Departamento de Ciência de Computadores FCUP

Introdução à Programação (CC111) 2011/2012

Exame (11.02.2012) *duração: 3 horas*

(Cotação: 2+2, 1, 2.5+1+1.5, 2+2+1+3, 1+1)

Nas respostas, pode usar funções definidas no enunciado mesmo que não as tenha implementado. As perguntas 4 e 5 são resolvidas numa folha de exame (p.f. solicite-a, se necessário).

1. Pretende-se um programa completo para ler **dois** nomes da entrada padrão, cada um dado numa linha, e imprimi-los por ordem alfabética. Admita que cada linha tem um nome bem formado e que o nome tem menos de `MAXN` caracteres (devendo arbitrar um valor para a constante). No programa deve usar as funções seguintes:

- `int compara_nomes(char x[],char y[])` para comparar dois nomes, representados por sequências de caracteres bem formadas dadas por `x` e `y`, e retornar `-1` se o primeiro preceder o segundo na ordem alfabética (sem distinguir letras minúsculas de maiúsculas), `0` se forem iguais, e `1` se o segundo preceder o primeiro. Os nomes podem conter espaços (apenas um entre cada duas palavras) e letras maiúsculas e minúsculas. O código do espaço é inferior ao de qualquer letra.
- `int ler_nome(char nome[])` que, fazendo uso da função `getchar()`, leia para o vetor `nome` uma sequência de caracteres terminada por fim de input ou pelo caracter mudança de linha (caracter que já não guardará em `nome`). Deve ignorar todos os espaços e mudanças de linha que possam ocorrer no início, verificar que os restantes caracteres são letras ou espaços, e manter apenas um espaço entre cada duas palavras se for introduzido mais do que um (e nenhum espaço no fim). No fim, deve colocar o terminador convencional para *strings* em C. A função retornará 1 se o nome estiver bem formado e 0 caso contrário. Admita que o vector tem espaço suficiente para guardar a sequência final. Não poderá fazer uso de outras funções de sistema além de `getchar()`.

No programa (completo) deve implementar **apenas uma dessas duas funções** e incluir comentários para indicar o local em que inseriria a definição da outra.

(CONTINUA, v.p.f.)

```
// continuação da resposta 1.
```

2. Seja `y` uma variável do tipo `char`. Quais das dez expressões `'0'`, `"0"`, `'\0'`, `0`, `'5'`, `'5'-'0'`, `'D'-'A'`, `"D"`, `"DA"` e `48` podem substituir β na atribuição `y=`β`;` e que valor ficaria em `y` em cada caso? Justifique (também os casos negativos).

(CONTINUA, v.p.f.)

CC111 Exame N.º _____ Nome _____

3. Pretende-se analisar o percurso realizado por um veículo de uma empresa de distribuição para determinar quantas vezes esteve na empresa num certo dia (contando também o início e fim do trajeto se for na empresa). Considera-se que o percurso foi realizado numa região plana, sendo a empresa representada por um ponto. O veículo pode partir da empresa ou de um outro local. São dadas as coordenadas da empresa, as coordenadas do ponto de partida, e a seguir uma sequência de pares de inteiros `dx dy` terminada por `0 0`. Cada par `(dx,dy)` corresponde a um deslocamento entre duas paragens consecutivas e será sempre não nulo (o par `0 0` é apenas terminador mas existirá sempre pelo menos um deslocamento). Pretende-se também indicar se o percurso analisado terminou no local de partida ou não, e se não tiver terminado, indicar onde terminou (como se ilustra abaixo).

Exemplo 1
Input

```
1 1 -1 3 3 -2 3 3 -6 -1 5 -1 -2 0 -5 0 2 1 0 0
```

Output

```
#vezes empresa: 0
Terminou no local de partida
```

Exemplo 2
Input

```
0 0 0 0 3 -2 3 3 -6 -1 5 -1 -2 0 -5 0 0 0
```

Output

```
#vezes empresa: 2
Terminou em (-2,1)
```

Implemente o programa sem impor quaisquer restrições adicionais. Indique a interpretação de cada variável que definiu. Inclua comentários que expliquem o algoritmo implementado e permitam justificar a correção do programa (assumindo que não existem erros de *overflow* ou de *underflow*).

(CONTINUA, v.p.f.)

