

Preparatório para Programação



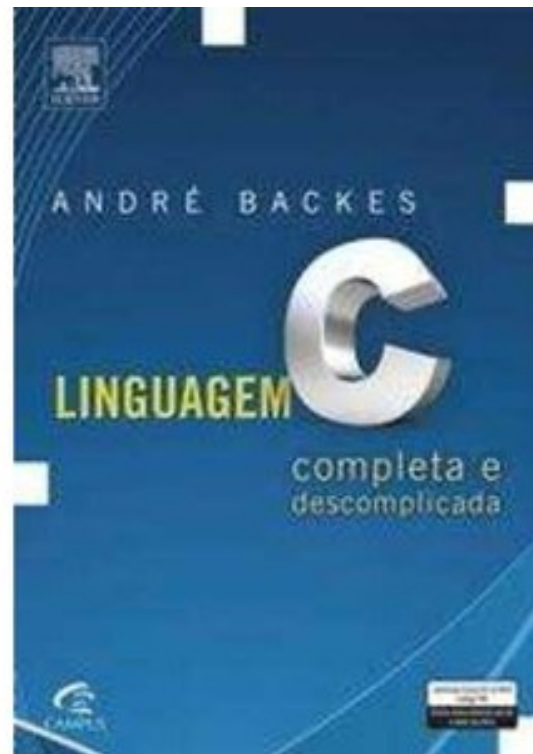
Aula 01 – Revisão da linguagem C e seus principais conceitos

Professor Murilo Zanini de Carvalho



Bibliografia Recomendada

- BACKES, André. **“Linguagem C: Completa e Descomplicada”**. Ed. Campus Elsevier, 2012



Bibliografia Recomendada

- Treinamento em Linguagem C – MIZHARI, Victorine Viviane – Ed. Pearson Prentice Hall



Bibliografia Recomendada

- Use a Cabeça! Programação – BARRY, Paul & GRIFFITHS, David – Editora Alta Books



Bibliografia Recomendada

- GRIFFITHS, David; GRIFFITHS, Dawn. **“Use a cabeça! C”**. Alta Books, 2013

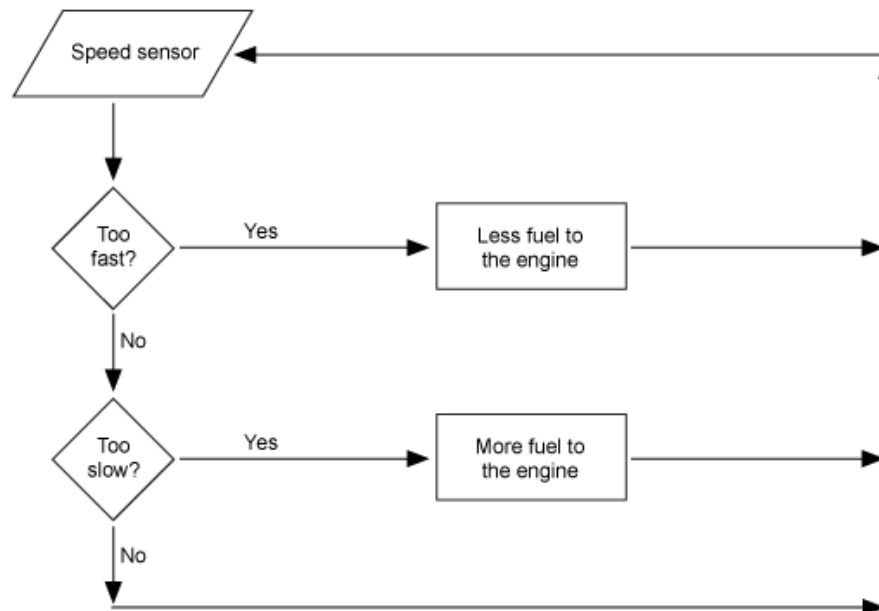


Algoritmos






- “(algoritmo) conjunto de regras necessárias para resolução de um problema ou cálculo; processo computacional bem definido, baseado num conjunto de regras, finito, que executa uma determinada tarefa” - <http://pt.wiktionary.org/wiki/algoritmo>
- “Algoritmo. Em matemática, este termo significa um procedimento padronizado para solução de determinada categoria de problema.” - http://www.escolanet.com.br/dicionario/dicionario_a.html

Fluxogramas

- Forma gráfica de representar um algoritmo.
- Utiliza símbolos padronizados para expressar uma sequência lógica de ações.

