



Algoritmos e Programação de Computadores

Disciplina 113476

Prof. Alexandre Zaghetto
<http://alexandre.zaghetto.com>
zaghetto@unb.com

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

O presente conjunto de *slides* não pode ser reutilizado ou republicado sem a permissão do instrutor.

Módulo 06

Algoritmos com Repetições

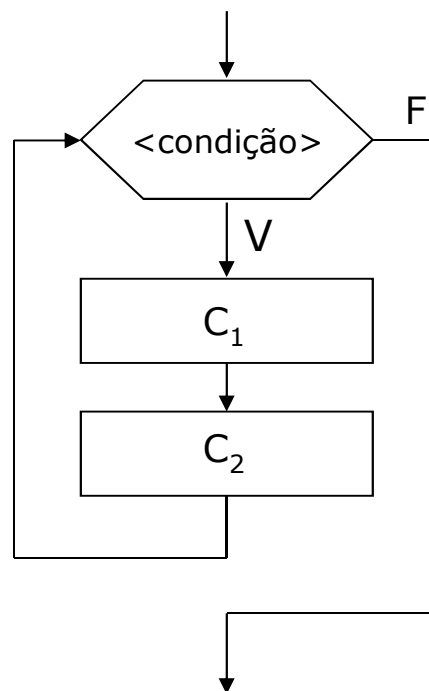
1. Repetições com Teste no Início

- Forma geral em Portugal:

```
enquanto <expressão> faça  
    <lista de comandos>  
fimenquanto
```

1. Repetições com Teste no Início

- Fluxograma:



1. Repetições com Teste no Início

Exemplo 1: Escreva um algoritmo que solicita ao usuário um valor inteiro positivo N e imprime na tela do computador todos os número inteiros de 0 a N .

1. Repetições com Teste no Início

- Portugal:

algoritmo "mostraN"

var

inteiro: N, contador;

inicio

escreva ("Entre com o número N:");

leia (N);

 contador = 0;

enquanto contador <= N faca

escreva (contador);

 contador = contador + 1;

fimenquanto

fimalgoritmo

1. Repetições com Teste no Início

- Forma geral em C:

while (<condição>) {



<comandos>

}



1. Repetições com Teste no Início

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, contador;

    printf("Digite o valor de N:");
    scanf("%d", &N);
    contador = 0;
    while(contador <= N) {
        printf("%d\n", contador);
        contador = contador + 1;
    }

    return 0;
}
```

1. Repetições com Teste no Início

Exemplo 2: Escreva um programa que solicita ao usuário N valores reais positivos, calcula e imprime na tela do computador o somatório dos números digitados. O programa deve continuar solicitando valores até que o valor -1 seja digitado pelo usuário.



1. Repetições com Teste no Início

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    float numero, soma;

    soma = 0;

    printf("Escreva um novo valor:");
    scanf("%f", &numero);

    while(numero != -1) {
        soma = soma + numero;
        printf("Escreva um novo valor:");
        scanf("%f", &numero);
    }
```

1. Repetições com Teste no Início

```
printf("O somatorio vale: %f \n", soma);  
  
return 0;  
}
```

1. Repetições com Teste no Início

Exemplo 3: Escreva um programa que solicita ao usuário N notas, calcula e imprime na tela do computador a média da turma. O programa deve continuar solicitando notas até que o valor -1 seja digitado pelo usuário.

1. Repetições com Teste no Início

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    float numero, soma = 0, media = 0;
    int N = 0;

    printf("Escreva uma nota:");
    scanf("%f", &numero);

    while(numero != -1) {
        N++;
        soma = soma + numero;
        printf("Escreva uma outra nota:");
        scanf("%f", &numero);
    }
```

1. Repetições com Teste no Início

```
media = soma/N;  
printf("A media eh: %f \n", media);  
  
return 0;  
}
```

1. Repetições com Teste no Início

Exemplo 4: O número 3025 possui a seguinte característica :

$$30 + 25 = 55$$

$$55^2 = 3025$$

Escreva um programa que pesquise e imprima todos os números de quatro dígitos que apresentam tal característica.



1. Repetições com Teste no Início

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    int N, N1, N2;
    float Soma2;

    N = 1000;
    while (N <= 9999) {

        N1 = N/100;
        N2 = N-(100*N1);

        Soma2 = pow(N1+N2, 2);
```



1. Repetições com Teste no Início

```
if (Soma2 == N) {  
    printf("N1: %d \n", N1);  
    printf("N2: %d \n", N2);  
    printf("Soma2: %.0f \n", Soma2);  
}  
  
    N++;  
}  
  
return 0;  
}
```

1. Repetições com Teste no Início

Exemplo 5: Achar o maior e o menor número de uma série de números positivos fornecidos pelo usuário via teclado. O programa deve solicitar valores até que o número -1 seja fornecido.



