o valor de  $x \in \mathbb{R}$  nas equações abaixo.

- |                               |  |
|-------------------------------|--|
| a) $\log_3 x = \log_3 8$      | c) $\log_{0,03} x = -3$                  |
| b) $\log_4(8^x) = \log_2(64)$ | d) $\log_5(x) \cdot \log_{x^2-6}(5) = 1$ |

**Exercício 36.10.** Resolva as equações.

- (a)  $\log_{21}(x+2) + \log_{21}(x+6) = 1$   
 (b)  $\log_2(x-2) - \log_2(2x-7) = 1 - \log_2(x-3)$

**Exercício 36.11.** Resolva a equação  $\log_{10}(x^2) = (\log_{10} x)^2$

**Exercício 36.12.** Resolva o sistemas abaixo.

- |   |  |
|---|--|
| (a) $\begin{cases} \log_{10}(x) + \log_{10} y = \log_{10} 2 \\ x^2 + y^2 = 5 \end{cases}$ | (c) $\begin{cases} x + y^2 = 50 \\ \log_y x = 2 \end{cases}$                   |
| (b) $\begin{cases} 3^{x+y} = 729 \\ \log_{10} x + \log_{10} y = \log_{10} 8 \end{cases}$  | (d) $\begin{cases} \log_2(x+y) - \log_3(x-y) = 1 \\ x^2 - y^2 = 2 \end{cases}$ |

**Exercício 36.13.** Sejam  $a$  e  $b$  números reais satisfazendo  $a^2 + b^2 = 7ab$ . Mostre que

$$\log_{10} \left( \frac{a+b}{3} \right) = \frac{1}{2} (\log_{10} a + \log_{10} b).$$

**Exercício 36.14.** Resolva a equação

$$(\log_{\sqrt{5}} x) \sqrt{\log_x(5\sqrt{5}) + \log_{\sqrt{5}}(5\sqrt{5})} = -\sqrt{6}.$$

**Exercício 36.15.** Se  $a = \log_{10}(x - 1/x)$  e  $b = \log_{10}(x^2 + 1/x^2 + 1)$ . Calcule  $\log_{10}(x^3 - 1/x^3)$  em função de  $a$  e  $b$ .



# 37 | Matrizes

As matrizes são um conceito fundamental na matemática e possuem uma grande importância em diversas áreas do conhecimento, como a física, a engenharia, a economia, a estatística e a computação, entre outras.

As matrizes também são amplamente utilizadas na área da computação, especialmente em programação. Elas são fundamentais para a representação de imagens, para a resolução de sistemas de equações lineares e para a criação de algoritmos de processamento de dados. Além disso, as matrizes são frequentemente utilizadas em aprendizado de máquina e inteligência artificial, como uma forma de representar dados complexos de forma mais simples e eficiente.

**Definição 37.1.** Chamamos uma **matriz de ordem  $m \times n$**  qualquer tabela de  $m \cdot n$  elementos dispostas em  $m$  linhas e  $n$  colunas. Em geral, escrevemos a matriz  $M$  de ordem  $m \times n$  como

$$M = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix}$$

Podemos ainda escrever resumidamente  $M = [a_{ij}]$ , onde  $a_{ij}$  é o elemento da linha  $i$  e coluna  $j$ .

Como exemplos de matrizes temos

$$A = \begin{bmatrix} 1 & 3 & -5 \\ 0 & 2 & 3 \end{bmatrix}_{2 \times 3}, \quad B = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}_{3 \times 3}, \quad C = [2023 \quad 2022 \quad 2021]_{1 \times 3}, \quad D = \begin{bmatrix} \pi \\ e^\pi \\ \pi^e \end{bmatrix}_{3 \times 1}.$$

harafterx 1

**Exemplo 37.0.1.** Vamos encontrar a matriz  $M = [a_{ij}]$  de ordem  $2 \times 3$ , onde

$$a_{ij} = \begin{cases} i + j, & \text{se } i \geq j \\ i - j, & \text{se } i < j \end{cases}.$$

Vejamos que esta matriz é da forma  $M = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$ . Assim, usando a condição dada temos

$$M = \begin{bmatrix} 1+1 & 1-2 & 1-3 \\ 2+1 & 2+2 & 2-3 \end{bmatrix} = \begin{bmatrix} 2 & -1 & -2 \\ 3 & 4 & -1 \end{bmatrix}$$

**Definição 37.2.** Dizemos que as matrizes  $A = [a_{ij}]$  e  $B = [b_{ij}]$  são iguais, quando estas possuem a mesma ordem e ainda vale  $a_{ij} = b_{ij}$  para qualquer que seja o par  $(i, j)$ .

**Exemplo 37.0.2.** Podemos procurar se existem valores para  $x$  e  $y$  tais que

$$\begin{bmatrix} 2x & 3y \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} x+1 & 2y \\ 3 & y+4 \end{bmatrix}.$$

Pela igualdade entre matrizes devemos ter as entradas de mesma posições iguais, assim, obtemos as equações  $2x = x+1$ ,  $3y = 2y$  e  $4 = y+4$ . Veja que de  $4 = y+4$  segue que  $y = 0$  e a mesma conclusão podemos obter da equação  $3y = 2y$ . Já de  $2x = x+1$  nós obtemos

$x = 1$ . Assim, pondo  $x = 1$  e  $y = 0$  obtemos a igualdade procurada.

### §37.1 Operações com matrizes

**Definição 37.3.** Dadas as matrizes  $A = [a_{ij}]$  e  $B = [b_{ij}]$  de mesma ordem, definimos a soma destas matrizes por  $A + B = [a_{ij} + b_{ij}]$ . Além disso, se  $c$  é um número real, definimos  $cA = [c \cdot a_{ij}]$ . Portanto, a soma de duas matrizes é obtida ao somarmos as entradas de mesma posição de cada uma dessas matrizes e o produto de uma matriz por um número real é apenas a matriz cuja as todas as entradas foram multiplicadas por esse número.

**Exemplo 37.1.1.** (i) Considerando as matrizes  $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$  e  $\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$ , podemos calcular sua soma do seguinte modo

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

(ii) Se  $A = \begin{bmatrix} 2 & 5 \\ -4 & 6 \end{bmatrix}$  e  $B = \frac{1}{2} \begin{bmatrix} 5 & 10 \\ -2 & -7 \end{bmatrix}$ , então a matriz  $4A + \frac{1}{2}B$  é calculada do seguinte modo

$$\begin{aligned} 4 \begin{bmatrix} 2 & 5 \\ -4 & 6 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 5 & 10 \\ -2 & -7 \end{bmatrix} &= \begin{bmatrix} 4 \cdot 2 & 4 \cdot 5 \\ 4 \cdot (-4) & 4 \cdot 6 \end{bmatrix} + \begin{bmatrix} 1/2 \cdot 5 & 1/2 \cdot 10 \\ 1/2 \cdot (-2) & 1/2 \cdot (-7) \end{bmatrix} \\ &= \begin{bmatrix} 8 & 20 \\ -16 & 24 \end{bmatrix} + \begin{bmatrix} 5/2 & 5 \\ -1 & -7/2 \end{bmatrix} = \begin{bmatrix} 21/2 & 25 \\ -17 & 55/2 \end{bmatrix} \end{aligned}$$

**Definição 37.4.** Dadas as matrizes  $A = [a_{ij}]_{m \times n}$  e  $B = [b_{ij}]_{n \times p}$ , podemos definir a matriz produto  $AB$ , pela matriz  $C = [c_{ij}]_{m \times p}$  tal que

$$c_{ij} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + a_{i3} \cdot b_{3j} + \cdots + a_{in} \cdot b_{nj} = \sum_{k=1}^n a_{ik} \cdot b_{kj}.$$

Em outras palavras, estamos definindo o elemento da linha  $i$  e coluna  $j$  da matriz produto  $AB$  como a soma de todos os produtos dos termos correspondentes da linha  $i$  de  $A$  com a coluna  $j$  de  $B$ .

Apesar da definição ser pouco intuitiva, a multiplicação de matrizes satisfaz algumas propriedades que já estamos acostumados e vamos expor estas na seguinte proposição.

**Proposição 37.1.2** – Se as matrizes  $A, B$  e  $C$  possuem ordens tais que as operações de soma e multiplicação estejam definidas, então valem:

- (i)  $A(BC) = (AB)C$
- (ii)  $A(B + C) = AB + AC$
- (iii)  $(A + B)C = AC + BC$

**Exemplo 37.1.3.** Vamos considerar as matrizes

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \text{ e } B = \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix}.$$

Como  $A$  é uma matriz  $2 \times 3$  e  $B$  é uma matriz  $3 \times 2$ , então podemos calcular a matriz produto  $AB$  e esta será de ordem  $2 \times 2$ . Escrevemos por simplicidade  $AB = [c_{ij}]_{2 \times 2}$ , onde  $c_{ij}$  é o elemento da linha  $i$  e coluna  $j$ . Para determinar, por exemplo, a entrada de linha 1 e coluna 1 de  $AB$  vamos multiplicar as entradas correspondentes da primeira linha de  $A$  pela primeira coluna de  $B$ , assim,  $c_{11} = 1 \cdot 7 + 2 \cdot 9 + 3 \cdot 11 = 58$ . Já para encontrarmos  $c_{12}$  vamos multiplicar a primeira linha de  $A$  com a segunda coluna de  $B$ , assim, obtemos  $c_{12} = 1 \cdot 8 + 2 \cdot 10 + 3 \cdot 12 = 64$ . Já para as demais entradas temos

$$c_{21} = 4 \cdot 7 + 5 \cdot 9 + 6 \cdot 11 = 139$$

$$c_{22} = 4 \cdot 8 + 5 \cdot 10 + 6 \cdot 12 = 154$$

Portanto, a matriz resultante  $AB$  é

$$AB = \begin{bmatrix} 1 \cdot 7 + 2 \cdot 9 + 3 \cdot 11 & 1 \cdot 8 + 2 \cdot 10 + 3 \cdot 12 \\ 4 \cdot 7 + 5 \cdot 9 + 6 \cdot 11 & 4 \cdot 8 + 5 \cdot 10 + 6 \cdot 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix}$$

**Exemplo 37.1.4.** A multiplicação de matrizes, em geral, não é *comutativa*, ou seja, o produto de duas matrizes depende da ordem em que elas são multiplicadas, assim, podemos ter  $AB$  e  $BA$  sendo duas matrizes diferentes. Isto talvez seja claro quando  $BA$  talvez nem faça sentido para ser calculada ou que possua uma ordem diferente. Por exemplo, se  $A$  possui ordem  $2 \times 3$  e  $B$  é de ordem  $3 \times 4$ , então  $AB$  é possível de ser calculada mas  $BA$  não, pois suas ordens são incompatíveis. Mesmo quando as matrizes  $AB$  e  $BA$  existam e são do mesmo tamanho é possível que tenhamos  $AB$  diferente de  $BA$ . De fato, considere as matrizes  $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$  e  $B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$ , assim,

$$AB = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \cdot 5 + 2 \cdot 7 & 1 \cdot 6 + 2 \cdot 8 \\ 3 \cdot 5 + 4 \cdot 7 & 3 \cdot 6 + 4 \cdot 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

e

$$BA = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 5 \cdot 1 + 6 \cdot 3 & 5 \cdot 2 + 6 \cdot 4 \\ 7 \cdot 1 + 8 \cdot 3 & 7 \cdot 2 + 8 \cdot 4 \end{bmatrix} = \begin{bmatrix} 23 & 34 \\ 31 & 46 \end{bmatrix}$$

**Definição 37.5.** Chamamos a matriz de ordem  $m \times n$  cujas as entradas são todas nulas de **matriz nula** ou **matriz zero** e denotaremos esta por  $0$ . Caso seja necessário enfatizar sua ordem, podemos escrever  $0_{m \times n}$ .

Observe que a matriz nula desempenha o mesmo papel do número zero quando estamos considerando a operação de soma, pois  $A + 0 = 0 + A = A$  para qualquer que seja a matriz  $A$ .

**Exemplo 37.1.5.** Existem várias curiosidades e peculiaridades sobre o produto de matrizes e vamos agora mostrar duas dessas. Considerando o produto de matrizes

$$\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 3 & -4 \\ -6 & 8 \end{bmatrix} = \begin{bmatrix} 4 \cdot 3 + 2 \cdot (-6) & 4 \cdot (-4) + 2 \cdot 8 \\ 2 \cdot 3 + 1 \cdot (-6) & 2 \cdot (-4) + 1 \cdot 8 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

observamos que nenhuma das matrizes operadas é a matriz nula e mesmo assim o seu produto é a matriz nula. A outra curiosidade que mostraremos é que a igualdade  $AB = AC$  não implica  $B = C$ . Recomendamos ao leitor cuidadoso que ao considerar as matrizes  $A = \begin{bmatrix} 0 & 1 \\ 0 & 2 \end{bmatrix}$ ,  $B = \begin{bmatrix} 1 & 1 \\ 3 & 4 \end{bmatrix}$  e  $C = \begin{bmatrix} 2 & 5 \\ 3 & 4 \end{bmatrix}$  conclua  $AB = AC = \begin{bmatrix} 3 & 4 \\ 6 & 8 \end{bmatrix}$ .

**Definição 37.6.** Dada a matriz  $A = [a_{ij}]$  de ordem  $m \times n$ , consideremos a matriz  $A^T = [b_{ij}]$  de ordem  $n \times m$  cujo os elementos são dados por  $b_{ij} = a_{ji}$ . Chamamos a matriz  $A^T$  de **matriz transposta** de  $A$ . Quando ocorre  $A = A^T$ , dizemos que  $A$  é uma **matriz simétrica** e quando  $A^T = -A$  dizemos que  $A$  é uma **matriz anti-simétrica**.

**Exemplo 37.1.6.** Vamos expor alguns exemplos de matrizes e suas matrizes transpostas.

(i) A matriz transposta da matriz  $\begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & -2 \\ 3 & 4 & 5 \end{bmatrix}$  é a matriz  $\begin{bmatrix} 1 & 0 & 3 \\ 2 & -1 & 4 \\ 3 & -2 & 5 \end{bmatrix}$ .

