INF 1620 – P3 - 22/11/02	Questão 1
Nome:	
Matrícula:	Turma:

Considere um arquivo texto com as notas dos alunos de uma disciplina. Cada linha do arquivo contém a matrícula de um aluno (cadeia de nove caracteres), seguida pelos valores de suas três notas (P1, P2 e P3). Considere ainda que não existem linhas em branco no arquivo. Um exemplo desse formato é mostrado abaixo.

```
9010087-2 2.0 4.3 6.5
8820324-3 7.0 8.2 8.6
9210478-5 6.0 7.5 7.8
9020256-8 3.0 0.5 4.2
```

Escreva um programa completo que leia o arquivo "entrada.txt", que contém as notas de uma disciplina no formato descrito acima, e escreva no arquivo texto "saida.txt" o número de matrícula de cada aluno com sua respectiva média ((P1+P2+P3)/3) e um caractere indicando se o aluno foi aprovado ou reprovado na disciplina (use 'A' para indicar aprovação e 'R' para indicar reprovação). Considere que um aluno está aprovado se tiver média maior ou igual a 5.0. Para o exemplo acima, o programa deve criar um arquivo "saida.txt" com o seguinte conteúdo:

```
9010087-2 4.3 R
8820324-3 7.9 A
9210478-5 7.1 A
9020256-8 2.6 R
```

Observe que as médias são escritas no arquivo "saida.txt" com apenas uma casa decimal. Se não for possível abrir um dos arquivos, o programa deve imprimir a mensagem "ERRO" e terminar sua execução

---

Protótipos de funções que podem ser úteis:

```
FILE* fopen (char* nome, char* modo);
int fclose (FILE* fp);
int fscanf (FILE* fp, char* formato, ...);
char* fgets (char* s, int n, FILE* fp);
int fgetc (FILE* fp);
```

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 - 22/11/02	Questão 2
Nome:	
Matrícula:	Turma:

Considere o tipo que representa um funcionário de uma empresa:

```
struct funcionario {
   char nome[81];     /* nome do funcionário */
   float valor_hora; /* valor da hora de trabalho em Reais */
   int horas_mes;     /* horas trabalhadas em um mês*/
};
typedef struct funcionario Funcionario;
```

Utilizando a função `qsort` da biblioteca padrão, escreva uma função para ordenar um vetor com os funcionários de uma empresa, em ordem crescente de salário (o salário de um funcionário é dado pelo número de horas trabalhadas vezes o valor da sua hora de trabalho). Se dois funcionários tiverem o mesmo salário, use o número de horas trabalhas no mês, em ordem decrescente, como critério de desempate. A função recebe como parâmetros o número de funcionários e um vetor que armazena ponteiros para estruturas com os dados dos funcionários da empresa. O protótipo da função deve ser:

```
void ordena (int n, Funcionario** vet);
```

Obs: a função `ordena` consistirá apenas da chamada da função `qsort`, fazendo uso de uma função auxiliar para comparar os elementos do vetor, que também deve ser escrita.

Protótipos de funções que podem ser úteis:

```
void qsort (void* vet, int n, int tam, int (*comp) (const void*, const void*));
```

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 - 22/11/02	Questão 3
Nome:	
Matrícula:	Turma:

Considere um vetor que armazena estruturas do tipo `Funcionario` (definida na questão anterior) ordenadas pelo nome do funcionário (ordem alfabética crescente). Sem usar funções da biblioteca padrão, escreva uma função que faça uma *busca binária* nesse vetor. Essa função recebe como parâmetros de entrada o número de funcionários, o vetor e o nome do funcionário que se deseja buscar. A função deve ter como valor de retorno o índice do elemento do vetor que armazena os dados do funcionário com o nome fornecido. Se não existir um funcionário com o nome fornecido, a função deve retornar –1. O protótipo da função deve ser:

```
int busca (int n, Funcionario* vet, char* nome);
```

Protótipos de funções que podem ser úteis:

```
int strcmp (char* s, char *t);
```

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 - 22/11/02	Questão 4
Nome:	
Matrícula:	Turma:

Considere uma árvore binária de busca que armazena ponteiros para estruturas do tipo `Funcionario`, que foi definido na segunda questão. O tipo que representa um nó da arvore é dado por:

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

Sabendo-se que numa árvore binária de busca as informações associadas aos nós da sub-árvore à esquerda são menores (no caso, ordem alfabética dos nomes dos funcionários) que a informação associada à raiz e que as informações dos nós da sub-árvore à direita são maiores, escreva uma função que procure um nome na árvore de busca. A função recebe como parâmetros a raiz da árvore e o nome a ser procurado, e deve ter como retorno o ponteiro para a estrutura com os dados do funcionário procurado, ou `NULL` caso o nome não seja encontrado. O protótipo dessa função deve ser:

```
Funcionario* busca (Arv* a, char* s);
```

Protótipos de funções que podem ser úteis:

```
int strcmp (char* s, char *t);
```

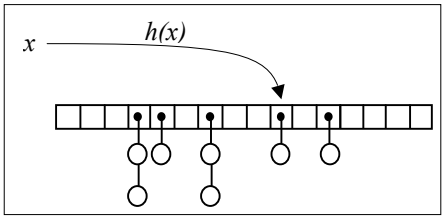
Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1620 – P3 - 22/11/02	Questão 5
Nome:	
Matrícula:	Turma:

Considere um cadastro de alunos armazenado numa tabela de dispersão (*hash*) que usa como chave de busca o nome do aluno. Considere também a existência de uma função de *hash* que, dado o nome do aluno, retorna o seu índice na tabela. O protótipo dessa função é dado por:

```
int hash (char* nome);
```

O tratamento de colisão é feito utilizando-se listas encadeadas para armazenar os elementos que colidem. Dessa forma, cada posição na tabela representa um ponteiro para o primeiro nó de uma lista encadeada, conforme ilustra a figura ao lado.



Considere as seguintes declarações, relativas à tabela de *hash* dos alunos:

```
struct aluno {
   char* nome;            /* nome do aluno */
   float cr;              /* coeficiente de rendimento */
   struct no* prox;       /* próximo elemento da lista*/
};
typedef struct aluno Aluno;

#define N 200
typedef Aluno* Hash[N];
```

Considerando que posições não preenchidas da tabela de *hash* armazenam o valor `NULL` (lista vazia), escreva uma função que insira um novo aluno na tabela de *hash*. A função recebe como parâmetros a tabela de *hash*, o nome do aluno e o seu coeficiente de rendimento (CR), e deve retornar um ponteiro para a estrutura `Aluno` que foi criada para armazenar os dados do novo aluno. O protótipo da função deve ser:

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

Obs: Para simplificar, considere que a função estará sempre adicionando um novo aluno ao cadastro, isto é, não existe a hipótese do nome do aluno a ser inserido já constar no cadastro.

Protótipos de funções que podem ser úteis:

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

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.

Não separe as folhas deste caderno.