UNISANTOS
Universidade Católica de Santos

Centro de Ciências Exatas, Arquitetura e Engenharia

Professor:	Ciro Cirne Trindade
Disciplina:	Introdução à Computação-II
Cursos:	Ciência da Computação/Sistemas de Informação

Lista de Exercícios: Arquivos

1. Escreva um programa que recebe o nome de dois arquivos do tipo texto via argumentos da função *main()*. O programa deve gerar uma cópia o primeiro arquivo no segundo, entretanto, todos os caracteres do segundo arquivo devem estar em letras maiúsculas. Use a função *toupper()* do *ctype.h* para converter todos os caracteres para maiúsculo e gravá-los no segundo arquivo.

2. Escreva um programa para criptografar e descriptografar um arquivo do tipo texto usando a cifra de César (https://pt.wikipedia.org/wiki/Cifra_de_César). O programa deve receber como argumentos da função da *main()* o nome do arquivo a ser criptografado ou descriptografado, *-c* (criptografar) ou *-d* (descriptografar) e a chave (rotação).

3. Escreva um programa que recebe dois valores via argumentos da função *main()*. O primeiro é uma string e o segundo o nome de um arquivo do tipo texto. O programa deve percorrer o arquivo imprimindo no vídeo todas as linhas que contêm a string. Use a função *strstr()* para procurar a palavra em uma linha.

4. A CETESP lhe contratou para implementar uma aplicação em C para manter em um arquivo chamado "chuvas.dat", o índice pluviométrico diário na região da Grande São Paulo. O programa deve armazenar estruturas do tipo `chuva`, mostrada abaixo, neste arquivo. Além de permitir esse cadastro, seu programa deve possuir uma opção que gere um arquivo do tipo texto, chamado "dias_chuvosos.txt", contendo, em cada linha, a data (no formato dd/mm/aaaa) e o índice pluviométrico, separados por um espaço em branco, de todos os dias de um determinado ano em que esse índice superou os 60 mm.

```
typedef struct {
   int dia;
   int mes;
   int ano;
} data;

typedef struct {
   data dt;
   int ind_pluviometrico;
} chuva;
```

5. Escreva uma aplicação em C que crie um cadastro contendo o nome, telefone e e-mail de seus amigos. Essas informações são representadas pela estrutura `amigo` mostrada a seguir. Todo amigo cadastrado recebe um *id* sequencial e gerado automaticamente pelo sistema. A aplicação deve possuir opções para cadastrar um novo amigo, listar todos os amigos cadastrados, alterar o telefone ou e-mail de um amigo dado seu *id* ou nome e consultar um amigo dado seu *id* ou nome.

```
typedef struct {
   int id; // identificador do amigo
   char nome[40];
   char fone[14];
   char email[30];
} amigo;
```