

PROGRAMAÇÃO COMPUTACIONAL PARA ENGENHARIA

BIBLIOTECAS

Maurício Moreira Neto¹

¹**Universidade Federal do Ceará**
Departamento de Computação

30 de janeiro de 2020

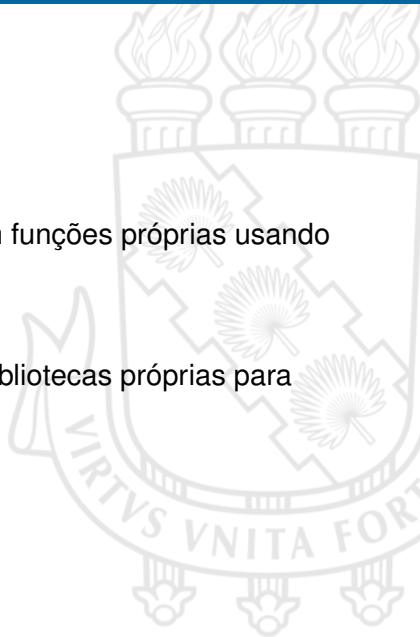
Sumário

- 1 Objetivos
- 2 Relembrando
- 3 Criando Biblioteca



Objetivos

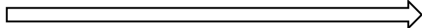
- Aprender a criar bibliotecas com funções próprias usando a linguagem C
- A partir disso, é possível criar bibliotecas próprias para utilização futura



Relembrando

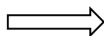
- Vamos relembrar como os códigos são feitos até então...

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



Diretivas de include
(bibliotecas)

```
void somar(float valor1, float valor2, float *r);  
void subtrair(float valor1, float valor2, float *r);  
void multiplicar(float valor1, float valor2, float *r);  
void dividir(float valor1, float valor2, float *r);
```



Declarações de
funções

Relembrando

```
int main(){
    float a, b, res = 0;
    int op;
    printf("Numero 1: ");
    scanf("%f", &a);
    printf("Numero 1: ");
    scanf("%f", &b);
    printf("Escolha opcao de operacao: ");
    scanf("%d", &op);
    switch (op){
        case 1: somar(a, b, &res); break;
        case 2: subtrair(a, b, &res); break;
        case 3: multiplicar(a, b, &res); break;
        case 4: dividir(a, b, &res); break;
        default: printf("Nenhuma operacao foi
escolhida");
    }
    printf("O resultado da operacao foi: %.2f", res);
    return 0;
}
```

- Função principal que normalmente chama as funções declaradas anteriormente

Relembrando

```
void somar(float valor1, float valor2, float *r){
    *r = valor1 + valor2;
}
void subtrair(float valor1, float valor2, float *r){
    *r = valor1 - valor2;
}
void multiplicar(float valor1, float valor2, float *r){
    *r = valor1 * valor2;
}
void dividir(float valor1, float valor2, float *r){
    if (valor2 == 0){
        printf("Nao pode divisao por zero!");
    } else{
        *r = valor1/valor2;
    }
}
```

- Implementação (definição) das funções declaradas anteriormente

Criando Biblioteca

- Regras para criação de arquivos de cabeçalhos:
 - 1 Este arquivo deve conter apenas as declarações que serão visíveis ao programa
 - 2 Deve conter apenas as diretivas **#include** necessárias para suas declarações
 - 3 Proteja-o de declarações duplicadas usando uma guarda de cabeçalho
 - 4 Nomeie corretamente o símbolo de seu arquivo de cabeçalho

- Vamos criar um arquivo chamado **calculadora.h**

Criando Biblioteca

■ Arquivo **calculadora.h**

```
#ifndef CALCULADORA_HEADER_  
#define CALCULADORA_HEADER_  
  
void somar(float valor1, float valor2, float *r);  
void subtrair(float valor1, float valor2, float *r);  
void multiplicar(float valor1, float valor2, float *r);  
void dividir(float valor1, float valor2, float *r);  
  
#endif
```


