Atividade Avaliativa

Entrega:	16/04/2025	**Observações:**	*Entrega individual (repositório Github)*
Nota:	1 ponto		

Estrutura do Projeto (Separação modular e Makefile)

Organizaremos as pastas e arquivos da seguinte forma:

```
strings/
├── src/
│   ├── main.c              # Arquivo principal com o menu
│   ├── strings_func.c      # Implementação das operações sobre strings
│   ├── menu.c              # Implementação do menu e interação com o usuário
├── lib/
│   ├── strings_func.h      # Declaração das funções de string
│   ├── menu.h              # Declaração das funções do menu
├── obj/
│   ├── strings             # Arquivo objeto [string] gerado pelo gcc
├── Makefile                # Compilação do projeto
```

1. `main.c`

O arquivo principal ficará super limpo, chamando o menu / limites.

```
#include <stdio.h>
#include <stdlib.h>
#include "../lib/menu.h"

int main(int argc, char const *argv[])
{

    showMenu();
    exibirLimites();

    return EXIT_SUCCESS;
}
```

2. Implementação das funções do `menu.c`

Aqui faremos as seguintes melhorias:

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
#include "../lib/menu.h"
#include "../lib/strings_func.h"
#include <limits.h>
#include <float.h>

#define MAX_STRING CHAR_MAX // Tamanho máximo da string baseado em CHAR_MAX

void showMenu() {
    printf("Dentro menu...\n");
    char input[MAX_STRING + 1] = {0}; // String manipulada pelo usuário
    bool rodando = true; // Controle do loop principal

    while (rodando) {
        printf("\n--- Menu ---\n");
        printf("1. Inserir uma string\n");
        printf("2. Transformar em maiúsculas (uppercase)\n");
        printf("3. Transformar em minúsculas (lowercase)\n");
        printf("4. Remover espaços extras (trim)\n");
        printf("5. Capitalizar palavras\n");
        printf("6. Exibir limites da linguagem C\n");
        printf("7. Sair\n");
        printf("Escolha uma opção: ");

        int opcao;
        if (scanf("%d", &opcao) != 1) { // Evita entrada inválida
            printf("Entrada inválida! Por favor, tente novamente.\n");
            while (getchar() != '\n'); // Limpa o buffer
            continue;
        }
        getchar(); // Limpa o buffer após o número

        switch (opcao) {
            case 1:
                printf("Digite sua string (máx. %d caracteres): ", MAX_STRING);
                fgets(input, sizeof(input), stdin);
                input[strcspn(input, "\n")] = '\0'; // Remove o \n
                break;
            case 2:
                uppercase(input);
                printf("Resultado: %s\n", input);
                break;
            case 3:
```

```
            lowercase(input);
            printf("Resultado: %s\n", input);
            break;
        case 4:
            trimSpaces(input);
            printf("Resultado: %s\n", input);
            break;
        case 5:
            capitalize(input);
            printf("Resultado: %s\n", input);
            break;
        case 6:
            exibirLimites();
            break;
        case 7:
            printf("Encerrando o programa.\n");
            rodando = false; // Encerra o loop
            break;
        default:
            printf("Opção inválida! Tente novamente.\n");
    }
  }
}

void exibirLimites() {
   printf("\n--- Limites Inteiros ---\n");
   printf("INT_MAX: %d\n", INT_MAX);
   printf("INT_MIN: %d\n", INT_MIN);
   printf("CHAR_MAX: %d\n", CHAR_MAX);
   printf("CHAR_MIN: %d\n", CHAR_MIN);

   printf("\n--- Limites de Ponto Flutuante ---\n");
   printf("FLT_MAX: %e\n", FLT_MAX);
   printf("FLT_MIN: %e\n", FLT_MIN);
   printf("DBL_MAX: %e\n", DBL_MAX);
   printf("DBL_MIN: %e\n", DBL_MIN);
}
```

3. Funções de Manipulação `strings_func.c`

```
#include <ctype.h>
#include <string.h>


#include "../lib/strings_func.h"
```

```
// Transforma a string em maiúsculas
void uppercase(char *str) {
    for (int i = 0; str[i]; i++) {
        str[i] = toupper((unsigned char)str[i]);
    }
}

// Transforma a string em minúsculas
void lowercase(char *str) {
    for (int i = 0; str[i]; i++) {
        str[i] = tolower((unsigned char)str[i]);
    }
}

// Remove espaços extras no início e final da string
void trimSpaces(char *str) {
    char *inicio = str;
    while (isspace((unsigned char)*inicio)) inicio++;

    char *fim = str + strlen(str) - 1;
    while (fim > inicio && isspace((unsigned char)*fim)) fim--;

    memmove(str, inicio, fim - inicio + 1);
    str[fim - inicio + 1] = '\0'; // Finaliza a string
}

// Capitaliza as palavras na string
void capitalize(char *str) {
    int capitalizeNext = 1;
    for (int i = 0; str[i]; i++) {
        if (isspace((unsigned char)str[i])) {
            capitalizeNext = 1;
        } else if (capitalizeNext) {
            str[i] = toupper((unsigned char)str[i]);
            capitalizeNext = 0;
        } else {
            str[i] = tolower((unsigned char)str[i]);
        }
    }
}
```

4. libs/ `menu.h`

```
#ifndef MENU_H
#define MENU_H

void showMenu();
void exibirLimites();

#endif
```

5. libs/ `strings_func.h`

```
#ifndef STRINGS_FUNC
#define STRINGS_FUNC

void uppercase(char *str);
void lowercase(char *str);
void trimSpaces(char *str);
void capitalize(char *str);
#endif
```

6. Makefile

```
CC = gcc
CFLAGS = -Wall -Werror -Wno-unused-result -O2
O = ./obj
SRC = ./src/*.c
EXEC = $(O)/strings

all:
	$(CC) -o $(EXEC) $(SRC)

run:
	$(EXEC)

asm:
	for file in $(SRC); do \
		$(CC) $(CFLAGS) -S -o $(O)/$$(basename $$file .c).s $$file; \
   done


clear:
	rm -f $(O)/*
```