1. Repetições com Teste no Início

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, maior, menor;

    printf("Digite um numero:");
    scanf("%d", &N);
    maior = N;
    menor = N;
```



1. Repetições com Teste no Início

```
while(N != -1) {  
    if (N < menor) {  
        menor = N;  
    } else if (N > maior) {  
        maior = N;  
    }  
    printf("Digite outro numero:");  
    scanf("%d", &N);  
}  
printf("Maior: %d \n", maior);  
printf("Menor: %d \n", menor);  
  
return 0;  
}
```



2. Pausa em Repetições

- **stdlib.h**: esse cabeçalho define várias funções de propósito geral, incluindo gerenciamento dinâmico de memória, geração de números pseudo-aleatórios, comunicação com o ambiente, aritmética de inteiro, busca, ordenação e conversão.

- ✓ O nome "stdlib" vem de *standard library* (biblioteca padrão, em inglês).
- ✓ <http://www.cplusplus.com/reference/clibrary/cstdlib/>
- ✓ Exemplo: **system()**.



2. Pausa em Repetições

- **stdio.h**: esse cabeçalho define várias funções de entrada e saída de dados.
 - ✓ O nome "stdio" vem de C *standard input and output library*.
 - ✓ <http://www.cplusplus.com/reference/clibrary/cstdio/>
 - ✓ Exemplos: **scanf()** e **printf()**.

2. Pausa em Repetições

- **math.h:** esse cabeçalho define várias funções matemáticas.

- ✓ <http://www.cplusplus.com/reference/clibrary/cmath/>

- ✓ Exemplos: **pow()** e **sqrt()**.

2. Pausa em Repetições

“Ora, está longe de ser óbvio, de um ponto de vista lógico, haver justificativa no inferir enunciados universais de enunciados singulares, independentemente de quão numerosos sejam esses; com efeito, qualquer conclusão colhida desse modo sempre pode revelar-se falsa: independentemente de quantos casos de cisnes brancos possamos observar, isso não justifica a conclusão de que *todos* os cisnes são brancos.” Karl Popper

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

Tipo	bits	Precisão
int	32 bits	$-(2^{31})$ a $(2^{31} - 1)$ -2.147.483.648 a 2.147.483.647
char	8 bits	-128 a 127
float	32 bits	6 a 7 dígitos significativos
double	64 bits	15 a 16 dígitos significativos
unsigned int	32 bits	0 a $(2^{32} - 1)$ 0 a 4.294.967.295
unsigned char	8 bits	0 a 255
short int	16 bits	$-(2^{15})$ a $(2^{15} - 1)$ -32.768 a 32.767
unsigned short int	16 bits	0 a $(2^{16} - 1)$ 0 a 65.535

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

Códigos de formatação para o printf()	Significado
%c	Caractere simples
%d ou %i	Inteiro decimal com sinal
%e ou E	Notação científica ("e" minúsculo ou "E" maiúsculo)
%f	Ponto flutuante
%g	O mais curto entre: %e ou %f
%G	O mais curto entre: %E ou %f
%o	Inteiro octal sem sinal
%s	String de caracteres
%u	Inteiro sem sinal
%x ou %X	Inteiro hexadecimal sem sinal (minúsc. ou maiúsc.)
%%	Imprime o caractere "%"

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

Tipo	Códigos de formatação para o printf()
int	%d ou %i
char	%c
float	%f
double	%lf
unsigned int	%u
unsigned char	%c
short int	%hi
unsigned short int	%hu

- Em geral:
 - %ld, %li, %lo, %lu, %lx e %lf → **long**.
 - %hd, hi, ho, hu, hx → **short**.
 - %Le, %Lf e %Lg → **long** double (não funciona no DevC++).

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

