AULA 03

{introcomp}

CONCEITOS BÁSICOS II

Estruturas de entrada e saída

- A linguagem C utiliza de algumas funções para tratamento de entrada e saída de dados.

- A maioria dessas funções estão presentes na biblioteca **<stdio.h>**.

- As funções mais comuns para entrada e saída de dados são o **scanf()** e o **printf()**. Ambas presentes na stdio.h.

Entrada de dados

scanf():

- Função de entrada de dados.
- Utilizada quando precisamos que algum dado seja submetido pelo usuário (de um teclado, por exemplo).
- Necessita de argumentos corretos para cada tipo de dado lido.

Saída de dados

printf():

- Função de saída de dados.
- Utilizada quando precisamos que algum dado seja impresso na saída de dados padrão.
- Necessita de argumentos corretos para cada tipo de dado impresso.

Entrada e saída de dados

- Como na linguagem C existem diferentes tipos de dados, devemos sempre explicitar qual o tipo queremos ler ou escrever.

- Esses parâmetros devem ser considerados nas funções scanf() e printf().

Entrada e saída de dados

Argumento de controle	Uso na linguagem C

%d – Usado para referenciar tipo inteiro.	`int a;`
	`scanf("%d", &a);`
	`printf("%d", a);`

%f – Usado para referenciar tipos flutuantes/reais.	`float a;`
	`scanf("%f", &a);`
	`printf("%f", a);`

%c – Usado para referenciar caracteres.	`char a;`
	`scanf("%c", &a);`
	`printf("%c", a);`

%lf – Usado para referenciar tipos flutuantes/reais com maior precisão.	`double a;`
	`scanf("%lf", &a);`
	`printf("%lf", a);`

Entrada e saída de dados

Pseudocódigo	Linguagem C
Leia(a)	`scanf("%d", &a);`
Escreva(a)	`printf("%d\n",a);`

Entrada e saída de dados – Exemplo 1

- Escreva um programa que leia dois números inteiros do teclado e mostre seus valores na tela.

Entrada e saída de dados – Exemplo 1

- Escreva um programa que leia dois números inteiros do teclado e mostre seus valores na tela.

```
int main()
{
   int a, b;
   scanf("%d %d", &a, &b);


   printf("%d %d", a, b);

   return 0;
}
```

Estruturas Condicionais

Objetivos:

- Compreender diversos tipos de controle de fluxo disponíveis na linguagem C.

- Aprender a implementar algoritmos em C com tomadas de decisões.

Controle de Fluxo

Por que utilizar controles de fluxo?

- Necessidade de tomada de decisões baseado em fatos passados;
- Repetição de certos trechos de código;
- Proporciona inteligência ao programa;

