

CT62A

COMPUTAÇÃO 1

Aula 05 - Estruturas de Seleção

Profs. Rafael Mantovani e Adalberto Lazarini



Apucarana - PR, Brasil

Universidade Tecnológica Federal do Paraná (UTFPR)

Roteiro

- 1 Introdução**
- 2 Declarações IF, IF-ELSE**
- 3 Declaração SWITCH**
- 4 Referências**

Roteiro

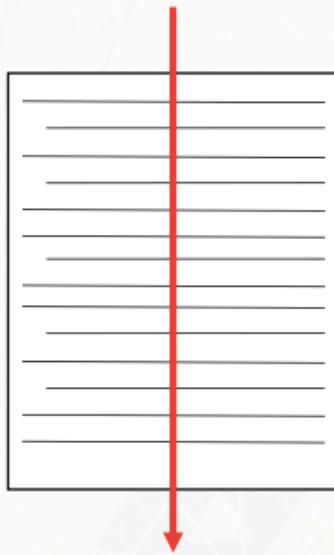
1 Introdução

2 Declarações IF, IF-ELSE

3 Declaração SWITCH

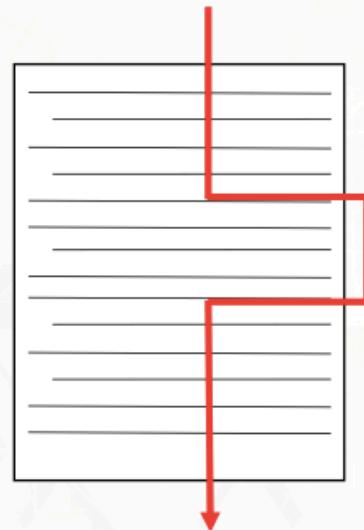
4 Referências

Introdução



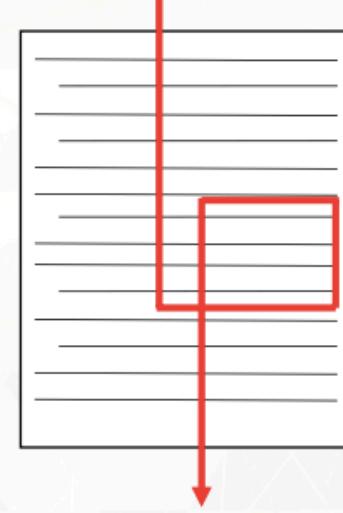
**Fluxo de execução
Sequencial**

Comandos são executados um após o outro



**Fluxo de execução
com desvio**

Comandos são executados dependendo do valor de uma condição



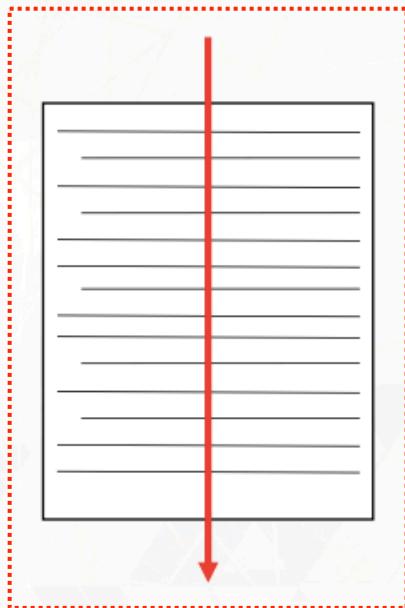
**Fluxo de execução
repetitivo**

Comandos são executados de forma repetida

Introdução

- Estruturas de controle dividem-se em:
 - Estruturas de decisão/seleção (condicionais)
 - Estruturas de repetição (loops de repetição)
- As estruturas de controle estão vinculadas às condições que determinam se instruções serão ou não executadas
- Uma condição de controle está relacionada aos operadores relacionais e lógicos

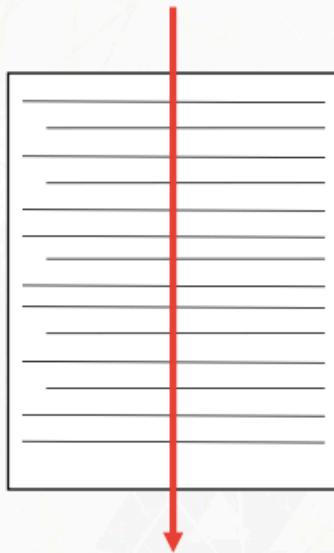
Introdução



Fluxo de execução
Sequencial

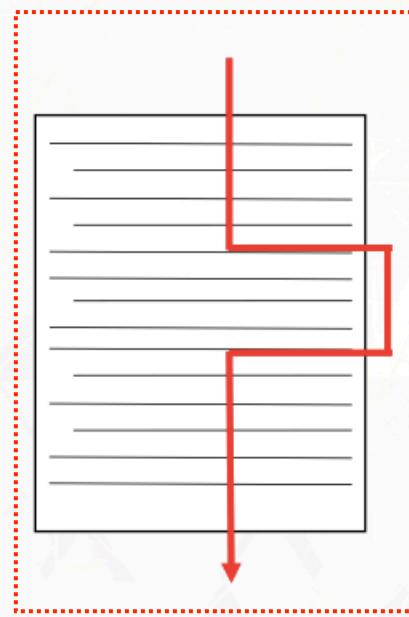
*Comandos são executados um após
o outro*

Introdução



**Fluxo de execução
Sequencial**

*Comandos são executados um após
o outro*



**Fluxo de execução
com desvio**

*Comandos são executados
dependendo do valor de uma
condição*

Exemplo

Pedra:



Papel:



Tesoura:



Exemplo

Pedra:



Papel:



Tesoura:



Vamos jogar? :)

Exemplo

- O que precisamos definir para ter um programa que jogue pedra-papel-tesoura?

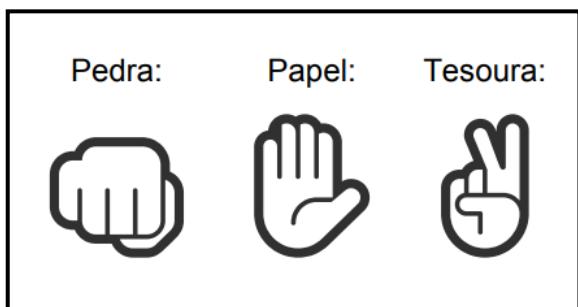
Pedra: Papel: Tesoura:



Exemplo

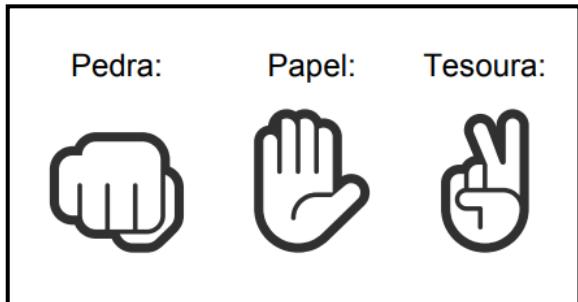
- O que precisamos definir para ter um programa que jogue pedra-papel-tesoura?

⌚ 2 jogadores (2 valores/dados)



Exemplo

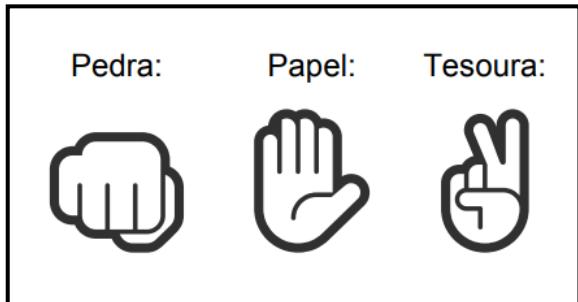
- O que precisamos definir para ter um programa que jogue pedra-papel-tesoura?



- ⌚ 2 jogadores (2 valores/dados)
- ⌚ regras (quem ganha?)
- ⌚

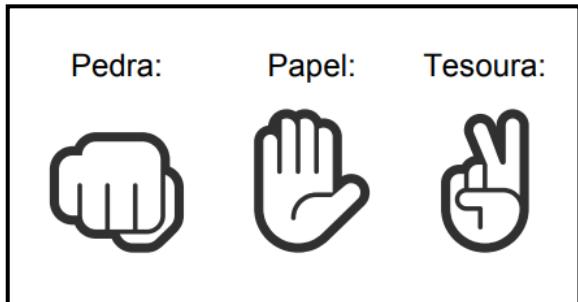
Exemplo

- O que precisamos definir para ter um programa que jogue pedra-papel-tesoura?
 - ⌚ 2 jogadores (2 valores/dados)
 - ⌚ regras (quem ganha?)
 - ⌚ encontrar um jeito do computador jogar (no hands!)



Exemplo

- O que precisamos definir para ter um programa que jogue pedra-papel-tesoura?
 - ⌚ 2 jogadores (2 valores/dados)
 - ⌚ regras (quem ganha?)
 - ⌚ encontrar um jeito do computador jogar (no hands!)

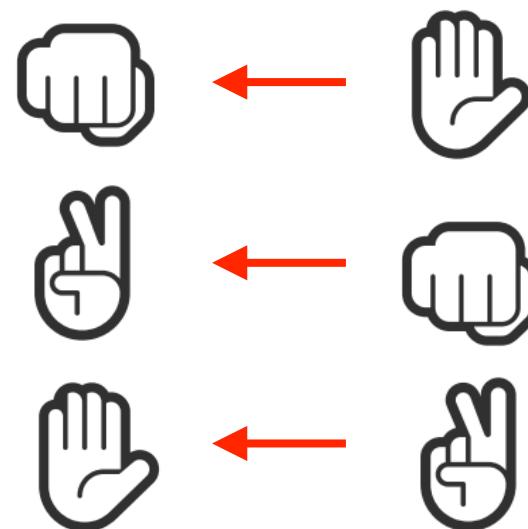


Exemplo

□ Regras

PERDE

GANHA

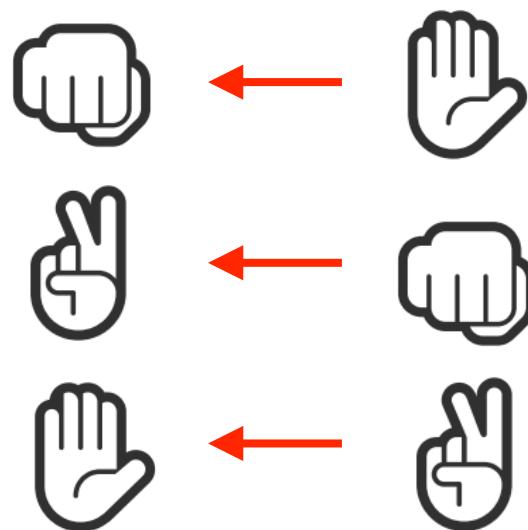


Exemplo

□ Regras

PERDE

GANHA



Se ambos os jogadores fazem
a mesma escolha, é um empate

Exemplo

- **Como jogar contra o computador?**

Exemplo

□ Como jogar contra o computador?

- Não tem mãos, então temos que mudar o funcionamento



Ideia: computador pré-seleciona sua escolha (pedra, papel e tesoura), e não nos diz nada.

Em sequência fazemos nossa escolha.

Por fim, o programa compara as jogadas e determina quem venceu.

Exemplo

□ Como jogar contra o computador?

- Não tem mãos, então temos que mudar o funcionamento



Ideia: computador pré-seleciona sua escolha (pedra, papel e tesoura), e não nos diz nada.

Em sequência fazemos nossa escolha.

