Faculdade projeção

Introdução a Programação Estruturada e Algoritmos

com implementação em linguagem C

AULA 2

mferoc

Tópicos

- Estruturas de Decisão
  - Se... então...(if else)
    - Operadores de Comparação
    - Operadores lógicos
    - If - else encadeado
  - Switch-Case

- Referências

- A biblioteca math.h

- A biblioteca string.h

Se... então...(if else)

As estruturas de decisão (testes condicionais) e de repetição (laços) são usados para controlar o fluxo, ordem e a maneira como os programas são executados em linguagem C e diversas linguagens que aceitam o paradigma estruturado, sendo conceitos fundamentais para a programação de computadores.

Em programas de computador em que há interação com o usuário podemos ter diversos ciclos de execução dependendo das ações do usuário. Para tal, precisamos dar instruções para o processador afim de que se comporte de determinadas maneiras e decida o que fazer em diversas situações.

Assim, com a estrutura **if - else**, o ’if’, que do inglês quer dizer ’se’, no sentido de condição, temos a estrutura de tomada de decisão **’SE isto ENTÃO aquilo’**.

A condição do comando **if** é uma expressão que será avaliada. Se o resultado for verdadeiro (qualquer coisa diferente de zero), as instruções dentro do if serão executadas se o resultado for falso (zero), as instruções dentro do if não serão executadas e o fluxo de execução do programa passará para as instruções dentro do **else**.

Se... então...(if else)

Figura encontrada em: http://w3programmers.com/bangla/wp-content/uploads/2017/10/php-if-else-flowchart.png

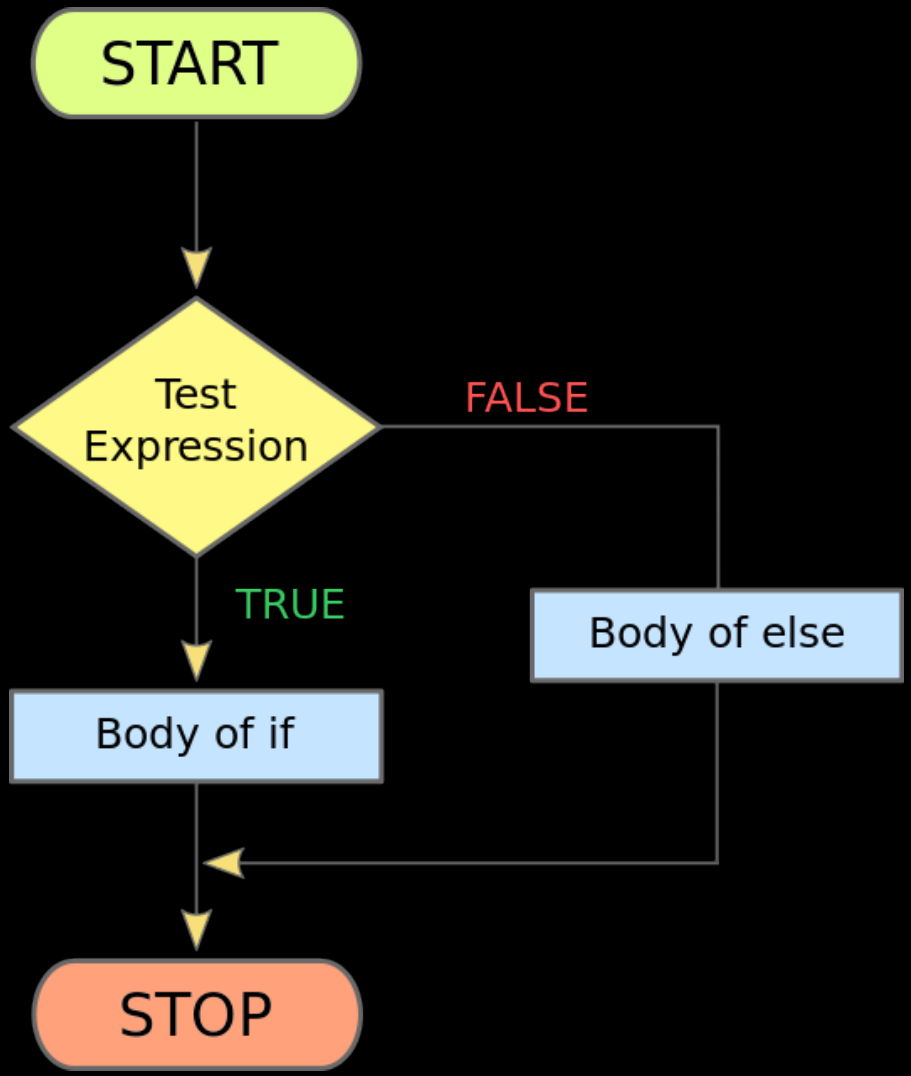

Figura: Diagrama do fluxo da estrutura de decisão if - else.