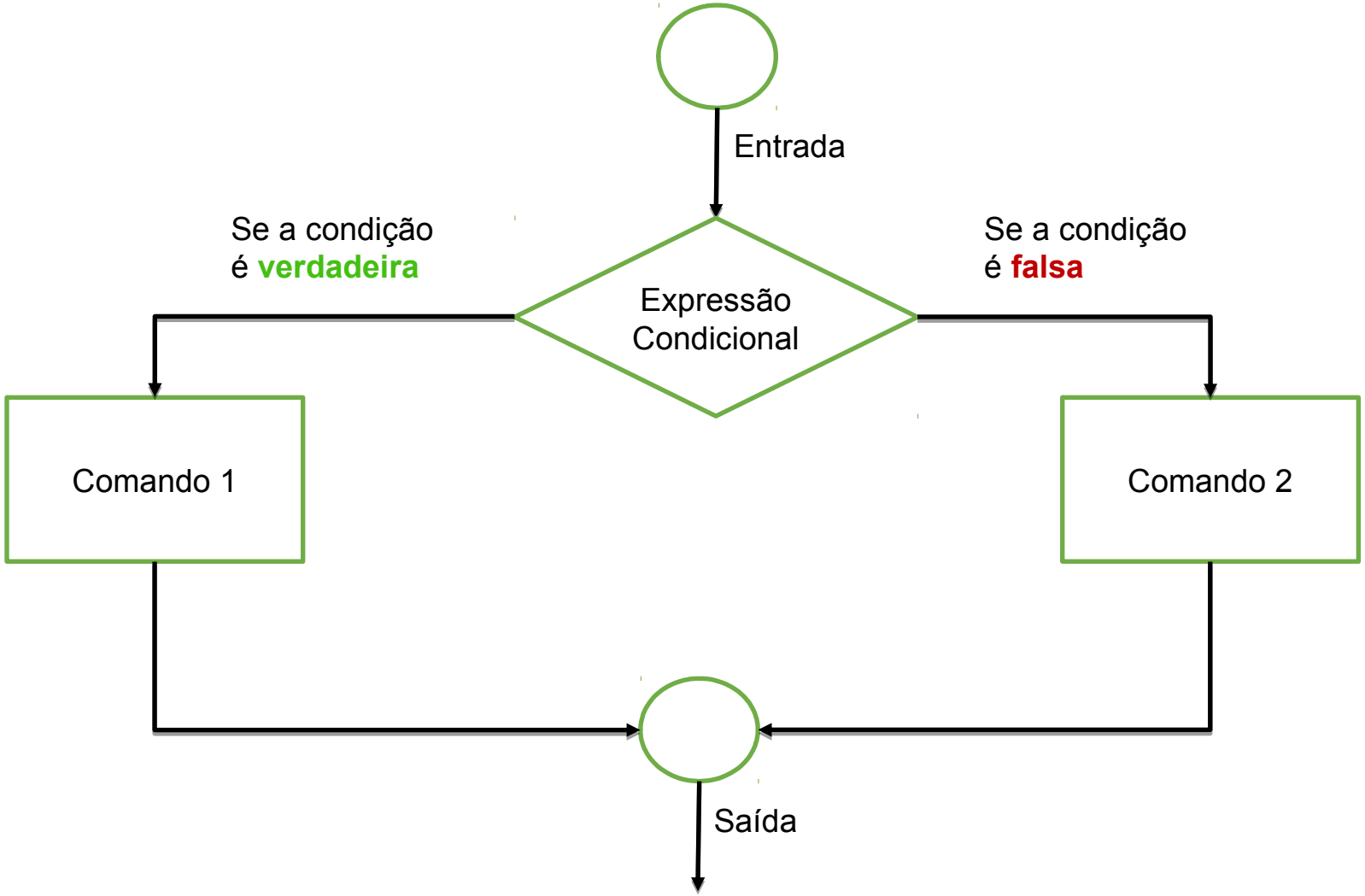
Controle de Fluxo

Existem três maneiras de controlar o fluxo de um programa:

- **Estruturas condicionais;**
- Estruturas de repetição;
- Desvios incondicionais;

Na aula de hoje, veremos **instruções condicionais**!

Controle de Fluxo Condicional



E na linguagem C?

Comando de seleção

Pseudocódigo	Linguagem C

se (Condição) **então**
 	
Códigos...
 	
senao
 	
Códigos...
 	
fim-se
	if(Condição)
	{
	Códigos...
	}
	else
	{
	Códigos...
	}

Instruções Condicionais - Expressões

- A linguagem C admite inúmeras formas de se criar expressões condicionais.

- Assim como aprendemos em pseudocódigo, essas expressões podem ser **verdadeiras** ou **falsas**.

- **Operadores relacionais** e **operadores lógicos** nos ajudam a criar expressões condicionais para tomada de decisão.

Operadores relacionais

Símbolo	Nome do operador	Exemplo	Significado
>	Maior que	if(x>y)	X é maior que Y
>=	Maior ou igual que	if(x>=y)	X é maior ou igual a Y
<	Menor que	if(x<y)	X é menor que Y
<=	Menor ou igual que	if(x<=y)	X é menor ou igual a Y
!=	Diferente de	if(x!=y)	X é diferente de Y
==	Igualdade	if(x==y)	X é igual a Y

Operadores relacionais – Exemplo 1

```
int main()
{
   int x = 3, y = 4;

   if(x>y)
   {
      x = x + 1;
   }
   printf("%d %d\n", x, y);
   return 0;
}
```

O código ao lado está **INCORRETO**?

Operadores relacionais – Exemplo 1

```
int main()
{
   int x = 3, y = 4;

   if(x>y)
   {
      x = x + 1;
   }
   printf("%d %d\n", x, y);
   return 0;
}
```

O código ao lado está **INCORRETO**?

O programa está **correto**.

O comando **if** não requer **necessariamente** de um **else**.

Operadores relacionais – Exemplo 2

```
int main()
{
   int x = 4, y = 4;

   else
   {
      x = x - 1;
      y = y + 1;
   }
   printf("%d %d\n", x, y);
   return 0;
}
```

O código ao lado está **INCORRETO**?

