

ESTRUTURA DE DECISÃO

Prof. Muriel Mazzetto
Algoritmos 1

Programação estruturada

2

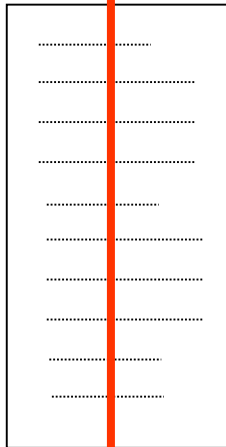
- Define que os programas podem ser resumidos em conjuntos de três estruturas diferentes:
 - ▣ Estrutura sequencial.
 - ▣ Estrutura de decisão.
 - ▣ Estrutura de repetição.

- Cada estrutura define a sequência e a quantidade de vezes que as instruções serão executadas.

Programação estruturada

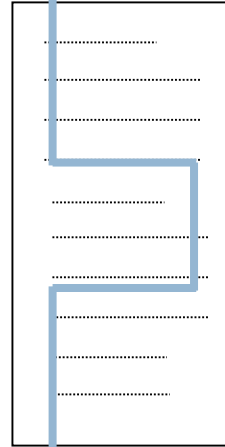
3

fluxo
de execução
sequencial



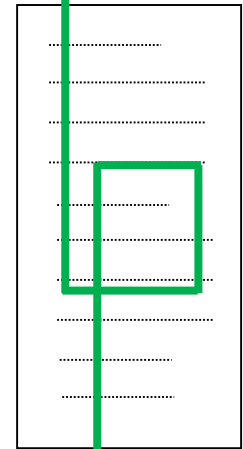
os comandos são executados um após o outro sequencialmente do início ({} até o final {})

fluxo
de execução
com decisão



os comandos são executados dependendo do valor de uma condição, ou expressão lógica:
if, if else e switch case

fluxo
de execução
com repetição



os comandos são executados, de forma repetida, um determinado número de vezes:
for, while e do while

Estrutura sequencial

4

- As instruções são executadas da primeira até a última, na ordem que foram descritas.
 - ▣ Cima para baixo, esquerda para direita, respeitando regra de precedência em operação matemática.
- **Todas as instruções do código serão executadas.**

Estrutura sequencial

5

```
#include <stdio.h>

int main(void)
{
    int a, b;
    int r1;
    float r4;

    printf("Informe o valor de a: ");
    scanf("%d", &a);
    printf("Informe o valor de b: ");
    scanf("%d", &b);

    r1 = a + b;
    printf("Soma de %d e %d = %d\n", a, b, r1);

    r4 = ((float) a) / ((float) b);
    printf("Divisao de %d e %d = %.2f\n", a, b, r4);

    return 0;
}
```

Estruturas de decisão

6

- Determina se um conjunto de instruções será ou não executado, dependendo de um teste lógico.
- Testes lógicos são realizados entre valores (variáveis e constantes) através de *operadores relacionais* e *operadores lógicos*.
 - ▣ O resultado de testes lógicos é representado por apenas dois valores:
 - Verdadeiro: 1;
 - Falso: 0;

Operadores relacionais

7

- Possibilitam **comparar** valores.
- As comparações são realizadas utilizando os seguintes símbolos:

Operação	Símbolo
Maior	>
Maior ou igual	>=
Menor	<
Menor ou igual	<=
Igual	==
Diferente	!=

Operadores relacionais

8

- O resultado de uma comparação é 0 (falso) ou 1 (verdadeiro):

$4 * 5 == 40 / 2$
 $20 == 20$
VERDADEIRO
1

$17 \% 2 > 5 \% 2$
 $1 > 1$
FALSO
0

$27 / 3 != 9$
 $9 != 9$
FALSO
0

$6 / 2 >= 9 / 3$
 $3 >= 3$
VERDADEIRO
1

Operadores lógicos

9

- Possibilitam formar **expressões** com comparações.
- Os símbolos da linguagem C para operações lógicas são:

Operação	Símbolo
Conjunção (E)	&&
Disjunção (OU)	
Negação	!

