SESC ESCOLA

MARCOS VINICIUS PEREIRA

PESQUISA SOBRE ALGORITMO E LOGICA DE PROGRAMAÇÃO

DIA 06/032024

Cuiabá

2024

MARCOS VINICIUS PEREIRA

PESQUISA SOBRE ALGORITMO E LOGICA DE PROGRAMAÇÃO

DIA 06/032024

trabalho sobre algoritmo e logicas

de programação,

SESC ESCOLA

APRESENTA O CURSO DE JOGOS DIGITAIS

Orientador(a)Wanderson Timoteo

Cuiabá

2024

Sumário

[função do algoritmo 1](#_Toc160610419)

[A importância da programação é relevante. 1](#_Toc160610420)

[Lógica de Programação e sua Relação com Algoritmos: 1](#_Toc160610421)

[As estruturas de controle: 2](#_Toc160610422)

[Exemplos de Organismos que utilizam Estruturas de Controle: 2](#_Toc160610423)

[Definição dos tipos de dados básicos: 2](#_Toc160610424)

[Aplicações e Modularidade 3](#_Toc160610425)

[Valor da Modularização na Escrita de Algoritmos: 3](#_Toc160610426)

[Exemplos de funções básicas e sua utilização em algoritmos mais complexos: 3](#_Toc160610427)

[Aplicação da função em um algoritmo mais complexo 4](#_Toc160610428)

## função do algoritmo

Um algoritmo é um conjunto limitado de instruções bem definidas e ordenadas que descrevem um processo computacional ou uma solução para um problema. Ele é o fundamento para a resolução de problemas computacionais, estabelecendo uma série de passos lógicos que, se seguidos de forma correta, resultam na solução desejada.

# A importância da programação é relevante.

Os algoritmos são cruciais na programação, uma vez que proporcionam uma estrutura lógica e organizada para a implementação de soluções em software. Eles possibilitam que os programadores expressem suas ideias de maneira clara e precisa, o que torna o desenvolvimento, a compreensão e a manutenção do código mais fáceis.

## Lógica de Programação e sua Relação com Algoritmos:

A capacidade de pensar de maneira estruturada e lógica é crucial para a criação de algoritmos eficientes. Ela envolve o uso de técnicas como sequência, seleção e repetição para controlar o fluxo de execução do programa e alcançar os resultados desejados.

## As estruturas de controle:

As estruturas de controle definem como as instruções em um programa são executadas, permitindo que ele tome decisões e repita as ações de acordo com o necessário. São estruturas de sequência, seleção (como condicionais) e repetição (como loops)

## Exemplos de Organismos que utilizam Estruturas de Controle:

Algoritmo de ordenação (usando repetição para iterar os elementos e seleção para compará-los e trocá-los)

Algo semelhante a uma busca binária (que utiliza a repetição para dividir a lista e a seleção para determinar em qual metade buscar)

Algoritmo para calcular a média de notas (utilizando sequência para somar as notas, seleção para verificar se há notas inválidas e repetição para inferir sobre as notas)

Tipos de Informações e Variáveis:

Os tipos de dados definem os múltiplos valores que podem ser manipulados em um programa, tais como inteiros, pontos flutuantes, booleanos, entre outros. As variáveis são espaços de memória reservados para armazenar valores de determinados tipos de dados durante a execução de um programa.

## Definição dos tipos de dados básicos:

Inteiros (int): números inteiros, como 1, 2, -3, etc.

Pontos Flutuantes (float): números com parte decimal, como 3.14, -0.5, etc.

Booleanos (bool): valores verdadeiro (true) ou falso (false).

Aplicações e Modularidade

Uma função é um bloco de código que executa uma tarefa específica e pode ser chamado de outros elementos do programa. A modularização consiste em dividir um programa em partes menores (funções) para simplificar o desenvolvimento, a manutenção e a reutilização do código.

### Valor da Modularização na Escrita de Algoritmos:

A modularização ajuda a organizar o código, tornando-o mais claro, fácil de compreender e manter. Também estimula o uso de códigos, pois as funções podem ser usadas em diferentes partes do programa, reduzindo a redundância e facilitando a correção de erros.

## Exemplos de funções básicas e sua utilização em algoritmos mais complexos:

# Função simples para calcular o quadrado de um número

def calcular\_quadrado(numero):

return numero \*\* 2

# Utilização da função em um algoritmo mais complexo

def calcular\_media\_quadrados(lista\_numeros):

soma\_quadrados = 0

for num in lista\_numeros:

soma\_quadrados += calcular\_quadrado(num)

return soma\_quadrados / len(lista\_numeros)

# Exemplo de uso

numeros = [1, 2, 3, 4, 5]

media\_quadrados = calcular\_media\_quadrados(numeros)

print("A média dos quadrados dos números é:", media\_quadrados)

Aplicação da função em um algoritmo mais complexo.

# Declaração de variáveis

inteiro = 10

ponto\_flutuante = 3.14

booleano = True

# Utilização de variáveis

resultado = inteiro \* ponto\_flutuante

se\_for\_verdadeiro = booleano and True