(ii) A matriz transposta da matriz  $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \end{bmatrix}$  é a matriz  $\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 2 \\ 4 & 3 \end{bmatrix}$ .

(iii) Observe que a transposta de  $\begin{bmatrix} 6 & 2 \\ 2 & 3 \end{bmatrix}$  é ela mesma, assim esta é um exemplo de matriz simétrica.

(iv) Já a transposta de  $\begin{bmatrix} 1 & -6 \\ 6 & 20 \end{bmatrix}$  é a matriz  $\begin{bmatrix} 1 & 6 \\ -6 & 20 \end{bmatrix} = -\begin{bmatrix} 1 & -6 \\ 6 & 20 \end{bmatrix}$ , logo esta é um exemplo de matriz anti-simétrica.

**Proposição 37.1.7** – Considerando as matrizes  $A$  e  $B$  e  $\lambda$  um número real, valem:

$$(i) \quad (A^T)^T = A$$

$$(iii) \quad (\lambda A)^T = \lambda A^T$$

$$(ii) \quad (A + B)^T = A^T + B^T$$

$$(iv) \quad (AB)^T = B^T A^T$$

## §37.2 A relação de matrizes com sistemas lineares

**Definição 37.7.** Vamos definir a matriz quadrada  $I = [\delta_{ij}]$  onde  $\delta_{ij} = \begin{cases} 1, & \text{se } i = j \\ 0, & \text{se } i \neq j \end{cases}$ , esta matriz é conhecida como **matriz identidade**. Caso seja necessário enfatizar seu tamanho escrevemos  $I_n$ . Alguns exemplos são

$$I_1 = [1], \quad I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \dots$$

A matriz identidade tem um papel importante no produto entre matrizes, pois é um elemento

neutro da multiplicação de matrizes. Vejamos isso considerando uma matriz  $A = [a_{ij}]_{3 \times 2}$ .

$$AI_2 = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} a_{11} \cdot 1 + a_{12} \cdot 0 & a_{11} \cdot 0 + a_{12} \cdot 1 \\ a_{21} \cdot 1 + a_{22} \cdot 0 & a_{21} \cdot 0 + a_{22} \cdot 1 \\ a_{31} \cdot 1 + a_{32} \cdot 0 & a_{31} \cdot 0 + a_{32} \cdot 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} = A$$

e por outro lado,

$$I_3 A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} = \begin{bmatrix} 1 \cdot a_{11} + 0 \cdot a_{21} + 0 \cdot a_{31} & 1 \cdot a_{12} + 0 \cdot a_{22} + 0 \cdot a_{32} \\ 0 \cdot a_{11} + 1 \cdot a_{21} + 0 \cdot a_{31} & 0 \cdot a_{12} + 1 \cdot a_{22} + 0 \cdot a_{32} \\ 0 \cdot a_{11} + 0 \cdot a_{21} + 1 \cdot a_{31} & 0 \cdot a_{12} + 0 \cdot a_{22} + 1 \cdot a_{32} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} = A$$

Essa propriedade pode sempre ser verificada, assim, se  $A$  é uma matriz de ordem  $m \times n$ , então vale

$$AI_n = A = I_m A.$$

**Exemplo 37.2.1.** Considere as matrizes  $A = \begin{bmatrix} 7 & 1 \\ 13 & 2 \end{bmatrix}$ ,  $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$  e  $b = \begin{bmatrix} 15 \\ 28 \end{bmatrix}$  e suponha que valha equação matricial  $A\mathbf{x} = b$ . Usando a definição de multiplicação de matrizes temos que

$$A\mathbf{x} = \begin{bmatrix} 7 & 1 \\ 13 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 7x + y \\ 13x + 2y \end{bmatrix}.$$

Observe que a equação  $A\mathbf{x} = b$ , por igualdade de matrizes, se torna um sistema linear pois

$$A\mathbf{x} = b \iff \begin{bmatrix} 7x + y \\ 13x + 2y \end{bmatrix} = \begin{bmatrix} 15 \\ 28 \end{bmatrix} \iff \begin{cases} 7x + y = 15 \\ 13x + 2y = 28 \end{cases}.$$

Este sistema é facilmente resolvido substituindo  $y = 15 - 7x$  na segunda equação, mas usaremos nossa teoria de matrizes para apresentar um novo método para resolver este problema. Vamos agora introduzir a matriz  $M = \begin{bmatrix} 2 & -1 \\ -13 & 7 \end{bmatrix}$ . Vejamos que

$$MA = \begin{bmatrix} 2 & -1 \\ -13 & 7 \end{bmatrix} \begin{bmatrix} 7 & 1 \\ 13 & 2 \end{bmatrix} = \begin{bmatrix} 2 \cdot 7 + (-1) \cdot 13 & 2 \cdot 1 + (-1) \cdot 2 \\ -13 \cdot 7 + 7 \cdot 13 & -13 \cdot 1 + 7 \cdot 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I.$$

Desse modo, ao multiplicarmos pela esquerda a equação matricial  $A\mathbf{x} = b$  por  $M$  temos

$$A\mathbf{x} = b \implies MA\mathbf{x} = Mb \implies I\mathbf{x} = Bb \implies \mathbf{x} = Mb.$$

Portanto a solução, desta equação é  $\mathbf{x} = Bb = \begin{bmatrix} 2 \cdot 15 + (-1) \cdot 28 \\ -13 \cdot 15 + 7 \cdot 28 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ , ou seja, as soluções desse sistema linear são  $x = 2$  e  $y = 1$ .

Utilizando as ideias expostas no [Exemplo 37.2.1](#) podemos generalizar o argumento para mostrar que qualquer sistema linear pode ser expresso como uma equação matricial, ou seja,

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \iff \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

**Definição 37.8.** Seja  $A$  uma matriz quadrada, se existir uma outra matriz  $B$  de mesma ordem tal que  $AB = BA = I$ , então diremos que  $A$  é uma **matriz invertível** e que  $B$  é a **matriz inversa** de  $A$ . Neste caso, normalmente denotamos  $B = A^{-1}$ . Caso não exista uma matriz com essa propriedade, diremos que  $A$  é uma *matriz singular* ou *matriz não invertível*.

Supondo que a matriz inversa de  $A$  exista, então esta é única. De fato, se  $B$  e  $C$  são inversas de  $A$ , então

$$B = BI = B(AC) = (BA)C = IC = C.$$

Observe que a técnica mostrada no [Exemplo 37.2.1](#) para resolver equação  $A\mathbf{x} = \mathbf{b}$  nada mais era que multiplicar esta igualdade pela esquerda pela matriz inversa de  $A$ . Veja que quando existe a inversa de  $A$  a equação  $A\mathbf{x} = \mathbf{b}$  é facilmente resolvida pois nesse caso temos,

$$A\mathbf{x} = \mathbf{b} \iff A^{-1}A\mathbf{x} = A^{-1}\mathbf{b} \iff I\mathbf{x} = A^{-1}\mathbf{b} \iff \mathbf{x} = A^{-1}\mathbf{b}$$

Neste momento ainda não é tão claro quais são as condições para que exista inversa de uma matriz, mas vamos continuar nossos estudos procurando jeitos para calcular esta matriz.

**Exemplo 37.2.2.** No caso de procurarmos a inversa de uma matriz  $2 \times 2$  as contas são relativamente curtas, por exemplo, considerando  $\begin{bmatrix} 5 & 4 \\ 6 & 5 \end{bmatrix}$  podemos procurar a sua matriz inversa buscando valores para  $x, y, z$  e  $w$  para que tenhamos  $\begin{bmatrix} 5 & 4 \\ 6 & 5 \end{bmatrix} \begin{bmatrix} x & y \\ z & w \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$  e isto se reduz a um sistema linear de 4 equações

$$\begin{cases} 5x + 4z = 1 \\ 5y + 4w = 0 \\ 6x + 5z = 0 \\ 6y + 5w = 1 \end{cases}$$

Multiplicando a segunda linha por  $-6$ , a última por  $5$  e somando estas equações obtemos

$$(-6) \cdot (5y + 4w) + 5 \cdot (6y + 5w) = (-6) \cdot 0 + 5 \cdot 1 \iff -30y - 24w + 30y + 25w = 5 \iff w = 5$$

Usando que  $w = 5$  na segunda linha obtemos  $5y = -20$ , ou seja,  $y = -4$ . Multiplicando a primeira linha por  $-6$ , a quarta por  $5$  e somando estas equações obtemos

$$(-6) \cdot (5x + 4z) + 5 \cdot (6x + 5z) = (-6) \cdot 1 + 5 \cdot 0 \iff -30x - 24z + 30x + 25z = -6 \iff z = -6$$

Usando que  $z = -6$  na quarta linha nos vem que  $x = 5$ . Portanto, acabamos de mostrar que

$$\begin{bmatrix} 5 & 4 \\ 6 & 5 \end{bmatrix}^{-1} = \begin{bmatrix} 5 & -6 \\ -4 & 5 \end{bmatrix}$$

**Observação 37.2.I.** De um modo mais geral, podemos adaptar os passos feitos no [Exemplo 37.2.2](#) para mostrar que existe inversa da matriz  $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$  se, e somente se,  $ad - bc \neq 0$  e ainda

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

**Exemplo 37.2.3.** Podemos verificar que a matriz  $A = \begin{bmatrix} 2 & 6 \\ 4 & 11 \end{bmatrix}$  satisfaz a equação  $A^2 - 13A - 2I = 0$ . Vejamos que esta equação pode ser reescrita como

$$A^2 - 13A - 2I = 0 \iff A^2 - 13A = 2I \iff A(A - 13I) = 2I \iff (A - 13I)A = 2I$$

Então podemos dizer que  $A^{-1} = \frac{1}{2}(A - 13I)$

### §37.3 Método para encontrar $A^{-1}$

Vamos definir três **operações elementares** com as linhas de uma matriz  $A$ .

1. Multiplicar uma linha por uma constante  $c$  não nula.
2. Trocar duas linhas entre si.
3. Somar a uma linha uma outra multiplicada por uma constante  $c$  não nula.

Podemos dizer que as matrizes  $A$  e  $B$  são *equivalentes por linhas* se uma delas pode ser obtida a partir da outra por meio de sequências de operações elementares com linhas.

Para fixar notações considere o seguinte exemplo. Considerando a matriz identidade  $I_2$ , multiplicando a segunda linha de  $I_2$  por  $-3$  obtemos

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \xrightarrow{(-3)L_2 \rightarrow L_2} \begin{bmatrix} 1 & 0 \\ 0 & -3 \end{bmatrix}$$

**Exemplo 37.3.1.** Considere a identidade  $I_3$  e a matriz que é resultado após somarmos 3 vezes a primeira linha na terceira, ou seja,  $B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}$ . Seja  $A = \begin{bmatrix} 1 & 0 & 2 & 3 \\ 2 & -1 & 3 & 6 \\ 1 & 4 & 4 & 0 \end{bmatrix}$  e assim

$$BA = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 3 \\ 2 & -1 & 3 & 6 \\ 1 & 4 & 4 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 3 \\ 2 & -1 & 3 & 6 \\ 4 & 4 & 10 & 9 \end{bmatrix}.$$

Note que esta última matriz é precisamente a matriz obtida ao somarmos 3 vezes a primeira linha de  $A$  na terceira linha de  $A$ .

Vamos assumir um resultado sobre matrizes invertíveis que mostraremos na próxima proposição.

**Proposição 37.3.2** – Se  $A$  é uma matriz quadrada, então as seguintes afirmações são equivalentes:

- (a)  $A$  é invertível.
- (b) A única solução de  $A\mathbf{x} = 0$  é  $\mathbf{x} = 0$ .
- (c) A matriz  $A$  é equivalente a matriz identidade.
- (d) A matriz  $A$  é o produto de matrizes elementares.

**Exemplo 37.3.3.** Estas operações são suficientes para encontrarmos a matriz inversa de  $A$ , pois podemos considerar a matriz  $[A \mid I]$  de ordem  $n \times 2n$  e vamos efetuar operações elementares com as linhas desta matriz aumentada de modo que o lado se transforma em  $I$ , assim, o lado direito se transforma em  $A^{-1}$ .

Desse modo, para encontrarmos a inversa da matriz  $A = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}$  faremos os seguintes passos

$$\begin{aligned}
 & \left[ \begin{array}{ccc|ccc} 2 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 \end{array} \right] \xrightarrow{\frac{1}{2}L_1 \rightarrow L_1} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 \end{array} \right] \\
 & \xrightarrow{L_2 - L_1 \rightarrow L_2} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 1/2 & -1/2 & 1 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 \end{array} \right] \xrightarrow{L_3 - L_1 \rightarrow L_3} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 1/2 & -1/2 & 1 & 0 \\ 0 & 1/2 & 3/2 & -1/2 & 0 & 1 \end{array} \right] \\
 & \xrightarrow{2L_2 \rightarrow L_2} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1 & 1 & -1 & 2 & 0 \\ 0 & 1/2 & 3/2 & -1/2 & 0 & 1 \end{array} \right] \xrightarrow{L_3 - \frac{1}{2}L_2 \rightarrow L_3} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1 & 1 & -1 & 2 & 0 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array} \right] \\
 & \xrightarrow{L_1 - \frac{1}{2}L_3 \rightarrow L_1} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array} \right] \xrightarrow{L_1 - \frac{1}{2}L_3 \rightarrow L_1} \left[ \begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 1/2 & -1/2 \\ 0 & 1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array} \right] \\
 & \xrightarrow{L_1 - \frac{1}{2}L_2 \rightarrow L_1} \left[ \begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array} \right]
 \end{aligned}$$

Portanto, a matriz inversa de  $A$  é dada por  $A^{-1} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 1 \end{bmatrix}$ .

O processo descrito no exemplo [Exemplo 37.3.3](#) é conhecido como **Método de Gauss-Jordan**.