Operadores lógicos

10

- Possibilitam formar **expressões** com comparações.
- Os símbolos da linguagem C para operações lógicas são:

Operação	Símbolo	Significado
Conjunção (E)	&&	O resultado é verdadeiro apenas se <u>todas as comparações</u> forem verdadeiras

$(A == B) \ \&\& \ (C > D)$

(A igual a B) E (C maior que D)

Operadores lógicos

11

□ Exemplo de Conjunção (&&):

$(4 * 5 == 40 / 2) \&\& (3 / 2 > 2 / 2)$

$(20 == 20) \&\& (1.5 > 1.0)$

VERDADEIRO && VERDADEIRO

1 && 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

12

□ Exemplo de Conjunção (&&):

A = 7

B = 5

(A > B) && (A / 2 == 1)

(7 > 5) && (7 / 2 == 1)

VERDADEIRO && FALSO

1 && 0

0

PORTANTO É FALSO

Operadores lógicos

13

□ Exemplo de Conjunção (&&):

A = 7

B = 5

(A > B) && (A % 2 == 0)

(7 > 5) && (7 % 2 == 0)

VERDADEIRO && FALSO

1 && 0

0

PORTANTO É FALSO

Operadores lógicos

14

□ Exemplo de Conjunção (&&):

A = 10

B = 5

$(A > B) \ \&\& \ (B / 2 \leq A / 4) \ \&\& \ (A \geq 2)$

$(10 > 5) \ \&\& \ (5 / 2 \leq 10 / 4) \ \&\& \ (10 \geq 2)$

VERDADEIRO && VERDADEIRO && VERDADEIRO

1 && 1 && 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

15

□ Exemplo de Conjunção (&&):

A = 8

B = 5

$(A > B) \ \&\& \ (B / 2 \leq A / 4) \ \&\& \ (A \geq 2)$

$(8 > 5) \ \&\& \ (5 / 2 \leq 8 / 4) \ \&\& \ (8 \geq 2)$

VERDADEIRO && **FALSO** && VERDADEIRO

1 && 0 && 1

0

PORTANTO É FALSO

Operadores lógicos

16

- Possibilitam formar **expressões** com comparações.
- Os símbolos da linguagem C para operações lógicas são:

Operação	Símbolo	Significado
Disjunção (OU)		O resultado é verdadeiro <u>se</u> <u>ao menos uma das</u> <u>comparações</u> for verdadeira

$(A == B) \ || \ (C > D)$

(A igual a B) OU (C maior que D)

Operadores lógicos

17

□ Exemplo de Disjunção (||):

$(4 * 5 == 40 / 2) || (3 / 2 > 2 / 2)$

$(20 == 20) || (1.5 > 1.0)$

VERDADEIRO || VERDADEIRO

1 || 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

18

□ Exemplo de Disjunção (||):

$(4 * 5 == 40 / 2) || (3 / 3 > 2 / 2)$

$(20 == 20) || (1 > 1)$

VERDADEIRO || FALSO

1 || 0

1

PORTANTO É VERDADEIRO

Operadores lógicos

19

□ Exemplo de Disjunção (||):

$A = 7$

$B = 5$

$(A > B) \ || \ (A / 2 == 1)$

$(7 > 5) \ || \ (7 / 2 == 1)$

VERDADEIRO || FALSO

1 || 0

1

PORTANTO É VERDADEIRO

Operadores lógicos

20

□ Exemplo de Disjunção (||):

A = 3

B = 5

(A > B) || (A % 2 == 0)

(3 > 5) || (3 % 2 == 0)

FALSO || FALSO

0 || 0

0

PORTANTO É FALSO

Operadores lógicos

21

□ Exemplo de Disjunção (||):

A = 10

B = 5

$(A > B) \ || \ (B / 2 \leq A / 4) \ || \ (A \geq 2)$

$(10 > 5) \ || \ (5 / 2 \leq 10 / 4) \ || \ (10 \geq 2)$

VERDADEIRO || VERDADEIRO || VERDADEIRO

1 || 1 || 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

22

□ Exemplo de Disjunção (||):

A = 8

B = 5

$(A > B) \ || \ (B / 2 \leq A / 4) \ || \ (A \geq 10)$

$(8 > 5) \ || \ (5 / 2 \leq 8 / 4) \ || \ (8 \geq 10)$

VERDADEIRO || FALSO || FALSO

1 || 0 || 0

1

PORTANTO É VERDADEIRO

Operadores lógicos

23

- Possibilitam formar **expressões** com comparações.
- Os símbolos da linguagem C para operações lógicas são:

Operação	Símbolo	Significado
Negação (Não)	!	O resultado é a inversão do valor da expressão.