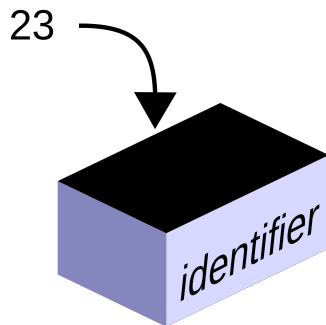


Fluxogramas

Symbol	Name	Function
	Start/end	An oval represents a start or end point.
	Arrows	A line is a connector that shows relationships between the representative shapes.
	Input/Output	A parallelogram represents input or output.
	Process	A rectangle represents a process.
	Decision	A diamond indicates a decision.

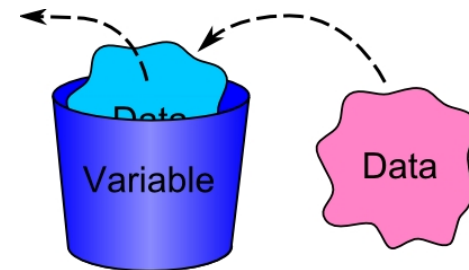
Variáveis

- Variáveis são espaços na memória utilizados para armazenar e manipular valores.
- Variáveis são utilizadas por seus identificadores, nomes.



Retirado de

(<https://upload.wikimedia.org/wikipedia/commons/thumb/8/80/CPT-programming-variable.svg/2000px-CPT-programming-variable.svg.png>), em 03.02.2016



Retirado de (<http://img.c4learn.com/2012/02/Variable-in-Java.jpg>), em 03.02.2016

Regras para nomes de variáveis

- Em linguagem C, os nomes de variáveis devem seguir algumas regras:
 - Sempre devem ser iniciados por uma letra ou o caractere '_';
 - Podem possuir números em seu nome, mas não começar por eles;
 - Não podem ser caracteres especiais, como acentos, espaços em branco, ou 'ç';
 - Não podem ser letras gregas;
 - Não podem ser palavras reservadas da linguagem.



Palavras Reservadas

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Retirado de (https://encrypted-tbn2.gstatic.com/images?q=tbn:ANd9GcQ9nXWV0U_RIkXu-eg4zIYU-PD76FEZOVoxsq4zYQEHDi_LQ3hd0), 03.02.2016

Exemplo

- Desenvolva o algoritmo para somar dois números.
- Desenvolva o algoritmo para converter um valor em metros, fornecido pelo usuário, para milímetros.
- Desenvolva um algoritmo que, dados os valores de tensão e corrente, calcule o valor da resistência de um resistor.
- Implemente um algoritmo que resolva uma equação de 1º grau: $Ax+B$.

Ambiente de Desenvolvimento

1

2

3

4

5

6

7

8

9

Fim

Nome do projeto e pasta

Seleção C

Retirado do material do Prof. Marco Furlan (Mauá, 2014)

Primeiro Programa em C

Programa em Linguagem C

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

int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Programa Compilado

Hello, World!