Operadores relacionais – Exemplo 2

```
int main()
{
   int x = 4, y = 4;

   else
   {
      x = x - 1;
      y = y + 1;
   }
   printf("%d %d\n", x, y);
   return 0;
}
```

O código ao lado está **INCORRETO**?

O programa está **incorreto**.

O comando **else** requer **necessariamente** de um **if**.

Outra forma de fazer um if

- Na linguagem C uma variável sem um operador relacional é considerada verdadeira se for diferente de zero.

Operadores relacionais – Exemplo 3

```
int main()
{
   int x = 4, y = 10;
   if (x){
      x = y;
   }
   printf (" x = %d\n",x);
}
```

O código ao lado está **INCORRETO**?

Operadores relacionais – Exemplo 3

```
int main()
{
    int x = 4, y = 10;
    if (x){
        x = y;
    }
    printf (" x = %d\n",x);
}
```

O código ao lado está **INCORRETO**?

O programa está **correto**.

A variável x é considerada como true nesse caso.

Expressões Condicionais

Observação:

O C não permite condições como:

```
if(x == y == z)
{
   // código...
}
```

Então como comparar mais de duas variáveis ?

Operadores Lógicos

- Para resolver o problema de comparar duas ou mais variáveis utilizaremos os **operadores lógicos**.

Operador	Significado
||	ou
&&	e
!	não

Operadores Lógicos

- Então para resolvermos o problema anterior de comparar 3 variáveis faremos:

```
if((x == y) && (x == z))
{
   // código...
}
```

Operadores Lógicos – Exemplo 1

```
int main()
{
   int x = 3, y = 4 , z = 5;

   if((x<y) || (x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

Operadores Lógicos – Exemplo 1

```
int main()
{
   int x = 3, y = 4 , z = 5;

   if((x<y) || (x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

Saída: x = 4

Operadores Lógicos – Exemplo 2

```
int main()
{
   int x = 3, y = 4 , z = 5;

   if((x<y) && (x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

Operadores Lógicos – Exemplo 2

```
int main()
{
   int x = 3, y = 4 , z = 5;

   if((x<y) && (x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

Saída: x = 3

Operadores Lógicos – Exemplo 3

```
int main()
{
   int x = 3, y = 4 , z = 5;

   if(!(x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

Operadores Lógicos – Exemplo 3

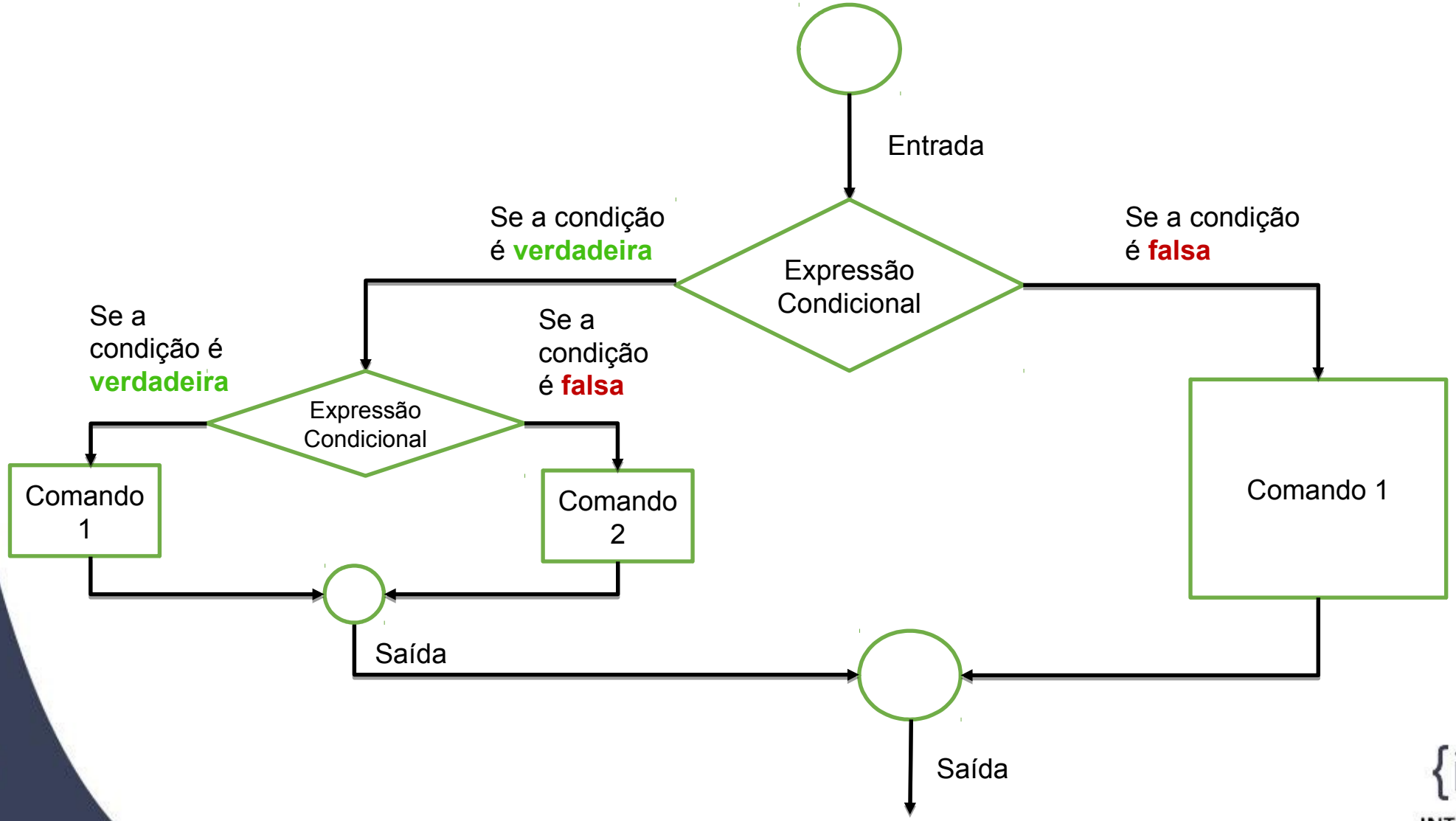
```
int main()
{
   int x = 3, y = 4 , z = 5;