Se... então...(if else)

Se você quiser que alguma instrução seja executada apenas se um determinado teste for verdadeiro poderá utilizar o comando **if**. Caso você queira que algumas instruções sejam executadas apenas quando um determinado teste for verdadeiro, e outras apenas quando determinado teste for falso, poderá utilizar o comando **if - else**, obedecendo a seguinte sintaxe em linguagem de programação:

```
if (teste) {

   instruções

} else {

   instruções
}
```

Sendo que:

- Teste são as comparações realizadas com os operadores de comparação (operadores relacionais);
- As primeiras { } delimitam as instruções do if que serão executadas se o teste for verdadeiro, e as últimas { } delimitam as instruções do else que serão executadas se o teste for falso;
- Se tiver apenas uma única instrução dentro do if ou do else, as chaves do referido if ou else serão opcionais.

Se... então...(if else)

Exemplo em diagrama: Verifique se um número é maior que 42.

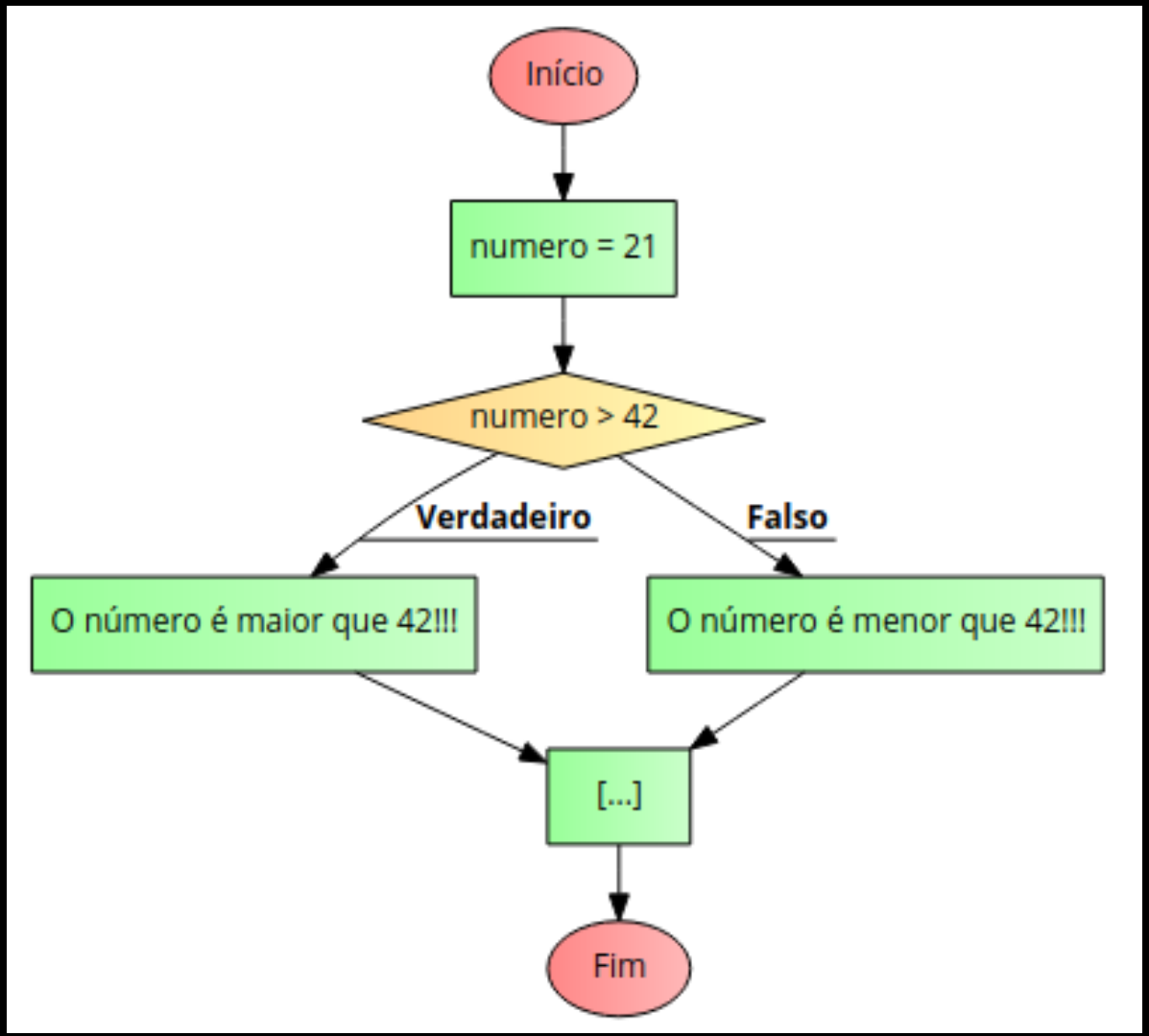

Figura: Diagrama do teste número maior que 42.

Se... então...(if else)

Exemplo em linguagem C: Verifique se um número é maior que 42.

```c
#include <stdio.h>

int main() {

    int numero = 21;

    if(numero > 42) {
        printf("O numero eh maior que 42!!!\n");
    } else {
        printf("O numero eh menor que 42!!!\n");
    }

    return 0;
}
```

Se... então...(if else)

Para realizar testes com o if você precisa também utilizar um dos **operadores que realizam comparações** entre variáveis, chamados de operadores de comparação ou operadores relacionais. São eles:

Sintaxe	Operador
a == b	Igual a
a > b	Maior que
a < b	Menor que
a >= b	Maior ou igual que
a <= b	Menor ou igual que
a != b	Diferente de

Tabela: Operadores de comparação em linguagem C

Se... então...(if else)

Exercícios:

Ex. 1

Leia 3 notas de um aluno, armazene cada nota em uma variável. Calcule a média aritmética dessas notas e informe se o aluno foi aprovado ou não, sendo que, a média (nota mínima) para aprovação é 6 (seis).

Ex. 2

Leia um número inteiro, armazene numa variável, em seguida determine e mostre na tela se o número é par ou ímpar.