```
#include <stdio.h>
#include <stdlib.h>

int main() {

    int          x1 = -2147483648;
    char         x2 = 'A';
    float        x3 = 3.141592;
    double       x4 = 3.141592653589793;
    unsigned int x5 = 4294967295;
    unsigned char x6 = 255;
    unsigned short int x7 = 65535;

    printf("x1: %d \n", x1);    // int
    printf("x2: %c \n", x2);    // char
    printf("x3: %f \n", x3);    // float
    printf("x4: %.15lf \n", x4); // double
    printf("x5: %u \n", x5);    // unsigned int
    printf("x6: %u \n", x6);    // unsigned char
    printf("x7: %hu \n", x7);   // unsigned short int

    return 0;
}
```

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

```
#include <stdio.h>
#include <stdlib.h>

int main() {

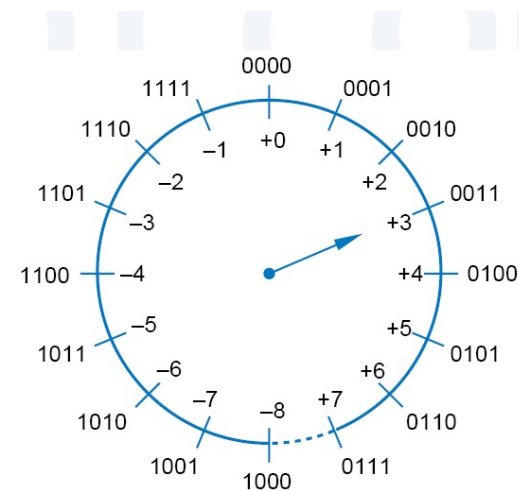
    printf("int: %d Bytes \n", sizeof(int));
    printf("char: %d Bytes \n", sizeof(char));
    printf("float: %d Bytes \n", sizeof(float));
    printf("double: %d Bytes \n", sizeof(double));
    printf("short int: %d Bytes \n", sizeof(short int));
    printf("unsigned int: %d Bytes \n", sizeof(unsigned int));
    printf("unsigned char: %d Bytes \n", sizeof(unsigned char));
    printf("unsigned short int: %d Bytes\n", sizeof(unsigned short int));

