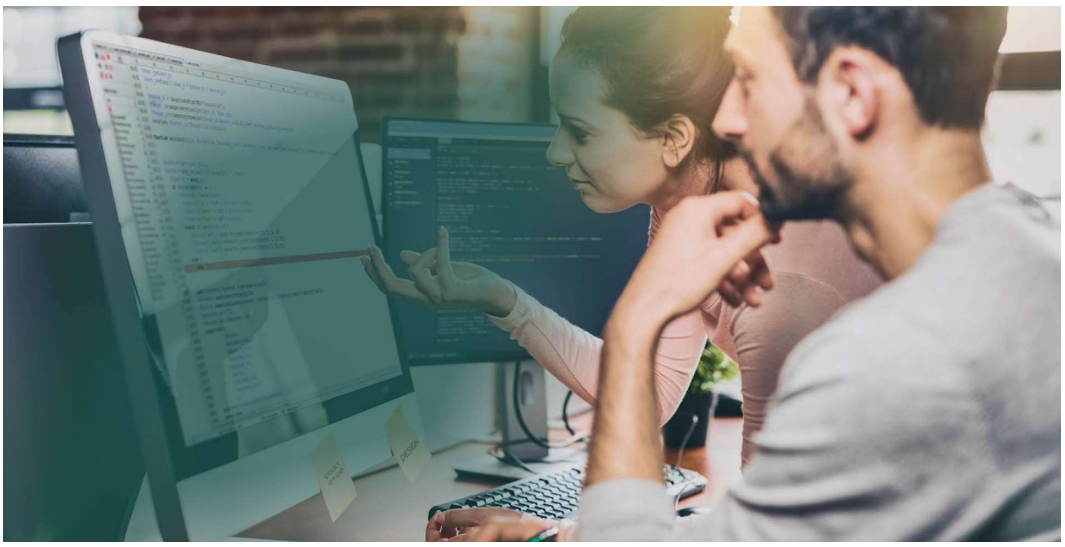


Estruturas de decisão

Prof. Anderson Fernandes Pereira dos Santos

Descrição

O emprego de estruturas de decisão na Linguagem C.

Propósito

Compreender os conceitos de estrutura de decisão suportados pela Linguagem C, de forma que as aplicações desenvolvidas sejam robustas e eficientes.

Objetivos

Módulo 1

Comandos condicionais simples

Aplicar os conceitos de estruturas de decisão simples e composta.

Módulo 2

Comandos condicionais aninhados

Aplicar os conceitos de estruturas de decisão encadeada e aninhada e de múltiplas alternativas.

Introdução

Em meados da década de 1940, surgem os primeiros computadores digitais no mundo. Naquela época, os computadores eram programados manipulando diretamente o hardware. Em meados da década de 1950, surgem os primeiros computadores com programa armazenado em memória, entretanto, a programação era feita em linguagem de máquina e o programador precisava ter conhecimento profundo do hardware.

Com a evolução da tecnologia, surgem, na década de 1960, as primeiras linguagens de segunda geração. A principal característica destas linguagens é empregar uma estrutura sintática próxima à estrutura sintática da linguagem natural. Assim, programar ficou mais fácil e o programador pode abstrair o hardware do computador. Destacam-se o as linguagens Cobol e Fortran.

Dentre os diversos conceitos introduzidos com estas linguagens, as estruturas de decisão, comandos condicionais, por exemplo, facilitaram muito a tarefa de programar. De lá pra cá, pouca coisa mudou neste aspecto e empregar estruturas de decisão é fundamental para elaborar um programa de computador em qualquer linguagem.

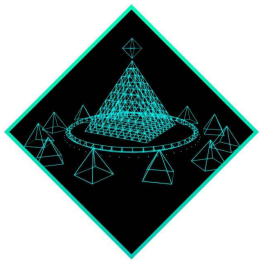
1 - Comandos condicionais simples

Ao final deste módulo, você será capaz de aplicar os conceitos de estruturas de decisão simples e composta.