Por fim, o programa compara as jogadas e determina quem venceu.

Exemplo

- **O jogo executando:**

```
> Pedra, papel ou tesoura? Pedra
```

Exemplo

- **O jogo executando:**

> Pedra, papel ou tesoura? Pedra



O computador pede sua escolha

Exemplo

- **O jogo executando:**

```
> Pedra, papel ou tesoura? Pedra  
> Você ganhou, eu escolhi tesoura.
```



○ computador determina o vencedor

Exemplo

□ O jogo executando:

```
> Pedra, papel ou tesoura? Pedra  
> Você ganhou, eu escolhi tesoura.
```

```
> Pedra, papel ou tesoura? Pedra  
> Ambos escolhemos pedra, um empate.
```

```
> Pedra, papel ou tesoura? Papel  
> Eu ganhei, eu escolhi tesoura.
```

← Execução gera um empate.

← Execução em que o computador vence.

Exemplo

- **Em descrição narrativa:**

Exemplo

□ Em descrição narrativa:

- 1 O usuário começa o jogo
- A O computador determina qual será sua escolha: Pedra, Papel ou Tesoura

Exemplo

□ Em descrição narrativa:

- 1 O usuário começa o jogo
 - A O computador determina qual será sua escolha: Pedra, Papel ou Tesoura
- 2 O jogo começa
 - A Obtém a escolha do usuário
 - B Examina a escolha do usuário. Se for inválida, mostrar uma mensagem e encerrar o jogo. Se for a mesma do computador, determina um empate e vai ao passo 3.
 - C Determina quem ganhou o jogo pelas regras

Exemplo

□ Em descrição narrativa:

- 1 O usuário começa o jogo
 - A O computador determina qual será sua escolha: Pedra, Papel ou Tesoura
- 2 O jogo começa
 - A Obtém a escolha do usuário
 - B Examina a escolha do usuário. Se for inválida, mostrar uma mensagem e encerrar o jogo. Se for a mesma do computador, determina um empate e vai ao passo 3.
 - C Determina quem ganhou o jogo pelas regras
- 3 O jogo termina. Diz ao usuário quem ganhou e qual foi a escolha do computador.

Exemplo

□ Em descrição narrativa:

1 O usuário começa o jogo

A O computador determina qual será sua escolha: Pedra, Papel ou Tesoura

2 O jogo começa

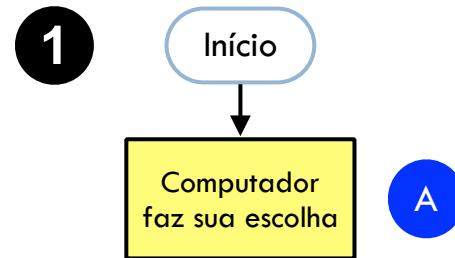
A Obtém a escolha do usuário

B Examina a escolha do usuário. Se for inválida, mostrar uma mensagem e encerrar o jogo. Se for a mesma do computador, determina um empate e vai ao passo 3.

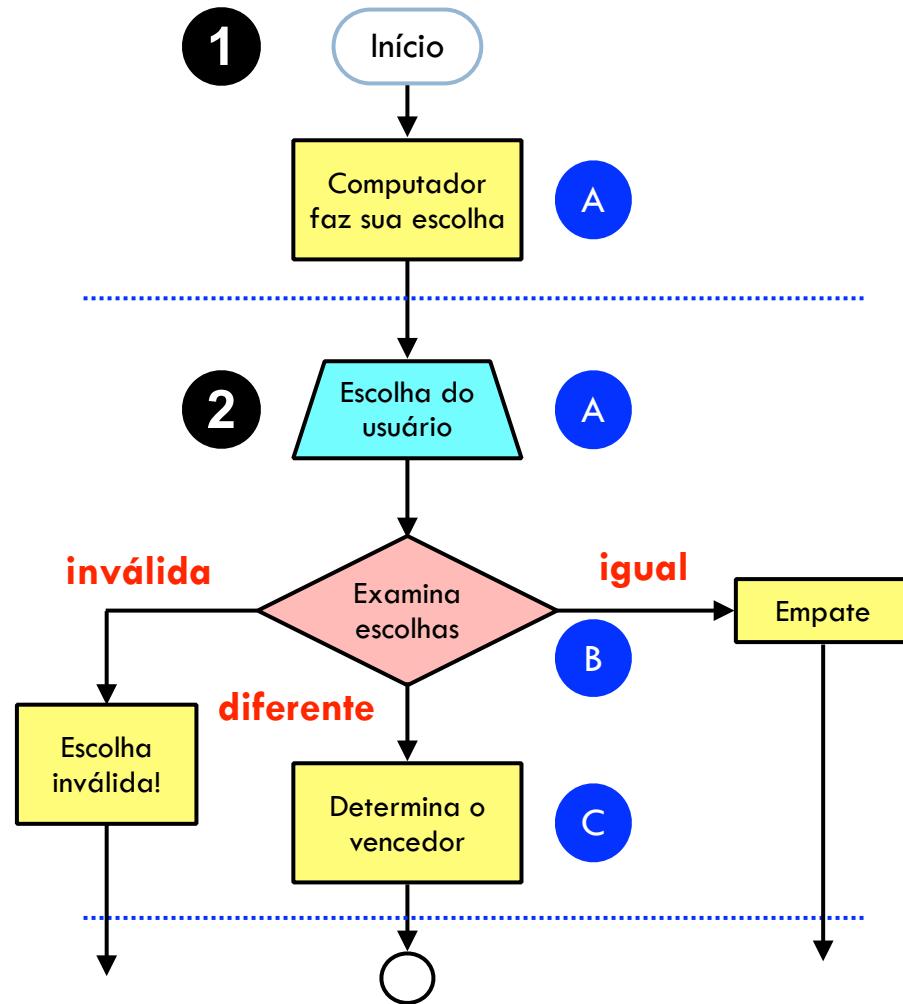
C Determina quem ganhou o jogo pelas regras