    return 0;
}
```

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação de números inteiros em complemento de 2:

Decimal	Two's Complement	Signed Magnitude
-8	1000	—
-7	1001	1111
-6	1010	1110
-5	1011	1101
-4	1100	1100
-3	1101	1011
-2	1110	1010
-1	1111	1001
0	0000	1000 or 0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111



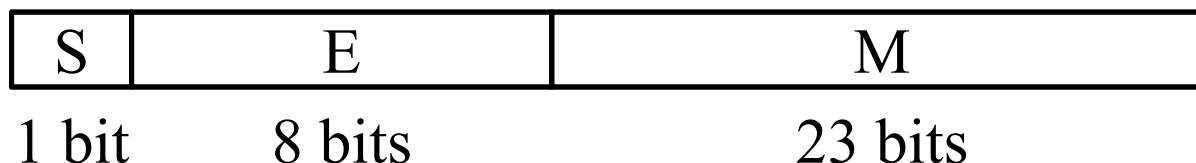
$$\begin{array}{rcl}
 119_{10} & = & 01110111 \\
 & & \Downarrow \text{complement bits} \\
 & & 10001000 \\
 & & +1 \\
 \hline
 & & 10001001^2 = -119_{10}
 \end{array}$$

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação por ponto flutuante (precisão simples → 32 bits): o padrão IEEE 754.

$$Y = (-1)^s(1+M) 2^{E-127}$$

$$1 \leq E \leq 254$$



- Considera-se que a representação utiliza a notação científica normalizada, ou seja, com exatamente um dígito diferente de zero antes do ponto binário.

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação por ponto flutuante (precisão simples → 32 bits): o padrão IEEE 754.
- Exemplo 1:

S	E	M
0	1 0 0 0 0 0 0 1	0 1 0 0 0
	$2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0$	$2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-23}$

$$S = 0, E = 129, M = 0,25$$

$$Y = (-1)^s(1+M)2^{E-127} = (-1)^0(1+0,25)2^{129-127}$$

$$= 1,25 * 2^2$$

$$= 5$$

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação por ponto flutuante (precisão simples → 32 bits): o padrão IEEE 754.
- Exemplo 2:

$$Y = 0,75$$

$$Y = \frac{1}{2} + \frac{1}{4} = \frac{3}{4} = 3 \cdot 2^{-2} = 11 \cdot 2^{-2} = 1,1 \cdot 2^{-1}$$

- Logo:

S	E	M
0	0 1 1 1 1 1 1 0	1 0 0 0 0
	$2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$	$2^{-1} \ 2^{-2} \ 2^{-3} \ 2^{-4} \ . \ . \ . \ . \ 2^{-23}$

$$E = e + 127 = 126_{10}$$

3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação por ponto flutuante (precisão simples → 32 bits): o padrão IEEE 754.
- Exemplo 3:

$$\begin{aligned}
 Y &= -2,625 \\
 Y &= -(2 + 1/2 + 1/8) = -21/8 = -21 \cdot 2^{-3} \\
 &= -10101 \cdot 2^{-3} = -1,0101 \cdot 2^1
 \end{aligned}$$

- Logo:

S	E	M
1	1 0 0 0 0 0 0 0	0 1 0 1 0
	$2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$	$2^{-1} \ 2^{-2} \ 2^{-3} \ 2^{-4} \ . \ . \ . \ . \ 2^{-23}$

$$E = 1 + 127 = 128_{10}$$

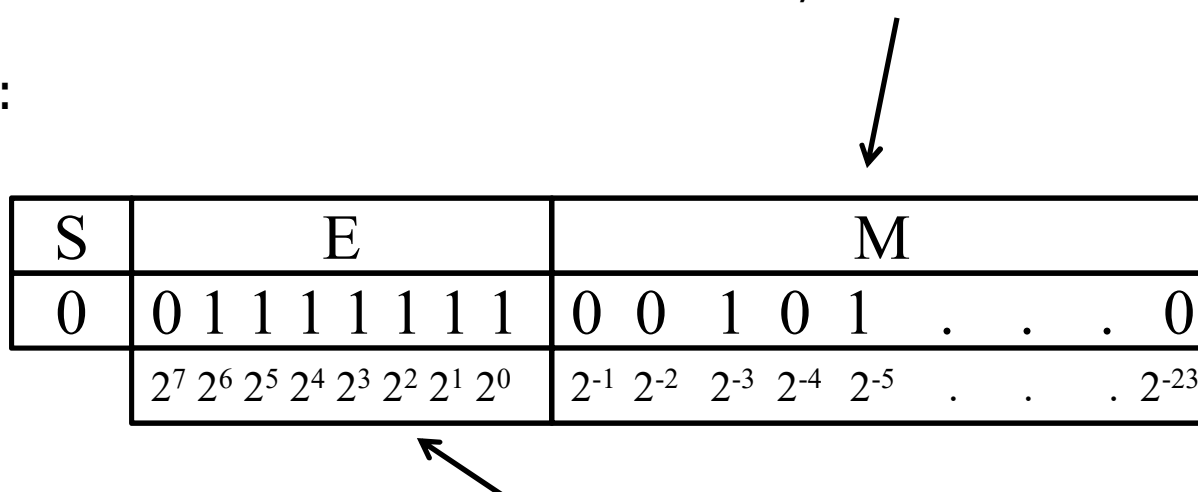
3. Mais sobre Tipos de Variáveis e Códigos de Formatação

- Representação por ponto flutuante (precisão simples → 32 bits): o padrão IEEE 754.
- Exemplo 4:

$$Y = 37/32$$

$$Y = 37 \cdot 2^{-5} = 100101 \cdot 2^{-5} = 1,00101 \cdot 2^0$$

- Logo:



S	E	M
0	0 1 1 1 1 1 1 1	0 0 1 0 1 . . . 0
	$2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$	$2^{-1} \ 2^{-2} \ 2^{-3} \ 2^{-4} \ 2^{-5} \ . \ . \ . \ 2^{-23}$

$$E = 0 + 127 = 127_{10}$$



4. Um Pouco mais Sobre Operadores

- O operador condicional ternário:
`<exp1> ? <exp2> : <exp3>`