Compilador

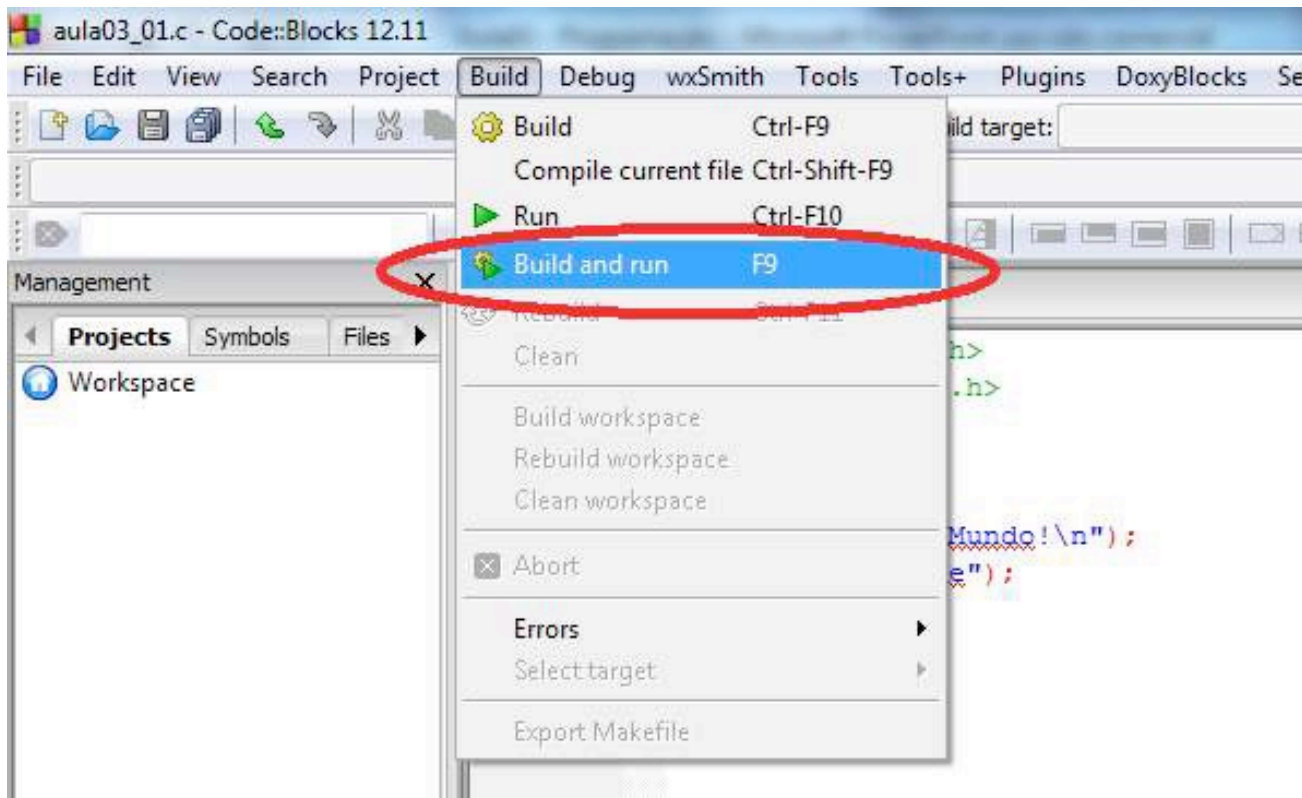


Primeiro Programa em C

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      printf("Ola Mundo!\n");
7      return 0;
8  }
```

Executar um Programa

- Para construir e executar seus projetos, selecionar a opção “**Build and Run**”, dentro do menu **Build**.



RELEMBRAR É VIVER:

Regras para Nomes de Programas

- As regras para criação de nomes de arquivos são:
 - NUNCA utilizar acentos;
 - Nomes sempre devem iniciar por letras;
 - Nenhum caractere especial, como vírgula ou espaço, pode ser utilizado, com exceção de _ (*underline*);
 - Nomes podem conter números, mas não começar com eles.

Programa Exemplo

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

