Lista 01

Strings

- **Atenção:**

  1. **Identificadores de variáveis:** escolha nomes apropriados;
  2. **Documentação:** inclua cabeçalho, comentários e indentação no programa.
  3. **Arquivo-base:** você deve usar o arquivo-fonte incompleto fornecido junto com a lista. É necessário completar as operações nos lugares indicados e você não deve realizar nenhuma alteração nas partes fornecidas. Inclusive, se houverem comandos de entrada (`scanf`) e saída (`printf`) definidos, estes não poderão ser alterados.

- **Exercícios:**

  1. Crie um programa para simular um jogo de caça-palavras. Ele precisará montar a grade de letras (informada pelo usuário) e buscar por palavras definidas pelo usuário tanto em linhas, colunas e diagonais.

     As entradas do programa são:

     - m: número de linhas da grade
     - n: número de colunas da grade
     - $\mathcal{G}$: grade com $m \times n$ letras
     - q: quantidade de palavras para buscar na grade
     - $\mathcal{P}$: conjunto com q palavras

     A saída do programa deverá ser a grade com as palavras encontradas e o caractere '#' substituindo as demais letras.

     Complete o arquivo `L01EX01.c`

     Detalhes

     (a) Nos casos de teste, todas as letras da grade já são informadas corretamente, portanto, não é necessária nenhuma verificação.
     (b) O seu programa deve permitir a busca por palavras invertidas.
     (c) As dimensões da grade m e n não podem ser superiores a 10.
     (d) Na leitura da grade, você não deve considerar espaços ou quebras de linha.

     Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	Saída
5 5 (dimensões da grade) A V I A O (preenchimento da grade) N L I J L A O G O F V E W I O E A V E I	
4 (número de palavras a serem procuradas)	
AVIAO (primeira palavra)	
ALGII (segunda palavra)	
NAVE (terceira palavra)	
FOGO (quarta palavra)	
	A V I A O N L # # # A O G O F V # # I # E # # # I

Casos de testes

(a) **Caso 1**: procura na horizontal.
(b) **Caso 2**: procura na vertical.
(c) **Caso 3**: procura na diagonal principal.
(d) **Caso 4**: procura na diagonal secundária.
(e) **Caso 5**: todos os casos + invertidos.

