Jornada Tech – Oficina "Lógica de Programação"

Sumário

1. Introdução à Lógica de Programação
    - 1.1. O que é Lógica de Programação – Definição e importância
    - 1.2. Pensamento Computacional

2. Algoritmos
    - 2.1. O que são Algoritmos
    - 2.2. Estruturas Básicas de Algoritmos
    - 2.3. Fluxogramas e Pseudocódigo

3. Decisões
    - 3.1. Instruções Condicionais
    - 3.2. Exemplo de Uso de Condições

4. Laços de Repetição
    - 4.1. O que são Laços de Repetição
    - 4.2. Tipos de Laços (for, while)
    - 4.3. Exemplos de Laços de Repetição

5. Resolução de Problemas com Lógica de Programação
    - 5.1. Abordagem para Resolver Problemas
    - 5.2. Exemplos Práticos de Problemas

6. Conclusão
    - 6.1. Importância da Lógica de Programação
    - 6.2. Como a Lógica de Programação é aplicada em diversas áreas
7. Referências Bibliográficas

1. Introdução à Lógica de Programação

1.1 O que é Lógica de Programação – Definição e Importância

A lógica de programação é um conjunto de regras e processos utilizados para criar programas de computador que executam tarefas específicas. Ela envolve analisar problemas e definir soluções eficientes. A lógica de programação é essencial para o desenvolvimento de qualquer software, seja para sistemas, jogos ou aplicativos.

1.2 Pensamento Computacional

O pensamento computacional é a habilidade de resolver problemas de forma estruturada e lógica, utilizando os conceitos da computação. Ele envolve a decomposição de problemas em partes menores, a criação de algoritmos e o uso de estruturas de dados para resolver problemas de forma eficiente.

2. Algoritmos

2.1 O que são Algoritmos

Algoritmos são sequências de passos ou instruções que visam resolver um problema. Eles são independentes da linguagem de programação e podem ser representados por fluxogramas, pseudocódigo ou diretamente em uma linguagem de programação, como C.

2.2 Estruturas Básicas de Algoritmos

Todo algoritmo possui três componentes principais:

- Entrada: Os dados fornecidos ao algoritmo.
- Processamento: As operações que o algoritmo realiza sobre os dados.
- Saída: O resultado gerado pelo algoritmo.

2.3 Fluxogramas e Pseudocódigo

- Fluxograma: Representação gráfica de um algoritmo. Utiliza símbolos como retângulos, círculos e losangos para representar ações e decisões.
- Pseudocódigo: Representação de um algoritmo usando uma linguagem próxima da natural, mas com estrutura definida para ser facilmente convertido em código.

3. Decisões

3.1 Instruções Condicionais

Instruções condicionais permitem que o programa tome decisões com base em condições específicas. As principais instruções condicionais em C são if, else if e else.

- Sintaxe em C:

```
C/C++
if (condição) {
    // Código a ser executado se a condição for verdadeira
} else {
    // Código a ser executado se a condição for falsa
}
```

3.2 Exemplo de Uso de Condições

Aqui está um exemplo de como verificar se um número é positivo ou negativo em C:

```
C/C++
#include <stdio.h>

int main() {
    int numero;
    printf("Digite um número: ");
    scanf("%d", &numero);

    if (numero > 0) {
        printf("Positivo\n");
    } else {
        printf("Negativo\n");
    }

    return 0;
}
```

4. Laços de Repetição

4.1 O que são Laços de Repetição

Os laços de repetição permitem executar um bloco de código várias vezes até que uma condição seja atendida. Em C, existem dois tipos principais de laços: for e while.

4.2 Tipos de Laços (for, while)

- Laço for: Utilizado quando se sabe de antemão quantas vezes o código precisa ser repetido. Sintaxe do laço for:

```
C/C++
for (inicialização; condição; incremento) {
    // Código a ser executado
}
```

- Laço while: Usado quando não se sabe o número exato de iterações e a repetição depende de uma condição. Sintaxe do laço while:

```
C/C++
while (condição) {
    // Código a ser executado enquanto a condição for verdadeira
}
```

4.3 Exemplos de Laços de Repetição

- Laço for para somar números de 1 a 5:

```
C/C++
#include <stdio.h>

int main() {
    int soma = 0;
    for (int i = 1; i <= 5; i++) {
        soma += i;
    }
    printf("A soma de 1 a 5 é: %d\n", soma);
    return 0;
}
```

- Laço while para contar de 10 até 1:

```
C/C++
#include <stdio.h>

int main() {
    int i = 10;
    while (i > 0) {
        printf("%d\n", i);
        i--;
    }
    return 0;
}
```

5. Resolução de Problemas com Lógica de Programação

5.1 Abordagem para Resolver Problemas

1. Compreender o Problema: Entenda o que é necessário resolver.
2. Dividir em Subproblemas: Decomponha o problema em partes menores.
3. Criar o Algoritmo: Desenvolva um algoritmo para cada subproblema.
4. Implementar e Testar: Escreva o código e teste para garantir que funcione corretamente.

5.2 Exemplos Práticos de Problemas

- Problema: Calcular a soma dos números de 1 a N.

```
C/C++
#include <stdio.h>

int main() {
    int soma = 0, N;
    printf("Digite o valor de N: ");
    scanf("%d", &N);

    for (int i = 1; i <= N; i++) {
        soma += i;
    }
```

```
    printf("A soma dos números de 1 até %d é: %d\n", N, soma);
    return 0;
}
```

6. Conclusão

6.1 Importância da Lógica de Programação

A lógica de programação é fundamental para qualquer área da computação, pois ela ensina como estruturar o pensamento e resolver problemas de forma eficiente e clara. Ela é a base para o desenvolvimento de qualquer software ou sistema.

6.2 Como a Lógica de Programação é aplicada em diversas áreas

A lógica de programação é usada em diversas áreas, como desenvolvimento de software, automação de processos, inteligência artificial, jogos, e muito mais. Ela facilita a resolução de problemas complexos e é aplicada tanto em sistemas simples quanto em tecnologias avançadas.

7. Referências Bibliográficas

1. Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language. 2nd ed. Prentice Hall.

2. Gaddis, T. (2013). Starting Out with C++: From Control Structures through Objects. 8th ed. Pearson Education.

3. Deitel, H. M., & Deitel, P. J. (2011). C How to Program. 8th ed. Pearson.

4. Zellweger, J. (2017). Programming in C: A Hands-on Introduction. W.W. Norton & Company.

5. King, K. N. (2013). C Programming: A Modern Approach. 2nd ed. W.W. Norton & Company.