```
#include <stdio.h>
#include <stdlib.h>

int main( )
{
    int a=5, b=2, maior;

    if (a > b)
        maior = a;
    else
        maior = b;
    printf("Maior: %d \n", maior);

    return 0;
}
```



4. Um Pouco mais Sobre Operadores

- O operador condicional ternário:
`<exp1> ? <exp2> : <exp3>`

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main( )
```

```
{
```

```
int a=5, b=2, maior;
```

```
if (a > b) →
```

```
    maior = a;
```

Onde foram parar as chaves?

```
else →
```

```
    maior = b;
```

```
printf("Maior: %d \n", maior);
```

```
return 0;
```

```
}
```



4. Um Pouco mais Sobre Operadores

- O operador condicional ternário:
`<exp1> ? <exp2> : <exp3>`

```
#include <stdio.h>
#include <stdlib.h>

int main( )
{
    int a=5, b=2, maior;

    maior = a>b ? a : b;

    return 0;
}
```

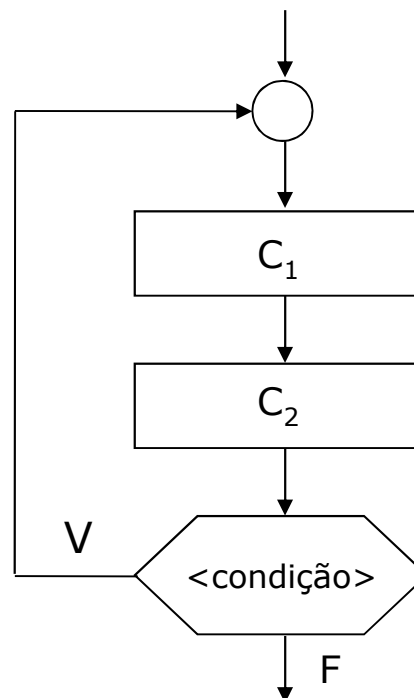
5. Repetições com Teste no Final

- Forma geral em Portugol:

```
faca  
    <lista de comandos>  
enquanto <expressão>
```


5. Repetições com Teste no Final

- Fluxograma:



5. Repetições com Teste no Final

Exemplo 1: Escreva um algoritmo que solicita ao usuário valores inteiros positivos N e conta a quantidade de número pares e a quantidade de números ímpares digitados. O usuário deve continuar fornecendo novos valores até que o -1 seja digitado. O algoritmo deve mostrar ao final quantos números pares e quantos números ímpares foram digitados.

5. Repetições com Teste no Final

algoritmo "facaenquanto"

var

inteiro: Npares, Nimpares, Numero;

inicio

Npares = 0;

Nimpares = 0;



5. Repetições com Teste no Final

```
faca
    escreva("Digite um numero: ");
    leia(Numero);
    se Numero > 0 entao
        se Numero%2 == 0 entao
            Npares = Npares +1;
        senao
            Nimpares = Nimpares +1;
        fimse
    fimse
enquanto numero != -1;
```

5. Repetições com Teste no Final

```
escreva("Numero de pares:", Npares);  
escreva("Numero de ímpares:", Nimpares);
```

```
fimalgoritmo
```



5. Repetições com Teste no Final

- Forma geral em C:

do{

<comandos>

} while (<condição>);

5. Repetições com Teste no Final

```
#include <stdio.h>
#include <stdlib.h>

int main( )
{
int Npares = 0, Nimpares = 0, Numero;
```



5. Repetições com Teste no Final

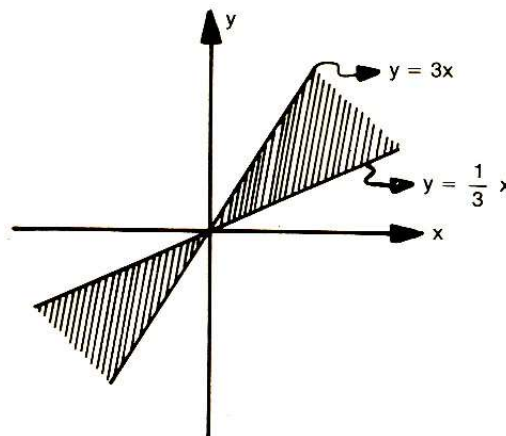
```
do{  
    printf("Digite um numero: ");  
    scanf("%d", &Numero);  
    if (Numero > 0)  
    {  
        if (Numero%2 == 0)  
            Npares++;  
        else  
            Nimpares++;  
    }  
} while (Numero != -1);
```