3 O jogo termina. Diz ao usuário quem ganhou e qual foi a escolha do computador.

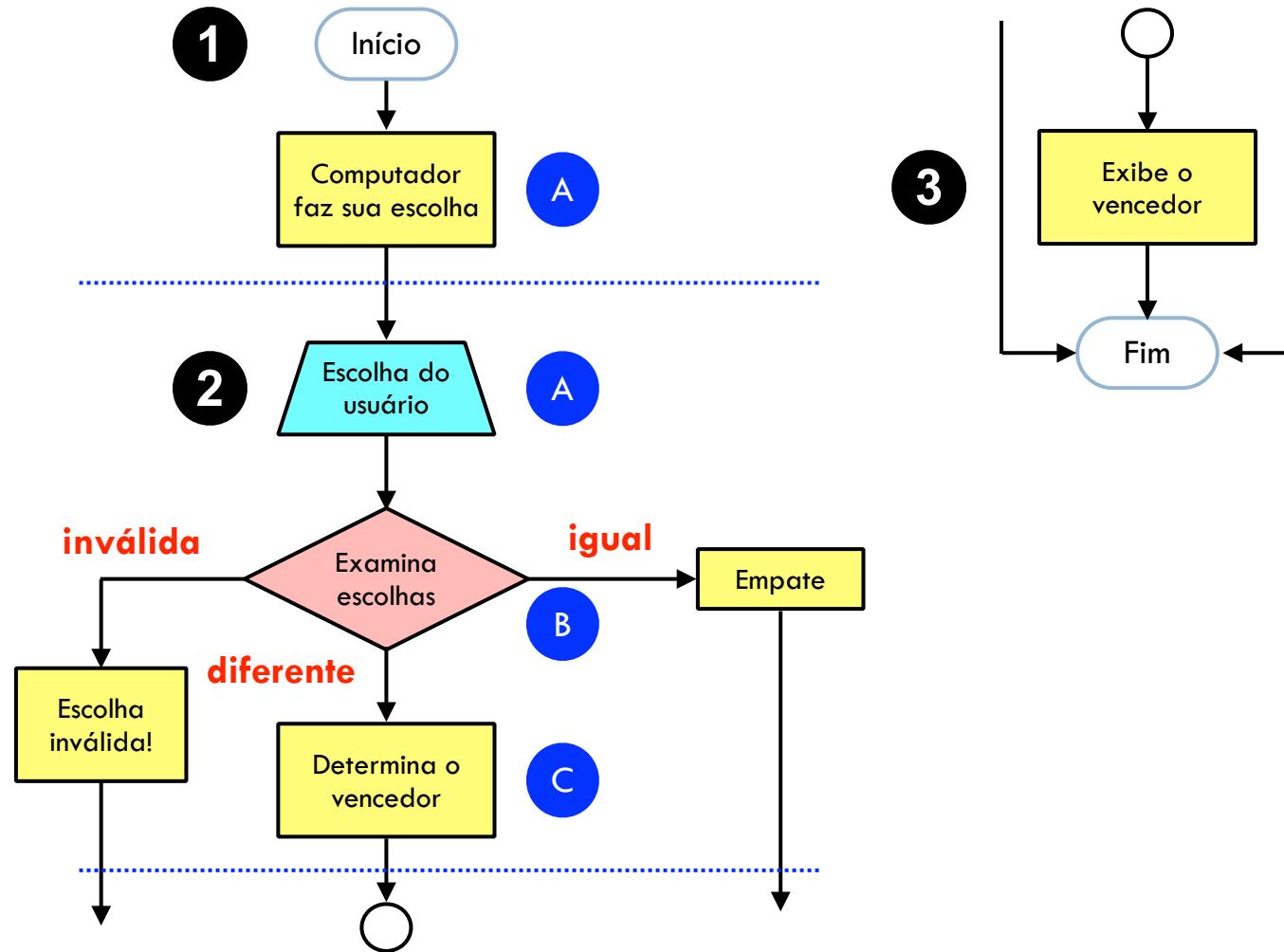
Fluxograma



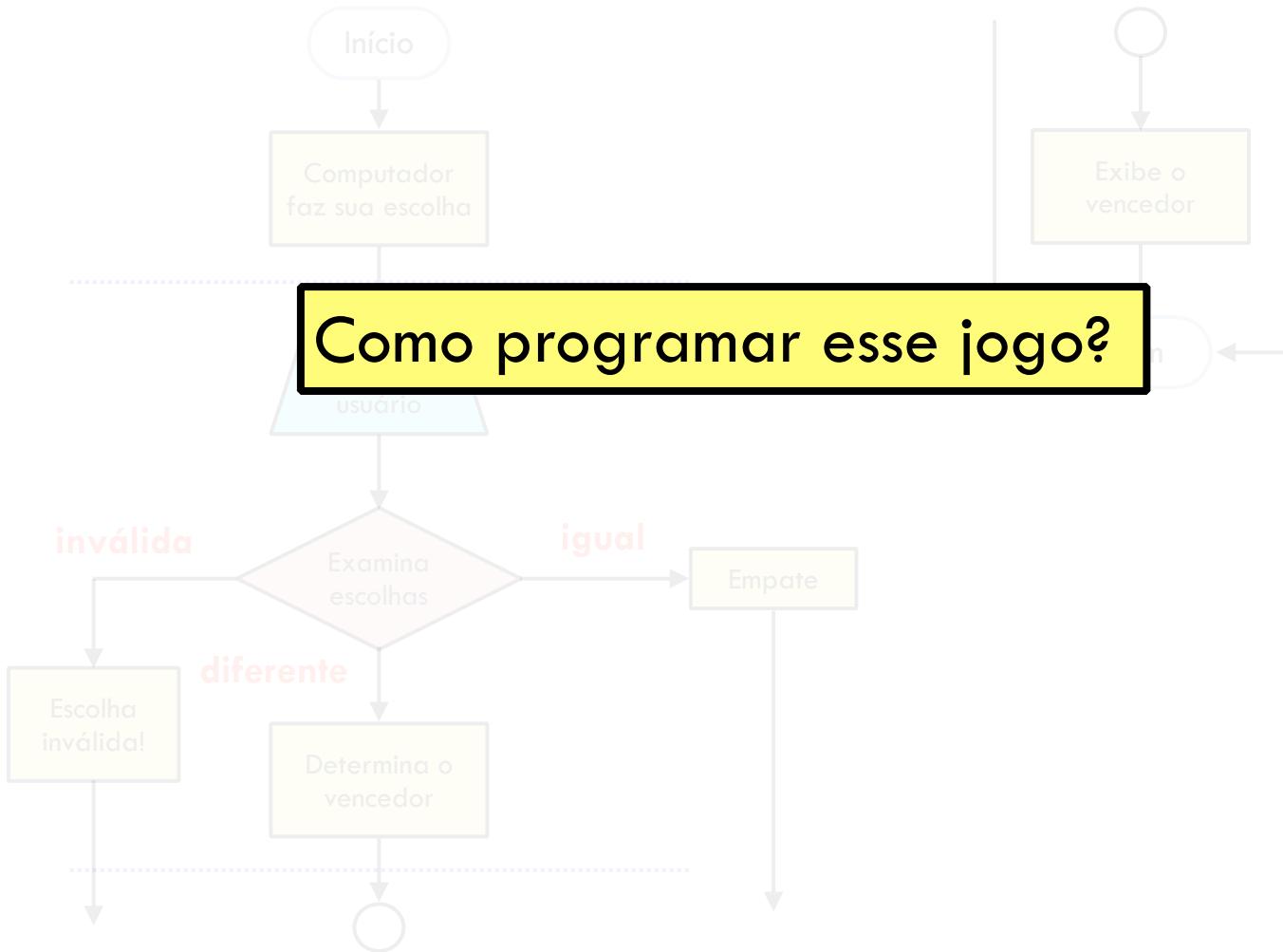
Fluxograma



Fluxograma



Fluxograma



Roteiro

1 Introdução

2 Declarações IF, IF-ELSE

3 Declaração SWITCH

4 Referências

Declarações Condicionais

- Uma estrutura de decisão permite decidir se um conjunto de instruções será ou não executado de acordo com determinadas condições
- A decisão é feita com base no resultado de um teste lógico que determina a condição
- Estruturas de decisão em C/C++:
 - `if`
 - `if-else`
 - `switch`

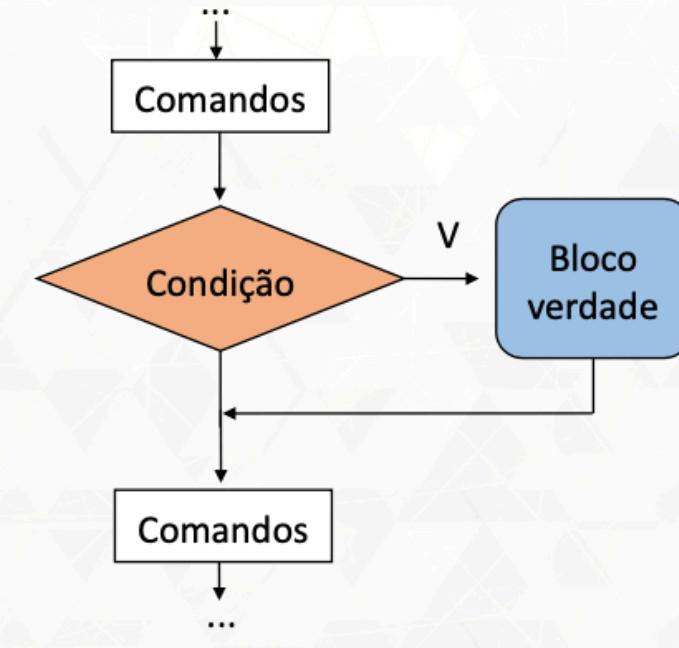
Declarações Condicionais

- Os operadores relacionais comparam dois valores e retornam um valor booleano
 - Verdadeiro (`true`)
 - Falso (`false`)

Operador	Descrição	X	Y	Lógico	Resultado
<code>==</code>	Igual a	2	3	<code>X == Y</code>	Falso
<code>!=</code>	Diferente de	2	3	<code>X != Y</code>	Verdadeiro
<code>></code>	Maior que	2	3	<code>X > Y</code>	Falso
<code>>=</code>	Maior ou Igual	2	3	<code>X >= Y</code>	Falso
<code><</code>	Menor que	2	3	<code>X < Y</code>	Verdadeiro
<code><=</code>	Menor ou igual	2	3	<code>X <= Y</code>	Verdadeiro