**Exemplo 37.3.4.** Este método também serve para mostrar quando uma matriz não é invertível. Considere  $A = \begin{bmatrix} 1 & 3 \\ 2 & 6 \end{bmatrix}$ . Assim,

$$\left[ \begin{array}{cc|cc} 1 & 3 & 1 & 0 \\ 2 & 6 & 0 & 1 \end{array} \right] \xrightarrow{L_2 - 2L_1 \rightarrow L_2} \left[ \begin{array}{cc|cc} 1 & 3 & 1 & 0 \\ 0 & 0 & -2 & 1 \end{array} \right]$$

Então como a matriz  $A$  é equivalente a uma matriz que possui uma linha nula, logo,  $A$  não é invertível.

## §37.4 Classificação de sistemas lineares

Já sabemos que um sistema linear pode ser escrito como uma equação matricial da forma  $A\mathbf{x} = \mathbf{b}$  que envolve produto de matrizes, agora vamos considerar a matriz  $[A \mid \mathbf{b}]$ , onde  $A$  é a matriz dos coeficientes e  $\mathbf{b}$  é coluna das igualdades do sistema linear, ou seja,



$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \Rightarrow \left[ \begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array} \right]$$

Chamamos essa matriz  $[A \mid b]$  de **matriz aumentada** do sistema  $Ax = b$ . Nos exemplos anteriores, lidamos com sistemas lineares que possuíam solução única. No entanto, existem outros casos a serem considerados, nos quais o sistema pode ter infinitas soluções ou até mesmo não ter solução. Abordaremos esses outros casos nos exemplos a seguir.

**Exemplo 37.4.1.** Podemos procurar uma solução para o sistema  $\begin{cases} x - 2y - z = 1 \\ 2x + y - 3z = 0 \\ x - 7y = 3 \end{cases}$  vindo que isto é equivalente a usar apenas operações elementares na matriz aumentada

$$\left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 2 & 1 & -3 & 0 \\ 1 & -7 & 0 & 3 \end{array} \right].$$

Desse modo,

$$\begin{aligned} \left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 2 & 1 & -3 & 0 \\ 1 & -7 & 0 & 3 \end{array} \right] &\xrightarrow{L_2 - 2L_1 \rightarrow L_2} \left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 0 & 5 & -1 & -2 \\ 1 & -7 & 0 & 3 \end{array} \right] \xrightarrow{L_3 - L_1 \rightarrow L_3} \left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 0 & 5 & -1 & -2 \\ 0 & -5 & 1 & 2 \end{array} \right] \\ &\xrightarrow{L_3 + L_2 \rightarrow L_3} \left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 0 & 5 & -1 & -2 \\ 0 & 0 & 0 & 0 \end{array} \right] \xrightarrow{\frac{1}{5}L_2 \rightarrow L_2} \left[ \begin{array}{ccc|c} 1 & -2 & -1 & 1 \\ 0 & 1 & -1/5 & -2/5 \\ 0 & 0 & 0 & 0 \end{array} \right] \\ &\xrightarrow{L_1 + 2L_2 \rightarrow L_1} \left[ \begin{array}{ccc|c} 1 & 0 & -7/5 & 1/5 \\ 0 & 1 & -1/5 & -2/5 \\ 0 & 0 & 0 & 0 \end{array} \right] \end{aligned}$$

Isto significa que  $x - \frac{7}{5}z = \frac{1}{5}$  e  $y - \frac{1}{5}z = -\frac{2}{5}$ , ou seja, para qualquer  $z$  real temos que  $x = \frac{1+7z}{5}$  e  $y = \frac{z-2}{5}$  são as soluções do sistema considerado. Então neste caso temos infinitas soluções e dizemos que  $z$  é uma variável livre, pois qualquer que seja o valor de  $z$  as soluções do sistema são dadas pela tripla  $[(1+7z)/5, (z-2)/5, z]$ .

**Exemplo 37.4.2.** Considerando o sistema  $\begin{cases} 3x + 2y + 2z = -1 \\ 2x - y - z = -3 \\ x + y + z = 1 \end{cases}$ , vamos fazer operações elementares com as linhas da matriz aumentada para tentar encontrar soluções deste sistema linear. Desse modo,

$$\left[ \begin{array}{ccc|c} 3 & 2 & 2 & -1 \\ 2 & -1 & -1 & -3 \\ 1 & 1 & 1 & 1 \end{array} \right] \xrightarrow{L_1 \leftrightarrow L_3} \left[ \begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 2 & -1 & -1 & -3 \\ 3 & 2 & 2 & -1 \end{array} \right] \xrightarrow{L_2 - 2L_1 \rightarrow L_2} \left[ \begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -3 & -3 & -5 \\ 3 & 2 & 2 & -1 \end{array} \right]$$

$$\xrightarrow{L_3 - 3L_1 \rightarrow L_2} \left[ \begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -3 & -3 & -5 \\ 0 & -1 & -1 & -4 \end{array} \right] \xrightarrow{L_2 - 3L_3 \rightarrow L_2} \left[ \begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 7 \\ 0 & -1 & -1 & -4 \end{array} \right]$$

Vejamos que a segunda linha da última matriz aumentada nos diz que  $0x + 0y + 0z = 7$ , o que é um absurdo. Portanto, este sistema não possui solução.

Vejamos que nos dois últimos exemplos encontramos exemplos de sistemas lineares cuja as matrizes dos coeficientes não possuem inversa, mas em um dos casos o sistema admite infinitas soluções e no outro não possui.

Observe ainda que em todos os casos estamos usando operações elementares para obter um sistema linear que é mais simples de ser resolvido e estamos fazendo isso forçando alguns coeficientes do sistema serem iguais a zeros. O nome dado a esse processo é **escalonamento** de matrizes ou sistemas lineares. Visualmente este processo deixa a matriz aumentada com uma “escada” de valores não nulos, como por exemplo,

$$\left[ \begin{array}{cccc|c} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & * \end{array} \right], \quad \left[ \begin{array}{cccc|c} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right], \quad \left[ \begin{array}{cccc|c} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right],$$

onde \* representa uma entrada não-nula. Na verdade, os exemplos de matrizes aumentadas nos exemplos anteriores são os únicos casos possíveis após um escalonamento de um sistema.

No primeiro caso, nós conseguimos resolver a última equação. Substituindo este resultado na linha a cima encontramos a solução da penúltima linha e repetindo esse processo conseguimos resolver o sistema todo, ou seja, ele possui solução única e assim dizemos que o sistema é um *Sistema Possível e Determinado (SPD)*.

No segundo caso, nós obtemos um sistema equivalente que possui mais variáveis do que equações e assim aparecem *variáveis livres*, ou seja, encontramos soluções que vão depender dessas variáveis e portanto obtemos infinitas soluções. Quando isto ocorre, chamamos o sistema de um *Sistema Possível Indeterminado (SPI)*.

Já no terceiro caso obtemos uma linha que é uma contradição matemática, pois um lado de uma das linhas é igual a zero enquanto que o outro é um número não nulo. Daí o sistema não possui soluções e dizemos que este é um *Sistema Impossível (SI)*.

**Exemplo 37.4.3.** Vejamos como classificar o sistema  $\begin{cases} 4x + (m+1)y = 0 \\ (m-1)x + 2y = 0 \end{cases}$  conforme o parâmetro  $m$ . Note que a matriz aumentada é equivalente a

$$\left[ \begin{array}{cc|c} 4 & m+1 & 0 \\ m-1 & 2 & 0 \end{array} \right] \xrightarrow{L_2 - \frac{m-1}{4}L_1 \rightarrow L_2} \left[ \begin{array}{cc|c} 4 & m+1 & 0 \\ 0 & \frac{9-m^2}{4} & 0 \end{array} \right]$$

Assim, se  $m^2 - 9 \neq 0$  temos que o sistema é SPD pois nesse caso teríamos  $y = 0$  e substituindo na primeira equação nos vem que  $x = 0$ . Porém se  $m^2 - 9 = 0$ , ou seja, se  $m = 3$  ou  $m = -3$  nós temos que a segunda linha é toda nula e portanto obtemos  $y$  como uma variável livre, nesse caso temos que o sistema é SPI.

**Exemplo 37.4.4.** Para quais valores de  $a$  e  $b$  temos que o sistema

$$\begin{cases} x + y + z = 0 \\ x - y + az = 2 \\ 2x + y - z = b \end{cases}$$

pode ser classificado como:

- (i) Sistema Possível Determinado,                      (iii) Sistema Impossível.  
 (ii) Sistema Possível Indeterminado,

Realizando as operações  $L_2 - L_1 \rightarrow L_2$ ,  $L_3 - 2L_1 \rightarrow L_3$  e  $L_3 - \frac{1}{2}L_2 \rightarrow L_3$  obtemos o sistema

$$\begin{cases} x + y + z = 0 \\ -2y + (a - 1)z = 2 \\ \frac{-a - 5}{2}z = b - 1 \end{cases}.$$

Assim, se  $a \neq -5$  então podemos resolver a última equação e para o valor encontrado para  $z$  substituímos na segunda linha e encontramos o valor para  $y$  e portanto também podemos encontrar o valor de  $x$ . Portanto o sistema é Sistema Possível Determinado se  $a \neq -5$ .

Se  $a = -5$  e  $b = 1$  a última equação se torna totalmente nula e o nosso sistema passa a ter mais variáveis do que equações, portanto neste caso temos um sistema Sistema Possível Indeterminado.

Já se  $a = -5$  e  $b \neq 1$  temos uma contradição na última linha, portanto o sistema é classificado como Sistema Impossível.

## §37.5 Exercícios

### Exercícios Introdutórios

**Exercício 37.1.** Encontre valores para  $a, b, c$  e  $d$  de modo que tenhamos  $\begin{bmatrix} a & 1 \\ 1 & 2 \end{bmatrix} + \begin{bmatrix} 2 & b \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ c & d \end{bmatrix}$ .

**Exercício 37.2.** Considere as matrizes

$$A = \begin{bmatrix} 3 & 0 \\ -1 & 2 \\ 1 & 1 \end{bmatrix}, B = \begin{bmatrix} 4 & -1 \\ 0 & 2 \end{bmatrix}, C = \begin{bmatrix} 3 & 0 \\ -1 & 2 \\ 1 & 1 \end{bmatrix}, D = \begin{bmatrix} 1 & 5 & 2 \\ -1 & 0 & 1 \\ 3 & 2 & 4 \end{bmatrix}, E = \begin{bmatrix} 6 & 1 & 3 \\ -1 & 1 & 2 \\ 4 & 1 & 3 \end{bmatrix}.$$

Quando for possível, realize as operações a seguir:

i)  $D + E$

ii)  $DC - 2A$

iii)  $2B - C$

iv)  $BC^T + 2023B^TB$

**Exercício 37.3.** Determine todos os valores de  $x$  e  $y$  para que a matriz  $A = \begin{bmatrix} x & x-y & 1 \\ y & 2 & 4 \\ 1 & 2y+x & 3 \end{bmatrix}$  seja simétrica.

**Exercício 37.4.** Uma pessoa dispõe de 17 moedas, sendo algumas de um real, outras de cinquenta centavos e outras de dez centavos. Ela percebe que, gastando todas as moedas de um real, fica com R\$ 1,50. Percebe, por outro lado, que se gastar todas de dez centavos, fica com R\$ 11,00. De quantas moedas de cada tipo a pessoa dispõe?

**Exercício 37.5.** Um par de tênis, duas bermudas e três camisas custam juntos R\$ 100,00. Dois pares de tênis, cinco bermudas e 8 camisas custam juntas R\$ 235,00. Quanto custam juntos um par de tênis, uma bermuda e uma camiseta?

### Exercícios de Aprofundamento

**Exercício 37.6.** Encontre a matriz inversa de  $\begin{bmatrix} (e^x + e^{-x})/2 & (e^x - e^{-x})/2 \\ (e^x - e^{-x})/2 & (e^x + e^{-x})/2 \end{bmatrix}$ .

**Exercício 37.7.** Seja  $A = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$ . Mostre que  $(I - A)^3 = 0$  e use que  $(I - A)^3 = I - 3A + 3A^2 - A^3$  para encontrar a matriz inversa  $A^{-1}$ .

**Exercício 37.8.** Calcule a expressão  $A^2 - 4A - 5I$ , onde  $A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$

**Exercício 37.9.** Dada a matriz  $A = \begin{bmatrix} 2 & -1 & 1 \\ 3 & -2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$ , calcule  $(I - A)(A^2 - I)$ .

**Exercício 37.10.** Resolva o sistema 
$$\begin{cases} x + 3y + 5z + 7w = 12 \\ 3x + 5y + 7z + w = 0 \\ 5x + 7y + z + 3w = 4 \\ 7x + y + 3z + 7w = 16 \end{cases}$$

### Exercícios Avançados

**Exercício 37.11.** Sejam  $A, B$  e  $P$  matrizes, onde  $P$  é invertível e satisfaz  $A = P^{-1}BP$ , mostre que  $A^n = P^{-1}B^nP$  para todo  $n$  natural.

**Exercício 37.12.** Se  $A$  é uma matriz quadrada tal que  $A^k = 0$  para algum inteiro positivo  $k$ , então

$$(I - A)^{-1} = I + A + A^2 + \cdots + A^{k-1}.$$

**Exercício 37.13.** Se  $A^2 = \lambda A$ , para algum  $\lambda \in \mathbb{R}$  diferente de 1, então  $(A + I)^{-1} = I - \frac{1}{1 + \lambda}A$

# 38

## Determinantes

Agora estamos imersos num contexto que apenas existem matrizes e sistemas lineares. Vamos introduzir o *determinante de uma matriz* como uma função que nos retorna um valor numérico que pode ser visto como a área ou volume de certa região ou que ainda nos gera um novo método para resolver sistemas lineares.

Nós vamos começar a trabalhar com matrizes pequenas e como caso inicial que vamos considerar são as matrizes  $2 \times 2$ .