CC111 Exame N.º _____ Nome _____

4. Nas alíneas seguintes supõe-se que a matriz `char nomes[][MAXN]` guardará nomes de alunos de um curso e que `int dnotas[][MAXD]` guardará as suas classificações a `n` disciplinas do plano de estudos, com `n` $\leq$ `MAXD`. Cada coluna da matriz `dnotas` tem notas de uma mesma disciplina e `-1` indica que o aluno não realizou a disciplina. Por exemplo, para sete disciplinas (`n=7`) poderia ter os valores representados à esquerda.

```
nomes[0][] = "Joao Diogo Matias"    dnotas[0][] = {12,-1,12,11,10,-1,-1}
nomes[1][] = "Jorge Maria Pereira"  dnotas[1][] = {16,17,18,16,18,-1,-1}
nomes[2][] = "Antonia Faria"        dnotas[2][] = {16,10,14,17,-1,15,16}
nomes[3][] = "Maria Dinis"          dnotas[3][] = {15,15,18,15,-1,15,-1}
nomes[4][] = "Martim Braga Sousa"   dnotas[4][] = {-1,-1,-1,-1,-1,-1,-1}
nomes[5][] = "Rui Simao Vieira"     dnotas[5][] = {20,10,-1,-1,-1,-1,-1}
```

```
                  #Alunos
------------------
Excelente     1
Muito Bom     2
Bom           1
Suficiente    1
------------------
Nao realizou  1
```

Implemente as funções seguintes:

a) `void resultados_disc(int disc,int na,int dnotas[][MAXD])` para produzir uma tabela da distribuição de classificações relativas à disciplina `disc` (identificada pelo índice de coluna), segundo níveis qualitativos definidos por `18-20 Excelente`, `16-17 Muito Bom`, `14-15 Bom`, e `10-13 Suficiente`, e ainda `-1 Nao realizou`. Acima, à direita, encontra a tabela que se obteria para os dados apresentados se `ndisc` fosse zero. Deve declarar e usar duas variáveis locais `int contadores[5];` e `char *Niveis[5] =` {`"Excelente","Muito Bom","Bom","Suficiente","Nao realizou"`};, entre outras.

b) `int melhores_notas(int disc,int na,int dnotas[][MAXD],int ipos[])` para determinar os identificadores dos alunos que obtiveram a melhor nota na disciplina `disc`. Retornará o número total de alunos que obtiveram tal nota e colocará os seus identificadores (índices de linha) no vetor `ipos`. Para o exemplo acima, com `ndisc = 2`, retornaria `2` e colocaria `1` e `3` no vetor.

c) a função principal de um programa que, para cada disciplina, escreve os nomes dos alunos que obtiveram a melhor nota. O formato de saída deve permitir distinguir os resultados obtidos para cada disciplina. Os dados são lidos da entrada padrão, e deve usar a função definida em b) e `void carrega(int na,char nomes[][MAXN],int nd,int dnotas[][MAXD])`, que se supõe já implementada, e que serve para ler e guardar em memória os nomes de `na` alunos e as suas classificações a `nd` disciplinas. Declare as matrizes supondo que o número de alunos não excede 100.

d) `void estatisticas(int na,int nd,int dnotas[][MAXD],STATS stats[])` para determinar o número de disciplinas realizadas por cada aluno, e, se tiver realizado alguma disciplina, a média e desvio padrão das suas notas. Comece por declarar o tipo `STATS` para uma estrutura com três campos que permita guardar esses três valores (um inteiro e dois em vírgula flutuante e precisão dupla). Recorde que a função `double sqrt(double a)` de `math.h` determina $\sqrt{\texttt{a}}$ e que média μ e o desvio padrão σ de k valores $x_1, \ldots, x_k$ são definidos por $\mu = \dfrac{x_1 + \cdots + x_k}{k}$ e $\sigma = \sqrt{\dfrac{(x_1-\mu)^2 + \cdots + (x_k-\mu)^2}{k}}$.

5. Apresente as ideias principais dos algoritmos de ordenação por seleção, ordenação por inserção e método da bolha, destacando o que os distingue (na ordenação de um vetor por ordem crescente). Na continuação de 4., baseando-se num desses métodos, implemente

`int obter_ordem(int na,char nomes[][MAXN],STATS stats[],int nconc,int ordem[])`

para colocar em `ordem[]` os identificadores dos alunos que realizaram pelo menos `nconc` disciplinas, de modo a traduzir uma ordenação de tais alunos por *média arredondada às unidades* (dada por `(int)` $(\mu + 0.5)$) e, em caso de empate, por *ordem alfabética*. A função retorna o número de alunos nessas condições. Supõe-se que o vetor `stats` contém os valores determinados em 6d) e que `nconc` é positivo. Note que `nomes[i]` equivale a `&nomes[i][0]` e que os elementos da mesma linha da matriz se encontram em posições consecutivas. Use a função definida no problema 1. (FIM)