DCC–IME–USP

Carlos Eduardo Ferreira
Sala 297A Tel.: 3818 6140
e-mail cef@ime.usp.br

José Coelho de Pina
Sala 164B Tel.: 3818 6295
e-mail coelho@ime.usp.br

MAC-115 – Introdução à Computação

Instituto de Química - Primeiro Semestre de 2000

Exercício-Programa de Recuperação — Entrega: **15 de agosto de 2000**

Fazendo média ☺

Todo fim de semestre cada professor tem que calcular a média final das suas turmas. Este ano, o Departamento de Ciência da Computação da USP resolveu uniformizar este processo, contratando vocês para implementarem um programa para o cálculo das médias.

Cada dois de vocês deve escrever um programa em `C` que leia de um arquivo os dados de uma turma e imprima a média final de acordo com o critério também descrito no arquivo de entrada.

No arquivo de entrada estão os seguintes dados, nesta ordem:

- três inteiros n, p e m, representando respectivamente o número de alunos da turma, o número de notas de provas e o número de notas de exercícios-programa;
- uma seqüência de p naturais representando o peso de cada prova;
- uma seqüência de m naturais representando o peso de cada exercício-programa;
- dois naturais pp e pep representando o peso da média de provas e da média de exercícios-programa para o cálculo da média final;
- uma seqüência de n linhas, cada uma com os dados de um dos n alunos da turma: nome, notas de provas e notas de exercícios-programa.

Vocês podem assumir que o nome de cada aluno contém no máximo 30 caracteres. Mais especificamente vocês podem assumir que os 30 primeiros caracteres de cada linha com os dados de um aluno contêm um nome (e eventuais brancos depois) e na coluna 31 começam as notas de provas deste aluno. Cada nota deve ser um número real entre 0 e 10.

Exemplo de arquivo de entrada:

```
3 3 5
1 2 2
1 1 2 2 3
1 2
Cassandra Avestruz da Silva   5.0  6.6  4.2 10.0  9.0  8.0  7.0  6.0
Roberto Marinho               2.0  1.5  3.2  0.0  5.0  0.0  6.0  6.5
Ronaldinho Gaúcho             8.0  3.0  7.7  9.5  8.0 10.0  9.5  9.0
```

O seu programa deve pedir que o usuário digite o nome do arquivo de entrada e deve ter como saída um arquivo de nome `"saida.txt"` com

- uma linha inicial com os rótulos das várias colunas a serem impressas (veja no exemplo abaixo);
- uma linha para cada aluno, contendo o seu nome, as suas notas de prova, a sua média de prova usando os pesos dados no arquivo de entrada, as suas notas dos exercícios-programa, a sua média de exercício-programa usando os pesos dados no arquivo de entrada e finalmente a sua média final;
- uma linha contendo a média da turma em cada prova, a média das médias das provas, a média de cada exercício-programa, a média das médias dos exercícios-programa e a média das médias finais;
- no final, o número de alunos aprovados (com média final maior ou igual a 5.0), o número de alunos que ficaram de recuperação (com média final maior ou igual a 3.0 e menor que 5.0) e o número de reprovados (com média final menor que 3.0).

Exemplo de saída para o arquivo de entrada acima:

```
Nome                            p1   p2   p3   mp  ep1  ep2  ep3  ep4  ep5  mep   mf
Cassandra Avestruz da Silva    5.0  6.6  4.2  5.3 10.0  9.0  8.0  7.0  6.0  8.0  7.1
Roberto Marinho                2.0  1.5  3.2  2.2  0.0  5.0  0.0  6.0  6.5  3.5  3.1
Ronaldinho Gaúcho              8.0  3.0  7.7  6.2  9.5  8.0 10.0  9.5  9.0  9.2  8.2
Medias                         5.0  3.7  5.0  4.6  6.5  7.3  6.0  7.5  7.2  6.9  6.1

Aprovados:    2
Recuperacao:  1
Reprovados:   0
```