**Definição 38.1.** O determinante da matriz  $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$  é definido como  $\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$ . O determinante de uma matriz  $1 \times 1$  é trivial, pois é definido como o valor dessa única entrada.

**Exemplo 38.0.1.** Como exemplos iniciais vamos considerar algumas matrizes simples.

$$\bullet \det \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = 1 \cdot 4 - 2 \cdot 3 = -2$$

$$\bullet \det \begin{bmatrix} 1 & 0 \\ 32 & 0 \end{bmatrix} = 1 \cdot 0 - 32 \cdot 0 = 0$$

$$\bullet \det \begin{bmatrix} 5 & 7 \\ 4 & 2 \end{bmatrix} = 5 \cdot 2 - 7 \cdot 4 = -4$$

$$\bullet \det \begin{bmatrix} 2 & 4 \\ 4 & 8 \end{bmatrix} = 2 \cdot 8 - 4 \cdot 4 = 0$$

### §38.1 Propriedades dos determinantes $2 \times 2$

Vamos resumir algumas propriedades importantes para o nosso estudo nas três proposições a seguir.

**Proposição 38.1.1** – Sejam  $\lambda \neq 0$  e  $a, b, c, d, a', b', c', d' \in \mathbb{R}$  números reais quaisquer. Valem os seguintes itens:

(i) Para qualquer matriz  $A$  de ordem  $2 \times 2$  temos que  $\det A = \det A^T$ .

$$(ii) \det \begin{bmatrix} \lambda a & \lambda b \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & b \\ \lambda c & \lambda d \end{bmatrix} = \lambda \det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$(iii) \det \begin{bmatrix} a + a' & b + b' \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} + \det \begin{bmatrix} a' & b' \\ c & d \end{bmatrix}$$

$$(iv) \det \begin{bmatrix} a & b \\ c + c' & d + d' \end{bmatrix} = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} + \det \begin{bmatrix} a & b \\ c' & d' \end{bmatrix}$$

(v) Se trocarmos a ordem de duas linhas (ou colunas) de uma matriz seu determinante muda de sinal.

*Demonstração.* (i) Este primeiro item decorre imediatamente da definição do determinante.

(ii) Para qualquer  $\lambda \neq 0$  temos que

$$\det \begin{bmatrix} \lambda a & \lambda b \\ c & d \end{bmatrix} = (\lambda a)d - (\lambda b)c = \lambda(ad - bc) = \lambda \det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

As outras igualdades são verificadas da mesma maneira.

(iii) Uma conta direta nos mostra que

$$\begin{aligned} \det \begin{bmatrix} a+a' & b+b' \\ c & d \end{bmatrix} &= (a+a')d + (b+b')c = (ad+bc) + (a'd+b'c) \\ &= \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} + \det \begin{bmatrix} a' & b' \\ c & d \end{bmatrix} \end{aligned}$$

(iv) A prova deste item é uma ligeira adaptação da prova do item anterior.

(v) Basta ver que  $\det \begin{bmatrix} c & d \\ a & b \end{bmatrix} = bc - ad = -(ad - bc) = -\det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

□

**Observação 38.1.I.** Os itens (iii) e (iv) da [Proposição 38.1.1](#) não devem ser confundidos com  $\det(A+B) = \det A + \det B$  pois esta última igualdade é falsa! De fato,

$$\det(I_2 + I_2) = \det \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = 4 \neq 2 = \det(I_2) + \det(I_2)$$

**Observação 38.1.II.** Podemos usar ainda a [Proposição 38.1.1](#) para verificar que se temos uma matriz  $2 \times 2$  onde uma linha (ou coluna) é múltipla da outra, então seu determinante é zero. De fato, basta observar que  $\det \begin{bmatrix} \lambda a & \lambda b \\ a & b \end{bmatrix} = \lambda \det \begin{bmatrix} a & b \\ a & b \end{bmatrix} = \lambda(ab - ab) = 0$ .

**Proposição 38.1.2** – Sejam  $\lambda \neq 0$  e  $a, b, c, d \in \mathbb{R}$  números reais quaisquer. Valem as seguintes igualdades

$$\det \begin{bmatrix} a+\lambda c & b+\lambda d \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & b \\ c+\lambda a & d+\lambda b \end{bmatrix} = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

*Demonstração.* Fazendo uso da [Proposição 38.1.1](#) e [Observação 38.1.II](#) nós obtemos que

$$\det \begin{bmatrix} a+\lambda c & b+\lambda d \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} + \det \begin{bmatrix} \lambda c & \lambda d \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

□

**Observação 38.1.III.** As proposições [38.1.2](#) e [38.1.1](#) nos mostra com o determinante é alterado após operações elementares em uma matriz.

**Proposição 38.1.3** – Para quaisquer matrizes  $A$  e  $B$  de ordem  $2 \times 2$  temos que

$$\det(AB) = \det(A)\det(B).$$

A prova desta proposição não é nada além do que uma conta direta, então deixamos a cargo do leitor.

**Exemplo 38.1.4.** É possível mostrar através de indução que  $\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \begin{bmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{bmatrix}$  para todo  $n \in \mathbb{N}$ , onde  $\{f_n\}$  é a sequência de Fibonacci. Usando a [Proposição 38.1.3](#) temos que

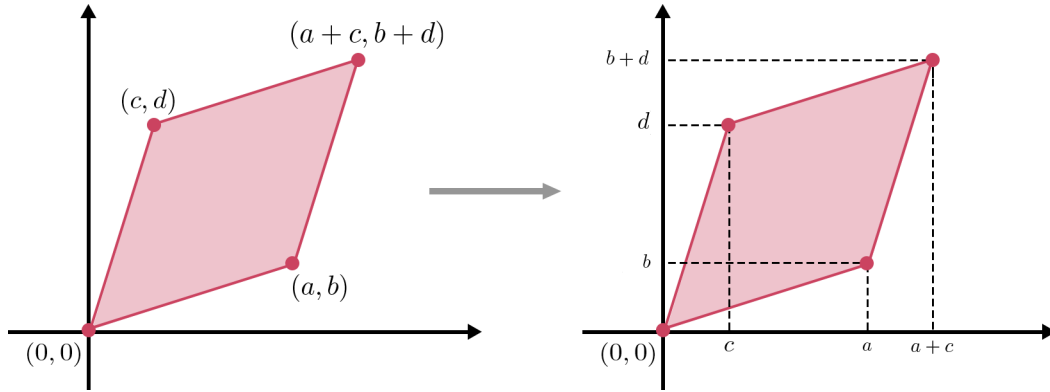
$$f_{n+1}f_{n-1} - (f_n)^2 = \det \begin{bmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{bmatrix} = \det \left( \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n \right) = (-1)^n$$

Esta igualdade é conhecida como *Identidade de Cassini* e pode ser usada para encontrarmos várias outras identidades envolvendo a sequência de Fibonacci.

## §38.2 Determinante visto como área

Veja que essa definição de determinante não se parece nada intuitivo, então vamos dar uma motivação geométrica para esse valor numérico.

Considere o paralelogramo formado pelos pontos  $(0, 0)$ ,  $(a, b)$ ,  $(c, d)$  e  $(a + c, b + d)$ . Estamos interessados em descobrir a área desta região. Faremos isso de um modo indireto, pois podemos calcular de um modo mais simples a área de uma região exterior ao paralelogramo considerado.



Veja que na figura anterior o paralelogramo está dentro do retângulo de vértices  $(0, 0)$ ,  $(a + c, 0)$ ,  $(a + c, a + d)$  e  $(b + d)$ . Note que podemos particionar a área deste retângulo como a soma da área do paralelogramo e a outra região que pode ser dividida em dois triângulos e dois trapézios.

Desse modo, área da região exterior ao paralelogramo e interior ao retângulo citado é dado

$$\text{Área}(\text{exterior}) = \frac{1}{2}ab + \frac{1}{2}(2b + d)c + \frac{1}{2}cd + \frac{1}{2}(a + 2c)b = \frac{1}{2} [2ab + 4bc + 2cd] = ab + 2bc + cd$$

Daí a área do paralelogramo é dado por

$$\begin{aligned} \text{Área}(\text{paralelogramo}) &= \text{Área}(\text{retângulo}) - \text{Área}(\text{exterior}) \\ &= (a + c)(b + d) - (ab + 2bc + cd) \\ &= ad - bc = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \end{aligned}$$

**Observação 38.2.I.** Poderíamos definir a igualdade acima como a **área orientada** do paralelogramo considerado, pois se mudarmos a ordem que consideramos os vértices encontramos o valor da área com o sinal trocado. Caso não quisermos se preocupar com a ordem considerada basta tomar a função módulo nesta última igualdade.

Então podemos concluir que apesar da definição genérica do determinante, ele carrega um significado geométrico. Usando a intuição geométrica podemos explorar os resultados das três proposições anteriores para entendermos geometricamente o que cada nos diz.

**Exemplo 38.2.1 (Área de um triângulo qualquer).** Sabemos da Geometria Euclidiana que uma diagonal do paralelogramo divide este quadrilátero em dois triângulos congruentes. Daí, podemos calcular a área de qualquer triângulo com vértices  $(0, 0)$ ,  $(a, b)$  e  $(c, d)$ , pois, este será dado por

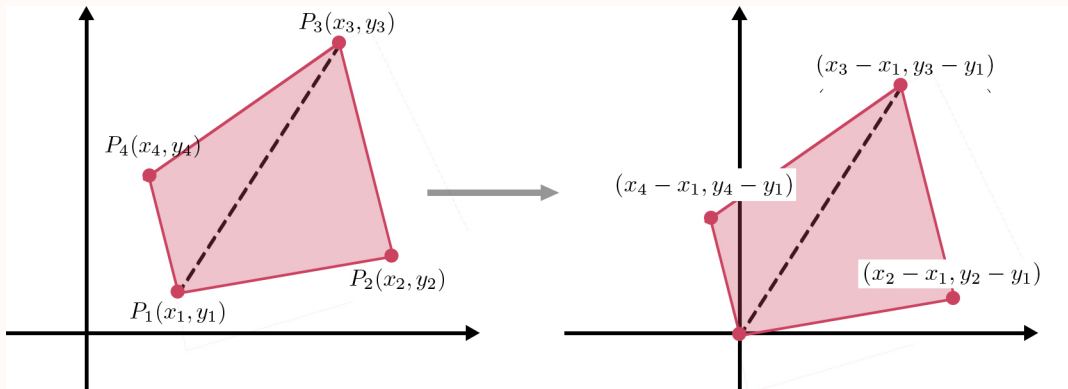
$$\frac{1}{2} \left| \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right|.$$

O uso da função modular nos permite a considerar qualquer ordem dos vértices e é mais apropriada para memorização.

Para o caso de um triângulo qualquer de vértices  $P_1(x_1, y_1)$ ,  $P_2(x_2, y_2)$  e  $P_3(x_3, y_3)$  vamos considerar a translação da figura de modo que o vértice  $P_3$  se torne a origem. Isto ocorre quando subtraímos  $x_3$  e  $y_3$  de cada uma das coordenadas dos demais vértices. [Exemplo 38.2.2](#) temos os vértices  $P_1, P_2$  e  $P_3 = P_4$ . Assim, temos que sua área é dada por

$$\text{Área}(P_1P_2P_3) = \frac{1}{2} \left| \det \begin{bmatrix} x_1 - x_3 & x_1 - y_3 \\ x_2 - x_3 & x_2 - y_3 \end{bmatrix} \right|.$$

**Exemplo 38.2.2 (Área de um quadrilátero qualquer).** Podemos nos perguntar como calcular a área de um quadrilátero qualquer usando um determinante, porém não podemos fazer isso diretamente por motivos bem simples. Primeiramente, veja que a área de qualquer quadrilátero não pode ser calculada com a teoria que vimos até agora, pois ela nos permite apenas a nos restringir ao paralelogramos com um dos vértices sendo a origem. Vamos então encontrar a área de um quadrilátero sabendo as coordenadas de seus vértices.



Veja que se considerarmos um quadrilátero de vértices  $P_1(x_1, y_1)$ ,  $P_2(x_2, y_2)$ ,  $P_3(x_3, y_3)$  e  $P_4(x_4, y_4)$ , então podemos considerar uma ‘recorte’ deste quadrilátero pelo segmento entre os vértices  $P_1$  e  $P_3$ . Para garantir que um dos vértices seja a origem, procedemos trasladando toda a figura de modo que o vértice  $P_1$  seja transformado na origem. Isso é alcançado

Juntando estas observações e a [Proposição 38.1.1](#) podemos concluir que a área do quadrilátero que estamos considerando é dado por

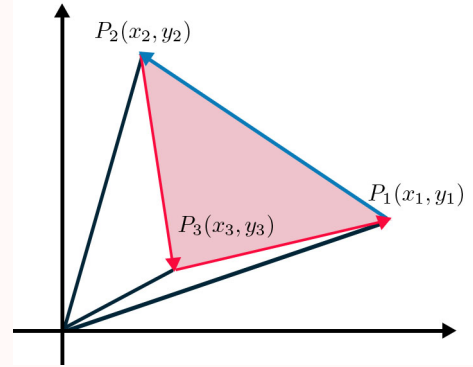
$$\begin{aligned} \text{Área}(P_1P_2P_3P_4) &= \frac{1}{2} \det \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{bmatrix} + \frac{1}{2} \det \begin{bmatrix} x_3 - x_1 & y_3 - y_1 \\ x_4 - x_1 & y_4 - y_1 \end{bmatrix} \\ &= \frac{1}{2} \det \begin{bmatrix} x_1 - x_3 & y_1 - y_3 \\ x_2 - x_3 & y_2 - y_3 \end{bmatrix} \end{aligned}$$

Novamente quero levantar a atenção para a ordem que consideramos os vértices e esta igualdade vale apenas nesta mesma ordem, caso contrário deve ser tomado a função modular.



