Estruturas de Decisão e Loops

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

Estruturas de Decisão

Estruturas de decisão são utilizadas em programa para que eles tenham algo parecido com as decisões que fazemos todo o dia, “o que comer?”, “o que vestir?”, “esse produto está barato, se sim eu preciso comprar ele?”.

Para nós é fácil realizarmos esse tipo de questionamento e responder, porém um computador precisa ter uma estrutura para tomar qualquer tipo de decisão. Cada linguagem de programação tem pelo menos uma forma de validação de valores com o intuito de realizar decisões.

Estruturas de Decisão

O Java tem algumas formas de escrevermos estruturas de decisão, entre elas temos o IF e o Switch. Podemos ter também o Operador Ternário que nos permite executar decisões em apenas uma linha de código.

Essas operações aparecem em várias linguagens de programação e todas seguem a mesma forma de pensar em decisões, se não, elas são bem parecidas.

IF

If = "se"
Essa é uma das operações mais conhecidas entre todas as linguagens de programação, pois se trata de uma forma simples de representar uma decisão.

O IF funciona da seguinte forma, se passamos dentro dos parênteses uma expressão verdadeira então ele executa o código entre chaves, se passamos uma expressão falsa então ele não executa o código entre parênteses. Dessa forma podemos validar valores.

```
boolean validacao = true;
if(validacao){
    // acontece se validacao for true
} else {
    // acontece se validacao for false
}
```

Expressões Booleanas

Para realizar a validação de valores precisamos entender o tipo Boolean do Java, esse tipo pode ser atribuído diretamente a uma variável da seguinte forma:

boolean a = true;

Essa variável pode ter 2 valores: true(verdadeiro) ou falso(falso). Sendo assim se trata de uma lógica binária, onde temos apenas 2 valores.

O IF pode receber esses valores diretamente, ou podemos “descobrir” esses valores.

Estruturas de Decisão

As expressões booleanas nos permitem validar valores e realizarmos as decisões, sendo assim podemos ter alguns operadores para essa tarefa com valores numéricos:

- "==" -> equals / igual
- ">" -> maior
- "<" -> menor
- ">=" -> maior ou igual
- "<=" -> menor ou igual
- "!=" -> diferente

Else e Else IF

O Else acontece caso o IF não seja Verdadeiro, sendo assim podemos ter um SE(if), e o SE NÃO (else).

Também temos o else if que nos permite realizar diversas validações dentro de um mesmo código.

```
int a = 20;
if(a < 10){
    // acontece se a for menor do que 10
}
else if(a == 10){
    // acontece se a for igual a 10
}
else {
    // acontece se a for maior do que 10
}
```

Operador Ternário

O operador ternário é uma forma de realizarmos um if diretamente em uma variável, sendo assim podemos fazer validações pequenas e que geram um resultado que pode ser armazenado.

Esse tipo de operador não é muito bom para executar códigos diferentes, mas sim para escolha entre valores

```
resultado = (expressão booleana) ? código 1 : código 2;
```

Switch

O SWITCH é uma estrutura de decisão que nos permite ter “caminhos múltiplos” de resultado, é muito parecido com IF, ELSE IF, ELSE, porém a validação é baseada em valores diretos.

Regras do SWITCH:

- Não podemos ter casos(CASE) duplicados;
- O tipo do caso/CASE deve ser o mesmo da variável do SWITCH
- O valor de cada caso/CASE deve ser constante ou literal, não pode ser uma variável
- Podemos ter o comando BREAK para sair do Switch, caso não haja BREAK execução vai continuar
- Podemos ter um DEFAULT que sempre será executado caso não haja um BREAK antes dele.

Switch

```
switch(variável){
    case valor1:
        //código
        break;
    case valor2:
        //código
        break;
    default:
        //código
}
```

INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20
Retorno: 20:40

Fixação

Vamos fazer alguns exercícios e exemplos para fixarmos os conceitos que acabamos de ver:

1. O número primo é aquele que só é divisível por ele mesmo e pela unidade. Faça uma algoritmo que determina se o número de uma variável é primo ou não

2. Faça um algoritmo para ler um número que é um código de usuário. Caso este código seja diferente de um código armazenado internamente no algoritmo (igual a 1234) deve ser apresentada a mensagem ‘Usuário inválido!’, caso seja o mesmo código deve ser apresentado a mensagem “Usuário válido”.

Fixação

3. Crie um programa que leia um valor numérico, o valor de um carro, e se o carro for mais caro do que 100000 deve ser aplicado um imposto de 30%, se o carro for mais barato deve ser aplicado um imposto de 20%.

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!

SENAI

<LAB365>