Estrutura de decisão

Conceito

No desenvolvimento de aplicações, estruturamos nossos algoritmos para realizar determinadas atividades. Uma destas estruturas é chamada **estrutura de decisão**. Esta estrutura permite que a aplicação possa ter mais de uma sequência a ser seguida, que será decidida a partir da inferência de um determinado valor. Este valor, na Linguagem C, será do tipo booleano, portanto, os valores poderão ser falsos (quando forem zero ou *null*) ou verdadeiros (nos demais casos).



Esta estrutura é conhecida como **SE-ENTÃO e SE-ENTÃO-SENÃO**. No primeiro caso, um determinado segmento de código será executado somente se a expressão lógica, que será inferida, for verdadeira, enquanto no segundo caso a inferência da expressão lógica determinará qual segmento de código será executado.

Graficamente, esta estrutura é representada por um losango, em que cada quina representa:

A decisão corresponde a uma expressão ou variável cujo valor será analisado, conforme já citado.

Exemplo

Expressões que são verdadeiras. Considere em todos os exemplos que a variável **a** é inteira e tem valor 1:

a. SE (a)

b. SE (a ==1)

c. SE (a > 0)

Na Linguagem C, o SE é representado pela palavra reservada *if*, assim os itens acima seriam representados nesta linguagem como:

a. if(a)

b. if(a==1)

c. if(a > 0)

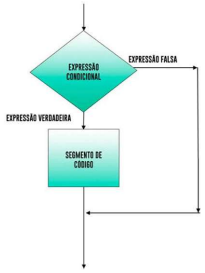
É importante relembrar que **a Linguagem C é sensível ao contexto**, assim deve-se usar *if* com todos os caracteres minúsculos. Comandos como *If*, *iF* e *IF* não são válidos. Além disso, os parênteses utilizados acima são obrigatórios.

Mais adiante, falaremos sobre quando tivermos uma estrutura destas dentro de outra estrutura, assim chamada de aninhada, e quando duas estruturas estão posicionadas de forma sequencial, denominada de encadeada.

Estrutura de decisão simples

Estrutura de decisão simples é caracterizada pela verificação de uma expressão lógica e, caso ela atenda aos requisitos estipulados, que neste caso será a expressão ser verdadeira, um determinado segmento de código é executado. Caso seja falso, ou seja, valor igual a zero, nulo ou vazio, nenhum segmento de código é executado e o programa continua no comando que sucede ao *if*.

Graficamente, esta estrutura pode ser apresentada através da figura:



No losango, que representa a estrutura de decisão, temos a **expressão lógica**. Esta expressão será avaliada. Caso o seu valor seja verdadeiro, um bloco de comandos (segmento de código) é executado logo após a expressão ter sido calculada. Caso contrário, será dada continuidade à execução da aplicação.

Na tabela a seguir, do lado esquerdo é exibido este código representado em PORTUGOL, e do lado direito o mesmo código, usando a Linguagem C.

PORTUGOL	LINGUAGEM C
SE EXPRESSÃO_CONDICIONAL	if(expressao_condicional)
ENTÃO	{
BLOCO DE COMANDOS	bloco_de_comandos;
FIM_ENTÃO	}
FIM_SE	

1

Na primeira linha é apresentado o início do comando. No Portugol, a expressão SE representa o comando *if*. A expressão condicional é apresentada obrigatoriamente entre parênteses. É recomendável que não haja espaço entre a palavra reservada *if* e o parêntese. Todavia é permitido, e até aconselhável, que haja espaço entre a expressão condicional e os parênteses.

PORTUGOL	LINGUAGEM C
SE EXPRESSÃO_CONDICIONAL	if(expressao_condicional)
ENTÃO	{
BLOCO DE COMANDOS	bloco_de_comandos;
FIM_ENTÃO	}
FIM_SE	

Na linha seguinte é apresentada a palavra ENTÃO, que interpretamos como o início do bloco de comandos, que na Linguagem C é representado pelo sinal de abre chaves {.