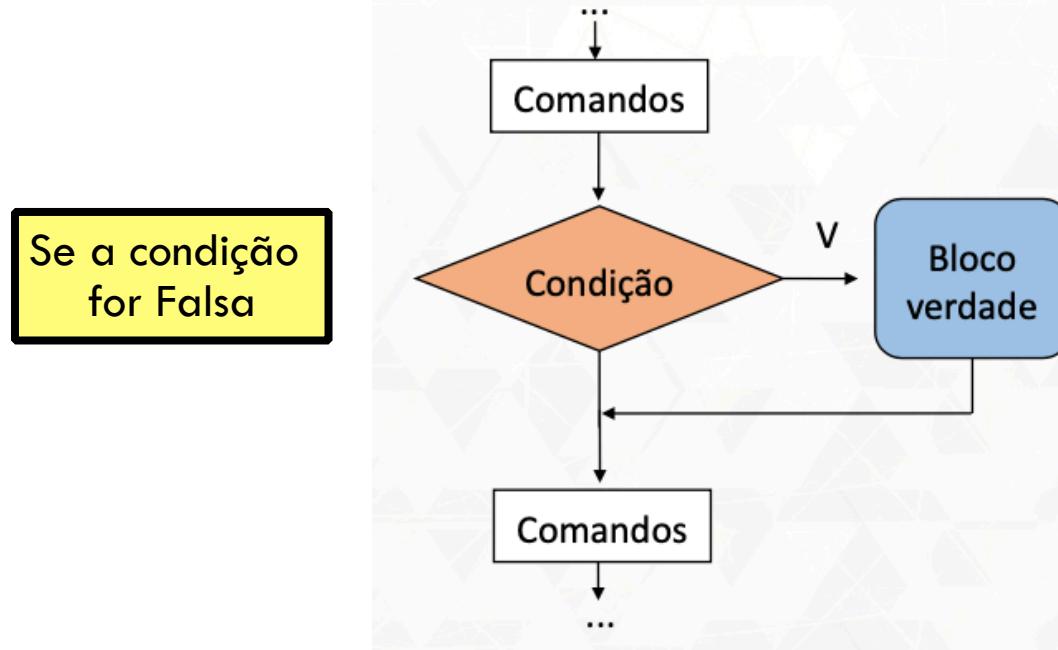
Declaração IF

Estrutura condicional simples



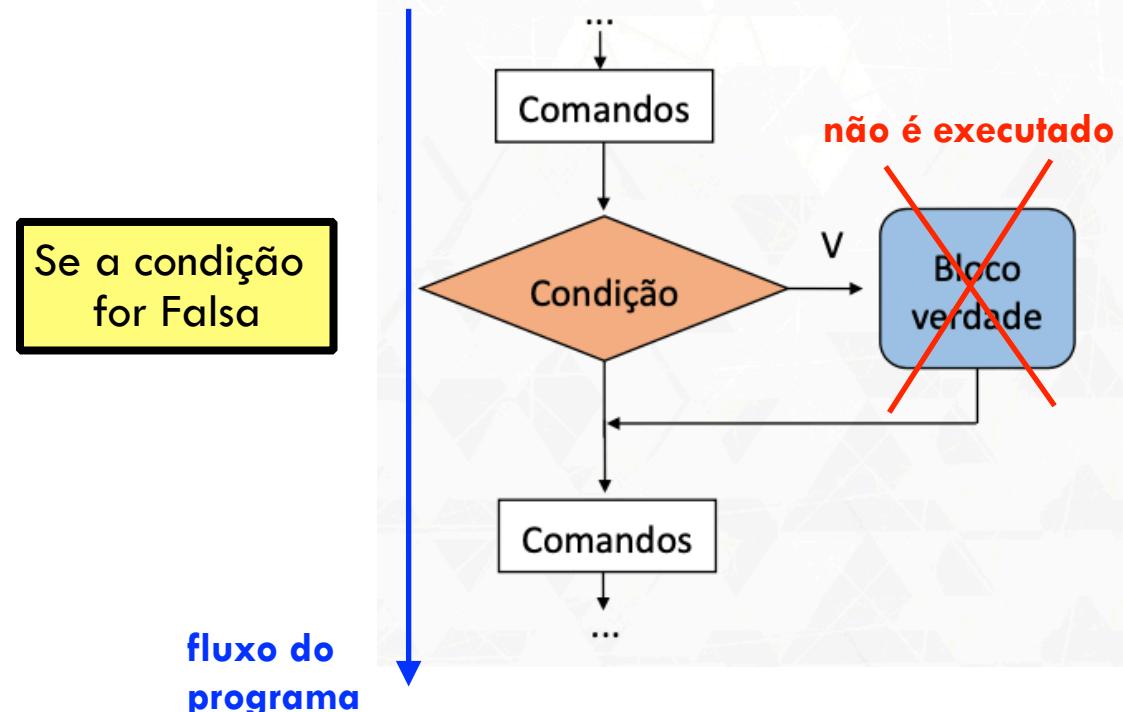
Declaração IF

Estrutura condicional simples



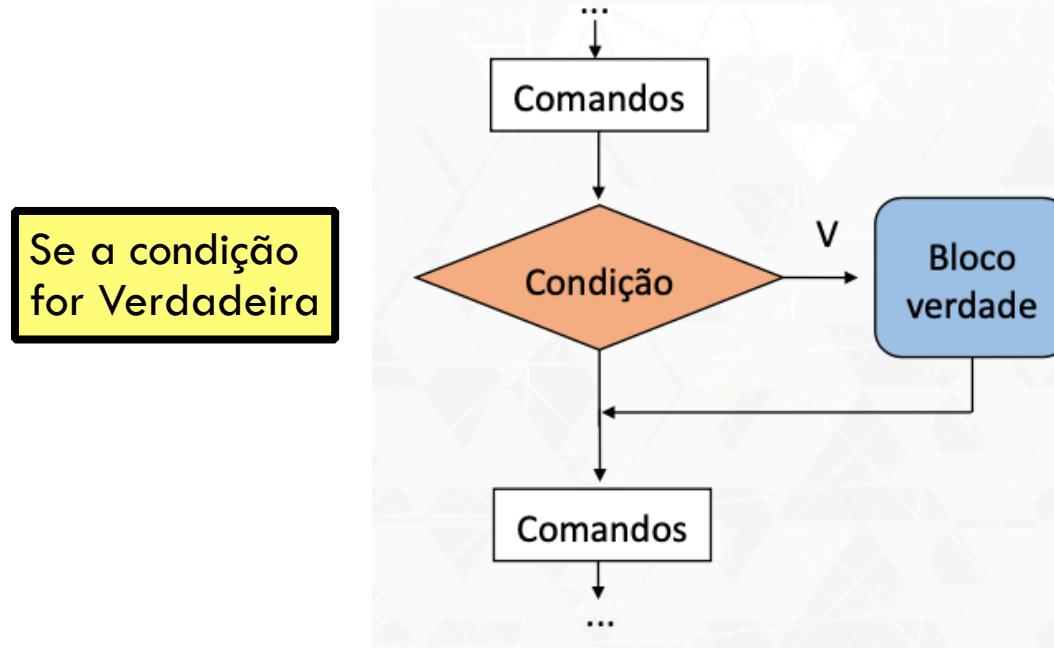
Declaração IF

Estrutura condicional simples



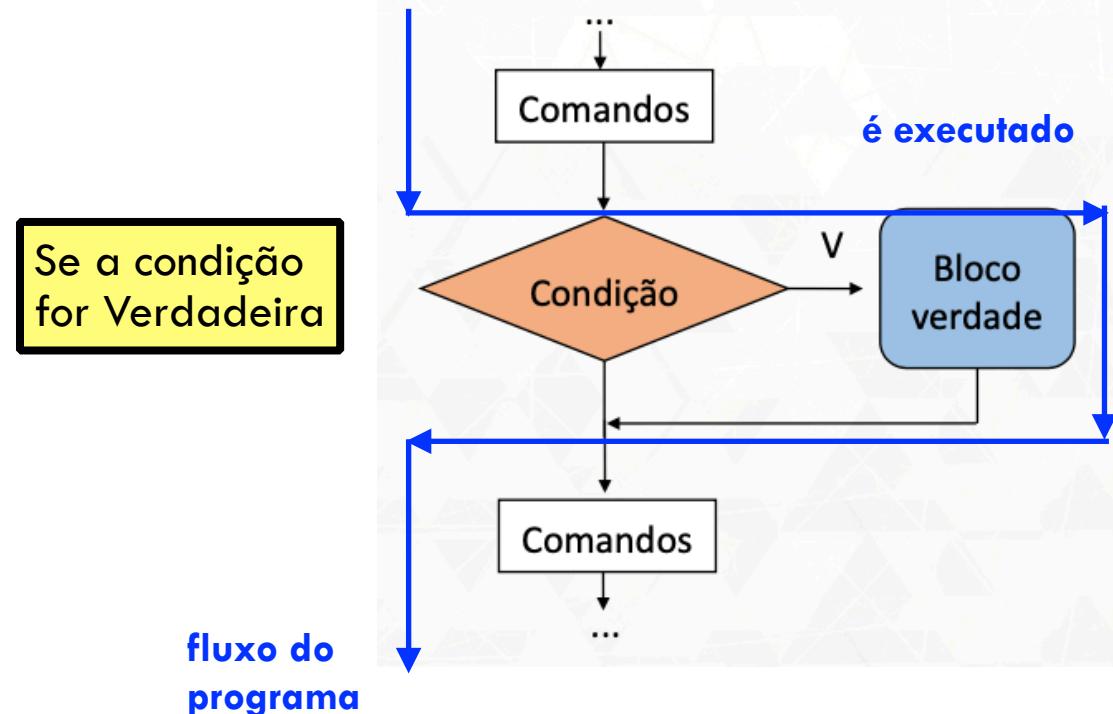
Declaração IF

Estrutura condicional simples



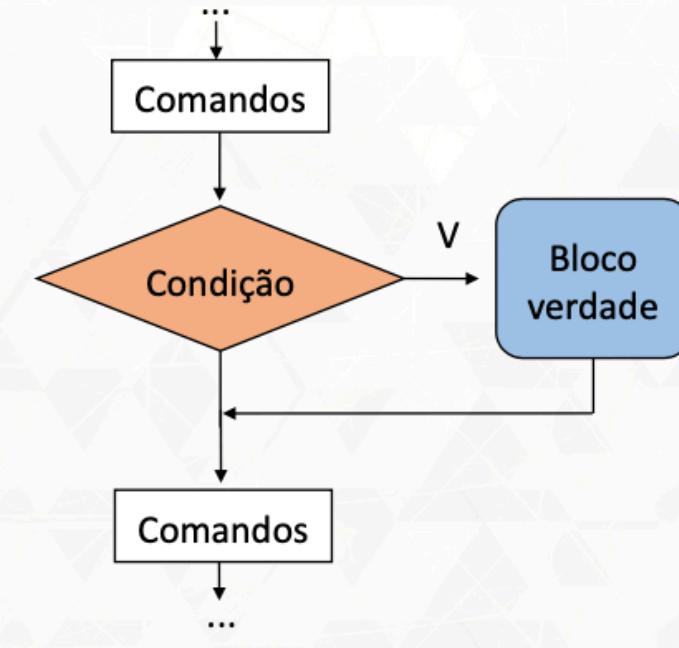
Declaração IF

Estrutura condicional simples



Declaração IF

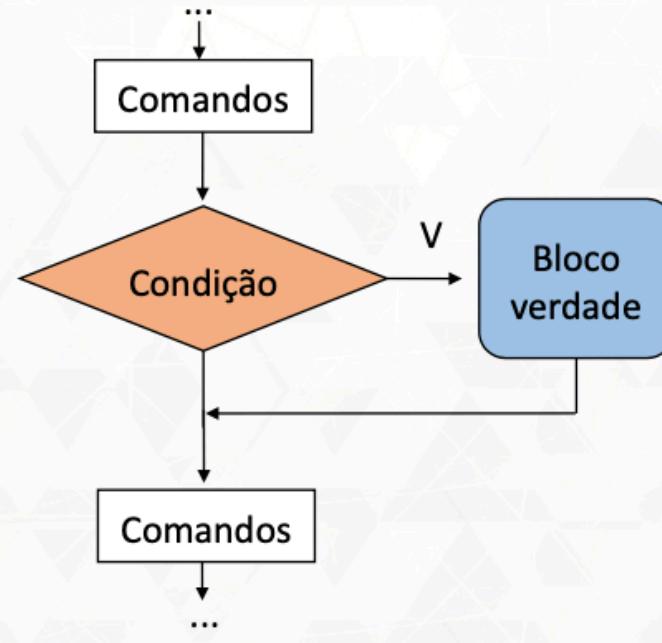
Estrutura condicional simples



Declaração IF

Estrutura condicional simples

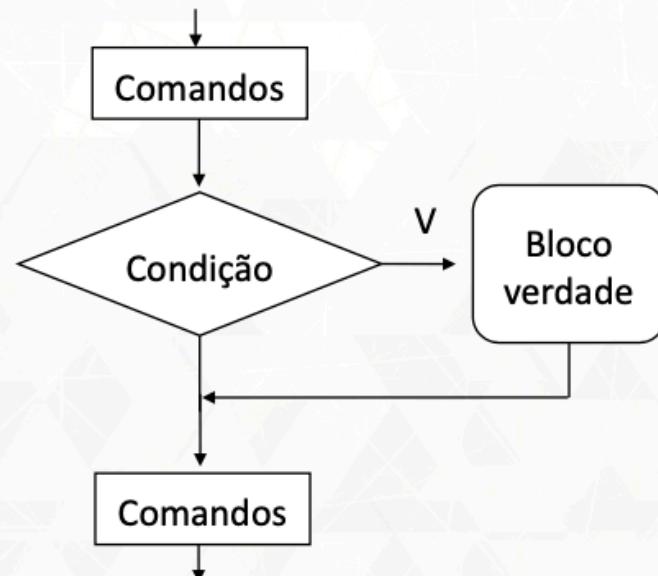
```
//comandos  
if(condição)  
{  
    //bloco verdade  
}  
//comandos
```



Declaração IF

Estrutura condicional Simples

```
int main(){  
  
    float nota = 7.6;  
  
    if(nota >= 7.0)  
        printf("APROVADO!");  
  
    ...  
    ...  
  
    return 0;  
}
```