$!(A == B)$

NÃO(A igual a B)

Operadores lógicos

24

□ Exemplo de Negação (!):

$!(4 * 5 == 40 / 2)$

$!(20 == 20)$

$!(\text{VERDADEIRO})$

$\text{NÃO}(\text{VERDADEIRO})$

$\text{NÃO}(1)$

0

PORTANTO É FALSO

Operadores lógicos

25

□ Exemplo de Negação (!):

$A = 3$

$!(A \% 2 == 0)$

$!(3 \% 2 == 0)$

NÃO(FALSO)

1

PORTANTO É VERDADEIRO

Operadores lógicos

26

- Tabela verdade dos operadores.
- Exibe as possibilidades de resposta de acordo com os operadores e os valores envolvidos.

A	B	A && B	A B	!A
V	V	V	V	F
V	F	F	V	F
F	V	F	V	V
F	F	F	F	V

- V: verdadeiro. (1)
- F: falso. (0)

Operadores lógicos

27

- É possível criar expressões lógicas com os operadores relacionais e lógicos.
- Existe uma regra de precedência dos operadores.

PRECEDÊNCIA DE OPERADORES EM EXPRESSÕES LÓGICAS	
Mais alta	!
	> >= < <=
	== !=
	&&
Mais baixa	

- A prioridade pode ser alterada utilizando parênteses.

Operadores lógicos

28

□ Exemplo de expressões lógicas:

A = 10

B = 5

$(A < B) \ || \ (B / 2 \leq A / 4) \ \&\& \ (A \geq 10)$

$(8 < 5) \ || \ (5 / 2 \leq 10 / 4) \ \&\& \ (10 \geq 10)$

FALSO $\ || \$ VERDADEIRO $\ \&\& \$ VERDADEIRO

0 $\ || \$ 1 $\ \&\& \$ 1

0 $\ || \$ 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

29

□ Exemplo de expressões lógicas:

A = 10

B = 5

**RESOLVER
SEMPRE NA
ORDEM DE
PRIORIDADE**

$(A < B) \ || \ (B / 2 \leq A / 4) \ \&\& \ (A \geq 10)$

$(8 < 5) \ || \ (5 / 2 \leq 10 / 4) \ \&\& \ (10 \geq 10)$

FALSO $\ || \$ VERDADEIRO $\ \&\& \$ VERDADEIRO

0 $\ || \$ 1 $\ \&\& \$ 1

0 $\ || \$ 1

1

PORTANTO É VERDADEIRO

Operadores lógicos

30

□ Exemplo de expressões lógicas:

A = 10

B = 6

$((A < B) \ || \ (B / 2 \leq A / 4)) \ \&\& \ (A \geq 10)$

$((8 < 5) \ || \ (6 / 2 \leq 10 / 4)) \ \&\& \ (10 \geq 10)$

$(\text{FALSO} \ || \ \text{FALSO}) \ \&\& \ \text{VERDADEIRO}$

$(0 \ || \ 0) \ \&\& \ 1$

$0 \ \&\& \ 1$

0

PORTANTO É FALSO

Operadores lógicos

31

□ Exercício:

VARIÁVEIS			EXPRESSÕES	
V1	V2	V3	$V1 * V1 + 10 > V2$	$V1 != V2 \&\& V3 == 'A'$
6	2	A		
-2	40	α		

VARIÁVEIS			EXPRESSÕES
V1	V2	V3	$V3 == 'A' V3 == 'α' \&\& V2 == V1$
M	M	A	
m	M	α	

Operadores lógicos

32

□ Exercício:

VARIÁVEIS			EXPRESSÕES	
V1	V2	V3	$V1 * V1 + 10 > V2$	$V1 != V2 \&\& V3 == 'A'$
6	2	A	1	1
-2	40	a	0	0

VARIÁVEIS			EXPRESSÕES
V1	V2	V3	$V3 == 'A' V3 == 'a' \&\& V2 == V1$
M	M	A	1
m	M	a	0

Operadores aritméticos

33

□ Operadores aritméticos:

Operação	Símbolo
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Resto da divisão inteira	%
Inversão de sinal	-
Decremento	--
Incremento	++

Operadores aritméticos

34

- Operadores aritméticos compostos com atribuição:

Operação	Símbolo
Adição e Atribuição	$+=$
Subtração e Atribuição	$-=$
Multiplicação e Atribuição	$*=$
Divisão e Atribuição	$/=$

Funções matemáticas

35

- ❑ Funções mais utilizadas já implementadas na biblioteca `math.h`:

FUNÇÃO	EXEMPLO	SIGNIFICADO
<code>pow</code>	<code>var = pow(x,y);</code>	Calcular x^y
<code>sqrt</code>	<code>var = sqrt(x);</code>	Calcular \sqrt{x}
<code>ceil</code>	<code>var = ceil(x);</code>	Arredonda o valor de x para cima. Ex: 3.2 se torna 4
<code>floor</code>	<code>var = floor(x);</code>	Arredonda o valor de x para baixo. Ex: 5.8 se torna 5