Para saber se um número é par, verificamos se o resto da divisão do número por 2 é igual a 0.

numero % 2 = 0 é par.

Se... então...(if else)

Agora, podemos trabalhar com funções e operações matemáticas e com strings de uma forma um pouco melhor, deixando nossos códigos mais otimizados e aumentando a possibilidade do que podemos fazer. Consulte as seguintes seções:

Funções matemáticas com a biblioeca math.h

Para conhecer a biblioteca math.h **clique aqui**.

Trabalhando com strings com a biblioteca string.h

Para conhecer a biblioteca string.h **clique aqui**.

Se... então...(if else)

Podemos usar o if com mais de um teste. Porém como os operadores relacionais binários comparam apenas dois dados de cada vez, precisamos utilizar também os operadores lógicos, cujos principais são estes:

Operador	**Representação em C**	**Exemplo simplificado**
E	&&	if(teste1 && teste2)
OU	&&	if(teste1 || teste2)

Tabela: Operadores lógicos em linguagem C

Se... então...(if else)

Lógica do uso do &&:
Se TODOS OS TESTES forem VERDADEIROS, serão executadas as instruções que estiverem dentro das chaves do if. Basta um teste ser falso para que as instruções do else sejam executadas.

Teste 1	Teste 2	Resultado
V	V	V
V	F	F
F	V	F
F	F	F

Figura: Lógica do E

Se... então...(if else)

Lógica do uso do ||:
Basta que um dos testes seja VERDADEIRO para executar as instruções do if.
Somente executará as instruções do else se TODAS FOREM FALSAS.

Teste 1	Teste 2	Resultado
V	V	V
V	F	V
F	V	V
F	F	F

Figura: Lógica do OU

Se... então...(if else)

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

int main( ){

 int a = 2, b = 5;

 if(a != 2 || b == 5)
    printf("Batman\n");
 else
    printf("Dr.Who\n");

 return 0;
}
```

Se... então...(if else)

Exercícios:

Ex. 1

Leia um número e verifique se o mesmo é divisível exato por 3 e por 7 ao mesmo tempo.

Ex. 2

Leia um número e informe se o mesmo está no intervalo entre 100 e 200, ou não.

Se... então...(if else)

Uso do if encadeado:

1 Uso de if encadeado

Em algumas situações você precisa colocar um if dentro de outro. Nesses casos, você deve analisar cada if como se fosse único. Importante ressaltar a importância da tabulação para destacar onde começa e termina uma estrutura.