```
3
4  int main()
```

```
5  {
```

```
6      int A,B,x,y;
```

Região de declaração
de variáveis

```
7
8      printf("Informe o valor de A:");
```

```
9      scanf("%i", &A);
```

Leitura do valor
da variável

```
10
11     printf("Informe o valor de B:");
```

```
12     scanf("%i", &B);
```

```
13
14     printf("Informe o valor de x:");
```

```
15     scanf("%i", &x);
```

```
16
17     y = A*x + B;
```

Processamento da lógica
do programa

```
18
19     printf("O Resultado: %i\n", y);
```

```
20     return 0;
```

Exibição do Resultado

```
21 }
```

RELEMBRAR É VIVER:

Regras para Nomes de Variáveis

- As regras para criação de nomes de variáveis e arquivos são iguais:
 - NUNCA utilizar acentos;
 - Nomes sempre devem iniciar por letras;
 - Nenhum caractere especial, como vírgula ou espaço, pode ser utilizado, com exceção de _ (*underline*);
 - Nomes podem conter números, mas não começar com eles.

Tipos de Variáveis

Tipo da Variável	Descrição	Leitura (scanf)	Escrita na tela (printf)
int	Números inteiros	scanf("%i", &nome);	printf("%i", nome);
float	Números reais	scanf("%f", &nome);	printf("%f", nome);
char	Letras	scanf("%c", &nome);	printf("%c", nome);

Observação: Existem outros tipos de variáveis, elas serão apresentadas ao longo do curso. Para maiores referências, consultar: <http://www.cplusplus.com/doc/tutorial/variables/>

Operadores Básicos Linguagem C

Operador	Descrição	Exemplo
Atribuição: =	Atribui o valor da direita à seu destino a esquerda	A = 10;
Adição: +	Realiza a soma de dois valores	C = A + B;
Subtração: -	Realiza a subtração entre dois valores	G = Y - 5;
Multiplicação: *	Realiza o produto entre dois valores	G = 7 * A;
Divisão: /	Realiza a divisão do valor da esquerda com o valor da direita. Entre números inteiros, retorna parte inteira.	H = G / 4;
Módulo (Resto da Divisão): %	Retorna o resto da divisão entre 2 números inteiros.	H = Y % 2;

Operadores Básicos Linguagem C

- Construa um programa que permita ao usuário calcular a média entre dois números.

```
1  #include <stdlib.h>
2  #include <stdio.h>
3
4  int main() {
5      int n1,n2;
6      float media;
7      printf("Informe 2 valores inteiros:");
8      scanf("%i%i",&n1, &n2);
9      media = (n1+n2)/2.0;
10     printf("Valor media: %f\n",media);
11     return 0;
12 }
```

Exemplo Problema

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int x;
    printf("Informe um número: ");
    scanf("%i", &x);
    printf("Valor lido: %i", x);
    return 0;
}
```

Função printf()

- Utilizada para escrever mensagens na saída padrão (monitor, no caso dos computadores).
- Permite utilizar texto formatado com variáveis.
- Para adicionar uma variável no texto formatado, utilizar o marcador %.

Marcador	Descrição
%i	Valores do tipo inteiro
%f	Valores do tipo real
%c	Apenas um caractere
%s	Para palavras

Função scanf()

- Utilizada para ler mensagens na entrada padrão (teclado, no caso dos computadores).
- Informar o tipo do valor que deve ser lido do teclado com o marcador % e informar o endereço da variável que deve armazenar esse valor com &

Marcador	Descrição
%i	Valores do tipo inteiro
%f	Valores do tipo real
%c	Apenas um caractere
%s	Para palavras

RELEMBRAR É VIVER:

Variáveis

- Cada variável deve possuir um nome e um tipo.
- Os nomes de variáveis são únicos no programa, não podem ser repetidos e devem respeitar as regras para os nomes das variáveis.

Tipo	Descrição
int	Números inteiros
float	Números reais
char	Caracteres

Exemplo – Problema I

- Elabore um programa que receba um valor em polegadas e faça sua conversão para milímetros. Lembrando que uma polegada vale 25.4 milímetros.

