**UNIVERSIDADE DE FRANCA – UNIFRAN**

FABIO TOLEDO BONEMER DE SALVI

**Monitoramento do Preço da Cesta Básica Usando Pseudocódigo**

Experiência Prática I

ARARAQUARA – SÃO PAULO

2025

FABIO TOLEDO BONEMER DE SALVI

**Monitoramento do Preço da Cesta Básica Usando Pseudocódigo**

Experiência Prática I

Trabalho apresentado ao curso de Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da Universidade de Franca – UNIFRAN, como requisito para obtenção de nota na disciplina Algoritmos e Pensamento Computacional.

ARARAQUARA – SÃO PAULO

2025

1. **Introdução**

O presente trabalho tem como objetivo desenvolver a habilidade de construção de algoritmos utilizando fluxograma e pseudocódigo. Para isso foi proposto a implementação de um problema do cotidiano real da vida do brasileiro envolvendo a análise da variação de preços. Este problema foi implementado em forma de fluxograma no software *Flowgorithm*.

1. **Descrição do Problema**

Desenvolver uma ferramenta que realize a análise da variação de preço dos produtos de uma cesta básica para ser utilizada por órgãos de defesa do consumidor.

O objetivo é verificar se os preços dos produtos estão aumentando de forma abusiva e gerar relatórios que ajudem na tomada de decisão.

* 1. **Requisitos que devem ser atendidos**

Para dada produto:

* + - **Ler**: nome, preço anterior e preço atual;
    - **Calcular:** Variação percentual;
    - **Exibir:** 
      * Se o preço **aumentou**, **diminuiu** ou **permaneceu estável**;
      * Se o aumento foi **superior a 10%**, sinalizar como **abuso**.
  1. **Definição das variáveis**

Todas as variáveis foram utilizadas em forma de **vetor(array)** para otimizar o algoritmo e permitir que o usuário defina o número de produtos que será passado para análise.

Segue as varáveis utilizadas neste algoritmo:

* + - **numeroProdutos**: tipo **real (número decimal)**.

Variável utilizada para definir o número de produtos que serão analisados.

* + - **nome**: tipo **caractere (texto/string)**.

Array utilizado para armazenar o nome dos produtos.

* + - **precoAnterior**: tipo **real (número decimal)**.

Array utilizado para armazenar o preço dos produtos antes da atualização.

* + - **precoAtual**: tipo **real (número decimal)**.

Array utilizado para armazenar o preço dos produtos após a atualização.

* + - **variacao**: tipo **real (número decimal)**.

Array utilizado para armazenar a variação percentual do preço dos produtos.

* + - **situacao**: tipo **caractere(texto/string)**.

Array utilizado para armazenar a classificação da variação, podendo ser “AUMENTO”, “AUMENTO ABUSIVO”, “QUEDA” ou “ESTÁVEL”.

* 1. **Definição das funções**
     + **estaEmBranco:** verifica se uma variável do tipo **caractere(texto/string)** possui apenas espaços em branco.

A função recebe como argumento uma variável **texto** do tipo **caractere(texto/string)** onde é passado o nome do produto e retorna uma variável do tipo **booleana** com o valor **true** para o caso do usuário não ter digitado o nome do produto e retorna **false** para o caso de o usuário ter digitado o nome do produto.

* + - **variacaoPercentual:** calcula a variação percentual de um produto utilizando seus valores de preço do mês anterior e do preço do mês atual.

A função recebe duas variáveis do tipo **real (número decimal)** representando os valores de **precoAnterior** e **precoAtual** e calcula a variação percentual do preço conforme a seguinte expressão:

|  |  |
| --- | --- |
|  | **(Eq. 1)** |

É retornado um valor **real (número decimal)** com o valor armazenado na variável **variacao**.

* + - **situacaoProduto:** determina o tipo de variação de preço utilizando o resultado o calcula da variação percentual do valor do produto.

As opções de variação de preço são:

* + - * QUEDA: para uma variação de preço negativa.
      * ESTÁVEL: para uma variação de preço nula.
      * AUMENTO: para uma variação de preço positiva e até 10%.
      * AUMENTO ABUSIVO: para uma variação de preço positiva acima de 10%.

1. **Fluxograma**
2. **Pseudocódigo**

O pseudocódigo foi gerado utilizando o software *Flowgorithm*(COOK, Devin).