PORTUGOL	LINGUAGEM C
SE EXPRESSÃO_CONDICIONAL	if(expressao_condicional)
ENTÃO	{
BLOCO DE COMANDOS	bloco_de_comandos;
FIM_ENTÃO	}
FIM_SE	

O bloco de comandos é apresentado na sequência, porém, na Linguagem C, caso haja apenas um comando neste bloco, o uso das chaves torna-se opcional. No caso de existirem duas ou mais instruções, estas chaves se tornam obrigatórias, uma vez que poderia haver confusão com o caso anterior.

Exemplo 1

Normalmente, ao preenchermos formulários na web, somos questionados se desejamos receber mais informação a respeito daquele assunto. Como isto poderia ser representado na parte da implementação do código?

c

```c

int Flag_Deseja_Receber_Mais_Informacoes;
if (Flag_Deseja_Receber_Mais_Informacoes){
Enviar_Mais_Informacoes();
}
```

Nossa expressão condicional é uma variável do tipo inteira: Flag_Deseja_Receber_Mais_Informacoes. Assim, ela recebe valores inteiros.

Já vimos que na Linguagem C é considerado falso se a variável possui valores 0, *null* ou vazio; e é considerado verdadeiro no caso contrário, ou seja, se é diferente dos valores 0, *null* e vazio, dado o tipo de dado.

Neste caso, a variável é inteira, então, caso o valor dela seja 0, será considerado falso, caso possua qualquer outro valor (positivo ou negativo), será considerado verdadeiro. Outra expressão condicional equivalente poderia ser colocada neste *if*, porém, o mais usual é como foi apresentado. Um exemplo equivalente seria Flag_Deseja_Receber_Mais_Informacoes!=0.

Exemplo 2

Veremos agora um emprego simples da estrutura de decisão condicional simples, para tal o programa em C abaixo pede a média de um aluno hipotético e decide se o aluno foi aprovado ou não, imprimindo uma mensagem na tela.

Exercício 1

TUTORIAL COPIAR

```
c
```

null

null

Observe que, ao digitar uma nota maior que 5, a estrutura condicional da linha 8 é verdadeira e a mensagem de aprovação é exibida, se for menor que 5, a da linha 10.

Assista ao vídeo abaixo e entenda mais sobre o emprego da estrutura de decisão:

Emprego de Estrutura de Decisão

Neste vídeo, iremos fazer uma breve contextualização do tema.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Teoria na prática

Estruturas de decisão correspondem a uma prática comum e estão presentes em praticamente todas as aplicações. Estas estruturas podem ser do tipo simples, composta, encadeada ou aninhada. Esta representação é tão comum, que até mesmo filmes que possuem algum teor computacional retratam esta situação.

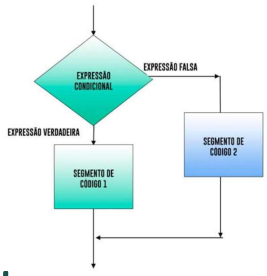
Uma das formas clássicas apresentadas está no filme Matrix, no qual o personagem Morpheus apresenta duas opções para o personagem Neo.

Mostrar solução

Estrutura de decisão composta

Como vimos anteriormente, a estrutura de decisão simples permite que um bloco seja executado caso uma expressão lógica seja verdadeira.

Agora acrescentaremos um novo bloco, que será executado no caso de a condição ser falsa. Assim, estamos nos referindo à estrutura **SE-ENTÃO-SENÃO**, conforme exibido na imagem a seguir.



> **A principal diferença deste tipo de estrutura condicional e a anterior reside no bloco marcado em azul na imagem anterior. Neste caso, quando a expressão condicional for avaliada como falsa, o bloco azul será executado.**

Dessa forma, quando uma expressão condicional é avaliada, caso o seu valor seja verdadeiro, um bloco de comandos 1 é executado, caso contrário, será executado o bloco de comandos 2.

Após a execução do bloco de comandos selecionado pela expressão, é dada continuidade à execução da aplicação.