5. Repetições com Teste no Final

```
printf("Numero de pares: %d \n", Npares);  
printf("Numero de ímpares: %d \n", Nimpares);  
  
return 0;  
}
```

5. Repetições com Teste no Final

Exemplo 2: Escreva um algoritmo que repetidamente leia coordenadas (x,y) fornecidas pelo usuário e escreva na tela do computador "INTERIOR" ou "EXTERIOR", caso o ponto pertença ou não à região sombreada abaixo, respectivamente. A solicitação das coordenadas deve prosseguir até que o usuário solicite a interrupção. Utilize `faca...enquanto` (`do...while`) na implementação do laço.



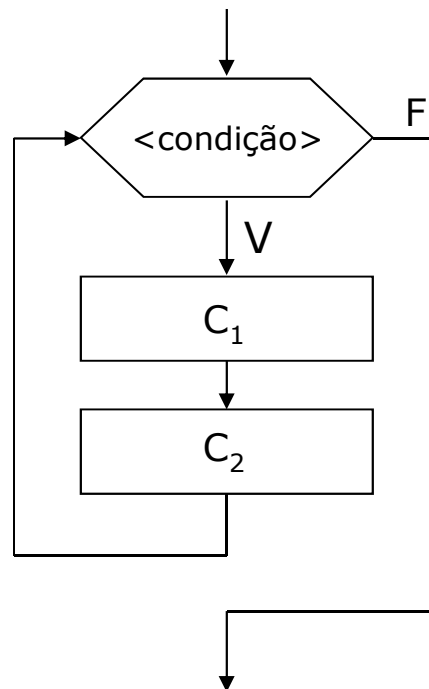
6. Repetições com Variável de Controle

- Forma geral em Portugol:

```
para <variável de controle> de <valor inicial> ate <valor final> [passo  
<incremento>] faca  
    <lista de comandos>  
fimpara
```

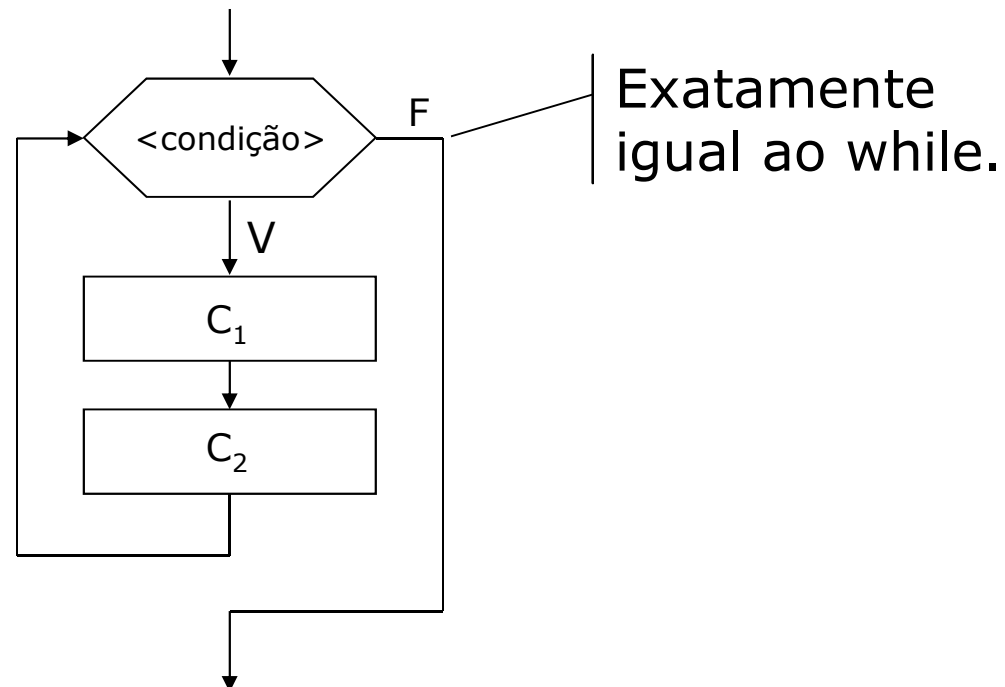
6. Repetições com Variável de Controle

- Fluxograma:



6. Repetições com Variável de Controle

- Fluxograma:



6. Repetições com Variável de Controle

Exemplo 1: Escreva um algoritmo que solicita ao usuário um valor inteiro positivo N e imprime na tela do computador todos os número inteiros de 0 a N .