Declaração IF

Estrutura condicional Simples

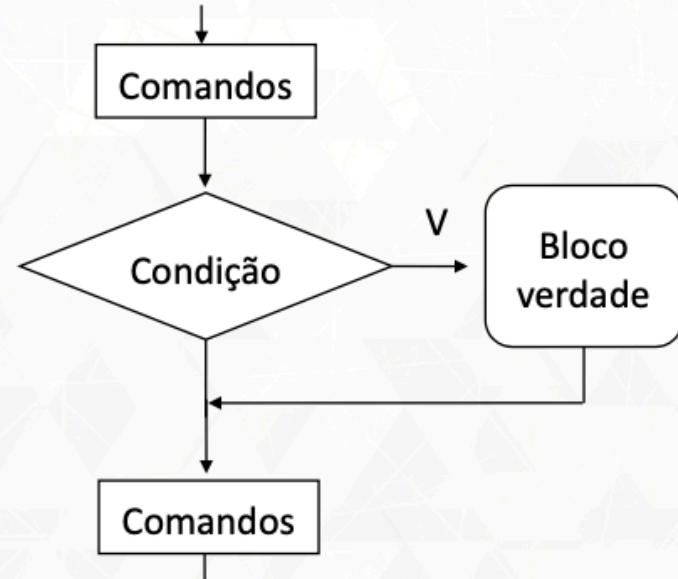
```
int main(){

    float nota1 = 7.6;
    float nota2 = 5.0;
    float media;

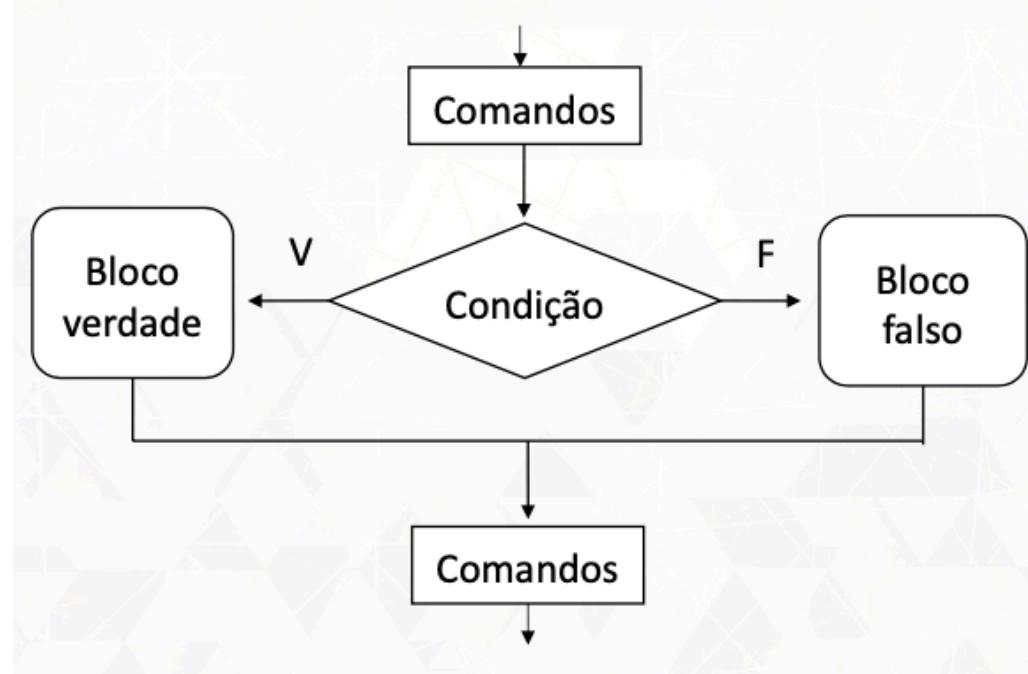
    if(nota1 == 10.0)
        printf("PARABÉNS!");

    media = (nota1 + nota2)/2;
    printf("Sua média é: %f", media);
    ...

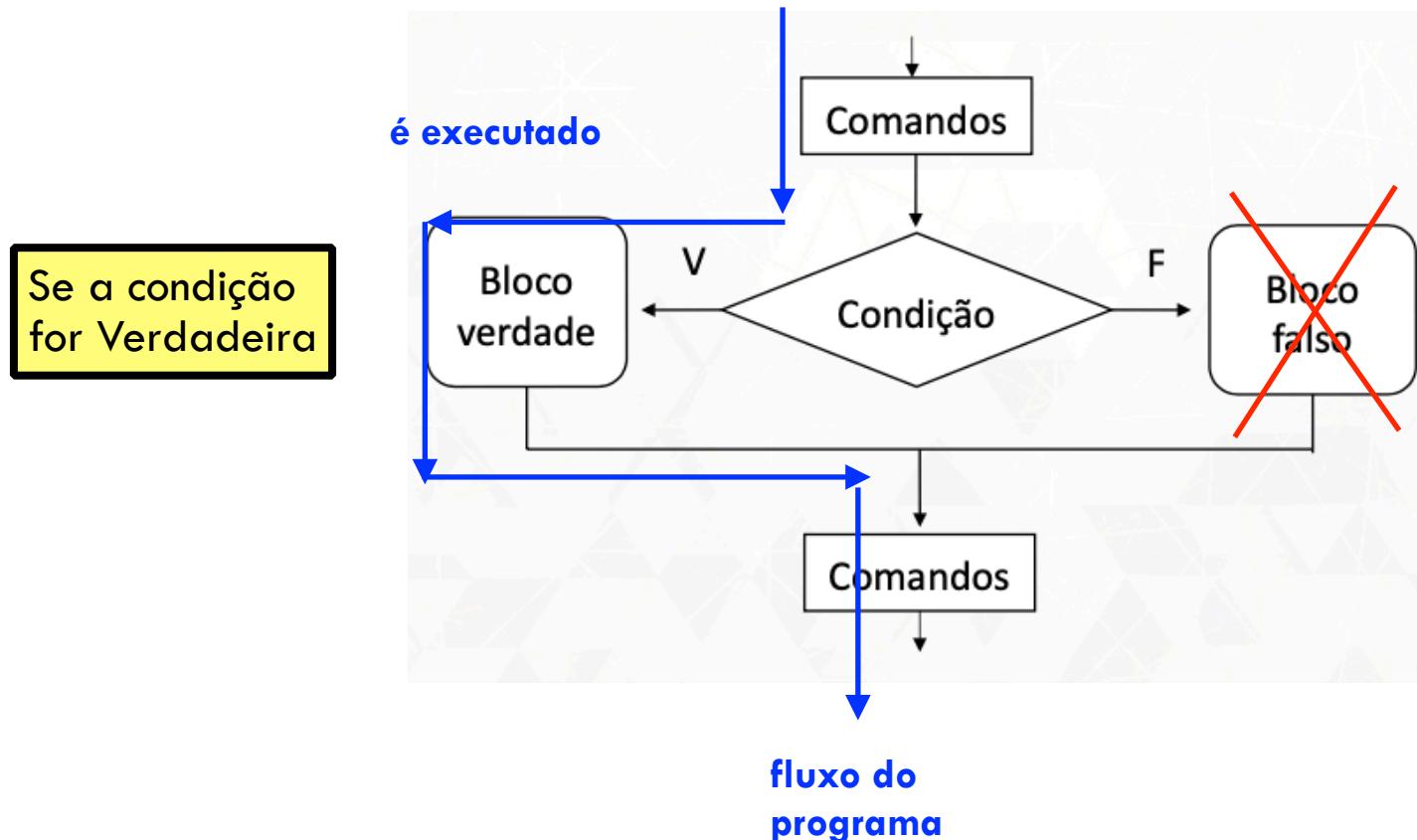
    return 0;
}
```



Declaração IF-ELSE

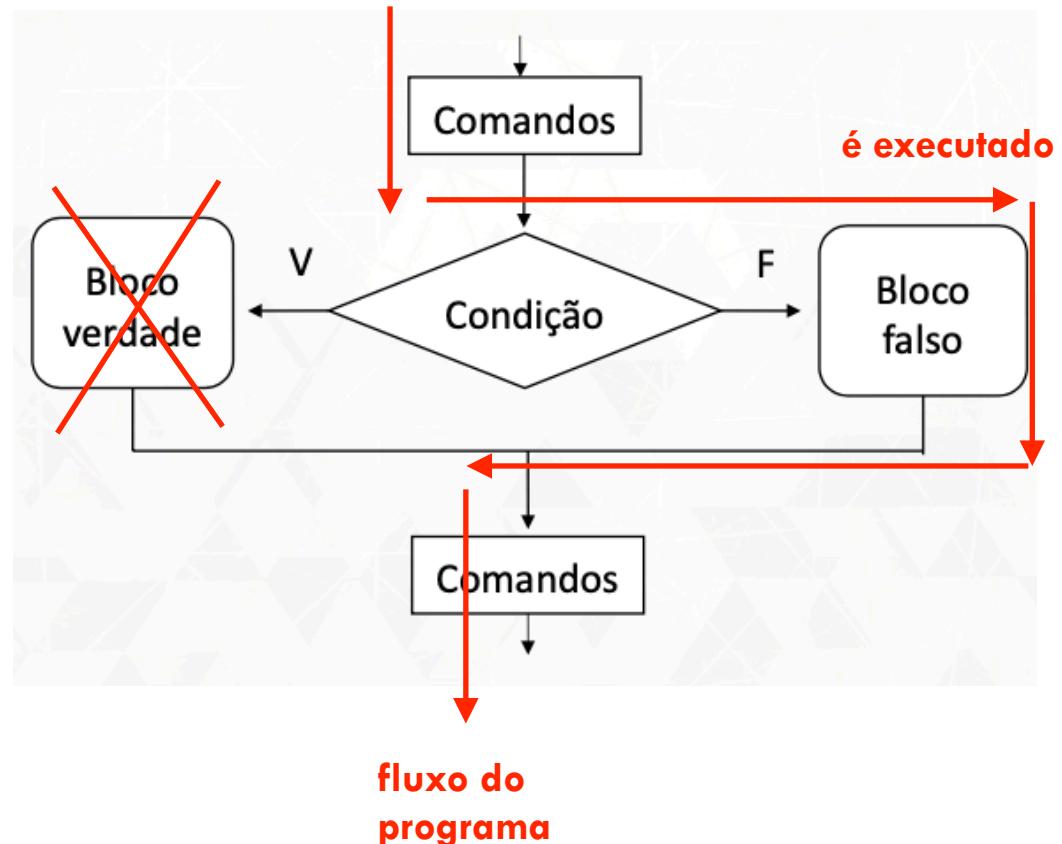


Declaração IF-ELSE



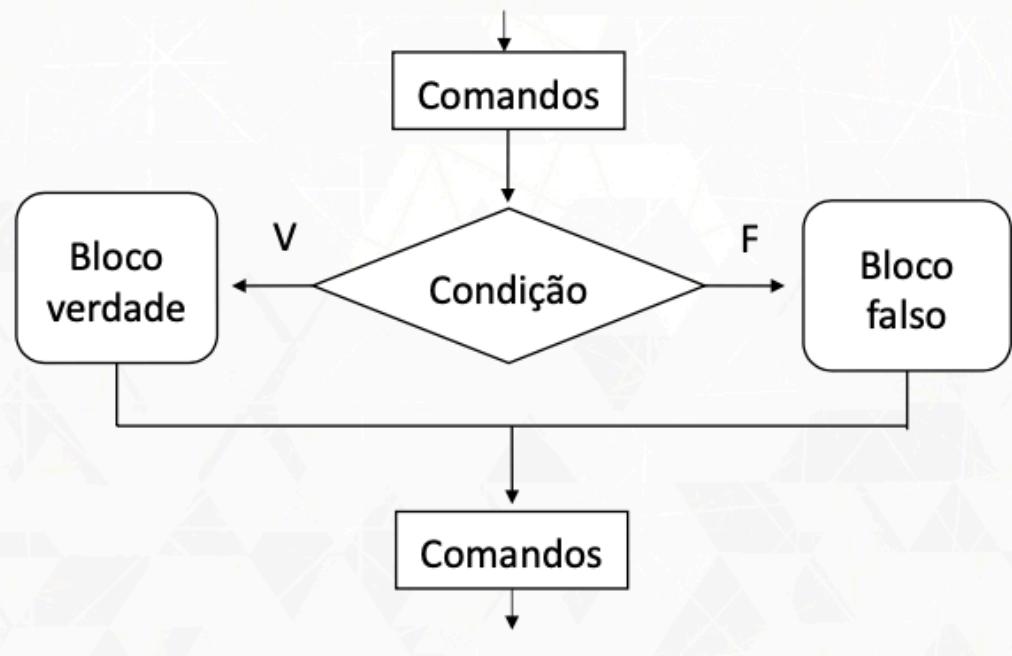
Declaração IF-ELSE

Se a condição
for Falsa



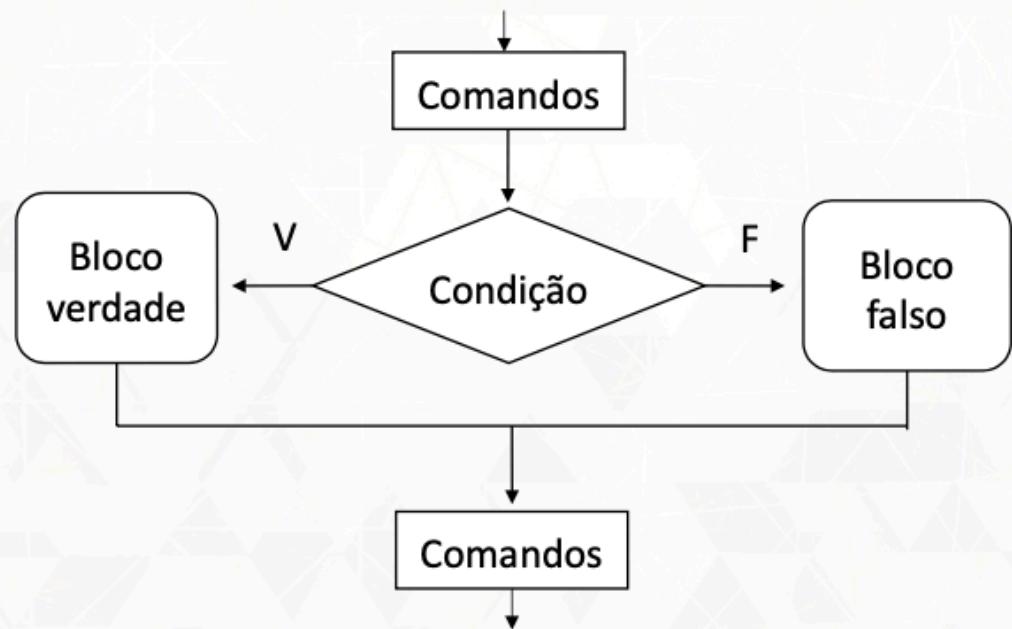
Declaração IF-ELSE

```
//comandos  
  
if(condição)  
{  
    //bloco verdade  
}  
else  
{  
    //bloco falso  
}  
  
//comandos
```