Funções matemáticas

36

- ❑ Funções mais utilizadas já implementadas na biblioteca `math.h`:

FUNÇÃO	EXEMPLO	SIGNIFICADO
<code>pow</code>	<code>var = pow(x,y);</code>	Calcular x^y
<code>sqrt</code>	<code>var = sqrt(x);</code>	Calcular \sqrt{x}
<code>ceil</code>	<code>var = ceil(x);</code>	Arredonda o valor de x para cima. Ex: 3.2 se torna 4
<code>floor</code>	<code>var = floor(x);</code>	Arredonda o valor de x para baixo. Ex: 5.8 se torna 5

`int var = (int)(x + 0.5); //Arredondar para o inteiro mais próximo`

Operadores relacionais

37

□ Operadores relacionais:

Operação	Símbolo
Maior	>
Maior ou igual	>=
Menor	<
Menor ou igual	<=
Igual	==
Diferente	!=

Operadores lógicos

38

□ Operadores lógicos:

Operação	Símbolo
Conjunção (E)	&&
Disjunção (OU)	
Negação	!

Operadores

39

- Ordem de precedência dos operadores:

Operadores	Prioridade
Parênteses mais internos	1 ^a
Aritméticos	2 ^a
Relacionais	3 ^a
Lógicos	4 ^a

Estruturas de decisão

40

- A linguagem C avalia o resultado de expressões lógicas através das estruturas:
 - if
 - if, else if, else
 - switch, case

Estruturas de decisão

41

- A linguagem C avalia o resultado de expressões lógicas através das estruturas:
 - ▣ if
 - ▣ if, else if, else
 - ▣ switch, case
- Valores **diferentes de zero** são considerados **verdadeiros**.
- Valor **zero** é considerado **falso**.

Estruturas de decisão

42

- **if (se)** – Estrutura de decisão de uma condição.
- **Sintaxe:**

```
if ( CONDIÇÃO )  
{  
    INSTRUÇÕES;  
}
```

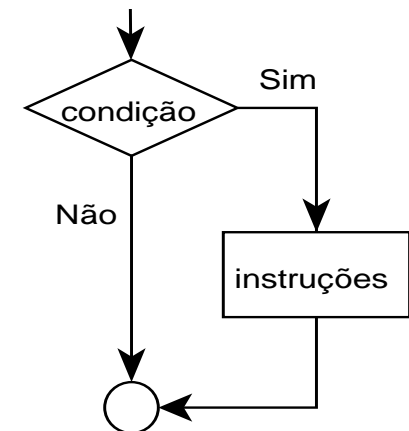
- **Condição:** expressão lógica.
- **Instruções:** conjunto de comandos que serão executados apenas se *Condição* for verdadeira.

Estruturas de decisão

43

- **if (se)** – Estrutura de decisão de uma condição.
- **Sintaxe:**

```
if ( CONDIÇÃO )  
{  
    INSTRUÇÕES;  
}
```



- ▣ **Condição:** expressão lógica.
- ▣ **Instruções:** conjunto de comandos que serão executados apenas se *Condição* for verdadeira.

Estruturas de decisão

44

□ **if** (se)

□ **Sintaxe**

□ **Código**

□ **Insira**

ap

```
int main(void)
```

```
{
```

```
    if( 1 )
```

```
    {
```

```
        printf("Entra aqui no 1.\n");
```

```
    }
```

```
    if( 18 )
```

```
    {
```

```
        printf("Entra aqui no 18.\n");
```

```
    }
```

```
    if( -53 )
```

```
    {
```

```
        printf("Entra aqui no -53.\n");
```

```
    }
```

```
    if( 0 )
```

```
    {
```

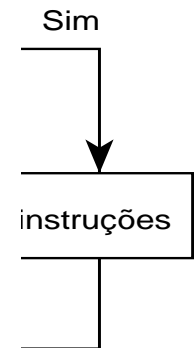
```
        printf("NÃO entra aqui no 0.\n");
```

```
    }
```

```
    return 0;
```

```
}
```

ão.



executados

Estruturas de decisão

45

- if (se)
- Sintax

□ Conc

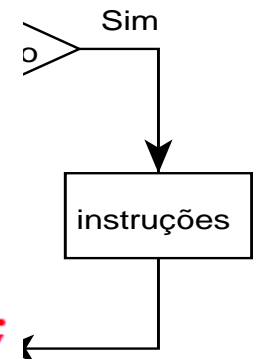
□ Instru

apei

```
int main(void)
{
    if( 1 && 15 )
    {
        printf("Entra aqui.\n");
    }
    if( 18 || 0 )
    {
        printf("Entra aqui.\n");
    }
    if( !-53 || !89 )
    {
        printf("Não entra aqui.\n");
    }
    if( !0 )
    {
        printf("Entra aqui.\n");
    }

    return 0;
}
```

ição.