Exemplo Problema

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    float pol, mm;
    printf("Informe o valor em polegadas: ");
    scanf("%f", &pol);
    mm = pol * 25.4;
    printf("Valor em mm: %f", mm);
    return 0;
}
```

Exemplo – Problema II

- Elabore um programa que receba duas frações, numerador e denominador de cada uma delas, realize a soma delas e exiba seu resultado, também no formato de fração (numerador e denominador).

Exemplo Problema

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int a,b,c,d,e,f;
    printf("A: ");
    scanf("%i", &a);
    printf("B: ");
    scanf("%i", &b);
    printf("C: ");
    scanf("%i", &c);
```

```
    printf("D: ");
    scanf("%i", &d);
    f = d*b;
    e = (a*f/b)+(b*f/d);
    printf("%i / %i",e,f);
    return 0;
}
```

Estruturas de Decisão

➤ As estruturas padrões na linguagem C são:

```
if(testeLogico)
```

Decisão Simples

```
if(testeLogico) {  
}  
else {  
}
```

Decisão
Composta

```
switch(variavel)  
{  
    case y:  
        break  
}
```

Decisão Múltipla

Estruturas de Decisão

- Em C, qualquer valor diferente de 0 é considerado verdadeiro.

```
int x, y, z;  
x = 0;  
y = 10;  
z = -4;  
if(x)  
    printf("Esse Teste foi Verdadeiro - X!\n");  
if(y)  
    printf("Esse Teste foi Verdadeiro - Y!\n");  
if(z)  
    printf("Esse Teste foi Verdadeiro - Z!\n");
```


Estruturas de Decisão em C

IF

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int idade;
    printf("Informe sua idade: ");
    scanf("%i", &idade);
    if (idade >= 18)
        printf("Idade maior de 18 anos!");
    printf("Idade: %i", idade);
    return 0;
}
```

Operadores Relacionais

Operador	Função	Exemplo
==	Compara se dois valores são iguais	A == 5
!=	Compara se dois valores são diferentes	B != 7.8
>	Compara se o valor da esquerda é maior que o valor da direita	C > 2
>=	Compara se o valor da esquerda é maior ou igual que o valor da direita	D >= 5
<	Compara se o valor da esquerda é menor que o valor da direita	E < 9
<=	Compara se o valor da esquerda é menor ou igual que o valor da direita	F <= 10
&&	Operador lógico “E”, testa se as condições testadas são verdadeiras	(F>=5) && (A==9)
	Operador lógico “OU”	(F>=5) (A==9)
!	Operador “NÃO”	!(G<135)

Estruturas de Decisão em C

IF-ELSE

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int idade;
    printf("Informe sua idade: ");
    scanf("%i", &idade);
    if (idade >= 18)
        printf("Idade maior de 18 anos!\n");
    else
        printf("Vai estudar!\n");
    printf("Idade: %i", idade);
    return 0;
}
```

Caracteres Especiais printf()

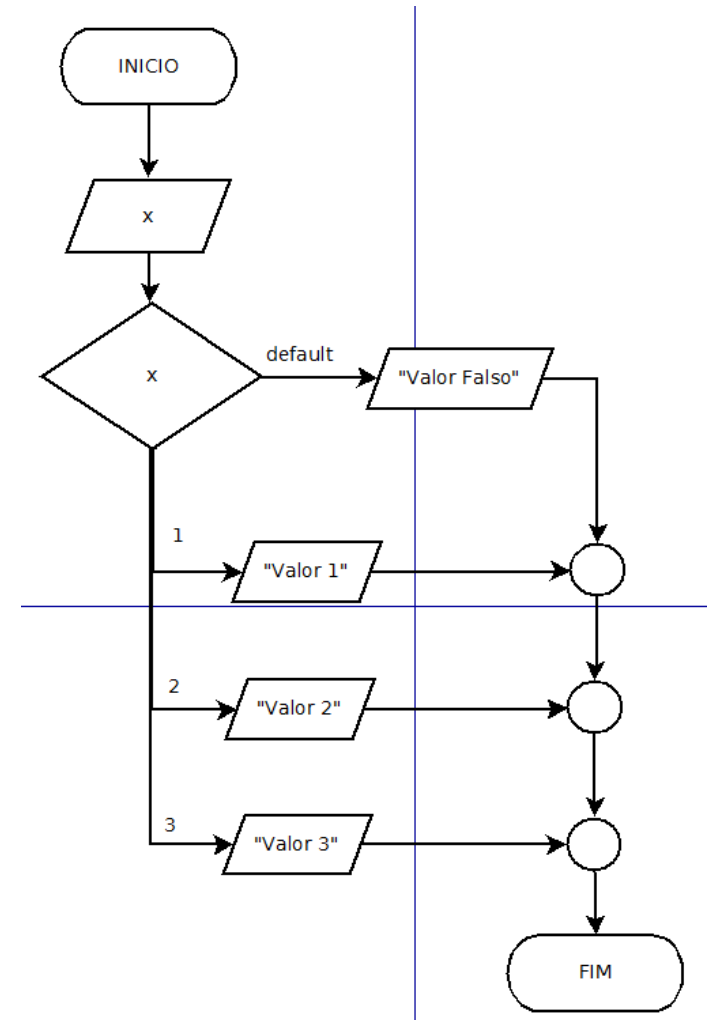
- A função printf() possui um conjunto de caracteres especiais para formatação do conteúdo apresentado.
- Deve estar dentro do texto exibido.