Declaração IF-ELSE

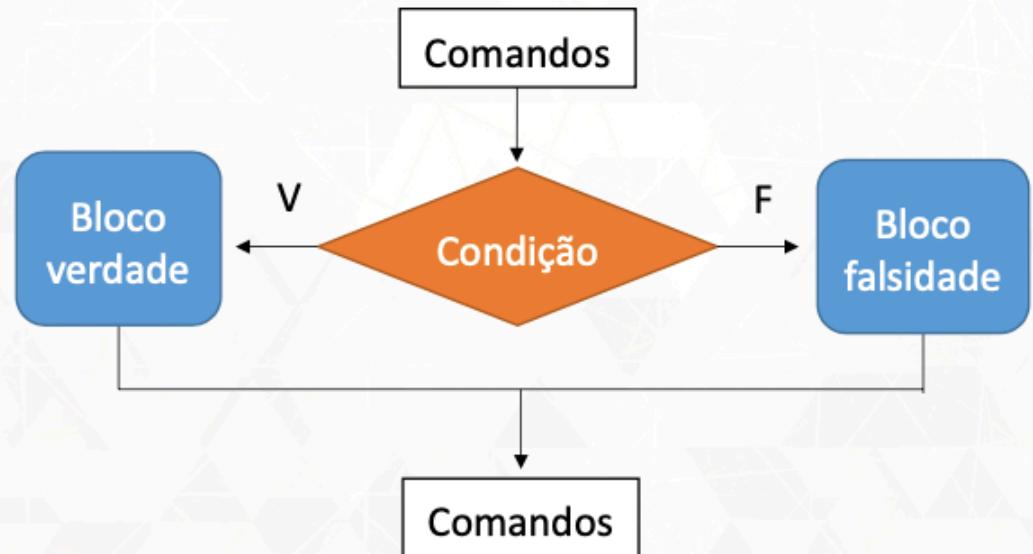
```
//comandos  
  
if(condição)  
{  
    //bloco verdade  
}  
else  
{  
    //bloco falso  
}  
  
//comandos
```



A declaração *else* é opcional: pode-se utilizá-la para determinar um conjunto de comandos que serão executados caso a condição testada seja falsa

Declaração IF-ELSE

```
int a = 8;  
  
if(a == 5)  
    printf("a vale 5");  
else  
    printf("a não vale 5");  
  
a = 3;
```



Somente um dos conjuntos de comandos (*if* ou *else*) será executado, nunca os dois!

Bloco de Instrução

- Com a adição de estruturas de controle segue um novo conceito: bloco de instruções
- Bloco:** Conjunto de instruções agrupadas e delimitadas por chaves { }
- Se o bloco for de apenas uma instruções, pode-se omitir as {}
- Exemplo:

```
if(x > 100)
    printf("x maior que 100");
}
else
    printf("x não maior que 100");
```

Bloco de instruções - Erros comuns I

```
int main(){
    float saldo = 150.0;
    float saque = 200.0;

    if(saldo - saque >= 0)
        saldo = saldo - saque;
        printf("Saque realizado com sucesso. Saldo atual = %f", saldo);
    else
        printf("Impossivel realizar o saque. Saldo insuficiente");

    return 0;
}
```

Onde está o erro?

Bloco de instruções - Erros comuns II

```
int main(){
    float saldo = 150.0;
    float saque = 50.0;

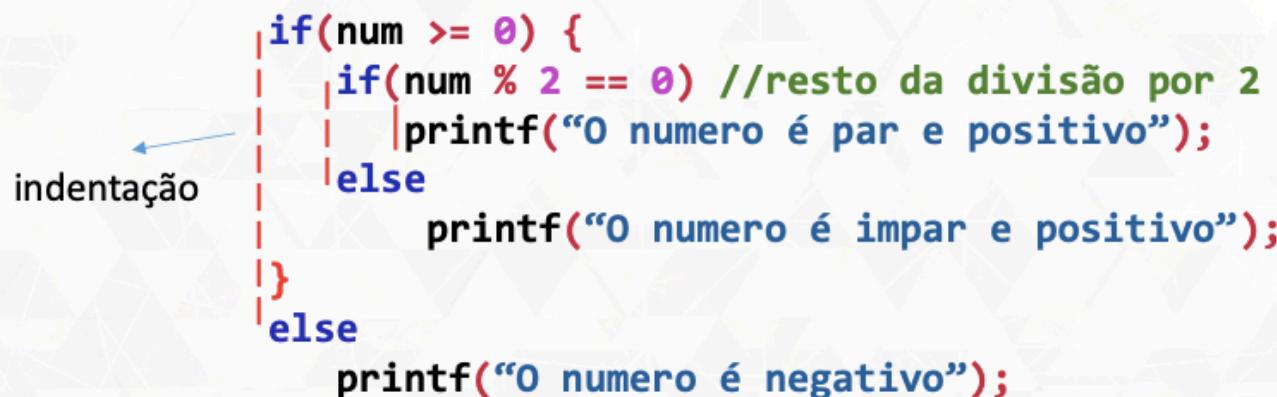
    if(saldo - saque >= 0){
        saldo = saldo - saque;
        printf("Saque realizado com sucesso. Saldo atual = %f", saldo);
    }
    else
        printf("Impossivel realizar o saque.");
        printf("Informe um valor menor ou igual a %f", saldo);

    return 0;
}
```

Onde está o erro?

Estrutura de seleção aninhadas

- É possível também aninhar comando **if**, ou seja, fazer uma declaração **if** dentro de outra declaração **if** anterior
- **Exemplo:** examinar se um número é positivo. Em caso afirmativo, verificar se o mesmo é divisível 2

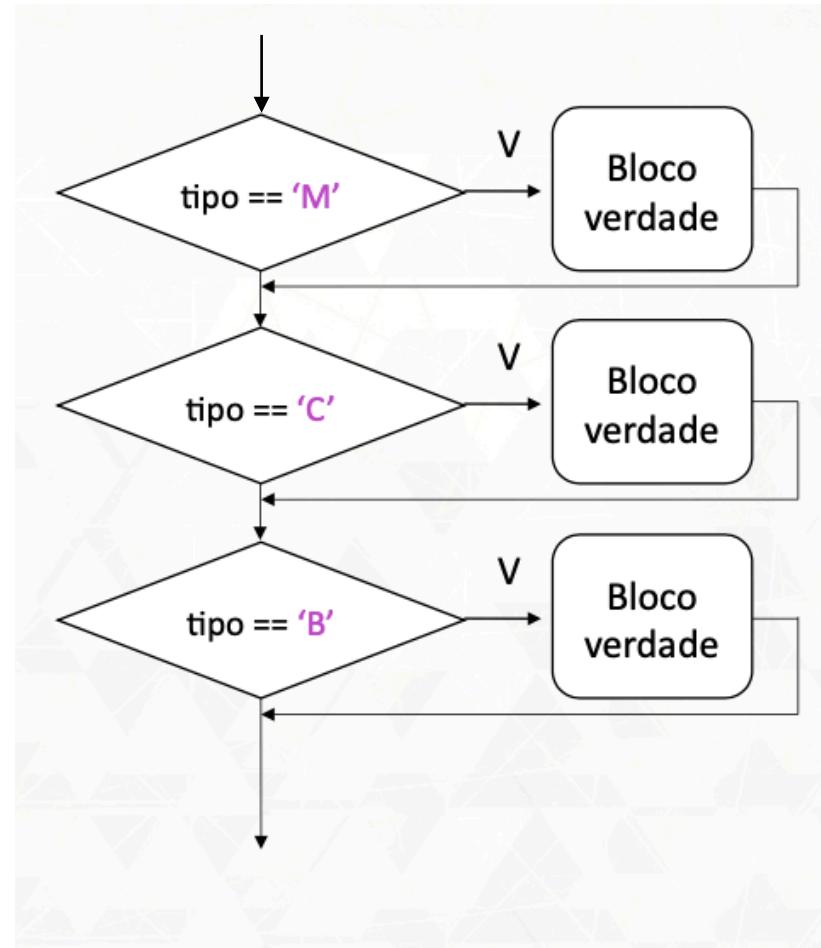


indentação

```
if(num >= 0) {  
    if(num % 2 == 0) //resto da divisão por 2  
        printf("O numero é par e positivo");  
    else  
        printf("O numero é impar e positivo");  
}  
else  
    printf("O numero é negativo");
```

Um **if** aninhado é um comando **if** que está dentro de um outro comando **if**.

Estrutura de seleção aninhadas

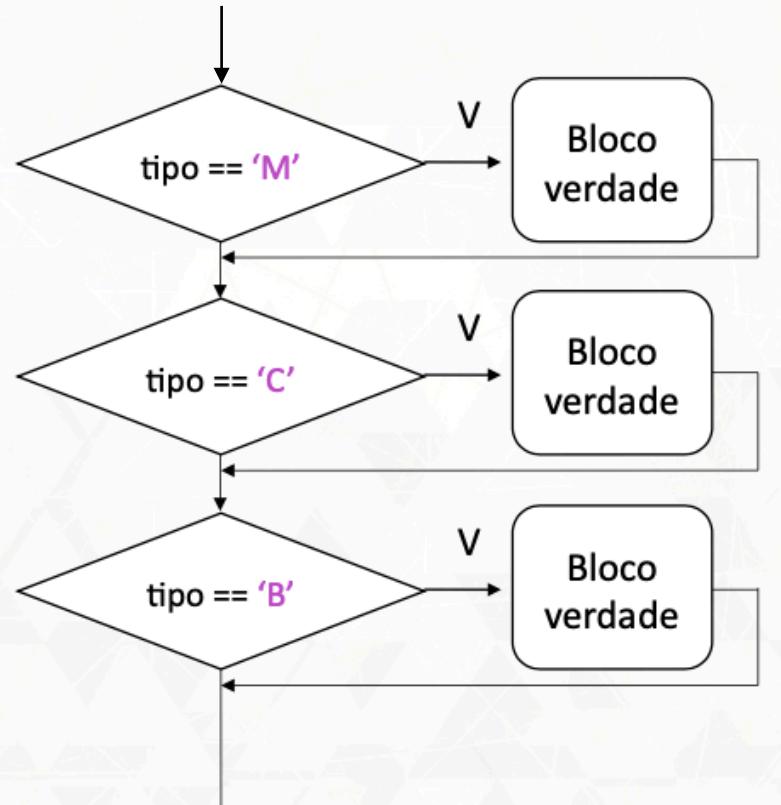


Estrutura de seleção aninhadas

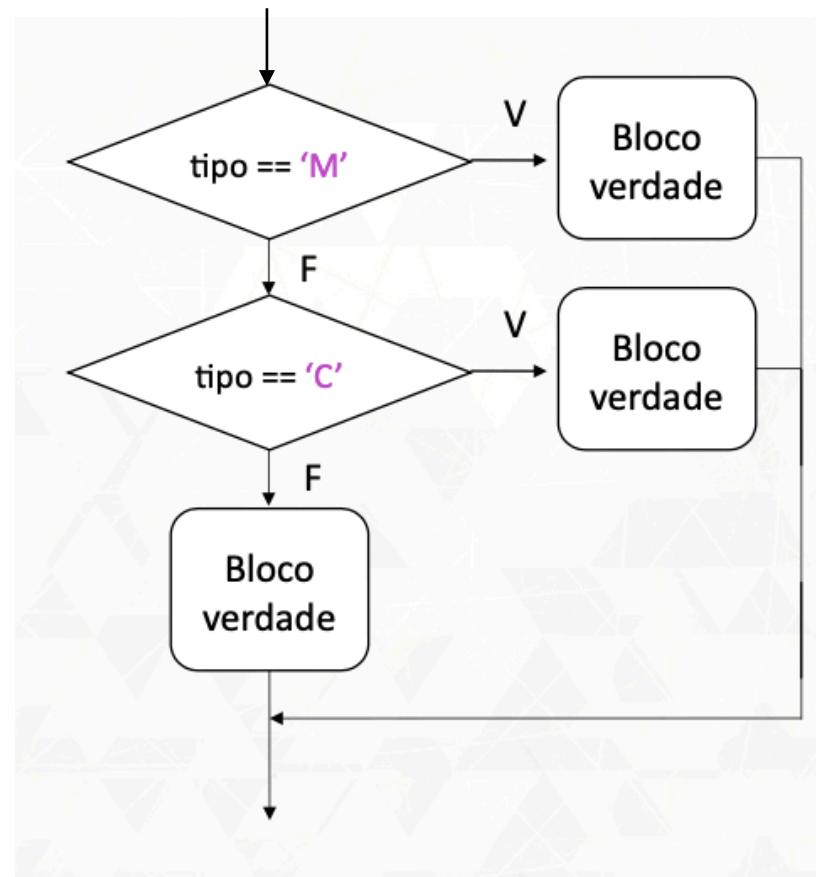
```
int main(){
    char tipo;
    scanf("%c", &tipo);

    if(tipo == 'M')
        printf("Mucarela");
    if(tipo == 'C')
        printf("Calabresa");
    if(tipo == 'B')
        printf("Bacon");

    return 0;
}
```