Na tabela a seguir, do lado esquerdo é exibido este código representado em PORTUGOL, e do lado direito o mesmo código, usando a Linguagem C. A principal diferença perante o caso anterior está relacionada nas linhas marcadas em amarelo, que representam o quadrado azul do diagrama. Assim, caso a expressão condicional seja falsa, o código que será executado será o presente na estrutura *else*.

PORTUGOL	LINGUAGEM C
SE EXPRESSÃO_CONDICIONAL	if(expressao_condicional)
ENTÃO	{
BLOCO DE COMANDOS 1	bloco_de_comandos_1;
FIM_ENTÃO	}
SENÃO	else

PORTUGOL	LINGUAGEM C
	{
BLOCO DE COMANDOS 2	bloco_de_comandos_2;
FIM_SENÃO	}
FIM_SE	

Elaborada por Anderson Fernandes Pereira dos Santos.

Atenção!

A mesma regra com relação à quantidade de instruções e às chaves é utilizada neste caso. Portanto, se o bloco possuir apenas um comando, as chaves são opcionais, porém normalmente utilizadas. No caso de haver mais de uma instrução, estas chaves tornam-se obrigatórias.

Teoria na prática

No final da década de 1990, a Internet foi desenvolvida nos laboratórios do CERN por Tim Berners-Lee . Desde aquele momento em que passamos a estar globalmente conectados com todas as pessoas do mundo, o avanço da tecnologia caminha a passos largos. Nos últimos anos, a Inteligência Artificial passou a fazer parte do nosso dia a dia. Quer seja em nosso aplicativo de streaming de música ou vídeo, quer seja no percurso que nos é sugerido para ir ao trabalho, nos tornamos usuários ferrenhos desta tecnologia. Mas o que, de fato, é Inteligência Artificial?

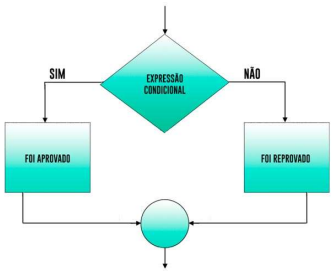
Mostrar solução

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

(Baseada em CESPE - 2017 - TRT - 7ª Região (CE) - Analista Judiciário - Tecnologia da Informação)



A estrutura lógica presente no diagrama apresentado é do tipo:

A SE ENTÃO

B CASO SELECIONE

C CASO REPITA

D SE ENTÃO SENÃO

E SE CASO CONTRÁRIO

Parabéns! A alternativa D está correta.

A estrutura apresentada é do tipo se então (indicado com o sim na figura) senão (indicado com o não da figura).

Questão 2

(Baseada em FCC - 2010 - SP - Agente de Defensoria - Analista de Sistemas) Marque a alternativa que apresenta o comando utilizado na estrutura de decisão composta, ou seja, que possua a opção a ser executada caso a condição seja verdadeira e caso a condição seja falsa.

A for(;;)

B if{}else{}

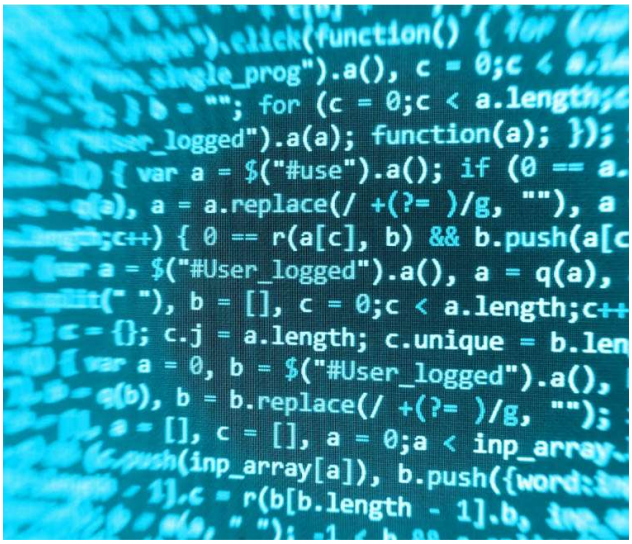
C while(){}

D do{}while()

E if {}

Parabéns! A alternativa B está correta.