   if(!(x > z))
   {
      x = x + 1;
   }
   printf("x = %d", x);
   return 0;
}
```

Qual a saída do código ao lado?

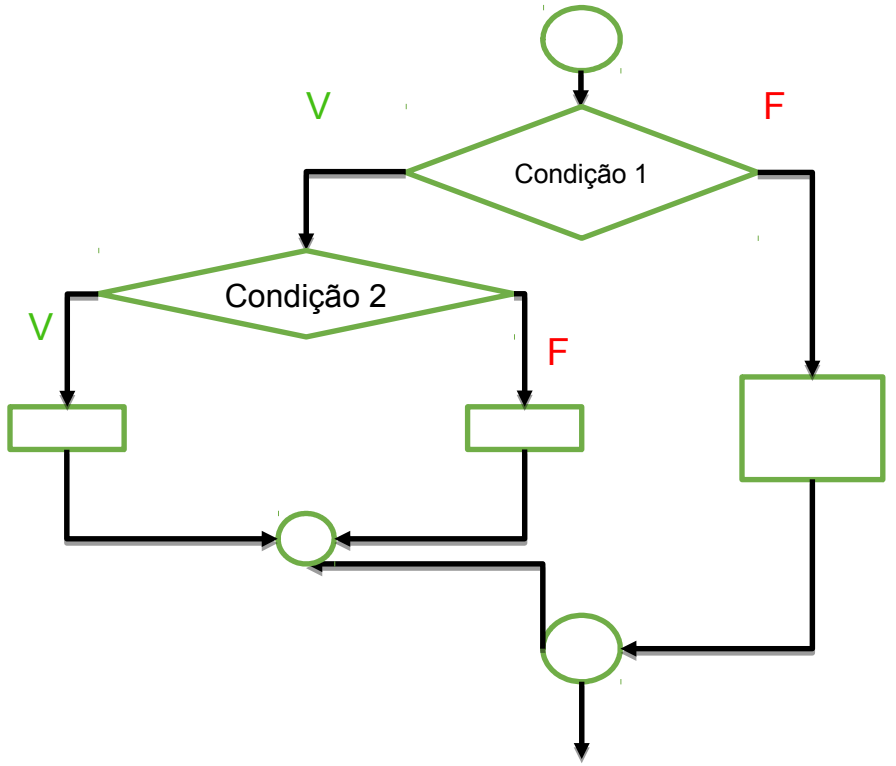
Saída: x = 4

Estruturas de seleção encadeadas

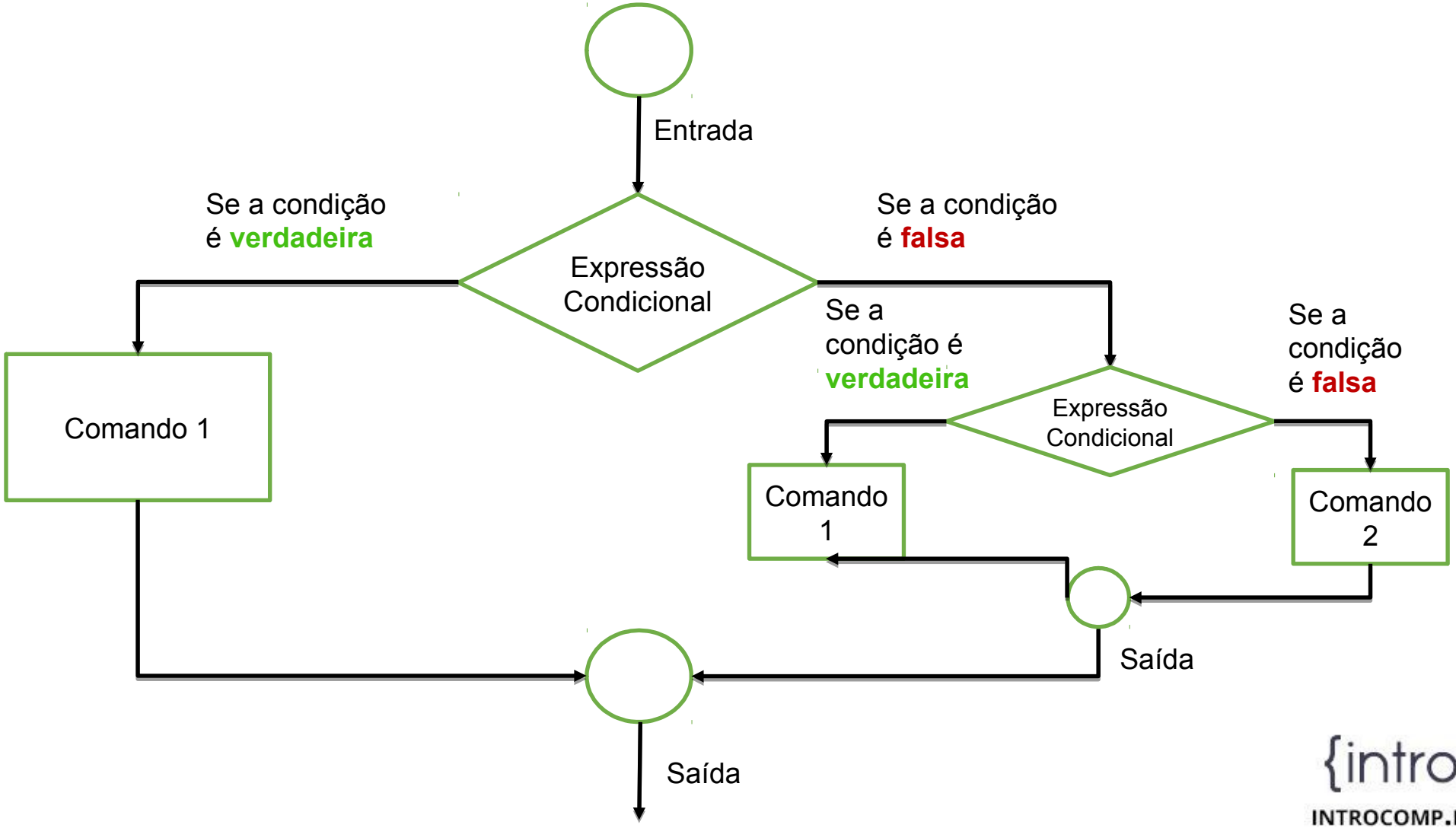


Estruturas de seleção encadeadas