**Exemplo 38.2.3.** O método que mostramos nos Exemplos 38.2.2 e 38.2.1 talvez não funcione para um polígono qualquer, então buscamos aqui um novo jeito para calcular a área de um polígono onde conhecemos as coordenadas de cada um de seus vértices. Vamos usar o fato de  $\frac{1}{2} \det \begin{bmatrix} a & b \\ c & d \end{bmatrix}$  ser a *área orientada* do triângulo de vértices  $(0, 0)$ ,  $(a, b)$  e  $(c, d)$ .

Assim, encontrar a área de um triângulo de vértices  $P_1(x_1, y_1)$ ,  $P_2(x_2, y_2)$  e  $P_3(x_3, y_3)$ , onde esses vértices estão em ordem anti-horária, é possível pois podemos considerar os triângulos  $OP_1P_2$ ,  $OP_2P_3$  e  $OP_3P_1$  e pela **Observação 38.2.1** o primeiro desses triângulos possui área positiva (por este motivo, na figura ao lado possuem arestas em vermelho e outra em azul). Logo, a área de  $P_1P_2P_3$  é dada por



$$\frac{1}{2} \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} + \frac{1}{2} \det \begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} + \frac{1}{2} \det \begin{bmatrix} x_3 & y_3 \\ x_1 & y_1 \end{bmatrix}$$

Podemos adaptar esse argumento para qualquer polígono sem auto-interseções cujo os pontos  $P_1(x_1, y_1), \dots, P_n(x_n, y_n)$  formam uma orientação anti-horária. Neste caso a área do polígono  $P_1P_2 \cdots P_n$  é dada por

$$\text{Área}(P_1P_2 \cdots P_n) = \frac{1}{2} \sum_{i=1}^n \det \begin{bmatrix} x_i & y_i \\ x_{i+1} & y_{i+1} \end{bmatrix}$$

onde estamos considerando que  $x_{n+1} = x_1$  e  $y_{n+1} = y_1$ .

### §38.3 Determinantes de ordens maiores

**Definição 38.2.** Se  $A$  for uma matriz quadrada, então o **determinante menor** da entrada  $a_{ij}$  é denotado por  $M_{ij}$  e definido como o determinante da submatriz que sobra quando excluimos a  $i$ -ésima linha e a  $j$ -ésima coluna de  $A$ . O número  $C_{ij} = (-1)^{i+j} M_{ij}$  é denominado **cofator** da entrada  $a_{ij}$ .

**Exemplo 38.3.1.** Vamos considerar a matriz seguinte  $3 \times 3$

$$A = \begin{bmatrix} 2 & 4 & 1 \\ -1 & 0 & 3 \\ 5 & -2 & 1 \end{bmatrix}$$

Para encontrar os cofatores das entradas da matriz, precisamos calcular o determinante de cada matriz  $2 \times 2$  formada pela exclusão da linha e coluna correspondentes à entrada em questão.

- Para a entrada  $a_{11} = 2$  temos que seu cofator é

$$C_{11} = (-1)^{1+1} M_{11} = (-1)^2 \det \begin{bmatrix} 0 & 3 \\ -2 & 1 \end{bmatrix} = 0 \cdot 1 - 3 \cdot (-2) = 6$$

- Para a entrada na posição  $(1, 2)$ :

$$C_{12} = (-1)^{1+2} M_{12} = (-1)^3 \cdot \det \begin{bmatrix} -1 & 3 \\ 5 & 1 \end{bmatrix} = -(-1 \cdot 1 - 3 \cdot 5) = 16$$

- Para a entrada  $a_{13} = 1$  que seu cofator é

$$C_{13} = (-1)^{1+1} M_{11} = (-1)^2 \det \begin{bmatrix} -1 & 0 \\ 5 & -2 \end{bmatrix} = (-1) \cdot (-2) - 0 \cdot 5 = 2$$

Podemos repetir o mesmo processo para as demais entradas da matriz que encontraremos

$$\begin{array}{lll} C_{11} = 6 & C_{12} = 16 & C_{13} = 2 \\ C_{21} = -6 & C_{22} = -3 & C_{23} = 24 \\ C_{31} = 12 & C_{32} = -7 & C_{33} = 4 \end{array}$$

**Observação 38.3.I.** Observe que o fator  $(-1)^{i+j}$  no cofator  $C_{ij}$  nos gera uma matriz com uma sequência alternada de sinais de  $+$  e  $-$ , semelhante ao padrão de um tabuleiro de xadrez.

$$\begin{bmatrix} + & - & + & - & + & \cdots \\ - & + & - & + & - & \cdots \\ + & - & + & - & + & \cdots \\ - & + & - & + & - & \cdots \\ + & - & + & - & + & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

**Observação 38.3.II.** Outra observação que podemos fazer é que o determinante menor pode ser calculado de uma forma visual. Por exemplo, para a matriz  $\begin{bmatrix} 0 & 2 & 1 \\ 3 & -1 & 2 \\ 4 & 0 & 1 \end{bmatrix}$  temos que as matrizes que geram os determinantes menores  $M_{11}$ ,  $M_{23}$  e  $M_{32}$  são, respectivamente,

$$\begin{bmatrix} \cancel{0} & \cancel{2} & \cancel{1} \\ 3 & -1 & 2 \\ 4 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 2 & \cancel{1} \\ \cancel{3} & \cancel{-1} & \cancel{2} \\ 4 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & \cancel{2} & 1 \\ 3 & \cancel{-1} & 2 \\ \cancel{4} & \cancel{0} & \cancel{1} \end{bmatrix}$$

Onde as linhas em vermelho representam a linha e coluna que foi suprimida em cada caso.

**Motivação** — Veja que para a matriz  $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$  os cofatores são  $C_{11} = a_{22}$ ,  $C_{12} = -a_{21}$ ,  $C_{21} = -a_{12}$  e  $C_{22} = a_{11}$ . Observe que

$$\begin{aligned} \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} &= a_{11} \cdot C_{11} + a_{12} \cdot C_{12} = a_{11} \cdot C_{11} + a_{21} \cdot C_{21} \\ &= a_{21} \cdot C_{21} + a_{22} \cdot C_{22} = a_{12} \cdot C_{12} + a_{22} \cdot C_{22} \end{aligned}$$

Ou seja, o determinante de uma matriz  $2 \times 2$  pode ser calculado como a soma dos produtos entre as entradas da matriz e seus respectivos cofatores, utilizando uma mesma linha ou

coluna fixada. A partir disso vamos definir o determinante de matrizes de ordens maiores como a seguir.

**Definição 38.3** (Expansão por Cofatores). Se  $A$  for uma matriz de tamanho  $n \times n$ , então o número obtido multiplicando as entradas de uma linha ou coluna qualquer de  $A$  pelos cofatores correspondentes e somando os produtos assim obtidos é denominado determinante de  $A$ , será denotado por  $\det A$ . Cada uma dessas somas são denominadas **expansões em cofatores** de  $\det A$ , ou seja,

$$\begin{aligned}\det A &= a_{i1} \cdot C_{i1} + a_{i2} \cdot C_{i2} + a_{i3} \cdot C_{i3} + \cdots + a_{in} \cdot C_{in} \\ &= a_{1j} \cdot C_{1j} + a_{2j} \cdot C_{2j} + a_{3j} \cdot C_{3j} + \cdots + a_{1n} \cdot C_{1n}\end{aligned}$$

A primeira das igualdades acima é chamada de **expansão em cofatores ao longo da linha  $i$**  e a segunda **expansão em cofatores ao longo da coluna  $j$** . Note que disto temos que  $\det A = \det A^T$ .

**Exemplo 38.3.2.** No Exemplo 38.3.1 mostramos que os cofatores da primeira linha da

matriz  $A = \begin{bmatrix} 2 & 4 & 1 \\ -1 & 0 & 3 \\ 5 & -2 & 1 \end{bmatrix}$  são dados por  $C_{11} = 6$ ,  $C_{12} = 16$  e  $C_{13} = 2$ . Vamos primeiramente usar a primeira linha para calcularmos o determinante da matriz  $A$ . Assim, a expansão por cofatores sobre esta linha nos mostra que

$$\det A = 2 \cdot C_{11} + 4 \cdot C_{12} + 1 \cdot C_{13} = 2 \cdot 6 + 4 \cdot (-16) + 1 \cdot 2 = 78$$

Vamos agora usar a segunda coluna para calcularmos o mesmo determinante.

$$\begin{aligned}\det A &= 4 \cdot C_{12} + 0 \cdot C_{22} + (-2) \cdot C_{32} = 4 \cdot C_{12} + (-2) \cdot C_{32} \\ &= 4 \cdot (-1)^{1+2} \det \begin{bmatrix} 0 & 3 \\ -2 & 1 \end{bmatrix} + (-2) \cdot (-1)^{3+2} \det \begin{bmatrix} 2 & 1 \\ -1 & 3 \end{bmatrix} = 78\end{aligned}$$

**Exemplo 38.3.3 (Regra de Sarrus).** Sabemos que o determinante de uma matriz  $2 \times 2$  é nada mais que o produto de uma das diagonais subtraído do produto da outra e para matrizes maiores definimos sendo a expansão por cofatores, mas será que existe algum método mais simples para matrizes de ordem  $3 \times 3$ ? Neste exemplo mostraremos a *Regra de Sarrus*, esta regra nada mais é que uma outra forma para calcularmos determinantes de matrizes  $3 \times 3$  e que se parece muito com a definição que demos para matrizes  $2 \times 2$ . Vamos dividir em passos:

(i) Utilizando a expansão em cofatores pela primeira linha temos que

$$\det \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = a \cdot \det \begin{bmatrix} e & f \\ h & i \end{bmatrix} - b \cdot \det \begin{bmatrix} d & f \\ g & i \end{bmatrix} + c \cdot \det \begin{bmatrix} d & e \\ g & h \end{bmatrix}$$

(ii) Continuando a conta do item anterior e arrumando a ordem dos termos nós obtemos

$$\det \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = aei + bfg + cdh - gec - hfa - idb.$$

(iii) Escrevendo a matriz original e copie as duas primeiras colunas da matriz à direita da 3ª coluna, de modo que seja obtida uma tabela de 3 linhas e 5 colunas. Nessa nova tabela, observe que os termos  $aei, bfg, cdh, gec, hfa$  e  $idb$  são os produtos de termos

que aparecem em uma mesma diagonal e marque essas diagonais com setas apontando para cima e outras apontando para baixo.

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \Rightarrow \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \begin{vmatrix} a & b \\ d & e \\ g & h \end{vmatrix} \Rightarrow \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \begin{vmatrix} a & b \\ d & e \\ g & h \end{vmatrix}$$

Portanto o determinante de uma matriz  $3 \times 3$  pode ser calculado através da soma dos produtos de cada uma das setas que apontam para baixo subtraído dos produtos das setas que apontam para cima. Esta é a Regra de Sarrus e para alguns é a maneira mais prática para calcular determinante de matrizes  $3 \times 3$ . Vale ainda observar que esta regra **não vale** para matrizes quadradas de qualquer outra ordem.

Como aplicação desse regra, vejamos que

$$\begin{vmatrix} 2 & 1 & 2 \\ 1 & 3 & 0 \\ 0 & 2 & -1 \end{vmatrix} \Rightarrow \begin{vmatrix} 2 & 1 & 2 \\ 1 & 3 & 0 \\ 0 & 2 & -1 \end{vmatrix} \begin{vmatrix} 2 & 1 \\ 1 & 3 \\ 0 & 2 \end{vmatrix}$$

Então temos

$$\det \begin{bmatrix} 2 & 1 & 2 \\ 1 & 3 & 0 \\ 0 & 2 & -1 \end{bmatrix} = 2 \cdot 3 \cdot (-1) + 1 \cdot 0 \cdot 0 + 2 \cdot 1 \cdot 2 - 0 \cdot 3 \cdot 2 - 2 \cdot 0 \cdot 2 - (-1) \cdot 1 \cdot 1 = -1$$

**Observação 38.3.III.** Por simplicidade, podemos representar o determinante de uma matriz substituindo o termo  $\det$  por duas barras verticais ao lado da matriz. Desse modo,

$$\det \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} = \begin{vmatrix} * & * & * \\ * & * & * \\ * & * & * \end{vmatrix}, \quad \det \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{vmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{vmatrix}$$

e assim por diante.

**Exemplo 38.3.4.** Infelizmente esta definição para calcular determinante não nos dá uma forma rápida de realizarmos esse processo, mas podemos fazer uma escolha esperta de linha ou coluna para que nossos cálculos sejam mais eficientes. Consideremos o seguinte exemplo e calculamos o determinante usando a linha ou coluna que está representada na cor vermelha

$$\begin{vmatrix} -1 & 0 & 0 & 3 & 1 & 0 \\ 2 & 3 & 1 & -1 & 0 & 2 \\ 1 & 0 & 3 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & -1 \\ 0 & 2 & 1 & 0 & 3 & 0 \\ -0 & -0 & -0 & 2 & 0 & 0 \end{vmatrix} = 2 \cdot C_{64} = 2 \cdot \begin{vmatrix} -1 & 0 & 0 & 1 & 0 \\ 2 & 3 & 1 & 0 & 2 \\ 1 & 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 & -1 \\ 0 & 2 & 1 & 3 & 0 \end{vmatrix} = 2[(-1) \cdot C_{11} + 1 \cdot C_{14}]$$

$$= 2 \cdot \left[ (-1) \cdot \begin{vmatrix} 3 & 1 & 0 & 2 \\ 0 & 3 & 0 & 0 \\ 0 & 2 & 0 & -1 \\ 2 & 1 & 3 & 0 \end{vmatrix} - 1 \cdot \begin{vmatrix} 2 & 3 & 1 & 2 \\ 1 & 0 & 3 & 0 \\ 0 & 0 & 2 & -1 \\ -0 & -2 & -1 & 0 \end{vmatrix} \right]$$

$$\begin{vmatrix} -1 & 0 & 0 & 3 & 1 & 0 \\ 2 & 3 & 1 & -1 & 0 & 2 \\ 1 & 0 & 3 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & -1 \\ 0 & 2 & 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \end{vmatrix} = 2 \left[ (-1) \left( -3 \begin{vmatrix} 3 & 1 & 2 \\ 0 & 3 & 0 \\ 0 & 2 & -1 \end{vmatrix} \right) - 1 \left( 2 \begin{vmatrix} 2 & 1 & 2 \\ 1 & 3 & 0 \\ 0 & 2 & -1 \end{vmatrix} - 1 \begin{vmatrix} 2 & 3 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{vmatrix} \right) \right]$$

$$= 2 \{ (-1)[(-3) \cdot (-9)] - 1[2 \cdot (-1) - 1 \cdot 3] \} = -44$$

Vejamos que cada uma das vezes que fizemos as escolhas de uma linha ou coluna preferimos aquelas que tem a maior quantidade de zeros, pois nesses casos não precisamos calcular seus cofatores e isto é um jeito um pouco mais eficiente para calcularmos o determinante de uma matriz grande.