Opção correta, letra B. A única opção que possui estrutura de condição é a alternativa b. As outras opções mostram estruturas de repetição que possuem caraterísticas e aplicações diferentes em programação.



2 - Comandos condicionais aninhados

Ao final deste módulo, você será capaz de aplicar os conceitos de estruturas de decisão encadeada e aninhada e de múltiplas alternativas.

Composição de estruturas de decisão

Compondo decisões

Cada bloco que compõe um *if*, seja simples ou composto, é formado por um ou mais comandos. Conforme já mencionado, caso seja apenas um comando, o uso de chaves é opcional. Caso seja utilizado mais de um comando, o uso de chaves torna-se obrigatório, para que seja determinado o bloco de comandos que ficará englobado pelo *if* ou *else*.

Mas, ainda, poderão ocorrer outros casos:

1

Quando um dos comandos que compõe o *if* (ou *else*) for outro comando *if* (simples ou composto), temos o que é chamado de *if* aninhado.

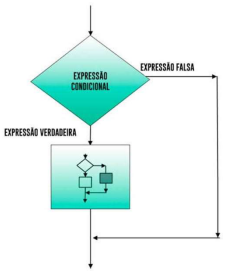
2

Quando dois comandos ifs forem sequenciais, teremos o caso de *if* encadeado.

Descreveremos estes dois tipos nas próximas seções.

Estruturas de decisão aninhadas

Este é o primeiro caso descrito acima, quando um dos comandos que compõem um *if* (ou *else*) é outro comando *if*, podendo estes comandos serem simples ou compostos. Ou seja, as estruturas são dispostas umas dentro das outras. A seguir, apresentamos uma **estrutura de decisão composta internamente de uma estrutura de decisão simples**.



Agora que já vimos a representação gráfica, partiremos para a representação na Linguagem C. Conforme já mencionado, temos um *if* composto internamente a um *if* simples, assim, esta figura representa o seguinte código em C:

c

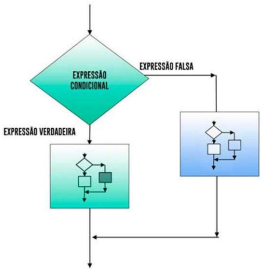
```c

if(EXPRESSAO_CONDICIONAL_1){
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_1;
    } else {
        BLOCO_INSTRUCAO_2;
    }
}
```

Se o *if* interno fosse simples, o código já seria expresso por:

c

```
if(EXPRESSAO_CONDICIONAL_1){
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_1;
    }
}
```

No caso de as duas estruturas de decisão serem compostas, estas são apresentadas dentro das instruções da estrutura de decisão. Convém salientar que **não há limitação com relação à quantidade de estruturas de decisão** que são dispostas internamente, independentemente do seu tipo, quer sejam simples ou compostas.



Um exemplo de estrutura de decisão aninhada, já na Linguagem C, semelhante ao diagrama anterior, é exibido a seguir.

c

```
if(EXPRESSAO_CONDICIONAL_1){
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_1;
    } else {
        BLOCO_INSTRUCAO_2;
    }
} else {
    EXPRESSAO_CONCIDIONAL_3
       BLOCO_INSTRUCAO_3;
} else {
    BLOCO_INSTRUCAO_4;
    }
}
```

De forma semelhante, no caso de um *if* simples ser um comando de uma estrutura *else* na Linguagem C, seria representado como:

c

```c

if(EXPRESSAO_CONDICIONAL_1){
    BLOCO_INSTRUCAO_1;
} else {
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_2;
    }
}
```

Ou ainda, no caso do *if* simples ser um comando do *if* na Linguagem C, seria representado como:

c

```c

if(EXPRESSAO_CONDICIONAL_1){
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_1;
    }
} else {
    BLOCO_INSTRUCAO_2;
}
```