Primeiro foi necessário montar um fluxograma que descrema o problema.

Após esta etapa foi possível obter o pseudocódigo do algoritmo desenvolvido.

Abaixo é apresentado uma descrição dos componentes que formam o pseudocódigo, exemplificando a forma de utilização.

Após esta descrição é apresentado o pseudocódigo com as funções que formam o fluxograma que soluciona o problema proposto.

* + - **Descrição dos componentes e estruturas que formam o pseudocódigo:**
      * **Comentário:** um comentário é inserido após o pseudocódigo **... (três pontos seguidos)**.

... Comentário.

* + - * **Definindo uma variável:** uma varável é definida utilizando o pseudocódigo **Definir** seguido do tipo de variável e nome da variável.

**Definir TipoDado** nomeVariaval

* + - * **Definindo uma variável vetor:** uma variável vetor é definida utilizando o pseudocódigo **Definir** seguido do tipo de variável, do pseudocódigo **Arranjo** e nome da variável.

**Definir TipoDado** **Arranjo** nomeVariaval

* + - * **Função saída:** uma função saída é definida utilizando o pseudocódigo **Saída** seguido da expressão que será analisada e mostrada em um prompt virtual.

**Saída** expressão

* + - * **Função entrada:** uma função entrada é definida utilizando o pseudocódigo **Entrada** seguido da variável que armazenará o dado digitado no prompt virtual.

**Entrada** variavel

* + - * **Laço de repetição enquanto (while):**  o laço de repetição enquanto (while) é delimitado pelos pseudcódigos **Enquanto** e **Fim**. A condição de execução do laço é inserida após o pseudcódigo **Enquanto**.

**Enquanto** condicao

**Fim**

* + - * **Função principal:** a função principal é denominada por “Principal”, sendo delimitada pelos pseudocódigos **Função** e **Fim**.

**Função** Principal

**Fim**

* + - * **Função que não retorna valor:** uma função que não retorna valor é delimitada pelos pseudocódigos **Função** e **Fim**.

**Função** nomeFuncao

**Fim**

* + - * **Função que retorna um valor:** uma função que retorna um valor é delimitada pelos pseudocódigos **Função** e **Retornar**, retornando uma variável declarada no corpo da função.

**Função** nomeFuncao

**Definir TipoDado** nomeVariaval

**Retornar TipoDado** nomeVariaval

* + - * **Função que recebe um argumento e retorna um valor:** uma função que retorna valor é delimitada pelos pseudocódigos **Função** e **Retornar**, retornando uma variável declarada no corpo da função.

**Função** nomeFuncao

**Retornar**

* + - **Função principal:**

**Função** Principal

**Saída** "Análise da Variação de Preços de Produtos 1.0"

**Saída** "Desenvolvido por Fabio Toledo Bonemer De Salvi"

**Saída** "Sobre: Ferramenta para análise da variação de preços dos produtos que compõem uma cesta básica."

... Número de produtos que serão analisados.

**Definir Inteiro** numProdutos

**Saída** "Digite o número de produtos que serão analisados (número positivo não nulo):"

... Solicita ao usuário que entre com um número de produtos que serão analisados.

**Entrada** numProdutos

**Enquanto** numProdutos <= 0

... O loop while certifica que seja digitado um número positivo não nulo solicitando ao usuário que entre com o valor correto.

**Saída** "Digite o número de produtos que serão analisados (número positivo não nulo):"

**Entrada** numProdutos

**Fim**

... Array para armazenar o nome dos produtos.

**Definir** **Caracteres** **Arranjo** nome[numProdutos]

... Array para armazenar os valores dos produtos no mês anterior.

**Definir Real Arranjo** precoAnterior[numProdutos]

... Array para armazenar os valores dos produtos no mês atual.

**Definir Real Arranjo** precoAtual[numProdutos]

... Array para armazenar a variação de preço de cada produto.

**Definir Real Arranjo** variacao[numProdutos]

... Array para armazenar a classificação da variação de preço de cada produto.

**Definir Caracteres Arranjo** situacao[numProdutos]

... Indice para acessar as variáveis dentro do laço for.

**Definir Inteiro** indiceProduto

**Para** indiceProduto = 0 **até** numProdutos-1

... Laço para o usuário entrar com os valores de nome do produto, preço no mês anterior e preço no mês atual referente aos produtos.