**Exemplo 38.3.5.** Usando o [Exemplo 38.2.3](#) sabemos que a área do triângulo de vértices  $P_1(x_1, y_1)$ ,  $P_2(x_2, y_2)$  e  $P_3(x_3, y_3)$ , onde esses vértices estão em ordem anti-horária, é dada por

$$\frac{1}{2} \left( \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} + \det \begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} + \det \begin{bmatrix} x_3 & y_3 \\ x_1 & y_1 \end{bmatrix} \right)$$

Por outro lado, expandindo pela última coluna obtemos que

$$\begin{aligned} \frac{1}{2} \det \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} &= \frac{1}{2} \left( \det \begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} - \det \begin{bmatrix} x_1 & y_1 \\ x_3 & y_3 \end{bmatrix} + \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} \right) \\ &= \frac{1}{2} \left( \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} + \det \begin{bmatrix} x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} + \det \begin{bmatrix} x_3 & y_3 \\ x_1 & y_1 \end{bmatrix} \right) = \text{Área}(P_1 P_2 P_3) \end{aligned}$$

### §38.4 Propriedades dos determinantes $n \times n$

Vamos agora estender as propriedades que tínhamos para os determinantes  $2 \times 2$ . Como o caso inicial, vamos considerar matrizes  $3 \times 3$  e vamos dividir em partes.

- (i) Em geral, **não** é verdade que  $\det(A + B) = \det(A) + \det(B)$  para quaisquer duas matrizes e é fácil mostrar que não ocorre a igualdade com um exemplo simples. Por outro lado, usando a expansão na primeira linha temos que

$$\begin{aligned} \det \begin{bmatrix} a+a' & b+b' & c+c' \\ d & e & f \\ g & h & i \end{bmatrix} &= (a+a') \det \begin{bmatrix} e & f \\ h & i \end{bmatrix} - (b+b') \det \begin{bmatrix} d & f \\ g & i \end{bmatrix} + (c+c') \det \begin{bmatrix} d & e \\ g & h \end{bmatrix} \\ &= a \det \begin{bmatrix} e & f \\ h & i \end{bmatrix} - b \det \begin{bmatrix} d & f \\ g & i \end{bmatrix} + c \det \begin{bmatrix} d & e \\ g & h \end{bmatrix} \\ &\quad + a' \det \begin{bmatrix} e & f \\ h & i \end{bmatrix} - b' \det \begin{bmatrix} d & f \\ g & i \end{bmatrix} + c' \det \begin{bmatrix} d & e \\ g & h \end{bmatrix} \\ &= \det \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} + \det \begin{bmatrix} a' & b' & c' \\ d & e & f \\ g & h & i \end{bmatrix} \end{aligned}$$



**Problema 38.1** — Mostre que

$$\det \begin{bmatrix} p+x & q+y & r+z \\ a+x & b+y & c+z \\ a+p & b+q & c+r \end{bmatrix} = 2 \det \begin{bmatrix} a & b & c \\ p & q & r \\ x & y & z \end{bmatrix}$$

**Solução:** Lembrando que operações de linha/colunas não alteram o determinante segue

$$\begin{aligned} \begin{bmatrix} p+x & q+y & r+z \\ a+x & b+y & c+z \\ a+p & b+q & c+r \end{bmatrix} &\xrightarrow{L_2-L_1 \rightarrow L_2} \begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ a+p & b+q & c+r \end{bmatrix} \xrightarrow{L_3+L_2 \rightarrow L_3} \begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ 2a & 2b & 2c \end{bmatrix} \\ &\xrightarrow{L_3+L_2 \rightarrow L_3} \begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ 2a & 2b & 2c \end{bmatrix} \end{aligned}$$

Daí temos que

$$\det \begin{bmatrix} p+x & q+y & r+z \\ a+x & b+y & c+z \\ a+p & b+q & c+r \end{bmatrix} = \det \begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ 2a & 2b & 2c \end{bmatrix} = 2 \det \begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ a & b & c \end{bmatrix}$$

Como ainda temos

$$\begin{bmatrix} p+x & q+y & r+z \\ a-p & b-q & c-r \\ a & b & c \end{bmatrix} \xrightarrow{L_2-L_3 \rightarrow L_2} \begin{bmatrix} p+x & q+y & r+z \\ -p & -q & -r \\ a & b & c \end{bmatrix} \xrightarrow{L_1+L_2 \rightarrow L_1} \begin{bmatrix} x & y & z \\ -p & -q & -r \\ a & b & c \end{bmatrix}$$

Donde concluímos que

$$\det \begin{bmatrix} p+x & q+y & r+z \\ a+x & b+y & c+z \\ a+p & b+q & c+r \end{bmatrix} = 2 \det \begin{bmatrix} x & y & z \\ -p & -q & -r \\ a & b & c \end{bmatrix} = -2 \det \begin{bmatrix} a & b & c \\ -p & -q & -r \\ x & y & z \end{bmatrix} = 2 \det \begin{bmatrix} a & b & c \\ p & q & r \\ x & y & z \end{bmatrix}$$

Como a expansão por cofatores de um determinante  $4 \times 4$  se reduz a soma de alguns determinantes  $3 \times 3$ , então todas as propriedades que fizemos no início da seção também valem para esse novo caso. Pelo mesmo argumento determinantes  $5 \times 5$  devem possuir as mesmas propriedades. Utilizando indução podemos mostrar o seguinte teorema

**Teorema 38.2** – Para quaisquer matrizes  $n \times n$  valem:

- (i) Podemos fazer operações elementares de linhas que o determinante se mantém o mesmo.
- (ii) Se uma linha está sendo multiplicada por uma constante  $k$ , então o determinante é  $k$  vezes o determinante da matriz cuja esta mesma linha foi dividida por esta constante.
- (iii) Se trocarmos duas linhas em uma matriz, então o determinante muda de sinal

**Observação 38.4.I.** Relembre que da definição, temos que  $\det A^T = \det A$  e assim o [Teorema 38.2](#) também vale se transferirmos as afirmações para afirmações sobre as colunas da matriz.

**Observação 38.4.II.** Quero chamar atenção ao leitor que o item (ii) do [Teorema 38.2](#) nos diz que se  $A$  é uma matriz  $n \times n$  e  $k \in \mathbb{R}$  qualquer, então  $\det(kA) = k^n \det A$ , pois cada uma das  $n$  linhas está sendo multiplicada pela mesma constante  $k$ .

**Observação 38.4.III.** Lembre que o escalonamento de uma matriz produz vários zeros nela e o [Teorema 38.2](#) nos diz que o determinante da matriz original e da matriz escalonada é o mesmo (a menos de um sinal). Então podemos utilizar esse truque para calcularmos determinantes de matrizes maiores.

**Definição 38.4.** Definimos a **diagonal principal** de uma matriz  $A = [A_{ij}]_{n \times n}$  pela diagonal que possui os elementos da forma  $a_{ii}$ , para  $i = 1, 2, \dots, n$ . Se para todo  $i > j$  temos  $a_{ij} = 0$ , ou seja, quando todos os elementos abaixo da diagonal principal são zeros dizemos que  $A$  é uma **matriz triangular superior**. Se para todo  $i < j$  temos  $a_{ij} = 0$ , ou seja, quando todos os elementos acima da diagonal principal são zeros dizemos que  $A$  é uma **matriz triangular inferior**.

**Exemplo 38.4.2.** Considere uma matriz triangular superior  $A = [a_{ij}]$ , assim, a matriz é da forma  $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{bmatrix}$ . Então pela [Observação 38.3.I](#) e sempre expandindo pela primeira coluna temos que  $\det A = a_{11} \cdot a_{22} \cdot a_{33} \cdots a_{nn}$ . O mesmo ocorre para matrizes triangulares inferiores.

**Problema 38.3 —** Calcule o determinante da matriz  $A = [a_{ij}]_{6 \times 6}$ , onde

$$a_{ij} = \begin{cases} 2, & \text{se } i = j \\ (-1)^{|i-j|}, & \text{se } i \neq j \end{cases}.$$

**Solução:** Vejamos que a matriz definida no enunciado é dada por  $A =$

$$\begin{bmatrix} 2 & -1 & 1 & -1 & 1 & -1 \\ -1 & 2 & -1 & 1 & -1 & 1 \\ 1 & -1 & 2 & -1 & 1 & -1 \\ -1 & 1 & -1 & 2 & -1 & 1 \\ 1 & -1 & 1 & -1 & 2 & -1 \\ -1 & 1 & -1 & 1 & -1 & 2 \end{bmatrix}.$$

Somando a última linha nas linhas ímpares e subtraindo das linhas pares, obtemos a seguinte matriz

$$\begin{bmatrix} 2 & -1 & 1 & -1 & 1 & -1 \\ -1 & 2 & -1 & 1 & -1 & 1 \\ 1 & -1 & 2 & -1 & 1 & -1 \\ -1 & 1 & -1 & 2 & -1 & 1 \\ 1 & -1 & 1 & -1 & 2 & -1 \\ -1 & 1 & -1 & 1 & -1 & 2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 1 & -1 & 1 & -1 & 2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 7 \end{bmatrix}$$

Onde realizamos no último passo as seguintes operações  $L_6 + L_1 \rightarrow L_6$ , depois  $L_6 + L_3 \rightarrow L_6$ , depois  $L_6 + L_5 \rightarrow L_6$ , seguido de  $L_6 - L_2 \rightarrow L_6$  e por fim  $L_6 - L_4 \rightarrow L_6$ . Desse modo, pelo [Exemplo 38.4.2](#), temos que  $\det A = 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 7 = 7$ .

Como última propriedade do determinante vamos expor, sem demonstração, o seguinte teorema



**Teorema 38.4** – Sejam  $A$  e  $B$  duas matrizes quadradas de mesma ordem, então

$$\det(AB) = \det(A) \det(B).$$

**Observação 38.4.IV.** O Teorema 38.4 nos diz que embora tenhamos  $AB \neq BA$ , vale que

$$\det(AB) = \det A \cdot \det B = \det B \cdot \det A = \det(BA).$$

**Exemplo 38.4.3.** Considerando para todo  $n \in \mathbb{N}$  a matriz dada por

$$M(n) = \begin{bmatrix} 2 & 2 & 2 & \cdots & 2 & 1-n \\ n+2 & 1 & 1 & \cdots & 1 & -n \\ 1 & n+2 & 1 & \cdots & 1 & -n \\ 1 & 1 & n+2 & \cdots & 1 & -n \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & 1 & \cdots & n+2 & -n \end{bmatrix}_{n \times n}$$

é possível mostrar que  $[M(n)]^n = 0$ , então segue pelo Teorema 38.4 que

$$0 = \det([M(n)]^n) = \underbrace{\det M(n) \cdot \det M(n) \cdots \det M(n)}_{n \text{ fatores}} = (\det M(n))^n$$

Logo,  $\det M(n) = 0$  para todo  $n$  inteiro.

**Exemplo 38.4.4.** Seja  $M$  uma matriz invertível. Assim,

$$1 = \det I_n = \det(M \cdot M^{-1}) = \det M \cdot \det M^{-1} \implies \det M^{-1} = \frac{1}{\det M}$$

logo, se  $M$  é invertível então  $\det M \neq 0$ . Considerando a matriz  $A = P^{-1}BP$  não é difícil mostrar por indução que  $A^n = P^{-1}B^nP$  para todo  $n \in \mathbb{N}$ . Assim, para todo  $n \in \mathbb{N}$  temos que

$$\begin{aligned} \det(A^n) &= \det(P^{-1}B^nP) = \det(P^{-1}) \det(B^n) \det P \\ &= \frac{1}{\det P} \det(B^n) \det P = \det(B^n) = [\det B]^n \end{aligned}$$

Isto é útil quando  $B$  é uma matriz triangular! Por exemplo, tomando

$$A = \begin{bmatrix} -5 & 10 & -6 & -7 & -5 \\ -4 & 8 & -4 & -4 & -2 \\ 6 & -7 & 7 & 4 & 4 \\ -2 & 3 & -2 & 1 & 0 \\ 2 & -3 & 2 & 3 & 4 \end{bmatrix}, P = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 \end{bmatrix} \text{ e } B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix}$$

Podemos mostrar que  $A = P^{-1}BP$  e isto nos implica que  $\det A = \det B = 5!$ .