Assista ao vídeo abaixo e entenda mais sobre o emprego da estrutura de decisão aninhada:

Emprego de Estrutura de Decisão Aninhada

Neste vídeo, iremos fazer uma breve contextualização do tema.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Exemplo

Numa escola, o aluno é aprovado se sua média é maior que 5,0. O código abaixo valida uma média lida e imprime a mensagem de aprovação. Teste a execução do código, digitando médias inválidas, isto é, menores que zero ou maiores que dez. Em seguida, teste com médias válidas de alunos aprovados e reprovados.

Exercício 1

TUTORIAL COPIAR

C

null

null

Estruturas de decisão encadeadas

Estruturas encadeadas são dispostas de forma sequencial. Independentemente de serem simples ou complexas, estas são dispostas sequencialmente, então, ao término de uma estrutura, é considerada a entrada da próxima estrutura, conforme o segmento de código a seguir.

C

```
1
2  if(EXPRESSAO_CONDICIONAL_1){
3      BLOCO_INSTRUCAO_1;
4  } else
5      if(EXPRESSAO_CONCIDIONAL_2){
6          BLOCO_INSTRUCAO_3;
7  } else {
8      BLOCO_INSTRUCAO_4;
9      }
```

Neste exemplo, uma estrutura *if* composta está disposta logo após o else da estrutura anterior. Se esta segunda estrutura fosse um *if* simples, na Linguagem C seria representado como:

C

```
if(EXPRESSAO_CONDICIONAL_1){
    BLOCO_INSTRUCAO_1;
} else
    if(EXPRESSAO_CONCIDIONAL_2){
        BLOCO_INSTRUCAO_3;
    }
```

Deve-se notar que em todos os exemplos apresentados até este ponto, BLOCO_INSTRUCAO pode ser representado por um ou mais comandos. Caso seja apenas um comando, o uso das chaves é opcional. Caso possua mais de um comando, o uso das chaves torna-se obrigatório.

Outro ponto que merece destaque é que um comando *if*, conforme o código abaixo, é dito ser apenas um comando, e os blocos de instrução que estão contidos neste código, mesmo que possuam mais de um comando, compõem apenas um, e assim não precisam de chaves antes e depois do *if*.

C

```
if(EXPRESSAO_CONDICIONAL_1){
    BLOCO_INSTRUCAO_1;
} else
    BLOCO_INSTRUCAO_2;
    }
```

Exemplo

Vamos repetir o exemplo da análise da aprovação dos alunos, agora com *if* encadeados.

Exercício 1

TUTORIAL COPIAR

c

null

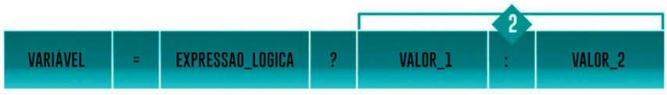
null

Operador ternário

A Linguagem C possui um operador que funciona de forma bem semelhante a um *if*. Este operador utiliza os símbolos de (?) e de (:) para representar a expressão lógica que é utilizada e os valores que comporão o campo do *if* e o campo do *else*, fazendo uma comparação com um *if* composto.

Este operador, conhecido como **operador ternário** por possuir três campos, é apresentado na imagem a seguir.

Esta expressão deverá retornar VERDADEIRO ou FALSO. No caso da Linguagem C, o valor falso deverá ser 0, *null* ou vazio. E verdadeiro será qualquer valor diferente deste.



Valores que serão retornados por este operador. O primeiro será retornado quando o operador for verdadeiro e o segundo no caso de falso.

Os símbolos (?) e (:) deverão ser utilizados nesta ordem, pois, caso contrário, serão identificados pelo compilador como um erro sintático.

VARIAVEL corresponde à variável que receberá os valores, VALOR_1 e VALOR_2, do operador ternário. Assim, caso EXPRESSAO_LOGICA seja verdadeira, VALOR_1 será atribuído à VARIAVEL, caso contrário, VALOR_2 é que será atribuído.

