Introdução ao Go

Aula 9: Compreendendo os Tipos de Valores

Nesta aula, o professor introduz os **tipos de valores** em Go, mostrando como o **sistema de tipos estático** da linguagem é importante. Ele demonstra como lidar com diferentes tipos de dados (como `int` e `float64`) e realizar operações matemáticas avançadas usando a **função** `math.Pow`.

Principais Conceitos

1. **Tipos de Dados**:
   - Em Go, **todo valor tem um tipo específico**, como `int`, `float64`, `string`, etc.
   - Exemplo:

```go
var investmentAmount = 1000        // Tipo: int (números inteiros)
var expectedReturnRate = 5.5       // Tipo: float64 (números com decimais)
```

2. **Erro de Tipos**:
   - Operações que misturam tipos diferentes (por exemplo, `int` e `float64`) não são permitidas em Go.
   - Exemplo de erro:

```go
var futureValue = investmentAmount * (1 + (expectedReturnRate / 100)) // Erro de tipos!
```

3. **Conversão de Tipos**:
   - Para resolver o erro, é necessário converter explicitamente o tipo.
   - Em Go, funções como `float64()` são usadas para conversão:

```go

```

```
var futureValue = float64(investmentAmount) * (1 + (expectedReturnRate / 100))
```

Adicionando Cálculo com Exponenciação

A fórmula completa para calcular o valor futuro é:

FV = Principal × (1 + Taxa)^t

- `t` é o número de anos (o **expoente** na fórmula).
- Go **não possui um operador de exponenciação** (como `^` em algumas linguagens).
- A função `math.Pow` da biblioteca padrão é usada para elevar um número a uma potência.

Importando o Pacote `math`

Para usar `math.Pow`, é necessário importar o pacote `math` no início do arquivo:

go

```
import "math"
```

Usando `math.Pow`

A função `math.Pow` aceita dois parâmetros:

1. **Base**: O número que será elevado à potência.
2. **Expoente**: O número que será usado como potência.

Exemplo de uso:

go

```
var futureValue = float64(investmentAmount) * math.Pow(1+(expectedReturnRate/100), float64(years))
```

Código Completo da Aula

```go
package main

import (
    "fmt"
    "math"
)

func main() {
    // Declaração das variáveis
    var investmentAmount = 1000         // Valor inicial do investimento (int)
    var expectedReturnRate = 5.5        // Taxa de retorno anual esperada (float64)
    var years = 10                      // Tempo de investimento em anos (int)

    // Cálculo do valor futuro com conversões necessárias
    var futureValue = float64(investmentAmount) * math.Pow(1+
(expectedReturnRate/100), float64(years))

    // Exibição do resultado
    fmt.Printf("O valor futuro do investimento é: %.2f\n", futureValue)
}
```

Explicação do Código

1. **Importação de `math`**:
   - Necessária para usar a função `math.Pow`.
2. **Conversões de Tipos**:
   - `float64(investmentAmount)`: Converte o valor inicial (`int`) para `float64` para ser usado no cálculo.
   - `float64(years)`: Converte o número de anos (`int`) para `float64`, porque `math.Pow` exige que ambos os argumentos sejam `float64`.
3. **Saída Formatada**:
   - A função `fmt.Printf` é usada para exibir o valor com duas casas decimais:

```go
```

```
fmt.Printf("O valor futuro do investimento é: %.2f\n", futureValue)
```

Pontos Importantes

1. **Conversões são Comuns em Go**:
   - Como Go não converte automaticamente tipos diferentes, você precisa gerenciar isso manualmente.
2. **Função `math.Pow`**:
   - Essencial para cálculos que envolvem expoentes.
   - Aceita apenas `float64` como entrada.
3. **Tipagem Forte e Estática**:
   - É um ponto forte de Go, ajudando a prevenir erros em tempo de execução, mas exige atenção ao lidar com tipos diferentes.

Se precisar de mais detalhes sobre o código ou explicação de algum conceito, é só perguntar!