Prova 2

Algoritmos e Estruturas de Dados I - turma TF

Professor: Pedro O.S. Vaz de Melo

29 de outubro de 2013

Nome: _____

escrevendo o meu nome eu juro que seguirei o código de honra

Código de Honra para este exame (baseado no *Honor Code* da Universidade de Stanford):

- Não darei ajuda a outros colegas durante os exames, nem lhes pedirei ajuda;
- não copiarei nem deixarei que um colega copie de mim;
- não usarei no exame elementos de consulta não autorizados.

Informações importantes:

- Em questões que pede um **programa**, este deve ser completo, com bibliotecas (incluindo, quando necessário, a biblioteca `prova2.h`), função main, etc. Se deve ser feita uma **função**, somente a função é suficiente. Se deve ser feito um **procedimento**, somente o procedimento é suficiente.
- A interpretação das questões da prova faz parte do critério de avaliação. Caso tenha dúvida sobre a sua interpretação de uma determinada questão, escreva as suas suposições na resolução da mesma.

Referências:

Função/Operador	Descrição	Exemplo
`printf("%05d", a)`	imprime zeros à esquerda da variável a até completar 5 caracteres impressos	se a==12, imprime 00012
`%`	retorna o resto da divisão	`20 % 3` retorna 2

1. (*7 points*) Uma rede social de amizades pode ser representada por uma matriz de adjacência $n \times n$ de n colunas e n linhas. Cada linha (ou coluna) i contém as relações da pessoa n_i. Considere a matriz de adjacência abaixo:

id	n_0	n_1	n_2	n_3	n_4
n_0	0	1	1	0	1
n_1	1	0	0	1	0
n_2	1	0	0	0	0
n_3	0	1	0	0	1
n_4	1	0	0	1	0

Esta matriz representa uma rede social entre 5 pessoas: n_0, n_1, n_2, n_3 e n_4. Além disso, quando a posição (i, j) da matriz é 1, então as pessoas n_i e n_j são amigas entre si. Caso a posição (i, j) da matriz é 0, então n_i e n_j não são amigas. Observe que a pessoa n_0 é amiga das pessoas n_1, n_2 e n_4, mas não é amiga da pessoa n_3.
Assim, implemente uma **função** que recebe uma matriz de adjacência M e o número de pessoas n contidas nela e que retorna o **identificador da pessoa** com maior número de amigos. Na matriz exemplo, a sua função deve retornar 0, que é o identificador da pessoa com maior número de amigos. Considere que essa pessoa será sempre única. Além disso, considere que existe uma definição para o número máximo de pessoas que a matriz comporta, chamado `MAX_PESSOAS`. O protótipo dessa função deve ser:

```
int pessoaMaisPopular(int M[ ][MAX_PESSOAS], int n);
```

2. (*13 points*)

a. (*2 pts*) Defina um novo tipo de dados para armazenar uma data válida. Essa estrutura deve conter os campos `dia`, `mes` e `ano`.

b. (*6 pts*) Escreva uma função de nome `dataValida` que recebe uma `Data` como parâmetro e retorna 1 se a data for válida ou 0 se ela for inválida. Lembrando que fevereiro pode ter 29 dias em anos bissextos. Um ano é bissexto quando ele é múltiplo de 400 (exemplo: ano 2000) ou então quando ele é múltiplo de 4 mas não é múltiplo de 100 (exemplo: 2004 é bissexto mas 2100 não é).

c. (*3 pts*) Escreva um procedimento de nome `leData` que recebe uma `Data` por passagem de parâmetro por referência e que lê os seus campos do teclado. Você deve pedir para o usuário reinserir os campos caso a data lida não seja válida. Conside que a função `dataValida` está implementada corretamente.

d. (*2 pts*) Escreva um programa para ler uma data válida e imprimir essa data no formato "aaaa-mm-dd" (Exemplo: 2013-09-01). Conside que a função `leData` está implementada corretamente.

3. (*6 points*) Escreva um procedimento RECURSIVO de nome `imprimeNaturais` para imprimir todos os números naturais de 0 até n em ordem decrescente. O procedimento não deve fazer uso de estruturas de repetição (while, for etc) nem de variáveis globais.