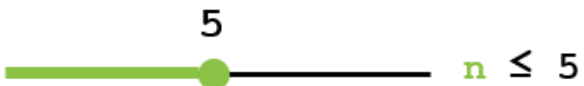


executados

Estruturas de decisão

46

Representação matemática:



Representação computacional:

```
if (n == 5)  
    printf("%d é igual a 5", n);
```

```
if (n < 5)  
    printf("%d é menor que 5", n);
```

```
if (n <= 5)  
    printf("%d é menor ou igual a 5", n);
```

```
if (n > 5)  
    printf("%d é maior que 5", n);
```

```
if (n >= 5)  
    printf("%d é maior ou igual a 5", n);
```


Estruturas de decisão


47


Representação matemática:

 $n \neq 5$

 $n \neq 5, \text{ e } n \neq 9$

 $n = 5, \text{ ou } n = 9$

 $n < 5, \text{ ou } n > 9$

 $n \geq 5, \text{ e } n \leq 9$

Representação computacional:

```
if (n != 5)
    printf("%d é diferente de 5", n);
```

```
if ((n != 5) && (n != 9))
    printf("%d é diferente de 5
           e é diferente de 9", n);
```

```
if ((n == 5) || (n == 9))
    printf("%d é igual a 5
           ou é igual a 9", n);
```

```
if ((n < 5) || (n > 9))
    printf("%d está fora do intervalo
           entre 5 e 9.", n);
```

```
if ((n >= 5) && (n <= 9))
    printf("%d pertence ao intervalo
           entre 5 e 9.", n);
```

Estruturas de decisão

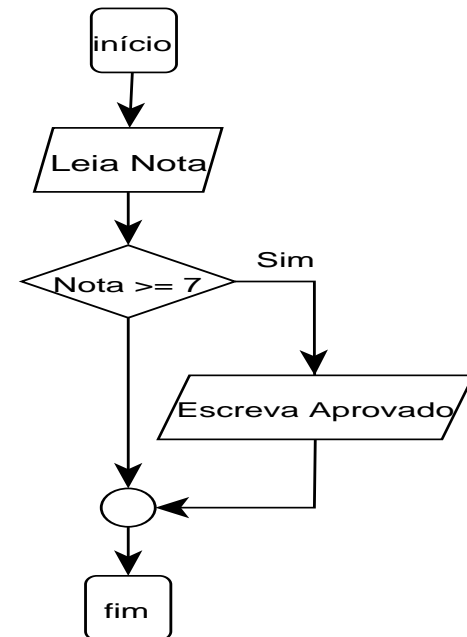
48

- Exemplo: Verificar se o valor da nota é maior ou igual a 7. Se sim, informar “Aprovado”.

```
int main(void)
{
    float Nota;

    printf("Informe a nota: ");
    scanf("%f", &Nota);

    if (Nota >= 7)
    {
        printf("Aprovado");
    }
}
```



Estruturas de decisão

49

- Exemplo: Leia 3 valores, verifique se o primeiro está dentro do intervalo definido pelos dois últimos.

Estruturas de decisão

50

- Exemplo: Leia 3 valores, verifique se o primeiro está dentro do intervalo definido pelos dois últimos.

```
int main(void)
{
    int n, inf, sup;

    printf("Informe o valor: ");
    scanf("%d", &n);
    printf("Informe o limite inferior: ");
    scanf("%d", &inf);
    printf("Informe o limite superior: ");
    scanf("%d", &sup);

    if (n >= inf && n <= sup)
    {
        printf("Esta dentro do intervalo.");
    }

    return 0;
}
```

Estruturas de decisão

51

- **if else** (se senão) – Estrutura de decisão com duas condições.
- **Sintaxe:**

```
if ( CONDIÇÃO )
{
    INSTRUÇÕES_1;
}
else
{
    INSTRUÇÕES_2;
}
```

- ▣ **Condição:** expressão lógica.
- ▣ **Instruções_1:** conjunto de comandos que serão executados apenas se *Condição* for verdadeira.
- ▣ **Instruções_2:** conjunto de comandos que serão executados apenas se *Condição* for falsa.