6. Repetições com Variável de Controle

algoritmo "mostraN"

var

inteiro: N, i;

inicio

escreva ("Entre com o número N:");

leia (N);

para i de 0 ate N passo 1 faca


escreva (i);

fimpara

fimalgoritmo

6. Repetições com Variável de Controle

- Forma geral em C:

```
for(<inicialização>; <condição>; <incremento>) {  
      
}
```


6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, i;

    printf("Digite o valor de N:");
    scanf("%d", &N);

    for(i = 0; i<=N; i++) {
        printf("%d\n", i);
    }

    return 0;
}
```



6. Repetições com Variável de Controle

```
for(i = 0; i<=N; i++) {  
    printf("%d\n", i);  
}
```

```
i = 0;  
while(i<=N) {  
    printf("%d\n", i);  
    i++;  
}
```



6. Repetições com Variável de Controle

- **time.h**

Exemplo 2: Complemente o exemplo anterior, mostrando na tela do computador o tempo total de execução do algoritmo.

6. Repetições com Variável de Controle

- **time.h**

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main()
{
    int N, i;
    double tempo;
    time t inicio, fim;
```

6. Repetições com Variável de Controle

- **time.h**

```
printf("Digite o valor de N:");  
scanf("%d", &N);  
  
time(&inicio);  
for(i = 0; i<=N; i++) {  
    printf("%d\n", i);  
}  
time(&fim);  
tempo = difftime(fim, inicio);  
  
printf("Tempo total: %.11f segundos", tempo);  
  
return 0;  
}
```

6. Repetições com Variável de Controle

Exemplo 3: Escreva um algoritmo que solicita ao usuário um valor inteiro positivo N e imprime na tela do computador todos os número inteiros de N a 0 .

6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, i;

    printf("Digite o valor de N:");
    scanf("%d", &N);

    for(i = N; i >= 0; i--) {
        printf("%d\n", i);
    }

    return 0;
}
```

6. Repetições com Variável de Controle

Exemplo 4: Escreva um algoritmo que solicita ao usuário um número inteiro positivo N , e mostra na tela do computador todos os números pares entre 1 e N .

6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, i;

    printf("Digite o valor de N:");
    scanf("%d", &N);

    for(i = 0; i<=N ; i++){
        if(i % 2 == 0) printf("%d\n", i);
    }
    return 0;
}
```

6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N, i;

    printf("Digite o valor de N:");
    scanf("%d", &N);

    for(i = 0; i<=N ; i+=2) {
        printf("%d\n", i);
    }

    return 0;
}
```

6. Repetições com Variável de Controle

Exemplo 5: Escreva um algoritmo que mostre a tabuada de 2 a 10 na tela do computador.

6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i, j;

    for(i = 2; i<=10 ; i++) {
        printf("*****\n");
        printf("Tabuada de %d\n", i);
        printf("*****\n\n");

        for(j = 1; j<=10 ; j++) {
            printf("%d x %d = %d \n", i, j, i*j);
        }
        printf("\n");
    }
    return 0;
}
```

6. Repetições com Variável de Controle

Exemplo 6: Calcule o $\cos(x)$ por meio de 5 termos da série abaixo e escreva a diferença entre o valor calculado por essa série e o valor dado pela função $\cos(x)$ de `<math.h>`.

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \dots$$

6. Repetições com Variável de Controle

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    float angulo, angrad, valcos, valserie, PI=3.14159265;
    float sinal, fatorial, termo;
    int i, j, Ntermos, Den;

    // Solicita o angulo em graus
    printf("Entre com o valor de um angulo:");
    scanf("%f", &angulo);

    // Converte para radianos
    angrad = angulo*PI/180;
```



6. Repetições com Variável de Controle

```
// Calcula o cosseno utilizando a função cos de math.h
valcos = cos(angrad);

/* Calcula o valor máximo do denominador,
dada a quantidade de termos a serem adicionados */
Ntermos = 5;
Den = (Ntermos*2)-2;

// Primeiro termo da serie
valserie = 1;