Por exemplo, considere o código abaixo:

c

```c

int a, b, c, d, e;

a=1;
b=2;
c=3;
d=4;

e=(a>b)?c:d;
```

Neste exemplo, temos que **a > b** é falso, pois **a = 1** e **b = 2**. Portanto, conforme já explicado anteriormente, a variável e irá receber o valor **d**.

Caso o comando fosse reescrito como e=(b>a)?c:d; teríamos o valor **c** sendo atribuído à variável **e**.

Atenção!

Operador ternário também permite que seja realizado alinhamento, porém não é uma boa prática de programação.

Estruturas de Múltiplas Alternativas

Esta estrutura, também conhecida como **switch-case**, permite que seja criada uma estrutura condicional que verificará o valor de uma variável de controle, no código abaixo identificado por VARIAVEL, e confrontará a determinados valores A, B e C para que sejam executados os blocos de

instrução BLOCO_INSTRUCAO_1, BLOCO_INSTRUCAO_2, BLOCO_INSTRUCAO_3 e BLOCO_INSTRUCAO_4.

c

```c

switch(VARIAVEL){
case A: BLOCO_INSTRUCAO_1;
    break;
case B: BLOCO_INSTRUCAO_2;
    break;
case c: BLOCO_INSTRUCAO_3;
    break;
default: BLOCO_INSTRUCAO_4;
}
```

O campo variável precisará, obrigatoriamente, ser do tipo **char**, **int** e **long**. O que significa que os valores A, B e C precisam também ser deste tipo.

Uma vez que VARIAVEL seja igual a A, por exemplo, a sequência de execução de tarefas continua até que o comando break seja encontrado. Assim, é executado BLOCO_INSTRUCAO_1. Caso a estrutura acima fosse conforme o segmento abaixo, no mesmo valor de VARIAVEL sendo igual a A, o resultado seria a execução de BLOCO_INSTRUCAO_1 e BLOCO_INSTRUCAO_2.

Isto porque o comando break não estava presente.

c

```c

switch(VARIAVEL){
    case A: BLOCO_INSTRUCAO_1;
    case B: BLOCO_INSTRUCAO_2;
        break;
    case c: BLOCO_INSTRUCAO_3;
        break;
    default: BLOCO_INSTRUCAO_4;
}
```

Outro ponto importante é que no caso de VARIAVEL não ser igual a nenhuma das opções, então o comando DEFAULT é executado. No entanto, deve-se salientar que este comando é opcional.

Exemplo

Imagine que você deseja implementar um menu para a escolha de uma operação em um sistema simples com interface textual. O menu terá as opções abaixo:

1. Inserir novo cliente
2. Consultar cliente por CPF
3. Consultar cliente por nome.
4. Remover cliente da base de clientes.

Qualquer outra tecla para sair do sistema.

Vejamos como podemos implementar isto com a estrutura de múltiplas alternativas

Exercício 1

TUTORIAL COPIAR

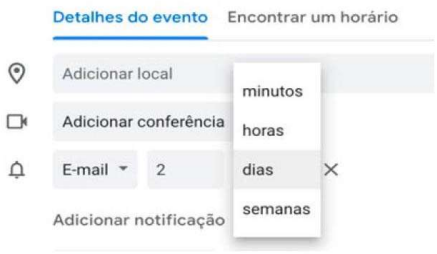
```
C
```

null

null

Teoria na prática

Normalmente, o comando switch é utilizado quando são ofertadas aos usuários diversas opções para que eles possam escolher, geralmente, no decorrer do preenchimento de um formulário, que hoje em dia está disponível na plataforma web ou na plataforma para dispositivos móveis. Estas opções podem ser apresentadas ao usuário em formato texto, assemelhando-se muito à estrutura da Linguagem C.