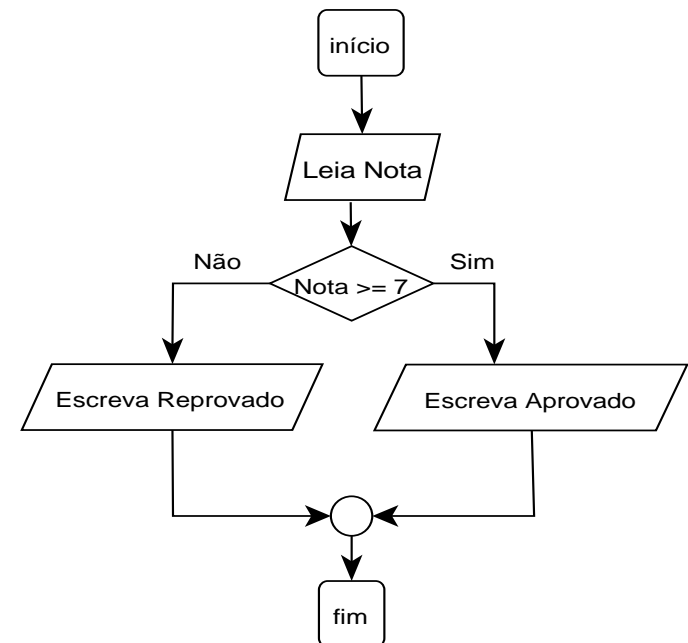
Estruturas de decisão

52

- Exemplo: Verificar se o valor da nota é maior ou igual a 7. Se sim, informar “Aprovado”, se não informar “Reprovado”.

```
int main(void)
{
    float Nota;
    printf("Informe a nota: ");
    scanf("%f", &Nota);

    if (Nota >= 7)
    {
        printf("Aprovado");
    }
    else
    {
        printf("Reprovado");
    }
}
```



Estruturas de decisão

53

- **if else if else** – Estrutura de decisão com mais de uma condição.

- **Sintaxe:**

```
if ( CONDIÇÃO_1 )  
{  
    INSTRUÇÕES_1;  
}  
else if ( CONDIÇÃO_2 )  
{  
    INSTRUÇÕES_2;  
}  
else  
{  
    INSTRUÇÕES_FINAIS;  
}
```

- ▣ **Condição_x**: expressão lógica.
- ▣ **Instruções_x**: conjunto de comandos que serão executados apenas se *Condição_x* for verdadeira E se condição anterior for falsa.
- ▣ **Instruções_Finais**: conjunto de comandos que serão executados apenas se todas as *Condição_x* forem falsas.

Estruturas de decisão

54

- **if else if else** – Estrutura de decisão com mais de uma condição.

- **Sintaxe:**

```
if ( CONDIÇÃO_1 )  
{  
    INSTRUÇÕES_1;  
}  
else if ( CONDIÇÃO_2 )  
{  
    INSTRUÇÕES_2;  
}  
else  
{  
    INSTRUÇÕES_FINAIS;  
}
```

- ▣ **Condição_x**: expressão lógica.
- ▣ **Instruções_x**: conjunto de comandos que serão executados apenas se Condição x for verdadeira E se condição anterior for falsa.
- ▣ **Instruções_Finais**: conjunto de comandos que serão executados apenas se todas as **Condição_x** forem falsas.

Estruturas de decisão

55

- **if else if else** – Estrutura de decisão com mais de uma condição.

- **Sintaxe:**

```
if ( CONDIÇÃO_1 )  
{  
    INSTRUÇÕES_1;  
}  
else if ( CONDIÇÃO_2 )  
{  
    INSTRUÇÕES_2;  
}  
else  
{  
    INSTRUÇÕES_FINAIS;  
}
```

- **Condição_x**: expressão lógica.
- **Instruções_x**: conjunto de comandos que serão executados apenas se *Condição_x* for verdadeira E se condição anterior for falsa.
- **Instruções_Finais**: conjunto de comandos que serão executados apenas se todas as Condição x forem falsas.

Estruturas de decisão

56

- Exemplo: Se a nota for maior ou igual a 7, informe “Aprovado”. Se a nota estiver entre 5 e 7, informe “Em Recuperação”. Se for menor que 5, informe “Reprovado”.