```
if(condição 1)
{
   if(condição 2)
   {
      // código...
   }
   else
   {
      // código...
   }
}
else
{
   // código...
}
```

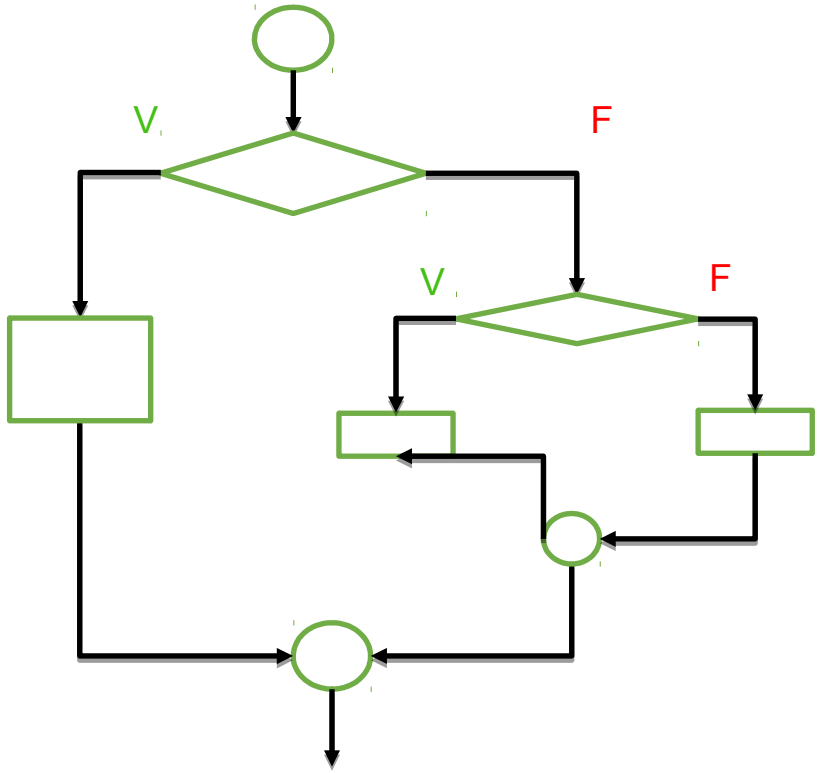


Estruturas de seleção encadeadas



Estruturas de seleção encadeadas