Caracter	Descrição
\n	Imprime uma nova linha
\t	Imprime uma tabulação do texto
\r	Volta o cursor para o início da linha
\a	Ativa o buzzer do sistema

Estruturas de Decisão em C

SWITCH

- Realiza diversos testes na mesma variável, procurando algum valor igual aos dos valores de referência.
- ATENÇÃO: apenas podem ser utilizados valores inteiros ou caracteres.



Estruturas de Decisão em C

SWITCH

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int x;
    printf("Informe x:");
    scanf("%i", &x);
    switch(x)
    {
        case 1:
            printf("Valor 1\n");
            break;
```

```
        case 2:
            printf("Valor 2\n");
            break;

        case 3:
            printf("Valor 3\n");
            break;

        default:
            printf("Valor Falso\n");
    }
    return 0;
```

```
}
```

Operador Ternário

- Realiza um teste em uma expressão, devolvendo um valor caso o resultado do teste for verdadeiro e outro caso o resultado for falso.

```
int x, y, z;  
x = 0;  
y = 10;  
z = -4;  
  
z = x > y ? 5 : -5;  
printf("Valor de Z:%i\n", z);  
  
z = z < 0 ? 10 : -5;  
printf("Valor de Z:%i\n", z);
```

Estruturas de Decisão em C

SWITCH – RELEMBRAR É VIVER

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int x;
    printf("Informe x:");
    scanf("%i", &x);
    switch(x)
    {
        case 1:
            printf("Valor 1\n");
            break;
```

```
        case 2:
            printf("Valor 2\n");
            break;

        case 3:
            printf("Valor 3\n");
            break;

        default:
            printf("Valor Falso\n");
    }
    return 0;
```

```
}
```


Operador Ternário

RELEMBRAR É VIVER

- Realiza um teste em uma expressão, devolvendo um valor caso o resultado do teste for verdadeiro e outro caso o resultado for falso.

```
int x, y, z;  
x = 0;  
y = 10;  
z = -4;  
  
z = x > y ? 5 : -5;  
printf("Valor de Z:%i\n", z);  
  
z = z < 0 ? 10 : -5;  
printf("Valor de Z:%i\n", z);
```

Exercícios - Operador Ternário

- Realize a leitura entre 2 valores, retorne o maior deles.

Estruturas de Repetição

- Utilizadas para repetir um trecho de código.
- Elas repetem enquanto a condição de teste for verdadeira.
- Existem 3 estruturas básicas de repetição em linguagem C:
 - `while(<condição>)`
 - `do { } while(<condição>);`
 - `for (<inicialização>; <teste>; <incremento>)`

RESUMO

- Estrutura WHILE – utilizada quando o número de repetições não é conhecido, apenas a causa que permite que a repetição continue.
- Estrutura FOR – utilizada quando o número de vezes que a repetição deve acontecer é conhecido.
- Estrutura DO-WHILE – utilizada no mesmo conjunto de condições que a estrutura WHILE, contudo, ele executa o código antes de testar a condição de repetição.