Estruturas de decisão

57

- Exemplo: Se a nota for maior ou igual a 7, informe “Aprovado”; Se a nota for maior que 5 e menor que 7, informe “Em Recuperação”; Se a nota for menor que 5, informe “Reprovado”.

```
int main (void)
{
    float Nota;
    printf("Informe um número: ");
    scanf("%f", &Nota);
    if (Nota >= 7)
    {
        printf("Aprovado");
    }
    else if (Nota > 5 && Nota < 7)
    {
        printf("em Recuperação");
    }
    else
    {
        printf("Reprovado");
    }
}
```

Exercício

58

- Qual o valor que X terá ao final da execução?

QUANDO FOR
APENAS UMA
INSTRUÇÃO,
NÃO PRECISA DE
CHAVES { }

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if (! (D>5))
        X = (A+B) *D;
    else
        X = (A - B) /C;
    printf("%f", X);
}
```

Exercício

59

- Qual o valor que X terá ao final da execução?

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if (! (D>5))
        X = (A+B) *D;
    else
        X = (A - B) /C;
    printf("%f", X);
}
```

Exercício

60

- Qual o valor que X terá ao final da execução?

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if( (A>2) && (B<7) )
        X = (A+2) * (B-2);
    else
        X = (A+B) / D * (C+D);
    printf("%f", X);
}
```

Exercício

61

- Qual o valor que X terá ao final da execução?

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if (!(A>2) || !(B<7))
        X = A + B;
    else
        X = A / B;
    printf("%f", X);
}
```

Exercício

62

- Qual o valor que X terá ao final da execução?

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if((A>2) || !(B<7))
        X = A + B - 2;
    else
        X = A - B;
    printf("%f", X);
}
```

Exercício

63

- Qual o valor que X terá ao final da execução?

```
#include <stdio.h>
void main() {
    int A=2, B=3, C=5, D=9;
    float X;

    if((C>=2) && (B<=7))
        X = (A + D)/2;
    else
        X = D * C;
    printf("%f", X);
}
```

Exercício

64

- Qual o valor que X terá ao final da execução?

```
int main(void)
{
    int A = 5, B = 10, C = 55, D = 5;
    float X = 0;

    if( A > 0 )
    {
        if( A < 10 )
            X = 10 / A;
    }
    else if( B >= 10 && C < 20)
        X = B + C / 5;
    else if( D != 5 )
        X = 4;
    else
        X = C + !D;

    printf("%f", X);

    return 0;
}
```


Exercício

65

- Qual o valor que X terá ao final da execução?

```
int main(void)
{
    int A = -5, B = 10, C = 55, D = 3;
    float X = 0;

    if( A > 0 )
    {
        if( A < 10 )
            X = 10 / A;
    }
    else if( B >= 10 && C < 20 )
        X = B + C / 5;
    else if( D != 5 )
        X = 4;
    else
        X = C + !D;

    printf("%f", X);

    return 0;
}
```

Exercício

66

- Qual o valor que X terá ao final da execução?

```
int main(void)
{
    int A = -35, B = 10, C = 55, D = 5;
    float X = 0;

    if( A > 0 )
    {
        if( A < 10 )
            X = 10 / A;
    }
    else if( B >= 10 && C < 20)
        X = B + C / 5;
    else if( D != 5 )
        X = 4;
    else
        X = C + !D;

    printf("%f", X);

    return 0;
}
```

Exercício

67

- Qual o valor que X terá ao final da execução?

```
int main(void)
{
    int A = 35, B = 10, C = 5, D = 3;
    float X = 0;

    if( A > 0 )
    {
        if( A < 10 )
            X = 10 / A;
    }
    else if( B >= 10 && C < 20)
        X = B + C / 5;
    else if( D != 5 )
        X = 4;
    else
        X = C + !D;

    printf("%f", X);

    return 0;
}
```

Exercício

68

- 1) Elaborar um algoritmo que lê dois valores, verifica se o primeiro é múltiplo do segundo e escreve a mensagem ‘São múltiplos’ ou ‘Não são múltiplos’ dependendo da condição.
- ▣ Verificar para que não seja realizada uma divisão por zero. Nesse caso, informar que não é possível realizar uma divisão por zero.

Exercício

69

□ 1) E

ver

esc

múl

□ V

Z

U

```
int main (void)
{
    int X, Y;

    printf("Informe o primeiro valor: ");
    scanf("%d", &X);
    printf("Informe o primeiro valor: ");
    scanf("%d", &Y);

    if(Y == 0)
    {
        printf("Nao eh possivel realizar divisao por zero!");
    }
    else if(X % Y == 0)
    {
        printf("O valor %d eh multiplo de %d.", X, Y);
    }
    else
    {
        printf("Nao sao multiplos.");
    }

    return 0;
}
```

io

o por
lizar

Estruturas de decisão

70

- **switch case** – O valor de uma variável é comparada com várias constantes, executando aquelas que forem satisfeitas.
- **Variavel:** uma variável numérica ou de caractere.