Criando Biblioteca

- A primeira linha verifica se o símbolo **CALCULADORA_HEADER_** está definido
- Caso não esteja definido, o conteúdo até **#endif** será incluído pelo compilador
- **CALCULADORA_HEADER_** é definido dentro desse intervalo, o que impede que o compilador duplique as duas declarações logo abaixo

Criando Biblioteca

■ O que deve ser evitado:

- Com dois sublinhados consecutivos (por `__` exemplo)
- Que comece com um sublinhado seguido por uma letra maiúscula (`CALCULADORA_HEADER` é reservado pela definição da linguagem)
- Que comece com um sublinhado no contexto global (`calculadora_header` é reservado, pois está no contexto global)
- Note que a regra para nomenclatura é simples:
 - O nome do arquivo em maiúsculas + `HEADER`

Criando Biblioteca

- Agora que temos o arquivo com as declarações das funções, é necessário criar as definições destas funções (sua implementação)
- Crie um arquivo um **calculadora.c** com as definições com das funções declaradas em **calculadora.h**
 - **Importante:** o nome é o mesmo, só muda a extensão

Criando Biblioteca

- As regras para criar o arquivo .C são:
 - 1 Deve conter apenas as definições, que não serão visíveis ao programa
 - 2 Deve conter apenas as diretivas **#include** necessárias para suas definições

Criando Biblioteca

■ Arquivo `calculadora.c`

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

void somar(float valor1, float valor2, float *r){
    *r = valor1 + valor2;
}
void subtrair(float valor1, float valor2, float *r){
    *r = valor1 - valor2;
}
void multiplicar(float valor1, float valor2, float *r){
    *r = valor1 * valor2;
}
void dividir(float valor1, float valor2, float *r){
    if (valor2 == 0){
        printf("Nao pode divisao por zero!");
    } else{
        *r = valor1/valor2;
    }
}
```



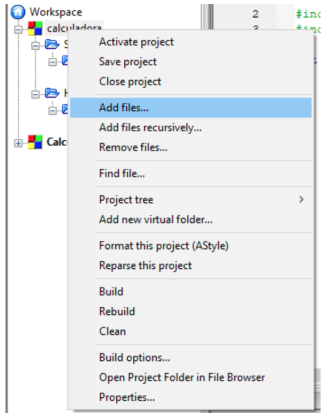
Criando Biblioteca

- Ainda não acabou...Provavelmente isso irar gerar um erro no Code Blocks
- Isso porque os arquivos **calculadora.h** e **calculadora.c** não foram unidas em um projeto para gerar um arquivo de extensão **.a**
- **Vamos ver o passo-a-passo para fazer isso no Code::Blocks**

Importante: esse procedimento depende da IDE!!

Criando Biblioteca

- Crie um novo projeto → **Static Library**
- Como título do projeto, coloque **calculadora**
- Agora adicione os arquivos **.h** e **.c** no projeto de biblioteca estática
- Exclua o arquivo **main.c** do projeto da biblioteca estática
- Por fim, gere o **build** deste projeto!



Criando Biblioteca

- Ao “buildar” o projeto de biblioteca estática, irá aparecer um arquivo novo chamado **libcalculadora.a**
- Deve-se renomear este arquivo para: **calculadora.a**

Nome	Data de modificaç...	Tipo	Tamanho
bin	08/05/2019 10:46	Pasta de arquivos	
obj	08/05/2019 10:46	Pasta de arquivos	
calculadora.cbp	08/05/2019 10:44	Arquivo CBP	2 KB
calculadora.depend	08/05/2019 10:46	Arquivo DEPEND	1 KB
main.c	24/12/2017 15:36	Arquivo C	1 KB

Nome	Data de modificaç...	Tipo	Tamanho
calculadora.a	08/05/2019 10:46	Arquivo A	3 KB

Criando Biblioteca

- Voltando para nosso projeto Console Application, faremos o seguinte passo-a-passo:
- Va em **Settings** → **Compiler** → **Linker Settings** à Add (após isso, deve-se colocar o arquivo calculadora.a que está no projeto da biblioteca estática)

Criando Biblioteca