2. 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 (ordem crescente de RA)
6. Listar por nome (ordem alfabética)
0. Sair
```

Todos os dados deverão ser mantidos com os registros ordenados em ordem crescente de RA. Mantenha os dados dos alunos em uma estrutura `Aluno` com os seguintes campos: RA (`int`), nome (*string* - 100 posições), data de ingresso (`registro` para armazenar datas do tipo `dd/mm/aaaa`) e quantidade de créditos cursados (`int`).

Cada opção deverá executar o seguinte procedimento:

- `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. Caso não haja espaço suficiente, informar na tela e retornar ao menu.
- `Alterar`: solicitar o RA do aluno. Caso ele seja encontrado, solicitar novamente os campos: data de ingresso, quantidade de créditos cursados e nome do aluno. Caso contrário, emitir mensagem de aluno pré-definida de aluno não cadastrado e retornar ao menu.
- `Remover`: solicitar o RA do aluno. Caso ele seja encontrado, remover o registro. Caso contrário, emitir mensagem pré-definida de aluno não cadastrado e retornar ao menu.

- `Buscar`: solicitar o RA do aluno. Caso ele seja encontrado, exibir todos os campos do registro. Caso contrário, emitir mensagem pré-definida de aluno não cadastrado e retornar ao menu.
- `Listar`: imprimir na tela todos os campos de todos os registros existentes em ordem crescente de RA.
- `Listar por nome`: imprimir na tela todos os campos de todos os registros existentes em ordem alfabética.
- `Sair`: finaliza o programa.

Complete o arquivo `L01EX02.c`

Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos)

1 (Inserir) 111111 01/01/2010 100 Joao da Silva 1 (Inserir) 222222 10/05/2012 222 Fulano Ferraz 1 (Inserir) 000000 30/01/2000 20 Sicrano Batista	
1 (Inserir) 111111 10/01/2011 111 Joao Silva	Aluno ja cadastrado!
5 (Listar)	000000 - Sicrano Batista - 30/01/2000 - 0020 111111 - Joao da Silva - 01/01/2010 - 0100 222222 - Fulano Ferraz - 10/05/2012 - 0222
2 (Alterar) 111111 10/01/2011 111 Joao Silva 2 (Alterar) 222222 10/01/2013 333 Fulano Ferraz da Silva	
5 (Listar)	000000 - Sicrano Batista - 30/01/2000 - 0020 111111 - Joao Silva - 10/01/2011 - 0111 222222 - Fulano Ferraz da Silva - 10/01/2013 - 0333
6 (Listar por nome)	222222 - Fulano Ferraz da Silva - 10/01/2013 - 0333 111111 - Joao Silva - 10/01/2011 - 0111 000000 - Sicrano Batista - 30/01/2000 - 0020
4 (Buscar) 123456	Cadastro nao encontrado!
4 (Buscar) 111111	111111 - Joao Silva - 10/01/2011 - 0111
3 (Remover) 123456	Cadastro nao encontrado!
3 (Remover) 111111	
5 (Listar)	000000 - Sicrano Batista - 30/01/2000 - 0020 222222 - Fulano Ferraz da Silva - 10/01/2013 - 0333
0 (Sair)	

Casos de testes

(a) **Caso 1**: Opções 1 e 5.
(b) **Caso 2**: Opções 1, 2 e 5.
(c) **Caso 3**: Opções 1, 2, 3 e 5.
(d) **Caso 4**: Opções 1, 2, 3, 4 e 5.
(e) **Caso 5**: Todas as opções.

3. Um amigo pediu para você criar um programa para validar endereços de e-mail que será usado em uma página de cadastros de um fórum criado por ele. Ele passou as seguintes anotações, contendo as regras para a validação dos endereços de e-mail.
   (a) Os endereços de e-mail deverão ter no máximo 50 caracteres;
   (b) Deve haver exatamente um `@`, separando o nome do usuário do nome do servidor;
   (c) O fórum é restrito, e portanto, só pode aceitar usuários que possuem e-mails dos seguintes servidores: `email.abc`, `email.abc.br`, `abcmail.xyz.br`, `abcweb.asd.br` e `asdmail.xyz`;
   (d) O nome do usuário poderá conter letras minúsculas, números e os seguintes 4 símbolos: {`.` `,` `_` `-`} (ponto, vírgula, *underline* e hífen);
   (e) Os símbolos não podem estar localizados no início ou no final do nome do usuário;
   (f) Não são permitidos símbolos consecutivos (por exemplo: `_-`);
   (g) Números não podem estar localizados no início do nome do usuário ou diretamente após um símbolo;

   Cada caso de teste possui duas linhas. A primeira contem um inteiro $N \leq 50$, representando a quantidade de caracteres do e-mail. A segunda linha contem uma string com N caracteres, com o e-mail a ser validado. As entradas serão finalizadas caso o programa receba $N = 0$. Para cada caso de teste imprima "`Valido`", caso o e-mail fornecido seja válido ou "`Invalido`", caso contrário.

   Complete o arquivo `L01EX03.c`

   Você deve apenas completar as operações nos lugares indicados e não deve realizar nenhuma alteração nas partes fornecidas. Inclusive, se houverem comandos de entrada (`scanf`) e saída (`printf`) definidos, estes não poderão ser alterados.

   Exemplos de E/S (os comentários entre parênteses não deverão ser exibidos):

Entrada	Saída
16	
abc@teste.xyz.br	
	Invalido
22	
test_user@email.abc.br	
	Valido
21	
user-.xyz@asdmail.xyz	
	Invalido
0	

   Casos de testes
   (a) **Caso 1**: número de arrobas inválido e servidor inválido.
   (b) **Caso 2**: caracteres inválidos.
   (c) **Caso 3**: símbolos consecutivos.
   (d) **Caso 4**: números precedidos de símbolos.
   (e) **Caso 5**: todos.