Estruturas de Repetição - while

```
#include<stdio.h>
#include <stdlib.h>
int main()
{
    int i, x;
    printf("Informe um valor para X:");
    scanf("%i", &x);
    i = 0;
    while(i < x)
    {
        printf("Valor digitado:%i\n", i);
        i++;
    }
    return 0;
}
```

Estruturas de Repetição – do/while

```
#include<stdio.h>
#include <stdlib.h>
int main()
{
    int i, x;
    printf("Informe um valor para X:");
    scanf("%i", &x);
    do
    {
        printf("Valor digitado:%i\n", i);
        i++;
    }
    while(i < x);
    return 0;
}
```

Estruturas de Repetição – for

```
#include<stdio.h>
#include <stdlib.h>
int main()
{
    int i;
    for(i = 0; i < 15; i++)
    {
        printf("Valor digitado:%i\n", i);
    }
    return 0;
}
```

Estruturas de Repetição – Comando *continue*

```
#include<stdio.h>
#include <stdlib.h>
int main()
{
    int i;
    for(i = 0; i < 15; i++)
    {
        if(i != 10)
            printf("Valor digitado:%i\n", i);
        else
            continue;
    }
    return 0;
}
```


Estruturas de Repetição – Comando *break*

```
#include<stdio.h>
#include <stdlib.h>
int main()
{
    int i;
    for(i = 0; i < 15; i++)
    {
        if(i != 10)
            printf("Valor digitado:%i\n", i);
        else
            break;
    }
    return 0;
}
```

Estruturas de Repetição

- A estrutura de repetição FOR é utilizada quando o número de repetições é conhecida pelo usuário.
- A estrutura automatiza a execução de um trecho do código.
- REFORÇANDO: quando o número de repetições for conhecido, utilizar a estrutura de repetição ***for***.

Estruturas de Repetição

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

int main()
{
    int i;
    for (i = 0; i < 3; i++)
        printf("Hello world!\n");
    return 0;
}
```

Estruturas de Repetição

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

```
int main()
```

```
{
```

Inicialização

Teste Lógico

Incremento (pós processamento)

```
    int i;
```

```
    for (i = 0; i < 3; i++)
```

```
        printf("Hello world!\n");
```

```
    return 0;
```

```
}
```

Programa para Somar 10 Valores

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

int main()
{
    int i;
    float soma = 0, n;
    for (i = 0; i < 10; i++){
        printf("Informe um valor:");
        scanf("%f", &n);
        soma += n;
    }
    printf("Soma dos 10 valores: %.2f\n", soma);
    return 0;
}
```

