INF 1007 - P4 - 12/12/09	Questão 1
Nome:	
Matrícula:	Turma

QUESTÃO OBRIGATÓRIA

[Valor: 2,5 pontos] Escreva um programa completo em C que leia do teclado um caractere e uma cadeia de caracteres e escreva quantas vezes aquele caractere aparece nessa cadeia. Assuma que a cadeia tem no máximo 20 caracteres, que serão apenas letras do alfabeto, maiúsculas ou minúsculas, e não contém espaços. O programa não deve fazer distinção entre maiúsculas e minúsculas. Por exemplo, para a entrada a seguir:

```
A Abracadabra
```

O programa deveria exibir a saída:

```
A cadeia Abracadabra contem 5 vezes o caractere A.
```

Já para a seguinte entrada:

```
x PROVA
```

A saída correspondente seria:

```
A cadeia PROVA nao contem o caractere x.
```

Considere que os valores fornecidos como entrada são sempre válidos, ou seja, não é necessário verificar a validade desses parâmetros.

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1007 - P4 - 12/12/09	Questão 2
Nome:	
Matrícula:	Turma

QUESTÃO OBRIGATÓRIA

[Valor: 2,5 pontos] Em uma prova de programação, o aluno deve implementar uma função que cria uma cópia de um vetor fornecido como parâmetro. Esse vetor, em que os elementos não estão em qualquer ordem especial, contém ponteiros para o tipo *Conteudo*, definido a seguir.

```
typedef struct conteudo {
   int valor;
} Conteudo;
```

O vetor cópia deve ser alocado dinamicamente e, além disso, ordenado em ordem crescente de acordo com o campo *valor*.

Assumindo que o aluno já tenha criado a função que copia e ordena o vetor e que a alocação do novo vetor tenha sido feita corretamente, escreva uma função em C que verifica se a função implementada pelo aluno funciona da forma esperada. Essa função deve receber como parâmetros o ponteiro *vet*, para o vetor original de ponteiros para *Conteudo*, o ponteiro *ordenado*, para o vetor de ponteiros para *Conteudo* que foi criado e ordenado pela função do aluno, e o inteiro *n*, que indica o número total de elementos de cada vetor. A função deve retornar 1, indicando que a ordenação foi executada corretamente ou 0, caso contrário, e deve ter o seguinte protótipo:

```
int Testa_ordenacao(Conteudo** vet, Conteudo** ordenado, int n);
```

Obs: O conteúdo e ordenação do vetor original e do vetor do aluno não devem ser modificados.

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1007 - P4 - 12/12/09	Questão 3
Nome:	
Matrícula:	Turma

QUESTÃO OBRIGATÓRIA

[Valor: 2,5 pontos] Um programa mantém as informações sobre os usuários de um sistema como uma lista encadeada onde cada elemento é do tipo *Elemento*, conforme definido a seguir:

```
typedef struct elemento {
   char login[21];        /*  login do usuario                */
   char senha[9];         /*  senha do usuario                */
   struct elemento *prox; /*  ponteiro para proximo elemento  */
} Elemento;
```

Os campos *login* e *senha* podem conter quaisquer letras e algarismos do alfabeto, maiúsculas ou minúsculas. Por exemplo, um usuário válido poderia ter *login* "jSilva" e *senha* "AbC123". Considere que não existem dois logins iguais na lista encadeada.

Escreva uma função em C que recebe como parâmetros o ponteiro *lst*, para o início da lista descrita, e os ponteiros para as cadeias de caracteres *log* e *sen*, representando o login e senha de um usuário. A função deve retornar um ponteiro para uma cadeia de caracteres alocada dinamicamente, contendo uma das seguintes mensagens:

- "Senha correta", se na lista for encontrado um usuário cujo conteúdo do campo *login* seja igual à cadeia de caracteres apontada por *log* e o conteúdo do campo *senha* correspondente seja igual à cadeia de caracteres apontada por *sen*
- "Senha incorreta", se for encontrado um usuário com *login* igual à *log*, mas cuja *senha* é diferente de *sen*
- "Login inexistente", se não for encontrado um usuário com *login* igual à *log*.

A função deve ter o seguinte protótipo:

```
char* Verifica_senha(Elemento* lst, char* log, char* sen);
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1007 - P4 - 12/12/09	Questão 4
Nome:	
Matrícula:	Turma

QUESTÃO OPCIONAL: O aluno deve escolher esta, a 5ª ou a 6ª questões.

[Valor: 2,5 pontos] Considere um programa de gerenciamento comercial que armazena as informações referentes aos dados sobre os produtos em estoque como um vetor em que cada elemento é um ponteiro para o tipo *Produto* definido a seguir.