```
if(teste1)
{
        instruções 1
        if(teste2)
        {
                instruções 2
        }
        else
        {
                if(teste3)
                {
                        instruções 3
                }
         }
}
else
{
        instruções 4
}
```

	teste1	teste2	teste3	Resultado
instruções 1	V	***	***	executa
instruções 2	V	V	***	executa
instruções 3	V	F	V	executa
instruções 4	F	***	***	executa

Figura: If encadeado, apostila do professor Salvador.

Se... então...(if else)

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

int main() {

    int n1, n2;

    printf("Digite o primeiro numero: ");
    scanf("%d", &n1);
    printf("Digite o segundo numero: ");
    scanf("%d", &n2);

    if(n1 == n2)
        printf("Os numeros sao iguais");
    else {
        if(n1 > n2)
            printf("O maior valor eh = %d", n1);
        else
            printf("O maior valor eh = %d", n2)
    }
    printf("\n");

    return 0;
}
```

Se... então...(if else)

Exercícios:

Ex. 1

Leia um número e informe se o mesmo é positivo, negativo ou nulo.

Ex. 2

Leia 3 números e informe o maior deles.

Switch-case

Switch/case é uma estrutura de condição que define o código a ser executado com base em uma comparação de valores. Usado para criar menus, por exemplo, onde iremos exibir uma série de opções, e o usuário vai escolher uma.
A sintaxe do Switch-case é a seguinte:

```
switch (opcao) {
    case opcao1:
        comandos caso a opcao 1 tenha sido escolhida
        break;
    case opcao2:
        comandos caso a opcao 2 tenha sido escolhida
        break;
    case opcaoN:
        comandos caso a opcao N tenha sido escolhida
        break;
    default:
        comandos caso nenhuma das opcoes anteriores tenha sido
            escolhida
}
```

Switch-case

Ex: De acordo com a entrada de dado do usuário, imprima na tela alguma frase, da seguinte forma: Se o usuário digitar 1, imprima "Bom dia!", se o usuário digitar 2, imprima "Boa tarde!"e se o usuário digitar 3, imprima "Boa noite!".

```c
#include<stdio.h>

int main() {

    int opcao = 0;
    printf("Digite a sua opcao: ");
    scanf("%d", &opcao);

    switch(opcao) {
        case 1:
            printf("Bom dia!\n");
            break;
        case 2:
            printf("Boa tarde!\n");
            break;
        case 3:
            printf("Boa noite!\n");
            break;
        default:
            printf("Opcao invalida\n");
    }

    return 0;
}
```

Se... então...(if else)

Exercícios:

Ex. 1

Faça um programa que recebe um dia da semana, indicado pelos números de 1 a 7, sendo que o número 1 representa o domingo, 2 representa a segunda-feira e assim por diante, e diga se o respectivo dia é um dia de semana ou se é final de semana.

Ex. 2

Faça uma calculadora básica em C. Peça para o usuário digitar 2 números e escolher uma operação, representadas da seguinte forma: 1 - soma, 2 - subtração, 3 - divisão e 4 - multiplicação. Após mostre na tela o resultado da operação desejada pelo usuário.

Referências

Site C progressivo

C PROGRESSIVO
https://www.cprogressivo.net/p/testes-condicionais-e-controle-de-fluxo.html

Site EXCRIPT

EXCRIPT
http://excript.com/curso-c.html

prof. Salvador Alves de Melo Júnior (2019)

Material da disciplina Algoritmos e Lógica de Programação, aulas 3 e 4, da Faculdade Projeção de Sobradinho


..Thats all Folks!..

A biblioteca math.h

O math.h é um arquivo cabeçalho que fornece protótipos para funções, macros e definição de tipos da biblioteca padrão da linguagem de programação C para funções matemáticas básicas. São disponibilizadas, das quais podemos citar:

- sin(ângulo): retorna o valor do seno. Recebe como argumento o valor dos graus em radianos
- cos(ângulo): retorna o valor do co-seno. Recebe como argumento o valor dos graus em radianos
- tan(ângulo): retorna o valor da tangente. Recebe como argumento o valor dos graus em radianos.
- pow(base,expoente): retorna o valor da base elevada ao expoente. Recebe dois argumentos, o primeiro é a base e o segundo o expoente.
- sqrt(valor): retorna o valor da raiz quadrada. Recebe como argumento um número do qual ele deve extrair a raiz quadrada.
- M_PI: essa constante fornece o valor do número π

