
Pequeno Manual de Estruturas e Padrões Clássicos (Listas 6 e 7)

Aqui estão as "receitas de bolo" que mais se repetiram nos exercícios, agora com as correções e exemplos práticos que discutimos.

1. O Laço de Leitura com Sentinela

Esta foi a estrutura mais comum para ler uma quantidade indefinida de dados.

- **O que é:** Um laço enquanto que continua lendo dados até que um valor especial, o "**sentinela**", seja digitado (como ' _ ' ou um espaço).
- **Como se faz:** Usamos a técnica da "leitura prévia". Lemos o primeiro dado *antes* do laço para ter o que testar na condição, e lemos o próximo dado sempre na *última linha dentro* do laço.
- **Exemplo Concreto (Exercício 4 da Lista 6 - ClassificaCaracter):**
leia(caracter_lido) // Leitura prévia
enquanto caracter_lido <> ' ' faca
 // ... classifica o caracter_lido ...
 leia(caracter_lido) // Lê o próximo para a próxima iteração
fimenquanto

2. O Padrão "Construtor de Strings"

Esta é a base de toda a manipulação de strings na Lista 7, respeitando o conceito de **imutabilidade**.

- **O que é:** Criar uma nova string que é o resultado de uma modificação (filtragem, adição, etc.) na string original.
- **Como se faz:**
 1. Inicializamos uma nova string como vazia (ex: nova_frase <- "").
 2. Percorremos a string original com um laço.
 3. A cada caractere que passa na nossa regra, nós o adicionamos à nova string usando a concatenação (nova_frase <- nova_frase + caractere).
- **Exemplo Concreto (Exercício 6 da Lista 7 - Retira Espaços):**
 - Começamos com nova_frase <- "". Dentro de um laço que percorria a frase original, a cada caractere c que não era um espaço, fazíamos nova_frase <- nova_frase + c. No final, a nova_frase continha a string original sem os espaços.

3. O Padrão "Fiscal de Sequência" (Validação)

Vários problemas pediram para verificar se uma sequência inteira obedecia a uma regra. Para isso, usamos bandeiras (flags) lógicas.

- **a) A Bandeira "Inocente até que se Prove o Contrário"**
 - **O que é:** A melhor técnica para validar regras. Começamos assumindo que a sequência é um "sucesso" e procuramos ativamente por uma única falha.
 - **Exemplo Concreto (Exercício 8 da Lista 6 - Sequência Crescente):**
 - Iniciamos com `sequencia_valida <- verdadeiro`. Dentro do laço, se encontrássemos um `caracter_atual <= caracter_anterior`, imediatamente fazíamos `sequencia_valida <- falso`. No final, a resposta era o valor desta bandeira.
- **b) O "Checklist" de Bandeiras**
 - **O que é:** Usado quando precisamos garantir que vários eventos diferentes aconteceram, em qualquer ordem.
 - **Exemplo Concreto (Exercício 9 da Lista 6 - Todas as Vogais):**
 - Criamos 5 bandeiras: `achou_a`, `achou_e`, etc., todas falso. Dentro do laço, ao encontrar um 'a' ou 'A', fazíamos `achou_a <- verdadeiro`. No final, o "sucesso" era a condição `achou_a e achou_e e achou_i e achou_o e achou_u`.

4. O Padrão "Máquina de Estados" (com Memória)

Esta é uma técnica um pouco mais avançada, para quando a regra do item *atual* depende do que aconteceu no passo *anterior*.

- **O que é:** Usar uma variável para guardar a "memória" do estado anterior.
- **Como se faz:** Dentro do laço, após processar o item atual, atualizamos uma variável de estado (ex: `ultimo_foi_espaco <- verdadeiro`) que será consultada na próxima iteração do laço.
- **Exemplo Concreto (Exercício 7 da Lista 7 - Transforma Espaços Duplicados):**
 - Usamos uma bandeira `anterior_foi_espaco`. Quando o caractere atual era uma letra, definíamos `anterior_foi_espaco <- falso`. Quando era um espaço, definíamos `anterior_foi_espaco <- verdadeiro`. A decisão de adicionar ou não um novo espaço na iteração seguinte dependia do valor dessa bandeira.

5. O Padrão "Cirurgião de Bits"

Na Lista 6, fomos fundo na manipulação de bits. A estratégia era quase sempre a de uma "cirurgia".

- **O que é:** Isolar e manipular bits específicos de um número (código ASCII) sem afetar os

outros.

- **Como se faz:** Usamos as "ferramentas cirúrgicas":
 - **E (AND):** Para "isolar" ou "desligar" bits.
 - **OU (OR):** Para "juntar" ou "ligar" bits.
 - **NAO (NOT):** Para inverter todos os bits.
 - **shr e shl (Deslocamento):** Para "mover" bits de uma posição para outra.
- **Exemplo Concreto (Exercício 21 da Lista 6 - Troca Bits do Caractere):**
 - Para trocar o bit 7 com o bit 1 de um `codigo_original`, fizemos uma 'cirurgia':
 1. `bit7 <- codigo_original e 128` (Isolamos o bit 7)
 2. `bit1 <- codigo_original e 2` (Isolamos o bit 1)
 3. `corpo <- codigo_original e 125` (Isolamos o resto)
 4. `bit7_movido <- bit7 shr 6` (Movemos o bit 7 para a posição 1)
 5. `bit1_movido <- bit1 shl 6` (Movemos o bit 1 para a posição 7)
 6. `novο_codigo <- corpo ou bit7_movido ou bit1_movido` (Recombinamos tudo)