Prova 2

Algoritmos e Estruturas de Dados I - turma TE

Professor: Pedro O.S. Vaz de Melo

9 de maio de 2014

Nome: _____

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

Código de Honra para este exame:

- 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
`rand()`	gera um número aleatório inteiro entre 0 e `RAND_MAX`	rand() pode gerar 41
`RAND_MAX`	o maior número possível que pode ser gerado por `rand()`	`RAND_MAX = 32767`

1. (*13 points*) Uma empresa o contratou para desenvolver um sistema social baseado em localizações. Assim, faça os exercícios abaixo:

a. (*3 pts*) Implemente os seguintes novos tipos de dados:

- `Local`, que tem os campos `idLocal` (um número inteiro maior ou igual a zero) e `categoria` (0 = restaurante, 1 = atração ao ar livre, 2 = casa de shows).
- `Data`, que tem os campos `dia`, `mes` e `ano`.
- `Checkin`, que tem os campos `localCheckin` (do tipo `Local`), `dataCheckin` (do tipo `Data`) e codigoUsuario (**um número inteiro maior ou igual a zero**).

b. (*4 pts*) Implemente uma função de nome `randCheckin` que recebe um `Checkin` como parâmetro (**por referência**) e preenche todos os seus campos com valores aleatórios. Considere que o a data do `Checkin` pode ser de 01/01/2012 a 30/12/2013. Para simplificar, considere que todos os meses têm 30 dias.

c. (*6 pts*) Implemente uma função de nome `numPessoasDistintas` que recebe um vetor de `Chekins` e a quantidade n de elementos desse vetor. Essa função deve retornar o número de pessoas distintas que visitaram restaurantes no mês de maio de 2013. Considere que o número máximo de usuários suportado pelo sistema é 100000.

2. (*3 points*) Escreva um programa para ler as informações de 10000 *Checkins* aleatórios e imprimir o número de pessoas distintas que visitaram restaurantes no mês de maio de 2013.

3. (*6 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 (já preenchida) e o número de pessoas n contidas nela e que retorna 1 se há nessa rede social uma pessoa sem amigos e 0 caso contrário. Na matriz exemplo, a sua função deve retornar 0, pois todas as pessoas tem amigos. Além disso, considere que o número máximo de usuários suportado pelo sistema é 100000 (este número só é necessário para descrever a matriz como parâmetro). O protótipo dessa função deve ser:

```
int existeIsolados(int M[ ][100000], int n);
```

4. (*6 points*) Escreva uma função RECURSIVA de nome `somaNaturais` que retorna a soma de todos os números naturais de 1 até n. O procedimento não deve fazer uso de estruturas de repetição (while, for etc) nem de variáveis globais. O protótipo dessa função deve ser:

```
int somaNaturais(int n);
```