Programa que realiza a média entre 5 valores

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

int main()
{
    int i;
    float soma = 0, n, media;
    for (i = 0; i < 5; i++){
        printf("Informe um valor:");
        scanf("%f", &n);
        soma += n;
    }
    media = soma / 5;
    printf("Media dos 5 valores: %.2f\n", media);
    return 0;
}
```

Observações

- Em algumas situações, é necessário, ou vantajoso, omitir alguma clausula do cabeçalho do for.
- É possível utilizar o operador vírgula no for para utilizar dois parâmetros no mesmo ponto do cabeçalho, como duas inicializações, por exemplo.
- CUIDADO: mesmo em loops for, quando algo é emitido, a chance de um loop infinito acontecer também existe.

Exercícios

- Construir um programa que calcule, se possível, as raízes de uma equação de grau 2. Utilize o método de Bhaskara para resolver a equação.
- Construa um programa que determine se um número é par ou ímpar.
- Construa um programa que verifique se um número inteiro entre 1000 e 9999 é um palíndromo. Exemplos de palíndromos: 1221, 9999, 3443.

Exercícios - Estrutura SWITCH

- Uma empresa vende o mesmo produto para quatro diferentes estados. Cada estado possui uma taxa diferente de imposto sobre o produto. Faça um programa em que o usuário entre com o valor e o estado de destino (representado por um valor inteiro) e o programa retorne o preço final do produto acrescido do imposto do estado em que ele será vendido. Se o estado digitado não for válido, mostrará uma mensagem de erro.

Estado	MG	SP	RJ	MS
Imposto	7%	12%	15%	8%

Exercícios - Estrutura SWITCH

- Faça um programa que leia três números inteiros positivos e efetue o cálculo de uma das seguintes médias de acordo com um valor numérico digitado pelo usuário e mostrado na tabela a seguir:

Número Digitado	Média	Exemplo
1	Geométrica	$X * Y * Z$
2	Ponderada	$\frac{X + 2 * Y + 3 * Z}{6}$
3	Harmônica	$\frac{1}{\frac{1}{X} + \frac{1}{Y} + \frac{1}{Z}}$
4	Aritmética	$\frac{X + Y + Z}{3}$

Exercícios

- Escreva um programa que solicita ao usuário para digitar três números e então determina sua média, sua soma e seu produto.
- Escreva um programa que solicita ao usuário para digitar a nota de P1, a nota de P2 e a nota de trabalho, para então apresentar o valor da média. $Média = ((P1+P2)/2) * 0.6 + T * 0.4$
- Elabore um programa que permita decompor um vetor no plano XY. O usuário irá informar o módulo do vetor e sua inclinação em relação ao eixo X. Considerar que o valor do ângulo fornecido está no intervalo 0 e 90° .

Exercícios com Estruturas de Repetição

- Elabore um programa que leia números até que o usuário digite o valor 0. Imprima a soma de todos os números digitados (utilizar ***while***).
- Repetir o programa acima utilizando o comando ***do-while***.
- Implemente um programa que realize a leitura de um número N inteiro e imprima na tela todos os números de 0 até N, de forma crescente.

Exercícios com Estruturas de Repetição

- Elabore um programa que realize a leitura de um número N e imprima todos os números naturais ímpares até N .
- Elabore um programa que realize a leitura de um número N e imprima todos os números naturais pares até N .

Exercícios com Estruturas de Repetição

- Implemente um programa que implemente um jogo de adivinhações. O seu código deve permitir o usuário tentar 3 vezes acertar um número aleatório de 1 à 10. Utilizar a função:

```
srand(time(NULL));
```

```
res = rand()%11 + 1;
```

para determinar o valor aleatório.

Exercícios

- Faça um programa que leia um número inteiro positivo N e imprima todos os números naturais de 0 até N em ordem crescente.
- Faça um programa que leia um número inteiro N e depois imprima os N primeiros números naturais ímpares.
- Escreva um programa que leia 10 números e escreva o menor e o maior valor lido.

Referências

- SALVETTI, Dirceu Douglas; BARBOSA, Lisbete Madsen. “Algoritmos”, Ed Makron Books, 1998
- BACKES, André. “Linguagem C: completa e descomplicada”, Ed Elsevier, 2013
- BARRY, Paul; GRIFFITHS, David. “Use a Cabeça! Programação”. Editora Alta Books - 2010

Exercícios

- Faça um programa que exiba a soma de todos os números naturais abaixo de 1000 que são múltiplos de 3 ou 5.
- Escreva um programa que calcule o fatorial de um número.
- Escreva um programa que exiba o Nésimo valor da sequencia de Fibonacci fornecido pelo teclado.

Referências

- BOLTON, W.. “Mecatrônica – Uma abordagem multidisciplinar”, 4 ed., Ed Bookman, Porto Alegre, 2010;
- SANTIAGO, Rafael de. DAZZI, Rudimar Luís. “Ferramenta de apoio ao ensino de algoritmos”, s.d.
- GONDIM, Halley Wesley A.S.. AMBRÓSIO, Ana Paula. “Esboço de Fluxogramas no Ensino de Algoritmos”, s.d.