Estrutura de seleção aninhadas

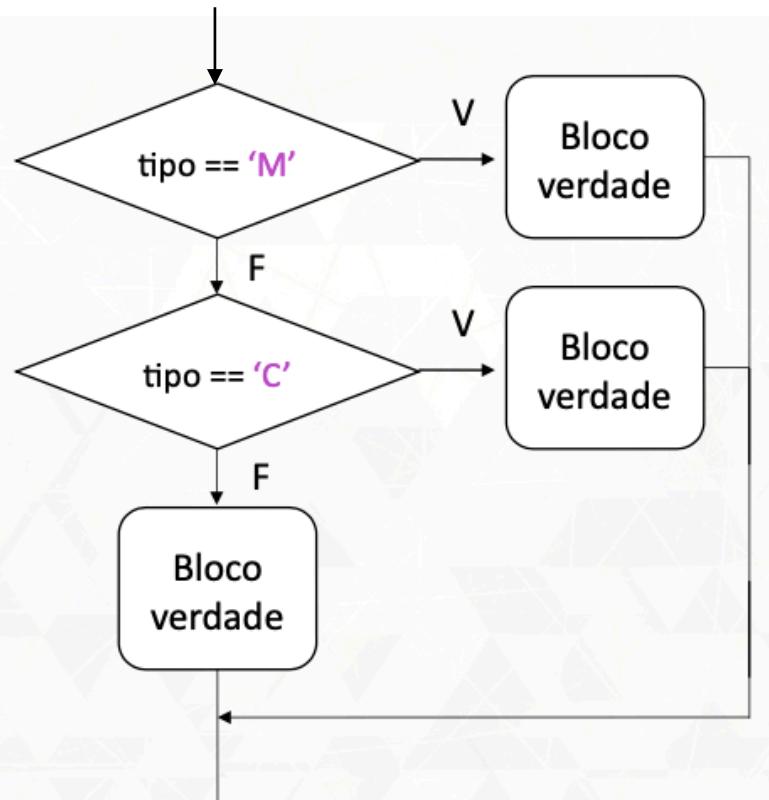


Estrutura de seleção aninhadas

```
int main(){
    char tipo;
    scanf("%c", &tipo);

    if(tipo == 'M')
        printf("Mucarela");
    else if(tipo == 'C')
        printf("Calabresa");
    else
        printf("Bacon");

    return 0;
}
```



Declaração IF-ELSE-IF

```
int num = 10;

if(num < 100)
    printf("Menor que 100");
if(num < 1000)
    printf("Menor que 1000");
if(num < 10000)
    printf("Menor que 10000");
if(num >= 10000)
    printf("Maior ou igual a 10000");
```

```
int num = 10;

if(num < 100)
    printf("Menor que 100");
else
    if(num < 1000)
        printf("Menor que 1000");
    else
        if(num < 10000)
            printf("Menor que 10000");
        else
            printf("Maior ou igual a 10000");
```

```
int num = 10;

if(num < 100)
    printf("Menor que 100");
else if(num < 1000)
    printf("Menor que 1000");
else if(num < 10000)
    printf("Menor que 10000");
else
    printf("Maior ou igual a 10000");
```

- As condições são avaliadas de cima para baixo;
- Assim que uma condição verdadeira é encontrada, o comando associado a ela é executado;
- O restante das condições não são executadas;
- Se nenhuma das condições for verdadeira, o último `else` é executado.

Operadores Lógicos

- Trabalham com valores booleanos e seu resultado também é booleano (verdadeiro ou falso)
- Eles são usados somente em expressões lógicas
- Usados para combinar condições simples, criando condições complexas

Operador	Significado	Equivalente
!	Negação	Não
&&	Conjunção	E
	Disjunção	OU

Operadores Lógicos

Operador **E**: gera um valor **VERDADEIRO** se ambas as condições forem verdadeiras



`Idade > 5 && idade < 10`

Quando idade é 6, qual o resultado?

Quando idade é 11, qual o resultado?

Quando idade é 5, qual o resultado?

Operadores Lógicos

Operador **E**: gera um valor **VERDADEIRO** se ambas as condições forem verdadeiras



`Idade > 5 && idade < 10`

Quando idade é 6, qual o resultado? **VERDADEIRO**

Quando idade é 11, qual o resultado? **FALSO**

Quando idade é 5, qual o resultado? **FALSO**

Operadores Lógicos

Operador **OU**: gera um valor **VERDADEIRO** se ao menos uma das condições for verdadeira



Idade > 5 || idade == 3

Quando idade é 6, qual o resultado?

Quando idade é 2, qual o resultado?

Quando idade é 3, qual o resultado?

Operadores Lógicos

Operador **OU**: gera um valor **VERDADEIRO** se ao menos uma das condições for verdadeira



Idade > 5 || idade == 3

Quando idade é 6, qual o resultado? **VERDADEIRO**
Quando idade é 2, qual o resultado? **FALSO**
Quando idade é 3, qual o resultado? **VERDADEIRO**

Operadores Lógicos

Operador **NÃO**: inverte o valor retornado por uma condição



`! (idade > 5)`

Quando `idade` é 6, qual o resultado?
Quando `idade` é 2, qual o resultado?

Operadores Lógicos

Operador **NÃO**: inverte o valor retornado por uma condição



`! (idade > 5)`

Quando idade é 6, qual o resultado? **FALSO**
Quando idade é 2, qual o resultado? **VERDADEIRO**

Operadores Lógicos

Operador **E**: gera um valor **VERDADEIRO** se ambas as condições forem verdadeiras



`idade > 5 && idade < 10`

Quando idade é 6, qual o resultado? **VERDADEIRO**

Quando idade é 11, qual o resultado? **FALSO**

Quando idade é 5, qual o resultado? **FALSO**

Operador **OU**: gera um valor **VERDADEIRO** se ao menos uma das condições for verdadeira



`idade > 5 || idade == 3`

Quando idade é 6, qual o resultado? **VERDADEIRO**

FALSO

VERDADEIRO

Quando idade é 2, qual o resultado?

Quando idade é 3, qual o resultado?

Operador **NÃO**: inverte o valor retornado por uma condição



`! (idade > 5)`

Quando idade é 6, qual o resultado? **FALSO**

VERDADEIRO

Quando idade é 2, qual o resultado?

Operadores Lógicos

- Se quisermos saber se um número **num** é positivo e par

```
if(num >= 0){  
    if(num % 2 == 0)  
        printf("Numero par e positivo");  
}
```

Combinando as duas condições em apenas uma...

```
if(num >= 0 && num % 2 == 0)  
    printf("Numero par e positivo");
```

Exercícios

- 1 Faça um programa que leia três números quaisquer e imprima o maior deles
- 2 Faça um programa que duas notas de um aluno. Calcule e mostre a média aritmética das notas e uma mensagem conforme a tabela a seguir

Média	Mensagem
[0, 4)	Reprovado
[4, 6)	Exame
[6, 10]	Aprovado

Exercícios

3 Desenvolver a lógica para um programa que efetua o cálculo do reajuste de salário de um funcionário

- Salário <= 500, reajuste 15%
- Salário > 500, mas Salário <= 1000, reajuste será de 10%
- Salário > 1000, reajuste será de 5%

Ao final do programa, mostrar o valor do salário somado ao reajuste.

Exercícios

4

Faça um programa que recebe a idade de um nadador e classifique-o numa das seguintes categorias:

- Adulto (idade ≥ 18)
- Juvenil (idade ≥ 14 e idade < 18)
- Infantil (idade ≥ 9 e idade < 14)
- Mirim (idade < 9)

Roteiro

- 1** Introdução
- 2** Declarações IF, IF-ELSE
- 3** Declaração SWITCH
- 4** Referências

Declaração SWITCH

- Construções **if-else** facilitar a escrita de programas que devem escolher uma entre duas alternativas
- Algumas vezes, entretanto, o programa necessita escolher uma entre várias alternativas
- Embora **if-else** possa ser alinhado para executar vários testes condicionais, elas podem tornar o código muito complexo
- Uma solução a ser considerada é utilizar uma construção para teste multi-condicional chamado **switch**
- Porém, só opera com tipos de variável **char** ou **int**

Declaração SWITCH

```
switch(variável do tipo char ou tipo int){  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    default:  
        ...comandos se nenhum case for atendido...  
        break;  
}
```

Declaração SWITCH

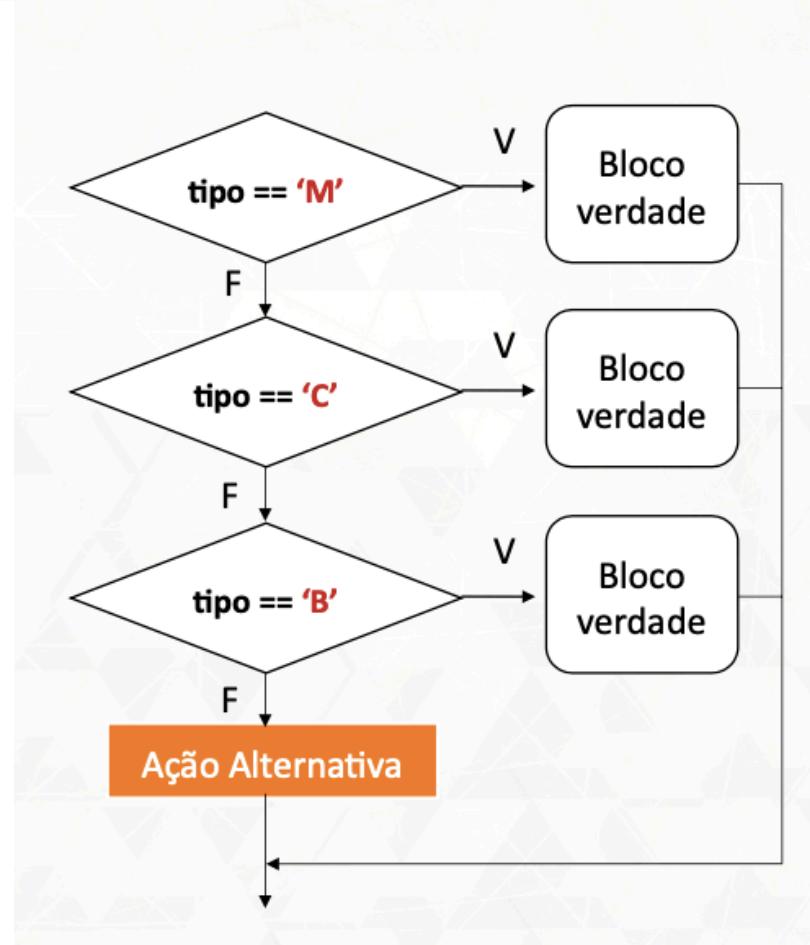
```
switch(variável do tipo char ou tipo int){  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    case <valor possível da variável>:  
        ...comandos...  
        break;  
    default:  
        ...comandos se nenhum case for atendido...  
        break;  
}
```