Você pode conhecer mais sobre o math.h em:
https://pt.wikipedia.org/wiki/Math.h

A biblioteca math.h

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

int main( ) {

    int numero = 49, raizq, potencia;
    double pi = M_PI;

    raizq = sqrt(numero);
    potencia = pow(numero, 2);

    printf("Raiz quadrada de 49: %d\n49 elevado ao quadrado: %d\
        nValor de pi: %f\n", raizq, potencia, pi );

 return 0;
}
```

A biblioteca math.h

Exercícios:

Ex. 1

Leia o valor da base e do expoente de uma potenciação e depois calcule e mostre na tela o valor da potência.

Ex. 2

Sejam a e b os catetos de um triangulo, faça um programa que receba os valores de a e b e calcule o valor da hipotenusa.Imprima o resultado dessa operacão. Calculamos a hipotenusa da seguinte forma:

$$hipotenusa = \sqrt{a^2 + b^2}$$

Para voltar a apresentação sobre if - else **clique aqui**.

A biblioteca string.h

Uma string é uma série de caracteres. Em programação, cada caractere armazenado na memória de uma string é representado por um valor numérico, e cada caractere é o mesmo que 1 byte (8 bits).
Para representar uma string em C você deve fazer o seguinte:

char nomeDaVariavel[tamanho];

A biblioteca padrão string da linguagem C contém várias funções de manipulação de strings. Para usar essas funções, o seu programa deve incluir o arquivo string.h (#include <string.h>).
Algumas funções da biblioteca string.h são:

- **strlen(string)**: recebe uma string e devolve o seu tamanho, ou seja, o número de caracteres armazenados.

A biblioteca string.h

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

int main( ) {

 int tamanho;
 char nome[20];

 printf("Entre com seu primeiro nome\n");
 scanf("%s", nome);

 tamanho = strlen(nome);

 printf("Seu nome tem %d caracteres\n", tamanho);

 return 0;
}
```

- **strcpy(string1,string2)**: recebe duas strings e copia o conteúdo da segunda string para o endereço de memória da primeira string. O conteúdo original da primeira string é perdido.

A biblioteca string.h

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

int main( ) {

 char nome[20];

 strcpy(nome,"Batman");

 printf("String aramezenada na variavel nome = %s\n", nome);

 return 0;
}
```

- **strcmp(string1,string2)**: recebe duas strings, compara as duas e retorna um valor.
  - devolve um número negativo se a primeira string for menor que a segunda;
  - devolve 0 se as duas strings são iguais;
  - devolve um número positivo se a primeira string for maior que a segunda.

A biblioteca string.h

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

int main( ) {

 char senha[13];
 int valor;

 printf("Entre com a senha\n");
 scanf("%s", senha);

 valor = strcmp(senha,"alaska");

 if(valor == 0)
    printf("senha correta\n");
 else
    printf("senha errada\n");

 return 0;
}
```

Você pode consultar todas as funções da biblioteca string.h no link:
https://www.cprogressivo.net/2013/03/Aprenda-a-usar-todas-as-funcoes-da-biblioteca-string-h-em-C.html

A bilioteca string.h

Outras formas de ler strings do teclado:
Quando você utiliza a função scanf( ) para ler uma string, você consegue armazenar na variável indicada apenas até que a referida função encontre um espaço em branco, quando então para de armazenar.
Para ler strings com espaço em branco e armazenar na memória o valor lido com o espaço podemos fazer das seguintes maneiras:
1 - Usando a função fgets():

```
#include<stdio.h>
#define STRSIZE 10

int main( ) {

    char str[ STRSIZE ];

        fgets( str, STRSIZE, stdin );

        printf( "%s\n", str );

 return 0;
}
```

Existe também a função gets(), porém seu uso é atualmente desaconselhado. Pode consultar com mais delathes no link:
http://rberaldo.com.br/c-por-que-usar-fgets-em-vez-de-gets/

A bilioteca string.h

2 - Usando uma expressão regular para o scanf():

```
#include<stdio.h>
#define STRSIZE 10

int main( ) {

    char str[ STRSIZE ];

        scanf("%[^\n]%*c", str);

        printf( "%s\n", str );

 return 0;
}
```

A biblioteca string.h

Exercícios:

Ex. 1

Leia seu nome completo do teclado, armazene numa variável e mostre o nome na tela.

Ex. 2

Leia um cpf no formato de string, e informe se o mesmo tem 11 digitos ou não.

Para voltar a apresentação sobre if - else **clique aqui**.