/* Dado que a cada interação o sinal
do termo muda faz-se necessário declarar
uma variável para controlar isso */
sinal = 1;
```



6. Repetições com Variável de Controle

```
// Inicializa o valor do fatorial com 1
fatorial = 1;

// Calcula um novo termo a cada valor de denominador
for(i = 2; i<=Den ; i+=2) {

    /* O calculo do novo termo depende de um fatorial.
    O fatorial do novo termo é calculado a partir do
    valor do fatorial anteriormente calculado.
    Isso aumenta a eficiente do programa
    */

    for (j = i-1; j<= i; j++){
        fatorial = fatorial*j;
    }
```




6. Repetições com Variável de Controle

```
// Inverte o sinal
sinal = -sinal;

// Calcula o novo termo
termo = sinal*pow(angrad,i)/fatorial;

// Acrescenta o novo termo ao somatório
valserie = valserie + termo;
}

printf("\nFuncao cos(x): %f \n", valcos);
printf("Calculado      : %f \n\n", valserie);

return 0;
}
```

7. Repetições com Variável de Controle e Teste no Início

Exemplo 7: Calcule o valor aproximado de $\cos(x)$ por meio da série abaixo e com uma precisão menor que 0.001. A precisão é aqui definida como o valor absoluto da diferença entre duas aproximações consecutivas de $\cos(x)$. Considere também que o cálculo do fatorial em uma iteração deve ser realizado a partir do fatorial calculado na iteração anterior. O valor do ângulo x deve ser fornecido em graus, mas o valor de $\cos(x)$ deve ser realizado para x em radianos. Inclua instruções para realizar a conversão.

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \dots$$

7. Repetições com Variável de Controle e Teste no Início

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    float angulo, angrad, valcos, valserie;
    float PI=3.14159265;
    float sinal, fatorial, termo, erro, precisao = 0.0001;
    int i, j;

    // Solicita o angulo em graus
    printf("Entre com o valor de um angulo:");
    scanf("%f", &angulo);
```

7. Repetições com Variável de Controle e Teste no Início

```
// Converte para radianos
angrad = angulo*PI/180;
printf("Angulo em rad: %f \n", angrad);

// Calcula o cosseno utilizando a função cos de math.h
valcos = cos(angrad);

// Primeiro termo da serie
valserie = 1;

/* Dado que a cada interação o sinal
do termo muda faz-se necessário declarar
uma variável para controlar isso */
sinal = 1;
```

7. Repetições com Variável de Controle e Teste no Início

```
// Inicializa o valor do fatorial com 1
fatorial = 1;

// Calcula um novo termo a cada valor de denominador
i = 2;

// Inicializa termo com o valor para entrar no laço
termo = 1;
while( fabs(termo) >= precisao) {

    /* O cálculo do novo termo depende de um fatorial.
    O fatorial do novo termo é calculado a partir do
    valor do fatorial anteriormente calculado.
    Isso aumenta a eficiência do programa. */

    for (j = i-1; j<= i; j++) fatorial = fatorial*j;
```

7. Repetições com Variável de Controle e Teste no Início

```
// Inverte o sinal
sinal = -sinal;

// Calcula o novo termo
termo = sinal*pow(angrad,i)/fatorial;
printf("termo: %f \n", termo);

// Acrescenta o novo termo ao somatório
valserie = valserie + termo;
printf("serie: %f \n", valserie);

/* Incrementa i para calculo do denominador
   e potência de x */
i +=2;

}
```

7. Repetições com Variável de Controle e Teste no Início

```
// Mostra os resultados na tela do computador
printf("\nFuncao cos(x)      : %f \n", valcos);
printf("Calculado pela serie : %f \n\n", valserie);

return 0;

}
```

“No momento, a Física está mais uma vez em terrível confusão. De qualquer modo, para mim é muito difícil. Gostaria de ter-me tornado um comediante de cinema ou algo do gênero e nunca ter ouvido falar de Física.”

Wolfgang Pauli, nos meses que precederam o artigo de Heisenberg que indicaria o caminho para uma nova teoria dos quanta.

“O tipo de mecânica proposta por Heisenberg devolveu-me a esperança e a alegria de viver. Sem dúvida alguma, ela não proporciona a solução para a charada, mas acredito que agora é possível avançar novamente.”

Wolfgang Pauli, cinco meses depois.