DEMAC – UNESP, Rio Claro
Projeto de Estruturas de Dados I – Noturno

SIMULADOR SIMPLIFICADO DE UM SISTEMA DE BANCO

O projeto abaixo visa a desenvolver algoritmos relacionados aos assuntos subprogramas/ registros/ *strings*/ árvores, e os correspondentes programas em C devidamente documentados. Data de entrega: até o dia **26/06/2014**, no Portfólio do grupo no TelEduc.

Descrição do Projeto: o BANCO de crédito "NOME A ESCOLHER" deseja informatizar o atendimento da sua agência e pediu para os alunos (futuros profissionais) da UNESP-Rio Claro do curso de Bacharelado em Ciência da Computação-Noturno para desenvolver um sistema que seja capaz de:

i) cadastrar uma conta corrente;
ii) excluir uma conta corrente;
iii) alterar os dados da conta corrente;
iv) consultar dados da conta corrente;
v) realizar depósito em conta corrente;
vi) efetivar saque em conta corrente;
vii) consultar saldos;
viii) realizar a transferência entre contas correntes;
ix) consultar os dados da última movimentação.

É importante esclarecer que qualquer funcionário do banco pode realizar todas as operações descritas acima, e que um cliente pode, em algum caixa eletrônico, realizar somente as operações v), vi), vii), viii) e ix).

A ficha para cadastro de conta corrente é:

Nro. da conta corrente:	Nome:	Senha:
CPF:	RG:	
Data de nascimento:	Data de abertura:	
Depósito inicial:	Tipo da conta:	
Saldo atual:	Data da última movimentação:	
Tipo da movimentação:	Valor da movimentação:	

Ficha Conta Corrente

Sendo:

Nro. da conta corrente: número da conta corrente do cliente cadastrado (formato: inteiro);
Nome: nome do cliente (formato: caracter com 40 posições);
Senha: senha do cliente (formato: caracter com 6 posições);
CPF: cadastro de pessoa física (formato: caracter com 12 posições);
RG: registro geral (formato: caracter com 10 posições);
Data de nascimento: data de nascimento do cliente (formato: caracter com 10 posições);
Data de abertura: data em que o cliente realizou a abertura da conta corrente (formato: caracter com 10 posições);
Depósito inicial: valor inicial de depósito (formato: real);
Tipo da conta: pode ser especial (E), quando o depósito inicial for superior a R$12.000,00, e comum (C), caso contrário (formato: 1 caracter);
Saldo atual: valor do saldo atual (formato: real);
Data da última movimentação: data da última movimentação realizada pelo cliente sobre a conta corrente (formato: caracter com 10 posições);
Tipo da movimentação: tipo da última movimentação (D – débito; C – crédito) (formato: 1 caracter);

Valor da movimentação: valor da última movimentação (formato: real).

Baseado nas informações fornecidas, desenvolver um programa que utiliza uma **árvore binária de busca balanceada (AVL)** para armazenar as informações de contas correntes, tendo por chave primária o número da conta corrente. Os números de conta corrente devem ser gerados seqüencialmente no intervalo de 1 a 999.

a) Definir o registro de conta corrente ***regCCorrente*** e declarar a variável tipo ponteiro **arvCCorrente** (que apontará para a raiz da árvore).

b) Fazer uma função (chamada *leCCorrente*) capaz de realizar a leitura dos dados, a partir do teclado, de uma conta corrente a ser cadastrada, retornando um ponteiro para o registro de conta corrente lido (nó). Parâmetros da função: nenhum.

c) Construir uma função (chamada *obtemCCorrente*) que seja capaz de retornar um ponteiro para um registro de conta corrente (nó) a partir do seu número de conta corrente. Caso o registro (nó) não exista, retornar **NULL**. Parâmetros: número da conta corrente e ponteiro para raiz da árvore.

