INF 1620 – P3 - 06/12/03	Questão 1
Nome:	
Matrícula:	Turma

Considere um cadastro com as médias por disciplina dos alunos de uma universidade. Esse cadastro é originalmente armazenado em um arquivo texto que contém, para cada disciplina, a lista dos alunos com suas respectivas médias. Nesse arquivo, cada disciplina é identificada por seu código alfanumérico, e seu código é seguido pelo número de alunos que cursaram a disciplina. Em seguida, são listados os nomes dos alunos com suas respectivas médias. O nome do aluno é representado por uma string delimitada por aspas simples. Um exemplo de um arquivo nesse formato é mostrado abaixo:

```
INF1001 2
  'fulano de tal'   7.3
  'sicrano silva'   8.7
CRE0700 2
  'beltrano alves' 10.0
  'fulano de tal'   9.2
INF1620 3
  'beltrano alves'  3.4
  'fulano de tal'   7.2
  'sicrano silva'   6.7
```

Escreva um programa completo que leia um arquivo "entrada.txt", no formato descrito acima, e crie um novo arquivo "saida.txt", composto por linhas com o código de uma disciplina, o nome de um aluno, e sua média na disciplina, na mesma ordem do arquivo de entrada. Para o exemplo acima, esse programa deve gerar o seguinte arquivo de saída:

```
INF1001 'fulano de tal'   7.3
INF1001 'sicrano silva'   8.7
CRE0700 'beltrano alves' 10.0
CRE0700 'fulano de tal'   9.2
INF1620 'beltrano alves'  3.4
INF1620 'fulano de tal'   7.2
INF1620 'sicrano silva'   6.7
```

Eventuais linhas em branco podem ocorrer no arquivo de entrada e devem ser desprezadas. Deve ser previsto o caso de um arquivo de entrada vazio ou inexistente. Se o arquivo for inexistente, não se deve gerar nenhum arquivo de saída. Se o arquivo de entrada for vazio, isto é, sem nenhum dado, a saída gerada deve ser também um arquivo vazio.

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 1620 – P3 - 06/12/03	Questão 2
Nome:	
Matrícula:	Turma

Considere um arquivo texto onde cada linha contém a matrícula de um aluno e seu respectivo CR, como ilustra o exemplo a seguir:

```
9010087 -2 7.3
8820324 -3 8.7
9210478 -5 9.2
9020256 -8 6.7
```

Implemente uma função que receba como parâmetros o nome de um arquivo no formato descrito acima e a matrícula de um aluno, e retorne o CR do aluno. Se não for encontrado no arquivo a matrícula procurada, a função deve retornar −1.0. Se não for possível abrir o arquivo, a função deve imprimir a mensagem "ERRO" e terminar a execução do programa. O protótipo dessa função deve ser:

```
float busca (char* arquivo, char* matricula);
```

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 1620 – P3 - 06/12/03	Questão 3
Nome:	
Matrícula:	Turma

Considere uma estrutura de árvore binária de busca definida pelo tipo abaixo:

```
struct arv {
   int info;
   struct arv* esq;
   struct arv* dir;
};
typedef struct arv Arv;
```

Tirando proveito da ordenação dos nós, escreva uma função que retorne quantos nós de uma árvore binária de busca armazenam valores contidos em um intervalo [x1, x2]. Essa função deve receber como parâmetros a árvore de busca e dois valores inteiros x1 e x2, de acordo com o protótipo definido a seguir:

```
int intervalo (Arv* a, int x1, int x2);
```

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 1620 – P3 - 06/12/03	Questão 4
Nome:	
Matrícula:	Turma

Considere um tipo que representa os dados de um aluno definido pela estrutura a seguir:

```
struct aluno {
   char nome[81];
   int matricula;
   float cr;
};
typedef struct aluno Aluno;
```

Usando alguma das técnicas de ordenação vistas no curso, escreva uma função para ordenar um vetor de alunos, em ordem decrescente de número de matrícula. A função deve receber como parâmetros o número de alunos e um vetor que armazena ponteiros para estruturas do tipo `Aluno`, de acordo com o protótipo definido a seguir:

```
void ordena (int n, Aluno** v);
```

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 1620 – P3 - 06/12/03	Questão 5
Nome:	
Matrícula:	Turma

Considere um cadastro de alunos armazenado em uma tabela de dispersão (*hash*) que usa listas encadeadas para o tratamento de colisões, isto é, a tabela é implementada como um vetor de listas encadeadas. Considerando as seguintes declarações relativas à tabela *hash* dos alunos:

```
struct aluno {
   char nome[81];
   int matricula;
   float cr;
   struct aluno* prox;
};
typedef struct aluno Aluno;

#define N 127
typedef Aluno* Hash[N];
```

Implemente:

a) Uma função de *hash* que, dado o número de matrícula do aluno, retorne o índice do aluno na tabela, de acordo com protótipo definido a seguir:

```
int hash (int mat);
```

Assumindo que os números de matrícula seguem o mesmo padrão utilizado na PUC, a função deve procurar a melhor dispersão dos alunos na tabela.

b) Uma função que insira um novo aluno na tabela *hash*. Para simplificar, considere que a função estará sempre adicionando um novo aluno ao cadastro, isto é, não existe a hipótese da matrícula do aluno a ser inserido já constar no cadastro. A função deve receber como parâmetros a tabela hash, o nome do aluno, sua matrícula e seu CR, e deve retornar um ponteiro para a estrutura criada para o novo aluno, de acordo com o protótipo definido a seguir:

```
Aluno* insere(Hash tab, char* nome, int mat, float cr);
```

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

RASCUNHO

Respostas nesta folha não serão consideradas.

Protótipos de funções que podem ser úteis:

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, ...);
```

math.h:

```
double sqrt (double x);
double pow (double x, double exp);
```

string.h:

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

sdtlib.h:

```
void* malloc (int nbytes);
void free (void* p);
void qsort (void *vet, int n, int tam, int (*comp) (const void*, const void*));
```

Não separe as folhas deste caderno.