... Entrar com o nome do produto.

**Saída** "Digite o nome do produto " & indiceProduto+1 & ":"

**Entrada** nome[indiceProduto]

**Enquanto** estaEmBranco(nome[indiceProduto])

... Loop executado até o usuário digitar o nome do produto. Este loop tenta simular a função trim().

**Saída** "Digite o nome do produto " & indiceProduto+1 & ":"

**Entrada** nome[indiceProduto]

**Fim**

... Entrar com o preço anterior do produto.

**Saída** "Digite o preço do produto " & indiceProduto+1 & " no mês anterior:"

**Entrada** precoAnterior[indiceProduto]

**Enquanto** precoAnterior[indiceProduto] <= 0

... Loop executado até o usuário digitar um valor positivo não nulo.

**Saída** "Digite o preço do produto " & indiceProduto+1 & " no mês anterior:"

Entrada precoAnterior[indiceProduto]

**Fim**

... Entrar com o preço atual do produto.

Saída "Digite o preço do produto " & indiceProduto+1 & " no mês atual:"

Entrada precoAtual[indiceProduto]

Enquanto precoAtual[indiceProduto] <= 0

... Loop executado até o usuário digitar um valor positivo não nulo.

Saída "Digite o preço do produto " & indiceProduto+1 & " no mês atual:"

Entrada precoAtual[indiceProduto]

**Fim**

... Com as informações sobre os precos de cada produto é calculado a variação percentual do preço e a classificação desta variação de preço.

Atribuir variacao[indiceProduto] = variacaoPercentual(precoAnterior[indiceProduto], precoAtual[indiceProduto])

Atribuir situacao[indiceProduto] = situacaoProduto(variacao[indiceProduto])

**Fim**

Para indiceProduto = 0 até numProdutos-1

... Mostra no terminal os produtos inseridos para análise destacando o nome do produto, o seu valor no mês anterior, o seu valor no mês atual, a variação percentual do seu valor e a análise desta variação.

Saída "Produto " & indiceProduto+1 & ": " & nome[indiceProduto]

Saída " Valor mês anterior: R$" & ToFixed(precoAnterior[indiceProduto], 2)

Saída " Valor mês atual: R$" & ToFixed(precoAtual[indiceProduto], 2)

Saída " Variação : " & ToFixed(variacao[indiceProduto], 2) & "%"

Saída " Situação: " & situacao[indiceProduto]

**Fim**

**Fim**

Função estaEmBranco (Caracteres texto)

Definir Lógico estaEmBranco

Atribuir estaEmBranco = true

**Definir Inteiro** indice

Para indice = 0 até len(texto)-1

Se Char(texto, indice) != " "

... Atribui falso para a variável isBlank se houver algum caracter diferente de espaço.

Atribuir estaEmBranco = false

**Fim**

**Fim**

Retornar Lógico estaEmBranco

**Fim**

Função situacaoProduto (Real variacao)

Definir Caracteres classificacao

Se variacao > 0

Se variacao > 10

Atribuir classificacao = "AUMENTO ABUSIVO"

Senão

Atribuir classificacao = "AUMENTO"

**Fim**

Senão

Se variacao < 0

Atribuir classificacao = "QUEDA"

Senão

Atribuir classificacao = "ESTÁVEL"

**Fim**

**Fim**

Retornar Caracteres classificacao

**Fim**

Função variacaoPercentual (Real precoAnterior, Real precoAtual)

Definir Real variacao

Atribuir variacao = ((precoAtual - precoAnterior)/precoAnterior)\*100

Retornar Real variacao

**Fim**

1. **Teste de Execução**
2. **Conclusão**
3. **Referências**

COOK, Devin. *Flowgorithm: Flowchart Programming Language*. Versão 4.5. Disponível em: https://www.flowgorithm.org/ Acesso em: 19 set. 2025.

UNIVERSIDADE DE FRANCA (UNIFRAN). *Experiências Práticas – Fundamentos da Introdução à Programação e ao Pensamento Computacional*. Franca: UNIFRAN, [2025]. Apostila da disciplina Algoritmos e Pensamento Computacional, Curso EAD. Disponível em: https://www.unifran.edu.br/. Acesso em: 19 set. 2025.