Leitura e Gravação de um Arquivo

Para fazer a leitura de um arquivo de entrada e a gravação da saída em um arquivo, utilize a seguinte receita no seu programa, dentro da função `main()`:

```
/* Declaração das variáveis para leitura e gravação em arquivos */
char nome_arq_entrada[40]; /* para o nome do arquivo de entrada */
FILE *entrada, *saida;

/* Primeiros comandos do seu programa */
/* Abertura do arquivo de entrada */
printf("Digite o nome do arquivo de entrada: ");
scanf("%s", nome_arq_entrada);
if ((entrada = fopen(nome_arq_entrada, "r")) == NULL) {
  printf("Arquivo de entrada nao encontrado!\n");
  exit(0);
}
/* Abertura do arquivo de saída */
if ((saida = fopen("saida.txt", "w")) == NULL) {
  printf("Erro na abertura do arquivo de saida!\n");
  exit(0);
}
```

No restante do seu programa, toda vez que você quiser ler alguma coisa do arquivo de entrada, utilize a função `fscanf` e para escrever no arquivo de saída, utilize a função `fprintf`. Estas funções funcionam de forma semelhante ao `scanf` e ao `printf`, exceto que elas possuem um parâmetro a mais: o nome do arquivo onde está sendo feita a leitura ou a gravação. Sinta-se a vontade de imprimir tanto no arquivo (com o `fprintf`) quanto na tela (com o `printf`): isso pode lhe ajudar na fase de teste do seu programa, por exemplo na verificação de que a leitura do arquivo está funcionando corretamente, etc.

Para fazer a leitura do nome, utilize a função `fgets`, que tem como parâmetros (nesta ordem) o nome da *string* onde será guardada a *string* lida, o número de caracteres a serem lidos (no nosso caso, 30 – lembre-se de utilizar o comando `#define MAX_NOME 30` no início do programa) e o nome do arquivo de onde a leitura deve ser feita.

Exemplos:

```
/* Lendo do arquivo de entrada */
fscanf(entrada, "%d %d %d ", &n, &p, &m);

/* Verificando os valores que foram lidos - impressão na tela */
printf("n = %d   p = %d   m = %d\n", n, p, m);

/* Leitura de um nome do arquivo de entrada */
fgets(nome, MAX_NOME, entrada);

/* Gravando no arquivo de saida */
fprintf(saida, "%s %4.1f\n", nome, mf);
```

Observações

1. Você mesmo pode criar o seu arquivo de entrada usando o próprio `djgpp` como um editor. Abra um arquivo com nome, por exemplo, `notas.txt` e digite os dados do exemplo acima. Salve o seu arquivo (não execute, pois isso não é um programa em `C`!). Volte ao arquivo do seu EP para digitar ou executar o seu programa.

2. Exercícios-programa com erro de sintaxe ou copiados recebem nota ZERO.

3. Não serão aceitos exercícios atrasados, portanto comece cedo a fazer o seu EP!

4. O seu programa deve estar bem documentado (com comentários explicando o que guarda cada variável e o que faz cada trecho de programa), bem indentado (o espaçamento em relação a primeira coluna deve destacar a estrutura dos comandos de repetição e seleção como exemplificado nos exercícios de simulação do Caderno de Exercícios) e o mais claro possível. Isto será levado em conta na sua nota.

5. O seu programa não precisa fazer consistência de dados, ou seja, não precisa verificar se os valores dados satisfazem as restrições que impussemos. Mais especificamente, você não precisa verificar se os nomes de fato tem no máximo 30 caracteres, se as notas de fato estão entre 0 e 10, etc. Assuma que o arquivo satisfaz estas restrições.

6. **IMPORTANTE!!!!!!!!** Para não ter problemas na leitura dos dados, deixe sempre um espaço em branco depois do formato de leitura de cada número. Isso é essencial para você não ter problemas com a leitura dos nomes. Assim, escreva

   `fscanf(entrada, "%d %d %d ", &n, &p, &m);`

   — note o branco depois do último `%d` — ao invés de

   `fscanf(entrada, "%d %d %d", &n, &p, &m);`.