O **break** indica o término das instruções de um **case**

O **default** é utilizado se nenhum dos **cases** for atendido

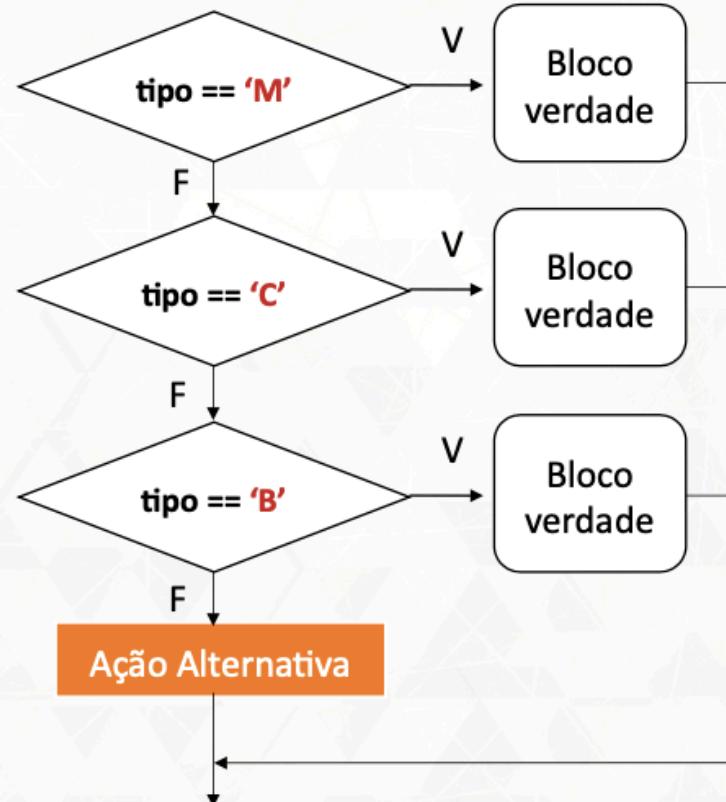
O comando **switch** só aceita variáveis do tipo **int** e **char**.

Declaração SWITCH



Declaração SWITCH

```
1 int main()
2 {
3     char tipo;
4     scanf("%c", &tipo);
5
6     switch(tipo)
7     {
8         case 'M':
9             printf("Mucarela");
10            break;
11         case 'C':
12             printf("Calabresa");
13             break;
14         case 'B':
15             printf("Bacon");
16             break;
17         default:
18             printf("opcao invalida");
19             break;
20     }
21     return 0;
22 }
```



Declaração SWITCH

```
1 int main()
2 {
3     int opcao;
4     printf("Digite uma opção (1, 2 ou 3)\n");
5     scanf("%i", &opcao);
6
7     switch(opcao)
8     {
9         case 1:
10            printf("Escolheu a opção 1");
11            break;
12        case 2:
13            printf("Escolheu a opção 2");
14            break;
15        case 3:
16            printf("Escolheu a opção 3");
17            break;
18        default:
19            printf("opção invalida");
20            break;
21    }
22    return 0;
23 }
```

Declaração SWITCH

```
1 int main()
2 {
3     int opcao;
4     printf("Digite uma opção (1, 2 ou 3)\n");
5     scanf("%i", &opcao);
6
7     switch(opcao)
8     {
9         case 1:
10            printf("Escolheu a opção 1");
11            break;
12        case 2:
13            printf("Escolheu a opção 2");
14            break;
15        case 3:
16            printf("Escolheu a opção 3");
17            break;
18        default:
19            printf("opção invalida");
20            break;
21    }
22    return 0;
23 }
```

O computador pede sua escolha

Declaração SWITCH

```
1 int main()
2 {
3     int opcao;
4     printf("Digite uma opção (1, 2 ou 3)\n");
5     scanf("%i", &opcao);
6
7     switch(opcao) ←
8     {
9         case 1:
10            printf("Escolheu a opção 1");
11            break;
12        case 2:
13            printf("Escolheu a opção 2");
14            break;
15        case 3:
16            printf("Escolheu a opção 3");
17            break;
18        default:
19            printf("opção invalida");
20            break;
21    }
22    return 0;
23 }
```

De acordo com o valor, dispara
o bloco de comandos correspondente

Exercícios

5

Crie um programa onde:

- o usuário deve fornecer um valor
- o programa deve responder com o nome do dia da semana correspondente
- O programa não deve aceitar valores fora da faixa convencional, e deve apresentar uma mensagem de erro

1 → Domingo
2 → Segunda-feira
3 → Terça-feira
4 → Quarta-feira
5 → Quinta-feira
6 → Sexta-feira
7 → Sábado

Exercícios

6 Crie um programa que inicialmente receba dois números inteiros. Depois disso, mostre um menu com as seguintes opções:

1. Adição
2. Subtração
3. Multiplicação
4. Divisão

Peça para o usuário informar um valor de 1 a 4. Baseado nesse valor, calcule e mostre o resultado da opção escolhida pelo usuário, levando em consideração os dois números inteiros recebidos.

Exercícios

7

Final Boss: implemente o jogo Pedra-Papel-Tesoura



Exercícios

7

Final Boss: implemente o jogo Pedra-Papel-Tesoura

Dica: usar números aleatórios para realizar a jogada do computador. Utilize os seguintes comandos no seu programa:

Exercícios

7

Final Boss: implemente o jogo Pedra-Papel-Tesoura

Dica: usar números aleatórios para realizar a jogada do computador. Utilize os seguintes comandos no seu programa:

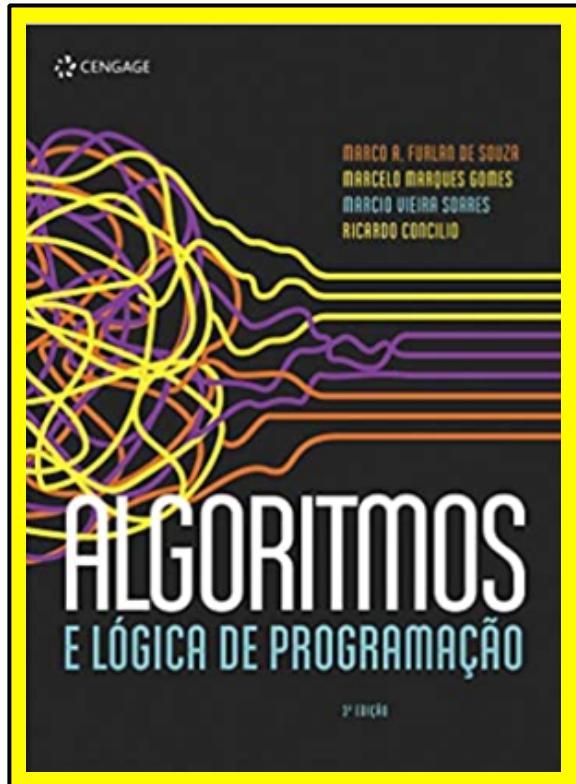
```
#include <stdlib.h> // contém a função srand, que gera números aleatórios
#include <time.h> /* trabalha com o tempo do computador, p sempre gerar
números diferentes */

int main () {
    int i;
    srand( (unsigned) time (NULL)); // inicia gerador de números aleatórios
    i = rand() % 3; // gera números aleatórios entre 0 e 2 {0, 1, 2}
    // ... seu programa continua daqui
}
```

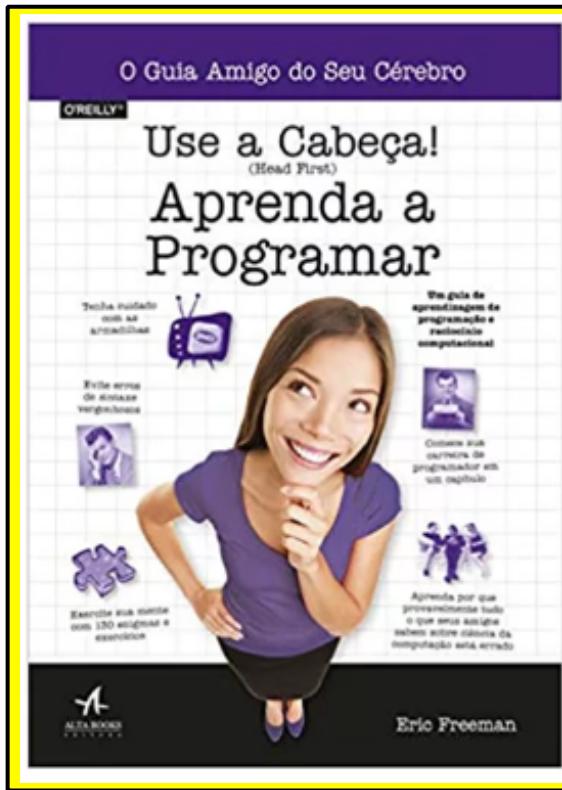
Roteiro

- 1 Introdução**
- 2 Declarações IF, IF-ELSE**
- 3 Declaração SWITCH**
- 4 Referências**

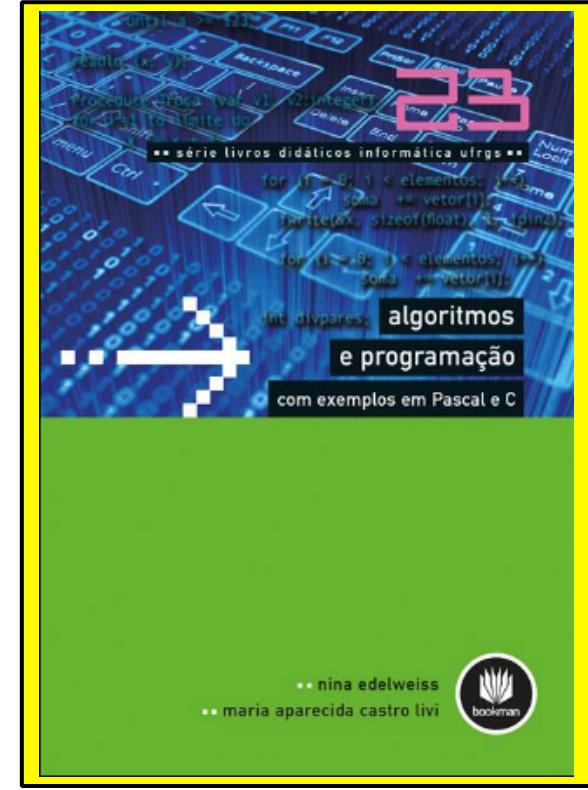
Referências sugeridas



[Souza et al, 2019]



[Freeman, 2019]



[Edelweiss & Livi, 2014]

Perguntas?

Prof. Rafael G. Mantovani

rafaelmantovani@utfpr.edu.br

Prof. Adalberto Lazarini

adalbertoz@utfpr.edu.br