### §38.4.5 Determinante de Vandermonde

Considerando  $x_1, x_2, \dots, x_n$  sendo números reais quaisquer, definimos a **Matriz de Vandermonde** de ordem  $n$  por

$$V_n = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{bmatrix}.$$

Neste momento estamos interessados em calcular  $\det V_4$ . Considerando as operações  $L_4 - x_4 L_3 \rightarrow L_4, L_3 - x_4 L_2 \rightarrow L_3, L_2 - x_4 L_1 \rightarrow L_2$ , temos que

$$\begin{aligned} \det V_4 &= \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ x_1^2 & x_2^2 & x_3^2 & x_4^2 \\ x_1^3 & x_2^3 & x_3^3 & x_4^3 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1 - x_4 & x_2 - x_4 & x_3 - x_4 & 0 \\ x_1(x_1 - x_4) & x_2(x_2 - x_4) & x_3(x_3 - x_4) & 0 \\ x_1^2(x_1 - x_4) & x_2^2(x_2 - x_4) & x_3^2(x_3 - x_4) & 0 \end{vmatrix} \\ &= \begin{vmatrix} x_1 - x_4 & x_2 - x_4 & x_3 - x_4 \\ x_1(x_1 - x_4) & x_2(x_2 - x_4) & x_3(x_3 - x_4) \\ x_1^2(x_1 - x_4) & x_2^2(x_2 - x_4) & x_3^2(x_3 - x_4) \end{vmatrix} = (x_1 - x_4)(x_2 - x_4)(x_3 - x_4) \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^2 & x_2^2 & x_3^2 \end{vmatrix} \end{aligned}$$

Note que agora precisamos calcular  $\det V_3$ . Realizando as operações  $L_3 - x_3 L_2 \rightarrow L_3$  e  $L_2 - x_3 L_1 \rightarrow L_2$  segue que

$$\begin{aligned} \det V_3 &= \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ x_1^2 & x_2^2 & x_3^2 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 \\ x_1 - x_3 & x_2 - x_3 & 0 \\ x_1(x_1 - x_3) & x_2(x_2 - x_3) & 0 \end{vmatrix} \\ &= (x_1 - x_3)(x_2 - x_3) \begin{vmatrix} 1 & 1 \\ x_1 & x_2 \end{vmatrix} = (x_1 - x_3)(x_2 - x_3)(x_2 - x_1) \end{aligned}$$

Portanto, temos que

$$\det V_4 = (x_1 - x_4)(x_2 - x_4)(x_3 - x_4)(x_1 - x_3)(x_2 - x_3)(x_2 - x_1).$$

Estas contas nos mostram que para calcular  $\det V_{n+1}$  se reduz a calcular  $\det V_n$ , ou seja, podemos tentar provar uma fórmula geral por indução.

**Afirmção 38.4.6** — Para todo  $n \in \mathbb{N}$  temos que  $\det V_n = \prod_{1 \leq i < j \leq n} (x_j - x_i)$ .

*Demonstração.* A prova será por indução em  $n \in \mathbb{N}$ . Já mostramos para  $n = 3$ , os casos  $n = 1$  e  $n = 2$  deixamos para o leitor. Assuma que o resultado vale para  $n - 1$  e considere qualquer escolha de  $x_1, x_2, \dots, x_n \in \mathbb{R}$ . Veja que se dois desses números escolhidos forem o mesmo, então imediatamente temos duas colunas iguais em  $V_n$ , logo o determinante é zero. Podemos então assumir que todos esses são dois-a-dois distintos. Considere o polinômio

$$p(y) = \begin{vmatrix} 1 & 1 & \cdots & 1 & 1 \\ x_1 & x_2 & \cdots & x_{n-1} & y \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_{n-1}^{n-1} & y^{n-1} \end{vmatrix} = (\det V_n) y^{n-1} + a_{n-2} y^{n-2} + \cdots + a_1 y + a_0$$

para certos  $a_0, a_1, \dots, a_{n-2} \in \mathbb{R}$ . Pela hipótese de indução temos que o coeficiente líder é  $\prod_{1 \leq i < j \leq n} (x_j - x_i)$ . Note que se  $1 \leq i \leq n$  é qualquer inteiro, então temos  $p(x_i) = 0$ , ou seja,  $p(x_1) = p(x_2) = \cdots = p(x_{n-1}) = 0$  são  $n - 1$  zeros de um polinômio de grau  $n - 1$ , logo podemos

fatorar o polinômio  $p(y)$  como  $p(y) = \prod_{1 \leq i < j \leq n} (x_j - x_i) \cdot \prod_{i=1}^{n-1} (y - x_i)$ . Tomando  $y = x_n$  nós temos que

$$\prod_{1 \leq i < j \leq n} (x_j - x_i) = p(x_n) = \begin{vmatrix} 1 & 1 & \cdots & 1 & 1 \\ x_1 & x_2 & \cdots & x_{n-1} & x_n \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_{n-1}^{n-1} & x_n^{n-1} \end{vmatrix} = \det V_n$$

□

## §38.5 Sistemas Lineares e Determinantes

Já sabemos que todo sistema linear pode ser expresso da forma  $A\mathbf{x} = \mathbf{b}$ , onde  $A$  é uma matriz quadrada. Lembre que se  $A^{-1}$  existe então a solução deste sistema é dada por  $\mathbf{x} = A^{-1}\mathbf{b}$ . Daí é importante podermos caracterizar quando  $A^{-1}$  existe ou não. No [Exemplo 38.4.4](#) mostramos a seguinte implicação

$$“A \text{ é invertível} \implies \det A \neq 0”$$

Será que a recíproca é verdadeira? Vamos estudar as relações entre determinantes e sistemas lineares e vamos mostrar uma forma explícita para calcularmos a matriz inversa. A resposta para este questionamento será uma consequência imediata deste estudo.

**Motivação 5** — Vamos considerar uma matriz  $A$  de ordem  $4 \times 4$ . Vamos criar uma outra matriz  $4 \times 4$  sendo que esta é uma cópia da anterior a menos das duas primeiras linhas, que agora serão iguais. Por exemplo, temos

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \implies B = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Assim, independente da matriz  $A$ , o determinante de  $B$  é zero pois temos duas linhas iguais. Daí, expandindo em cofatores pela segunda linha temos que

$$0 = \det B = 1 \cdot C_{21} + 2 \cdot C_{22} + 3 \cdot C_{23} + 4 \cdot C_{24}$$

Note que esta expressão se parece muito com o cálculo do termo da primeira linha e segunda coluna do produto entre  $A$  e outra matriz que envolvem os cofatores de  $A$ . De fato, basta observar que

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \begin{bmatrix} * & C_{21} & * & * \\ * & C_{22} & * & * \\ * & C_{23} & * & * \\ * & C_{24} & * & * \end{bmatrix} = \begin{bmatrix} * & 0 & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

Os asteriscos representam termos que não são importantes. Podemos repetir a mesma construção copiando a linha  $i$  e substituindo esta pela linha  $j$ . Pelo mesmo argumento, se  $i \neq j$ , teremos que esta nova matriz possui determinante zero e a expansão por cofator nesta linha copiada pode ser vista como o elemento da posição  $(i, j)$  do produto entre duas matrizes. Porém, se  $i = j$ , então o elemento da posição  $(i, i)$  do produto considerado será

simplesmente  $\det A$ . Em resumo, vale a seguinte igualdade

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \begin{bmatrix} C_{11} & C_{21} & C_{31} & C_{41} \\ C_{12} & C_{22} & C_{32} & C_{42} \\ C_{13} & C_{23} & C_{33} & C_{43} \\ C_{14} & C_{24} & C_{34} & C_{44} \end{bmatrix} = \begin{bmatrix} \det A & 0 & 0 & 0 \\ 0 & \det A & 0 & 0 \\ 0 & 0 & \det A & 0 \\ 0 & 0 & 0 & \det A \end{bmatrix}$$

Observe que na [Motivação 5](#) não há nada especial na matriz  $A$  que foi considerada, então podemos repetir as mesmas contas para uma matriz qualquer matriz de ordem  $n \times n$ . Desse modo, nos vem a seguinte definição

**Definição 38.5.** Dada a matriz  $A = [a_{ij}]_{n \times n}$ , onde  $C_{ij}$  é o cofator de  $a_{ij}$ , então consideramos  $\text{Adj}(A) = [C_{ij}]^T$ , isto é, a transposta da matriz dos cofatores. A matriz  $\text{Adj}(A)$  é conhecida como **matriz adjunta** da matriz  $A$ .

**Teorema 38.6** – Para qualquer matriz  $A$  de ordem  $n \times n$  temos que

$$A \cdot \text{Adj}(A) = \text{Adj}(A) \cdot A = (\det A)I_n.$$

*Demonstração.* A ideia da prova é repetir os passos da [Motivação 5](#). Considere a matriz  $A = [a_{ij}]$ , onde  $C_{ij}$  é o cofator de  $a_{ij}$ , então temos

$$\sum_{k=1}^n a_{ki} C_{kj} = \sum_{k=1}^n C_{kj} a_{ki} = \delta_{ij}(\det A),$$

onde  $\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$ . Então  $A \cdot \text{Adj}(A) = \text{Adj}(A) \cdot A = (\det A)I_n$ . □

**Corolário 38.7** – Se  $\det A \neq 0$ , então  $A^{-1} = \frac{1}{\det A} \text{Adj}(A)$ .

Veja que este corolário completa a equivalência que citamos no início da seção e vamos expor isto como o seguinte corolário:

**Corolário 38.8** – Seja  $A$  uma matriz quadrada.  $A$  é invertível se, e somente se,  $\det A \neq 0$ .

**Exemplo 38.5.1.** Podemos encontrar (novamente) a inversa de uma matriz  $2 \times 2$ . De fato, considere  $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ , assim  $C_{11} = d$ ,  $C_{12} = -c$ ,  $C_{21} = -b$  e  $C_{22} = a$ . Portanto,

$$\text{Adj}(A) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}^T = \begin{bmatrix} C_{11} & C_{21} \\ C_{12} & C_{22} \end{bmatrix} = \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

e usando o [Corolário 38.7](#) segue que

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

**Problema 38.9** — Encontre a matriz  $A$  que satisfaz

$$(4A^{-1} - I)^T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix}$$

Relembre que se  $X$  é invertível, então  $(X^T)^T = X$  e  $(X^{-1})^{-1} = X$ . Então

$$\begin{aligned} (4A^{-1} - I)^T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix} &\implies 4A^{-1} - I = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix}^T \\ &\implies 4A^{-1} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix}^T + I = \begin{bmatrix} 2 & 0 & 2 \\ 1 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} \\ &\implies A^{-1} = \frac{1}{4} \begin{bmatrix} 2 & 0 & 2 \\ 1 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 1/4 & 1/2 & 1/4 \\ 0 & 0 & 1/2 \end{bmatrix} \\ &\implies A = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 1/4 & 1/2 & 1/4 \\ 0 & 0 & 1/2 \end{bmatrix}^{-1} \end{aligned}$$

Para calcular a matriz inversa usaremos a fórmula  $B^{-1} = \frac{1}{\det(B)} \cdot \text{Adj}(B)$ . Podemos seguir os seguintes passos:

1. Calcular o determinante de  $B = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 1/4 & 1/2 & 1/4 \\ 0 & 0 & 1/2 \end{bmatrix}$  utilizando a expansão por cofatores utilizando a 3ª linha:

$$\det(B) = \frac{1}{2} \cdot \begin{vmatrix} 1/2 & 0 \\ 1/4 & 1/2 \end{vmatrix} = \frac{1}{8}$$

2. Calcular a adjunta de  $B$  encontrando a matriz transposta dos cofatores:

$$\begin{aligned} \text{Adj}(B) &= \begin{bmatrix} \begin{vmatrix} 1/2 & 1/4 \\ 0 & 1/2 \end{vmatrix} & -\begin{vmatrix} 1/4 & 1/4 \\ 0 & 1/2 \end{vmatrix} & \begin{vmatrix} 1/4 & 1/2 \\ 0 & 0 \end{vmatrix} \\ -\begin{vmatrix} 0 & 1/2 \\ 0 & 1/2 \end{vmatrix} & \begin{vmatrix} 1/2 & 1/2 \\ 0 & 1/2 \end{vmatrix} & -\begin{vmatrix} 1/2 & 0 \\ 0 & 0 \end{vmatrix} \\ \begin{vmatrix} 0 & 1/2 \\ 1/2 & 1/4 \end{vmatrix} & -\begin{vmatrix} 1/2 & 1/2 \\ 1/4 & 1/4 \end{vmatrix} & \begin{vmatrix} 1/2 & 0 \\ 1/4 & 1/2 \end{vmatrix} \end{bmatrix}^T = \begin{bmatrix} 1/4 & -1/8 & 0 \\ 0 & 1/4 & 0 \\ -1/4 & 0 & 1/4 \end{bmatrix}^T \\ &= \begin{bmatrix} 1/4 & 0 & -1/4 \\ -1/8 & 1/4 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \end{aligned}$$

3. Calcular a inversa de  $A$  utilizando a fórmula:

$$A = B^{-1} = \frac{1}{\det(B)} \cdot \text{Adj}(B) = \frac{1}{1/8} \cdot \begin{bmatrix} 1/4 & 0 & -1/4 \\ -1/8 & 1/4 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} = \begin{bmatrix} 2 & 0 & -2 \\ -1 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

Voltando ao [Corolário 38.8](#) temos que o sistema linear

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

o qual pode ser escrito de forma matricial como  $A\mathbf{x} = \mathbf{b}$  possui solução se, e somente se,  $\det A \neq 0$ , onde  $A = [a_{ij}]_{n \times n}$ ,  $\mathbf{b} = [b_i]_{n \times 1}$  e  $\mathbf{x} = [x_i]_{n \times 1}$ . Denote por  $A_i$  a matriz obtida a partir de  $A$  substituindo a  $i$ -ésima coluna de  $A$  pela matriz  $\mathbf{b}$ . Além disso, como a solução é dada por  $\mathbf{x} = A^{-1}\mathbf{b}$  e neste caso vale que  $A^{-1} = \frac{1}{\det A} \text{Adj}(A)$ , então como

$$\text{Adj}(A)\mathbf{b} = \begin{bmatrix} C_{11} & C_{21} & \cdots & C_{n1} \\ C_{12} & C_{22} & \cdots & C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{1n} & C_{2n} & \cdots & C_{nn} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} C_{11}b_1 + C_{21}b_2 + \cdots + C_{n1}b_n \\ C_{12}b_1 + C_{22}b_2 + \cdots + C_{n2}b_n \\ \vdots \\ C_{1n}b_1 + C_{2n}b_2 + \cdots + C_{nn}b_n \end{bmatrix}$$

e observando que a soma  $C_{1i}b_1 + C_{2i}b_2 + \cdots + C_{ni}b_n$  significa calcular o determinante da matriz  $A_i$  usando a expansão de cofatores sobre a  $i$ -ésima coluna, então cada variável do sistema  $A\mathbf{x} = \mathbf{b}$  é dada por  $x_i = \frac{\det A_i}{\det A}$ , para cada  $i = 1, 2, \dots, n$ .

