INF 1007/INF1006 – DI - PROVA 1– 27/04/2013	
Nome:	
Matrícula:	Turma:

Instruções: Leia as instruções até o final antes de realizar qualquer ação.

1. Esta prova deverá ser resolvida em até 1:45 minutos

2. A prova é sem consulta e sem perguntas. A interpretação do enunciado faz parte da prova.

3. Seguindo as instruções do fiscal, o aluno deve **CRIAR** o arquivo ***TURMA_MATRICULA_NOME*_P1.c** (**onde deverá ser colocada a solução da prova**) substituindo *TURMA* pela turma do aluno (com 3 dígitos) , *MATRICULA* pela matrícula do aluno (com 7 dígitos) e *NOME pelo último nome do aluno*. Por exemplo, "33B_0720870_MARTINS_**P1**.c" seria o arquivo contendo a solução da prova de um aluno da turma 33B, cuja matrícula é 0720870 , chamado, por exemplo Pedro da Silva Martins.
   Deve ser incluído no início desse arquivo, como comentário, o cabeçalho abaixo preenchido CORRETAMENTE com nome completo, matrícula e turma do aluno. Informações incorretas ou omitidas neste cabeçalho invalidam a prova.

```
/*
NOME:
MATRICULA:
TURMA:
*/
```

   Antes da solução de cada questão deve vir a identificação da questão, também como comentário:
   Exemplo: /* **Solucao da Questao 1.a** */

4. Por segurança, este arquivo .c deve ser salvo a cada 15 minutos, pelo próprio aluno. Somente esse arquivo será considerado a resposta da prova. EVITAR MANTER MAIS DE UMA CÓPIA, PARA NÃO ALTERAR UM ARQUIVO E ENTREGAR O ERRADO.

5. Esta é uma prova prática onde o aluno é suposto desenvolver as funções pedidas de maneira que compilem e funcionem corretamente conforme as especificações. **Portanto, recomendamos que o aluno crie o arquivo .c em um projeto no MS Visual Studio, na pasta de DESTINO indicada pelo fiscal.** O nome do projeto deve ter o formato projeto_TURMA_MAT (obviamente substituindo TURMA pela sua turma e MAT pela sua matrícula. Mas apenas as funções pedidas serão consideradas e o único arquivo considerado resposta é o arquivo .c do item 3. (Não entregar projeto ou mais de um arquivo em nenhuma hipótese).

   IMPORTANTE:
   As funções que você escrever irão compor um módulo **.c** que será colocado no projeto do professor para fins de correção. Portanto, respeite a ordem dos argumentos da função conforme o enunciado. Por exemplo, o texto "a função **f** recebe um vetor de inteiros e o tamanho desse vetor e retorna a soma dos elementos desse vetor" implica no seguinte protótipo **int f(int * v,int n);** e não int f(int n,in * v);. Em outras palavras, você tem a liberdade de nomear suas variáveis, mas não pode mudar a ordem (tipos) com que elas aparecem como argumentos. Isto é necessário para que o seu módulo possa ser usado pelos módulos de teste do professor.

6. Se você quiser usar o exemplo que está no enunciado em seus testes, o arquivo com a definição das structs e esse exemplo encontra-se na pasta de ORIGEM indicada pelo fiscal. Faça logo a cópia desse arquivo para a pasta de DESTINO indicada pelo fiscal, pois depois poderá não ter mais acesso a ela.
7. Se em alguma questão o aluno criar e usar uma função auxiliar, essa função deve também ser apresentada no arquivo de respostas.

8. Ao final, copie o seu arquivo.c que está dentro do projeto para a pasta de DESTINO indicada pelo fiscal.
9. Todo material entregue ao aluno em papel deverá ser devolvido, mas não terá efeito na nota.OBSERVAÇÂO:

Use o MS Visual Studio para testar as funções pedidas. Lembre-se que para compilar no VisualStudio é necessário que exista uma main, ainda que vazia:

```
int main (void)
{
  return 0;
}
```

QUESTÃO 1:

O cadastro dos eleitores de uma determinada zona eleitoral é mantido utilizando-se um contador com o número de eleitores e dois vetores de inteiros: um vetor de inteiros com todas as inscrições (inscrição é considerada um inteiro) dos eleitores dessa determinada zona eleitoral e um vetor de inteiros com as idades desses eleitores (onde idade[k] é a idade do eleitor de inscrição insc[k]).

1.a) (1.5 ponto) Escreva a função *contaEleitoresOpcionais* que :

- recebe um vetor de inteiros com as idades desses eleitores e o número de eleitores;
- devolve dois valores: o número de eleitores com idade estritamente menor do que 18 anos e o número de eleitores com idade estritamente maior do que 65 anos. O tipo de retorno da função é void. Observe que, nesse caso, a sua função deverá receber além do vetor e do número de eleitores, outros dois parâmetros para poder devolver esses dois valores.