```
typedef struct produto {
   int cod;                /* codigo do produto         */
   char* descr;            /* descricao do produto      */
   int quant;              /* quantidade em estoque     */
   float valor;            /* valor unitario do produto */
} Produto;
```

Sabendo que o vetor encontra-se em ordem crescente de acordo com o código de cada produto (valor do campo *cod*), escreva uma função em C que verifica a quantidade de um produto em estoque. Esta função recebe como parâmetros o ponteiro *estoque*, para o primeiro elemento do vetor, o inteiro *n*, representando o tamanho do vetor, e o inteiro *cod*, indicando o código de um produto, e retorna a quantidade em estoque deste produto (valor do campo *quant*). Se o produto com o referido código não for encontrado no vetor, a função deve retornar -1. A função deve empregar a técnica de busca binária e seu protótipo é:

```
int Consulta_estoque(Produto **estoque, int n, int cod);
```

Obs.: Sua função NÃO deve utilizar a função *bsearch* da biblioteca padrão.

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1007 - P4 - 12/12/09	Questão 5
Nome:	
Matrícula:	Turma

QUESTÃO OPCIONAL: O aluno deve escolher esta, a 4ª ou a 6ª questões.

[Valor: 2,5 pontos] Considere um TAD do tipo *Pilha* --- que armazena caracteres ---, para o qual foram implementadas as seguintes funções:

Pilha* pilha_cria (void);	Retorna o ponteiro para uma nova pilha alocada dinamicamente.
void pilha_push (Pilha* p, char c);	Insere um caractere no topo de uma pilha. O ponteiro da pilha e o valor a ser inserido são passados como parâmetros.
char pilha_pop (Pilha* p);	Retira um caractere do topo de uma pilha. O ponteiro da pilha é passado como parâmetro, e o retorno é o valor deste elemento. **Esta função não deve ser chamada se a pilha estiver vazia.**
int pilha_vazia (Pilha* p);	Retorna 1 se a pilha está vazia, ou 0, caso contrário.
void pilha_libera (Pilha* p);	Libera toda a memória alocada para uma pilha. O ponteiro da pilha é passado como parâmetro.

Sem conhecer a representação interna deste TAD e usando apenas as funções descritas acima, escreva uma função em C que compara duas pilhas. A função deve receber como parâmetro os ponteiros *p1* e *p2* para duas pilhas, e retornar 1 se o conteúdo das duas pilhas for idêntico, ou seja, se ambas contêm os mesmos valores armazenados na mesma ordem, ou 0, caso contrário. Após a execução dessa função, o conteúdo das pilhas originais deve continuar o mesmo. O protótipo da função é:

```
int pilha_compara(Pilha *p1, Pilha *p2);
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1007 - P4 - 12/12/09	Questão 6
Nome:	
Matrícula:	Turma

QUESTÃO OPCIONAL: O aluno deve escolher esta, a 4ª ou a 5ª questões.

[Valor: 2,5 pontos] Considere um TAD do tipo *Fila* --- para armazenar valores inteiros ---, para o qual foram implementadas as seguintes funções:

Fila* fila_cria (void);	Retorna o ponteiro para uma nova fila alocada dinamicamente.
void fila_insere (Fila* f, int x);	Insere um valor inteiro no fim de uma fila. O ponteiro para a fila e o elemento a ser inserido são passados como parâmetros.
int fila_retira (Fila* f);	Retira um valor do início de uma fila. O ponteiro da fila é passado como parâmetro, e o retorno é o valor inteiro retirado. **Esta função não deve ser chamada se a fila estiver vazia.**
int fila_tamanho (Fila* f);	Retorna o número de elementos na fila.
void fila_libera (Fila* f);	Libera toda a memória alocada para uma fila. O ponteiro da fila é passado como parâmetro.

Sem conhecer a representação interna deste TAD e usando apenas as funções descritas acima, escreva uma função em C que cria a cópia de uma fila. A função deve receber como parâmetro o ponteiro *f* para uma determinada fila e retornar o ponteiro para uma nova fila cujo conteúdo é cópia do conteúdo daquela fornecida como parâmetro, ou seja, ambas (a fila recebida como parâmetro e a fila nova) contêm os mesmos valores e na mesma ordem. Após a execução dessa função, o conteúdo da fila original deve continuar o mesmo. O protótipo da função é:

```
Fila *fila_copia(Fila *f);
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.