O que acabamos de mostrar é um método para resolver sistemas lineares através da matriz adjunta!

**Teorema 38.10** (Regra de Cramer) – Seja  $A$  uma matriz quadrada invertível e  $\mathbf{b}$  uma matriz coluna. Denote por  $A_i$  a matriz obtida a partir de  $A$  substituindo a  $i$ -ésima coluna de  $A$  pela matriz  $\mathbf{b}$ . Então o sistema linear  $A\mathbf{x} = \mathbf{b}$ , possui solução única  $\mathbf{x} = [x_i]_{n \times 1}$  dada por

$$x_i = \frac{\det A_i}{\det A},$$

para cada  $i = 1, 2, \dots, n$ .

**Observação 38.5.I.** Duas restrições para a aplicação da *Regra de Cramer* são imediatas.

- (i) Os sistemas em questão devem ter exatamente o mesmo número de equações que de incógnitas para garantir que todas as matrizes envolvidas sejam quadradas e, portanto, seja possível calcular seus determinantes.
- (ii) O determinante da matriz dos coeficientes não pode ser zero, uma vez que ele aparece no denominador de uma fração. Se o determinante de  $A$  for igual a zero, então a *Regra de Cramer* não pode ser aplicada.

**Observação 38.5.II.** Embora a *Regra de Cramer* forneça um método sistemático para a solução de equações lineares simultâneas, o número de cálculos envolvidos pode se tornar impressionante quando a ordem do determinante é grande. Assim, para sistemas grandes, a *Regra de Cramer* quase nunca é utilizada. A grande vantagem que este método oferece é quando queremos encontrar o valor de uma única variável do sistema linear dado.

**Exemplo 38.5.2.** Para resolvermos o sistema

$$\begin{cases} -3x + 4y + 6z = 30 \\ -x - 2y + 3z = 8 \\ x + 2z = 6 \end{cases}$$

Basta considerarmos as matrizes

$$A = \begin{bmatrix} -3 & 4 & 6 \\ -1 & -2 & 3 \\ 1 & 0 & 2 \end{bmatrix}, A_x = \begin{bmatrix} 30 & 4 & 6 \\ 8 & -2 & 3 \\ 6 & 0 & 2 \end{bmatrix}, A_y = \begin{bmatrix} -3 & 30 & 6 \\ -1 & 8 & 3 \\ 1 & 6 & 2 \end{bmatrix} \text{ e } A_z = \begin{bmatrix} -3 & 4 & 30 \\ -1 & -2 & 8 \\ 1 & 0 & 6 \end{bmatrix}$$

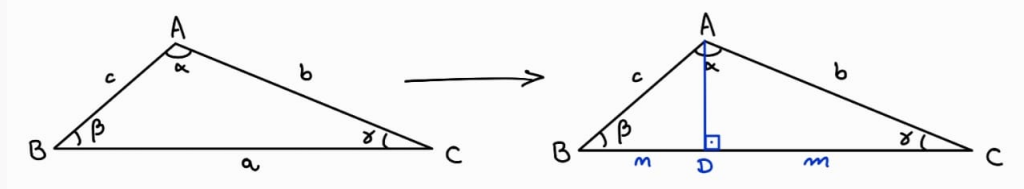
cujo os determinante são  $\det A = 44$ ,  $\det A_x = -40$ ,  $\det A_y = 72$  e  $\det A_z = 152$ . Pela Regra de Cramer as soluções desse sistema linear é dado por

$$x = \frac{\det A_x}{\det A} = \frac{-40}{44}, y = \frac{\det A_y}{\det A} = \frac{72}{44} \text{ e } z = \frac{\det A_z}{\det A} = \frac{152}{44}.$$

**Problema 38.11 (Lei dos Cossenos)** — Num triângulo qualquer  $\triangle ABC$ , mostre que

$$\overline{BC}^2 = \overline{AB}^2 + \overline{AC}^2 - 2\overline{AB} \cdot \overline{AC} \cos \hat{A}$$

**Solução:** Por simplicidade, denotaremos as arestas do triângulo por  $\overline{AB} = c$ ,  $\overline{AC} = b$  e  $\overline{BC} = a$  e os ângulos internos serão  $\hat{A} = \alpha$ ,  $\hat{B} = \beta$  e  $\hat{C} = \gamma$ . A partir do vértice  $A$  trace a altura do triângulo e divida o lado  $BC$  em segmentos de comprimento  $m$  e  $n$ .



Logo,  $\cos \beta = n/c$  e  $\cos \gamma = m/b$  e disto concluímos que vale  $c \cos \beta + b \cos \gamma = n + m = a$ . Repita o processo anterior para os outros vértices de  $\triangle ABC$  e obtemos as equações

$$\begin{cases} c \cos \beta + b \cos \gamma = a \\ a \cos \gamma + c \cos \alpha = b \\ b \cos \alpha + a \cos \beta = c \end{cases} \iff \begin{bmatrix} 0 & c & b \\ c & 0 & a \\ b & a & 0 \end{bmatrix} \begin{bmatrix} \cos \alpha \\ \cos \beta \\ \cos \gamma \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Pela Regra de Cramer devemos considerar  $A = \begin{bmatrix} 0 & c & b \\ c & 0 & a \\ b & a & 0 \end{bmatrix}$  e  $A_\alpha = \begin{bmatrix} a & c & b \\ b & 0 & a \\ c & a & 0 \end{bmatrix}$ , donde

$\det A = 2abc$  e  $\det A_\alpha = -a^3 + ab^2 + ac^2$ . Portanto,

$$\cos \alpha = \frac{\det A_\alpha}{\det A} = \frac{-a^3 + ab^2 + ac^2}{2abc} = \frac{-a^2 + b^2 + c^2}{2bc}$$

ou seja,  $a^2 = b^2 + c^2 - 2bc \cos \alpha$  e isto é justamente o que queremos mostrar.

## §38.6 Exercícios

### Exercícios Introdutórios

**Exercício 38.1.** Usando a forma escalonada matriz, calcule o determinante de cada um das matrizes abaixo:

$$(i) \begin{bmatrix} 2 & -1 \\ -4 & 3 \end{bmatrix}$$

$$(ii) \begin{bmatrix} 0 & -1 & -2 \\ -1 & 0 & -3 \\ 2 & 1 & 2 \end{bmatrix}$$

$$(iii) \begin{bmatrix} 0 & 1 & -5 \\ 2 & 6 & 3 \\ 0 & 3 & 5 \end{bmatrix}$$

$$(iv) \begin{bmatrix} 5 & -1 & 0 & 2 \\ 0 & -3 & -2 & 4 \\ 0 & 0 & -4 & 2 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

$$(v) \begin{bmatrix} 1 & -2 & 1 & 4 \\ 2 & -4 & 0 & 2 \\ 0 & 0 & 1 & 6 \\ 5 & 3 & 4 & 0 \end{bmatrix}$$

$$(vi) \begin{bmatrix} 1 & -2 & 1 & 4 & 2 \\ 2 & 4 & 2 & 0 & -4 \\ 3 & 8 & -6 & 5 & 1 \\ 0 & 0 & 4 & 1 & 3 \\ 2 & 2 & 0 & -3 & 4 \end{bmatrix}$$

**Exercício 38.2.** Usando a Regra de Cramer, resolva os seguintes sistemas lineares:

$$(i) \begin{cases} 3x + 6y = 5 \\ 12x - 2y = -12 \end{cases}$$

$$(ii) \begin{cases} 2x - 4y - 3z = -20 \\ 7x + 9z = 50 \\ -6x + 3y - 9z = -39 \end{cases}$$

$$(iii) \begin{cases} 4x + y + z + w = 6 \\ 3x + 7y - z + w = 1 \\ 7x + 3y - 5z + 8w = -3 \\ x + y + z + 2w = 3 \end{cases}$$

**Exercício 38.3.** Encontre para quais valores de  $\lambda \in \mathbb{R}$  temos  $\det(A - \lambda I) = 0$ , onde  $A = \begin{bmatrix} 3 & 1 \\ 4 & 6 \end{bmatrix}$ . Calcule a soma e o produto desses números e compare com  $\text{tr}(A)$  e  $\det(A)$ .

**Exercício 38.4.** Para quais valores de  $x \in \mathbb{R}$  temos  $\det \begin{bmatrix} x & -1 \\ 3 & 1-x \end{bmatrix} = \det \begin{bmatrix} 1 & 0 & -3 \\ 2 & x & -6 \\ 1 & 3 & 1-5x \end{bmatrix}$ .

**Exercício 38.5.** A função  $\min(i, j)$  calcula o mínimo entre os valores  $i$  e  $j$ . Considere a matriz  $A = [a_{ij}]_{5 \times 5}$ , onde  $a_{ij} = \frac{1}{\min(i, j)}$ . Calcule o determinante de  $A$ .

### Exercícios de Aprofundamento

**Exercício 38.6.** Decida quais das matrizes abaixo são invertíveis. Caso uma delas seja, encontre sua inversa através do uso da matriz adjunta.

$$(i) \begin{bmatrix} 2 & 5 & 5 \\ -1 & -1 & 0 \\ 2 & 4 & -3 \end{bmatrix}$$

$$(ii) \begin{bmatrix} 2 & 0 & 2 \\ 0 & 3 & 2 \\ -2 & 0 & -4 \end{bmatrix}$$

$$(iii) \begin{bmatrix} 2 & -3 & 5 \\ 0 & 1 & -3 \\ 0 & 0 & -4 \end{bmatrix}$$

$$(iv) \begin{bmatrix} 1 & 4 & 3 & 1 \\ 2 & 5 & 2 & 2 \\ 1 & 3 & 8 & 9 \\ 1 & 3 & 2 & 2 \end{bmatrix}$$

**Exercício 38.7.** Se  $\det \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = -5$ , então calcule:

$$(i) \det \begin{bmatrix} g & h & i \\ d & e & f \\ a & b & c \end{bmatrix}$$

$$(iii) \det \begin{bmatrix} a & b & c \\ d & e & f \\ 2a & 2b & 2c \end{bmatrix}$$

$$(v) \det \begin{bmatrix} a+d & b+e & c+f \\ -d & -e & -f \\ g & h & i \end{bmatrix}$$

$$(ii) \det \begin{bmatrix} d & e & i \\ g & h & i \\ a & b & c \end{bmatrix}$$

$$(iv) \det \begin{bmatrix} 3a & 3b & 3c \\ -d & -e & -f \\ 4g & 4h & 4i \end{bmatrix}$$

$$(vi) \det \begin{bmatrix} -3a & -3b & -3c \\ d & e & f \\ g-4d & h-4e & i-4f \end{bmatrix}$$



**Exercício 38.8.** Sem calcular diretamente os determinantes, mostre que:

(a)

$$\det \begin{bmatrix} 2a+p & 2b+q & 2c+r \\ 2p+x & 2q+y & 2r+z \\ 2x+a & 2y+b & 2z+c \end{bmatrix} = 9 \det \begin{bmatrix} a & b & c \\ p & q & r \\ x & y & z \end{bmatrix}$$

(b)

$$\det \begin{bmatrix} a_1+b_1 & a_1-b_1 & c_1 \\ a_2+b_2 & a_2-b_2 & c_2 \\ a_3+b_3 & a_3-b_3 & c_3 \end{bmatrix} = -2 \det \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix}$$

**Exercício 38.9.** Considerando a matriz  $A = \begin{bmatrix} 0 & 2 & 0 & 0 & 5 \\ 0 & 7 & 0 & 2 & 0 \\ 2 & 3 & 4 & 0 & 7 \\ 0 & 0 & 5 & 0 & 0 \\ 1 & 3 & 1 & 1 & 1 \end{bmatrix}$ . Encontre os seguintes determinantes:

(i)  $\det(A^T)$

(ii)  $\det(5A)$

(iii)  $\det(10A^{-1})$

### Exercícios Avançados

**Exercício 38.10.** Seja  $a$  um número real qualquer. Para todo  $n \geq 3$  defina  $D_n$  como o determinante da matriz

$$D_n = \det \begin{bmatrix} a & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & a & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & a & 1 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -1 & a & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 & a & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & -1 & a \end{bmatrix}_{n \times n}$$

(i) Calcule  $D_3, D_4$  e  $D_5$ .(ii) Mostre que para todo  $n \geq 5$  vale

$$D_n = a \cdot D_{n-1} + D_{n-2}.$$

**Exercício 38.11.** Considere  $x_1, x_2, \dots, x_n$  reais quaisquer. Considere a matriz  $A = [a_{ij}]_{n \times n}$ , onde

$$a_{ij} = \begin{cases} 1 + x_j, & \text{se } i = j \\ x_j, & \text{se } i \neq j \end{cases}.$$

Mostre que o determinante da matriz  $A$  é  $1 + x_1 + x_2 + x_3 + \cdots + x_n$ .

**Exercício 38.12.** Considere  $x_1 = 0$  e  $x_2, x_3, \dots, x_n$  reais quaisquer. Considere a matriz  $A = [a_{ij}]_{n \times n}$ , onde

$$a_{ij} = \begin{cases} 1 + x_i, & \text{se } i = j \\ 1, & \text{se } i \neq j \end{cases}.$$

Mostre que o determinante da matriz  $A$  é  $x_2 x_3 x_4 \cdots x_n$ .

**Exercício 38.13.** Se uma matriz quadrada é tal que a soma de cada linha é igual a zero, então esta matriz não é invertível.

**Exercício 38.14.** Se  $A$  for invertível, então  $\text{Adj}(A)$  é invertível e  $[\text{Adj}A]^{-1} = \frac{1}{\det A} A = \text{Adj}(A^{-1})$