```
switch (VARIABEL)
{
    case CONSTANTE_1:
    {
        INSTRUÇÕES_1;
    }
    case CONSTANTE_2:
    {
        INSTRUÇÕES_2;
    }
    case CONSTANTE_n:
    {
        INSTRUÇÕES_n;
    }
    default:
    {
        INSTRUÇÕES_FINAIS;
    }
}
```

Estruturas de decisão

71

- **switch case** – O valor de uma variável é comparada com várias constantes, executando aquelas que forem satisfeitas.

- **Constante_x**: um valor numérico constante ou um caractere. Não pode ser alterado.

```
switch (VARIÁVEL)
{
    case CONSTANTE_1:
    {
        INSTRUÇÕES_1;
    }
    case CONSTANTE_2:
    {
        INSTRUÇÕES_2;
    }
    case CONSTANTE_n:
    {
        INSTRUÇÕES_n;
    }
    default:
    {
        INSTRUÇÕES_FINAIS;
    }
}
```

Estruturas de decisão

72

- **switch case** – O valor de uma variável é comparada com várias constantes, executando aquelas que forem satisfeitas.

- **Instruções_x**: conjunto de instruções que serão executados apenas se a variável for igual a constante x.

```
switch (VARIÁVEL)
{
    case CONSTANTE_1:
    {
        INSTRUÇÕES_1;
    }
    case CONSTANTE_2:
    {
        INSTRUÇÕES_2;
    }
    case CONSTANTE_n:
    {
        INSTRUÇÕES_n;
    }
    default:
    {
        INSTRUÇÕES_FINAIS;
    }
}
```


Estruturas de decisão

73

- **switch case** – O valor de uma variável é comparada com várias constantes, executando aquelas que forem satisfeitas.

- **Instruções_finais:** conjunto de instruções que serão executados apenas se a variável for diferente de todas as constantes.

```
switch (VARIÁVEL)
{
    case CONSTANTE_1:
    {
        INSTRUÇÕES_1;
    }
    case CONSTANTE_2:
    {
        INSTRUÇÕES_2;
    }
    case CONSTANTE_n:
    {
        INSTRUÇÕES_n;
    }
    default:
    {
        INSTRUÇÕES_FINAIS;
    }
}
```

Estruturas de decisão

74

□ Exemplo:

```
int main (void)
{
    int valor;

    printf ("Digite um valor de 1 a 7: ");
    scanf ("%d", &valor);

    switch ( valor )
    {
        case 1 :
            printf ("Domingo\n");
            break;

        case 2 :
            printf ("Segunda\n");
            break;

        // ...

        case 7 :
            printf ("Sabado\n");
            break;

        default :
            printf ("Valor invalido!\n");
    }

    return 0;
}
```

Estruturas de decisão

75

□ Exemplo:

A INSTRUÇÃO *break* SERVE PARA IMPEDIR QUE SEJAM REALIZADAS AS INSTRUÇÕES SEGUINTE.

```
int main (void)
{
    int valor;

    printf ("Digite um valor de 1 a 7: ");
    scanf ("%d", &valor);

    switch ( valor )
    {
        case 1 :
            printf ("Domingo\n");
            break;

        case 2 :
            printf ("Segunda\n");
            break;

        // ...

        case 7 :
            printf ("Sabado\n");
            break;

        default :
            printf ("Valor invalido!\n");
    }

    return 0;
}
```

Estruturas de decisão

76

- **switch case** é muito utilizado para implementar menus.

```
int main (void)
{
    char op;

    printf("A: Adicao;\n");
    printf("S: Subtracao;\n");
    printf("Informe a opcao: ");
    scanf("%c", &op);

    switch ( op )
    {
        case 'A' :
        {
            int x, y;
            printf("Informe o valor de x: ");
            scanf("%d", &x);
            printf("Informe o valor de y: ");
            scanf("%d", &y);
            printf("A soma eh %d. \n", x + y);
            break;
        }
    }
}
```

```
        case 'B' :
        {
            int x, y;
            printf("Informe o valor de x: ");
            scanf("%d", &x);
            printf("Informe o valor de y: ");
            scanf("%d", &y);
            printf("A subtração eh %d. \n", x - y);
            break;
        }
        default :
        {
            printf ("Opcao invalida!\n");
        }
    }

    return 0;
}
```