1.b) (2.5 pontos) Escreva agora a função *obtemEleitoresObrigatorios* que :

- recebe um vetor de inteiros com todas as inscrições dos eleitores de uma determinada zona eleitoral, um vetor de inteiros com as idades desses eleitores (onde idade[k] é a idade do eleitor de inscrição insc[k]) e o número de eleitores; recebe também o endereço de uma variável inteira onde será devolvido o número de eleitores obrigatórios;
- retorna (o endereço do primeiro elemento de) um novo vetor de inteiros, do tamanho exato necessário, apenas com as inscrições dos eleitores que são obrigados a votar, ou seja, que têm idade maior ou igual a 18 anos e menor ou igual a 65 anos. A função também deve devolver na variável inteira cujo endereço foi fornecido como argumento na chamada da função o número de eleitores obrigatórios. A função pode utilizar a função do item 1.A. Caso não exista nenhum eleitor obrigatório, a função retorna NULL. Caso não seja possível alocar o espaço necessário, a função retorna NULL.

INF1007/INF1006- DI - PROVA1 – 27/04/2013

QUESTÃO 2:

Considere o tipo estruturado **PlanoDeSaude** abaixo, que representa um plano de saúde com o qual uma determinada clínica trabalha:

```
struct planoDeSaude
{    char nomeDoPlano[21];
     int  numDePacientes;  /* numero de pacientes com esse plano na clinica */
};
typedef struct planoDeSaude  PlanoDeSaude;
```

Considere também o tipo estruturado **Paciente** abaixo, que representa um paciente dessa mesma clínica:

```
struct paciente
{    char nome[51];
     int  idade;
     char plano[21];
};
typedef struct paciente Paciente;
```

2.a) (1.5 ponto) Escreva a função *criaNovoPaciente* que recebe dados de um paciente (um nome (cadeia de caracteres), uma idade (um inteiro) e um plano (cadeia de caracteres)), e cria, com alocação dinâmica de memória, um novo paciente, retornando-o (o seu endereço). Caso não seja possível criar o paciente, a função retorna NULL.

2.b) (3.0 pontos) Escreva a função *processaCadastroDosPlanos* que:

- recebe um vetor de estruturas do tipo PlanoDeSaude e o número de planos, um vetor de ponteiros para Paciente e o número de pacientes (se o número de pacientes for 7, teremos os 7 primeiros elementos desse vetor de ponteiros apontando pacientes válidos, ou seja, nenhuma das 7 posições terá valor NULL) . Inicialmente o campo numDePacientes de todos os planos do vetor de planos está zerado;

- a função tem como objetivo armazenar em cada um dos planos de saúde o número de pacientes desse plano. Para isso ela percorre o vetor de ponteiros para Paciente e, para cada paciente, busca no vetor de planos o plano correspondente, incrementando o número de pacientes desse plano. Caso um ou mais planos não sejam encontrados, após o processamento de todos os pacientes, a função retorna 0. Se o cadastro foi processado corretamente, a função retorna 1.

Por exemplo, para um vetor de planos com 4 planos “Amil”, “SulAmerica”, “Unimed” e “GoldenC”, e um vetor de ponteiros para os seguintes 10 pacientes:

{"Ana",50,”Amil”}, {"Paulo",55,”Amil”}, {"Abdala",30,”PlanoX”}, {"Beatrice",51,”SulAmerica”}, {"Braz",18,”GoldenC”}, {"Fred",75,”Unimed”}, {"Cid",80,”SulAmerica”}, {"Maria",30,”SulAmerica”}, {"Ada",15,”Unimed”}, {"Carlos",70,”Unimed”},

temos as seguintes quantidades de pacientes no vetor de planos atualizado:

{“Amil”,2}, {“SulAmerica”,3}, {“Unimed”,3} e {“GoldenC”,1}.

Neste caso, a função retorna 0 (porque o "PlanoX" do paciente "Abdala" não consta no vetor de planos).

2.c) (1.5 ponto) Escreva a função RECURSIVA *contaPlanosComMaisPacientes* que recebe os seguintes argumentos: um vetor de estruturas do tipo PlanoDeSaude, o número de planos e um limite de número de pacientes. Esta função retorna o número de planos com quantidade de pacientes maior do que o limite dado.
Por exemplo, para o vetor da questão anterior {"Amil",2}, {"SulAmerica",3}, {"Unimed",3} e {"GoldenC",1}, se o limite for 1, a resposta é 3, pois temos 3 planos com mais de 1 paciente , e no caso do limite ser 2, temos 2 planos com mais de 2 pacientes.

Protótipos de funções que podem ser úteis nesta prova:

stdio.h:

```
int scanf (char* formato, ...);
int printf (char* formato, ...);
FILE* fopen (char* nome, char* modo);
int fclose (FILE* fp);
int fscanf (FILE* fp, char* formato, ...);
int fprintf (FILE* fp, char* formato, ...);
char* fgets(char* str, int size, FILE* fp));
int sscanf(char* str, char* formato, ...);
```

stdlib.h:

```
void* malloc (int nbytes);
void* realloc (void *p, int nbytes);
void  free (void* p);
int atoi(char *s);
```

math.h:

```
double sqrt (double x);
double pow (double x, double exp);
double cos (double radianos);
double sin (double radianos);
```

string.h:

```
int strlen (char* s);
int strcmp (char* s, char *t);
char* strcpy (char* destino, char* fonte);
char* strcat (char* destino, char* fonte);
```

ctype.h:

```
int isdigit ( int c );
int toupper ( int c );
int tolower ( int c );
```