Algoritmos e Programação 2

Arquivos - Parte I

1. Faça um programa para ler um arquivo texto (o usuário deve fornecer o nome do arquivo junto com a chamada do programa) e imprimir o seu conteúdo na tela.

2. Faça um programa que leia um texto de um arquivo T_1 e copie o texto para um arquivo T_2 mantendo a estrutura de linhas.

3. Faça um programa que conte o número de caracteres e de palavras de um arquivo texto.

4. Faça um programa que lê um arquivo texto e copia apenas os caracteres alfabéticos (letras) para um arquivo de destino (ambos fornecidos na chamada do programa). Números e caracteres especiais devem ser desconsiderados.

5. Faça um programa que procura pelas ocorrências de uma *string* dentro de um arquivo texto e informa em quais posições (linha, coluna) foram encontradas tais ocorrências.

6. Faça uma função que gere um arquivo texto com n linhas e m colunas, sendo cada posição preenchida por um valor inteiro gerado aleatoriamente. A separação entre uma coluna e outra deverá ser feita por um espaço em branco. Faça outra função para ler e imprimir o arquivo gerado.

7. Faça um programa leia um arquivo de dados numéricos do tipo *float*, dispostos em uma coluna, e informe em quais linhas foram encontrados o maior valor e o menor.

8. Dado um arquivo numérico com duas colunas de valores inteiros, faça uma função que leia estes dados e gere um arquivo com 3 colunas, sendo a terceira coluna o valor da soma das outras duas. O número de linhas do arquivo deve permanecer o mesmo.

9. Faça um programa que processe um texto convertendo as letras maiúsculas em minúsculas. A inicial de cada frase deve permanecer maiúscula.

10. Faça um programa que leia um arquivo texto `TEXTO.TXT` e grave todas as palavras iniciadas com vogais no arquivo texto `VOGAIS.TXT`. O programa deve também mostrar na tela o conteúdo do arquivo `TEXTO.TXT`, durante a procura das palavras, e o número de palavras gravadas em `VOGAIS.TXT`.

11. Faça um programa para formatar um arquivo texto qualquer informado na chamada do programa. O seu programa deverá criar um novo arquivo com 60 caracteres por linha, páginas numeradas e com no máximo 60 linhas por página. Cada página deverá ser separada por uma linha (60 caracteres) de hifens '`-`'.

12. Crie um programa para manter um cadastro de Alunos. O seu programa deverá oferecer as seguintes funcionalidades (menu):

```
1. Cadastrar
2. Alterar
3. Remover
4. Buscar
5. Listar
6. Sair
```

Todos os dados deverão ser salvos no arquivo-texto `Alunos.dat` com os registros ordenados em ordem crescente de RA. A primeira linha do arquivo deve conter a quantidade de alunos cadastrados. Caso o arquivo já exista, o seu programa deverá ler os dados já existentes. Mantenha os dados dos alunos em uma estrutura `Aluno` com os seguintes campos: RA (`int`), nome (*string* - 100 posições), ano de ingresso (`int`) e quantidade de créditos cursados (`int`).

Cada opção deverá executar o seguinte procedimento:

(a) `Cadastrar`: solicita todos os dados do aluno. Caso o RA informado já exista, informar na tela que o aluno já está cadastrado e retornar ao menu;

(b) `Alterar`: solicitar o RA do aluno. Caso ele seja encontrado, solicitar novamente os campos: nome, ingresso e quantidade de créditos cursados. Caso contrário, emitir mensagem de aluno não cadastrado e retornar ao menu.

(c) `Remover`: solicitar o RA do aluno. Caso ele seja encontrado, remover o registro. Caso contrário, emitir mensagem de aluno não cadastrado e retornar ao menu.

(d) `Buscar`: solicitar o RA do aluno. Caso ele seja encontrado, exibir todos os campos do registro. Caso contrário, emitir mensagem de aluno não cadastrado e retornar ao menu.

(e) `Listar`: imprimir na tela todos os campos de todos os registros existentes em ordem crescente de RA.

(f) `Sair`: sobrescrever o arquivo com dados atualizados, liberar memória e fechar o programa.

O seu programa deverá carregar/criar o arquivo na inicialização, manipular os dados em memória (sempre mantendo os registros em ordem crescente de RA) e sobrescrever o arquivo na finalização.

Os campos dos registros deverão ser separados pelo delimitador @ e cada registro deverá ocupar uma linha do arquivo de dados.

Trate os possíveis erros que o usuário possa cometer, tais como: informar valores não numéricos nos campos `int`, deixar campos em branco, digitar uma opção inválida no menu, etc.