d) Desenvolver uma função (chamada *proximaCCorrente*) que seja capaz de encontrar o próximo número de conta corrente disponível para cadastro de um novo correntista. Deve retornar o próximo número encontrado, ou –1 caso não exista mais contas. Se algum número de conta já foi excluído, deve ser reaproveitado para a abertura de uma nova conta corrente. Parâmetro: ponteiro para raiz da árvore.

e) Construir um procedimento (chamado *incluiCCorrente*) capaz de incluir uma nova conta corrente. Para isto utilizar a função do item d) para encontrar o número da próxima conta corrente e a função de leitura do item b). Caso não haja nenhuma posição livre exibir a mensagem "Impossível cadastrar mais clientes". Parâmetro: ponteiro para raiz da árvore.

f) Fazer um procedimento (chamado *alteraCCorrente*) que, fornecida uma conta corrente válida, possa alterar os seguintes dados de uma conta corrente: *Nome, Depósito Inicial, Tipo da Conta e Senha*. Permitir ao usuário selecionar o item a ser alterado e realizar a operação. Parâmetros: número da conta corrente e ponteiro para raiz da árvore.

g) Construir um procedimento (chamado *depositoCCorrente*) capaz de realizar um depósito. Parâmetros: número da conta corrente; ponteiro para raiz da árvore e valor do depósito.

h) Desenvolver uma função (chamada *saqueCCorrente*) capaz de realizar um saque na conta corrente. Se o valor exceder o saldo atual e a conta corrente é comum, retornar Falso (indicando saque não efetuado) e não realizar o saque. Se a conta corrente for especial e o saldo, após o saque, for inferior a R$ -1.500,00, retornar Falso (0) e não efetivar o saque, caso contrário, efetivar o saque e retornar Verdadeiro (1). Parâmetros: número da conta corrente; ponteiro para raiz da árvore e valor do saque.

i) Desenvolver uma função (chamada *saldoAtualCCorrente*) que retorne o saldo atual para uma determinada conta corrente. Parâmetros: número da conta corrente e ponteiro para raiz da árvore.

j) Construir um procedimento (chamado *consultaUltimaMovimentacao*) que consulta e mostra os dados da última movimentação da conta corrente seguindo o formato apresentado a seguir. Parâmetros: número da conta corrente e ponteiro para raiz da árvore.

k) Fazer um procedimento (chamado *transfereValor*) que realiza a transferência entre contas correntes. Nesta função, observe que podem ser utilizadas as funções de saque e o procedimento de depósito desenvolvidos nos itens h) e g). Parâmetros: números das contas correntes de origem e destino; valor da transferência e ponteiro para raiz da árvore.

Banco “NOME A ESCOLHER”
Rua Sancho Pança, 430 – Mancha – CEP: 01506-870 – São Paulo, SP

Conta Corrente: 9999 – Tipo: (Especial ou Comum)
Data atual: DD/MM/AAAA
Nome do cliente: XXXXX... X
CPF: XXXXX... X RG: XXXXX... X

Saldo anterior: R$ 99999,99

Data Ult. Mov	Tipo Operação	Valor Movim.
DD/MM/AAAA	D/C	R$ 99999,99

Saldo atual: R$ 999999,99

l) Fazer um procedimento (chamado *excluiCCorrente*) que seja capaz de excluir uma conta corrente. A exclusão é realizada removendo-se o nó que possui o número da conta. Parâmetros: número da conta corrente e ponteiro para raiz da árvore.

m) Construir um programa que seja capaz de gerenciar o BANCO utilizando as funções e procedimentos desenvolvidos nos itens anteriores. Sabe-se que:

- inicialmente o usuário deve informar o número da conta corrente;
- o programa é finalizado quando o número da conta corrente fornecido for igual a –1;
- conta corrente 0 significa funcionário do banco habilitando todas as funções do projeto. A senha para o funcionário, inicialmente, é 12345. Um correntista do banco fornece um número de conta corrente válido/ senha correta o que disponibiliza as operações v), vi), vii), viii) e ix), definidas no início do projeto;

Obs.: preocupe-se com a interface desenvolvida para o usuário no programa C e documentação.