Lógica de Programação com Python - Teoria Expandida

1. Estrutura Básica: Entrada, Processamento e Saída

Todo programa tem uma estrutura básica, que pode ser dividida em três etapas principais:

Entrada de Dados:
A entrada é a parte do programa onde recebemos informações do usuário. Essas informações são capturadas para que o programa possa utilizá-las no processamento. Em Python, usamos a função input() para essa tarefa. Por padrão, tudo que é recebido pelo input() é uma string (texto), mesmo que o usuário digite um número. Por isso, é comum fazer a conversão de tipo com int() ou float().

Processamento:
O processamento é o "coração" do programa, onde as regras, operações, cálculos e decisões são executados. O objetivo é transformar os dados de entrada em um resultado significativo.

Saída de Dados:
A saída representa o resultado final, aquilo que o programa devolve ao usuário após processar os dados. Em Python, utilizamos a função print() para exibir mensagens na tela.

Essa sequência é fundamental para a construção de qualquer programa. Sem ela, não há como interagir com o usuário nem produzir um resultado útil.

2. Estruturas de Seleção (Tomadas de Decisão)

Na programação, muitas vezes precisamos que o sistema tome decisões com base em condições. Por exemplo: se um número é positivo ou negativo, se um aluno foi aprovado ou reprovado, se um usuário tem ou não acesso ao sistema.

Essas decisões são controladas pelas estruturas de seleção, que usam comparações lógicas e palavras-chave como if, elif e else no Python.

Uma estrutura condicional é um bloco de código que só será executado se uma condição for verdadeira.

Caso contrário, o programa pode seguir para outro caminho.

Sintaxe básica:

```
if condição:
    # código se a condição for verdadeira
elif outra_condição:
    # se a anterior for falsa e essa for verdadeira
else:
    # se nenhuma das anteriores for verdadeira
```

Operadores de comparação mais usados:

== igual, != diferente, > maior que, < menor que, >= maior ou igual, <= menor ou igual

As estruturas condicionais permitem que o programa seja inteligente e dinâmico, adaptando seu comportamento conforme os dados que recebe.

3. Estruturas de Repetição (Laços)

Na maioria dos programas, é comum ter que repetir ações várias vezes. Fazer isso manualmente seria ineficiente. As estruturas de repetição, também chamadas de laços, servem para automatizar essas tarefas repetitivas.

Laço while - Repetição Condicional:

O while executa um bloco de código enquanto uma condição for verdadeira. Ou seja, ele verifica a condição antes de cada execução.

Exemplo:

```
contador = 1
while contador <= 5:
    print(contador)
    contador += 1
```

Lógica de Programação com Python - Teoria Expandida

Usado quando não sabemos quantas repetições serão feitas. Requer uma condição de parada.

Laço for - Repetição Contada:

O for é utilizado quando sabemos exatamente quantas vezes queremos que algo aconteça. Ele é muito utilizado com a função range(), que cria uma sequência de números.

Exemplo:

```
for i in range(1, 6):
    print(i)
```

Também pode ter um passo:

```
for i in range(0, 10, 2):
    print(i)
```

O for é ideal para contagens, percorrer listas e gerar estruturas previsíveis.

Considerações Finais:

Dominar entrada, processamento, saída, estruturas de decisão e repetições é essencial para qualquer pessoa que está aprendendo a programar. Com a prática, o raciocínio lógico se desenvolve naturalmente.