```
if(condição 1)
{
   // código...
}
else
{
   if(condição 2)
   {
      // código...
   }
   else
   {
      // código...
   }
}
```



Estruturas de seleção encadeadas

```
if(condição 1)
{
   // código...
}
else if(condição 2)
      {
      // código...
      }
      else
      {
      // código...
      }
```

Estruturas Condicionais – Exemplos

- Dado um número do teclado, imprima este número e informe se ele é positivo, negativo ou nulo.

Estruturas Condicionais – Exemplos

```
#include <stdio.h>

int main(){
    int a;
    scanf(“%d”, &a);
    if(a > 0)
    {
        printf(“%d é positivo\n”, a);
    }
    else
    {
        if(a < 0)
        {
            printf(“%d é negativo\n”, a);
        }
        else
        {
            printf(“%d é nulo\n”, a);
        }
```

Estruturas Condicionais – Exemplos

Dado um número, diga se ele é:

- Menor ou igual a 100;
- Entre 100 e 200;
- Maior ou igual a 200.

Estruturas Condicionais – Exemplos

```
#include <stdio.h>

int main(){
    int a;
    scanf(“%d”, &a);
    if(a <= 100)
    {
        printf(“%d é menor ou igual a 100.\n”, a);
    }
    else
    {
        if(a > 100 && a < 200)
        {
            printf(“%d está entre 100 e 200\n”, a);
        }
        else
        {
            printf(“%d é maior ou igual a 200\n”, a);
        }
```

Estrutura condicional – Switch/case

- Uma outra estrutura condicional existente na linguagem C é o Switch/Case.

- O switch/case é usado de modo similar a um if com várias instruções, porém só pode verificar uma relação de igualdade.

Estrutura condicional – Switch/case

```
switch( opção )
{
    case opção1:
            comandos caso a opção 1 tenha sido escolhida
            break;

    case opção2:
            comandos caso a opção 2 tenha sido escolhida
            break;

    case opção3:
            comandos caso a opção 3 tenha sido escolhida
            break;

    default:
            comandos caso nenhuma das opções
anteriores tenha sido escolhida
```

Estrutura Condicional – Exemplos

Faça um código, utilizando switch/case, que simule uma urna eleitoral. Os números para votar nos candidatos são:

- 13 para Iriny Lopes
- 23 para Rezende
- 43 para Luiz Paulo

Caso seja digitado outro número, o voto deverá ser considerado nulo.

Estrutura Condicional – Exemplos

```
#include <stdio.h>

Int main(){

    Int candidato;
    printf(“Digite o numero do seu candidato e tecle ENTER: “);
    Scanf(“%d”, &candidato);

    Switch (candidato)
    {
        Case 13:
            printf (“Você votou na Iriny Lopes!\n”);
            break;
        Case 23:
            printf (“Você votou na Rezende!\n”);
            break;
        Case 43:
            printf (“Você votou na Luiz Paulo!\n”);
            break;
```

AULA 03

{introcomp}

CONCEITOS BÁSICOS II