Mostrar solução

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

Retornando ao caso da média, temos agora outra situação. O aluno é considerado aprovado se a média das duas provas AV1 e AV2 é maior do que 7, porém, será considerado “em recuperação” no caso de a média estar entre 5 e 7, e estará reprovado se a média for abaixo de 5. A nota máxima de um aluno é 10. Marque a alternativa que apresenta os testes que deverão ser completados no código abaixo.

Javascript

A
EXPRESSAO1: media>=7
EXPRESSAO2: media<=10
EXPRESSAO3: media>=5
EXPRESSAO4: media>=0

B
EXPRESSAO1: media>=10
EXPRESSAO2: media<=7
EXPRESSAO3: media>=5
EXPRESSAO4: media>=0

C
EXPRESSAO1: media>=7
EXPRESSAO2: media<=10
EXPRESSAO3: media>=0
EXPRESSAO4: media>=5

D
EXPRESSAO1: media>=0
EXPRESSAO2: media<=5
EXPRESSAO3: media>=7
EXPRESSAO4: media>=10

E
EXPRESSÃO 1: MEDIA = 7
EXPRESSÃO 2: MEDIA < 10
EXPRESSÃO 3: MEDIA = 5
EXPRESSÃO 4: MEDIA > 0

Parabéns! A alternativa A está correta.

A resposta correta é a letra A. Para ser aprovado, media >=7, portanto, isso somente poderá ocorrer se a EXPRESSAO1 tiver este valor. A média de uma avaliação necessariamente precisa ser até 10. Valores de médias maiores não são usuais, pois a média não pode ser maior do que 10. Caso a média não seja >=7, então o aluno poderá estar de recuperação, >=5 ou reprovado no caso contrário, portanto >=0. Valores estranhos a esta situação são considerados incorretos.

Questão 2

Considere o if abaixo.

Javascript

Marque a alternativa que apresenta o operador ternário que substitui este if.

A c = (a>b)?3:4;

B c = (a>b):4?3;

C c = (a<=b)?3:4;

D c = (a<=b):4?3;

E (a>b)?3?4;

Parabéns! A alternativa A está correta.

No operador ternário, o símbolo (?) vem antes do símbolo (:), assim, as letras B e D estão incorretas. Neste caso, como a > b é falso, o valor de C deverá ser 4. No exemplo da letra D, se a = b, a variável C receberá o valor 4, o que não corresponde ao if apresentado.

Considerações finais

Neste estudo, apresentamos os conceitos de estrutura de decisão suportados pela Linguagem C, que permite que as aplicações desenvolvidas sejam eficientes.

Em um primeiro momento, foram exibidas as estruturas de decisão simples e compostas representadas pelo comando *if* e *if-else*, respectivamente. Posteriormente, vimos o comportamento destas estruturas quando são utilizadas em forma aninhada e encadeada. Adicionalmente, foi apresentado o operador ternário que pode representar um *if* em alguns casos. Para terminar, a estrutura switch-case, que é muito utilizada hoje em dia.

Podcast

Neste podcast, o especialista irá falar sobre estrutura de decisão.

Para ouvir o *áudio*, acesse a versão online deste conteúdo.

Explore +

Para saber mais sobre este tema, sugerimos que você pesquise a linguagem HTML.

As linguagens de programação representam as estruturas de controle que determinarão o comportamento de diversas aplicações. As aplicações que representam o algoritmo que foi implementado, atualmente, revelam o que se encontra no backend de uma aplicação. Normalmente, as opções que são apresentadas aos usuários utilizam outras linguagens, porém não de programação, e sim, visuais, como o HTML.

Referências

ADHIKARI, S. **Artificial Intelligence, Machine Learning and Deep Learning** – Are they same? In: Code Heroku. 25 mar. 2019.

DAMAS, L. **Linguagem C.** 10. ed. Rio de Janeiro: LTC, 2016

SCHILDT, H. **C Completo e Total**. 3. ed. São Paulo: Pearson Education do Brasil, 1996.

Material para download

Clique no botão abaixo para fazer o download do conteúdo completo em formato PDF.

Download material

O que você achou do conteúdo?

Relatar problema