6º Laboratório de ECOP12 – Estrutura de Dados

Profª. Thatyana de Faria Piola Seraphim

Crie um arquivo chamado ***ecop12-lab06.c***, que será utilizado para implementar as funções para o laboratório de pilha. Esse arquivo que deverá ser postado no SIGAA para contabilizar a presença da disciplina ECOP12.

O objetivo do laboratório é implementar um simulador de caixa eletrônico usando pilhas dinâmicas encadeadas. As cédulas monetárias são: R$1, R$2, R$5, R$10, R$20, R$50 e R$100.

A variável ***pilha*** é definida como global para todas as funções desse programa, sendo que representa 7 pilhas de cédulas. Por convenção: pilha[0] armazena notas de R$1; pilha[1] armazena notas de R$2; pilha[2] armazena notas de R$5; pilha[3] armazena notas de R$10; pilha[4] armazena notas de R$20; pilha[5] armazena notas de R$50; pilha[6] armazena notas de R$100. Cada elemento nas pilhas representa uma única cédula monetária do tipo `noCedula`.

Abaixo são dadas algumas definições que serão usadas no programa.

```
#include <stdio.h>
#include <stdlib.h>

typedef enum{false, true} bool;

typedef struct no{
  int valor;
  struct no *prox;
}noCedula; //estrutura das cedulas

noCedula *pilha[7]; //pilhas cedulas

int valorNota[7] = {1,2,5,10,20,50,100};
```

a) Faça uma **função** chamada **inicializaPilha** que não recebe parâmetros e não retorna nada. Essa função deve inicializar cada elemento do vetor chamado pilha com o valor NULL.

b) A função **abastecePilha** adiciona cédulas nas pilhas correspondentes ao seu valor. Para isso, essa função não retorna nada e recebe como parâmetros: o valor de uma cédula (representado pela variável v) e a quantidade de cédulas que serão inseridas (representada pela variável `quant`). Complete a **função** abaixo **void abastecePilha(int quant, int v)** para que as cédulas possam ser inseridas na pilha correspondente.

```
void abastecePilha(int quant, int v){
  int i;
  int pos; //posicao da pilha a ser inserido o valor
  noCedula *novo;
  switch(valor){
    case 1:  pos=0; break;
    case 2:  pos=1; break;
    case 5:  pos=2; break;
    case 10: pos=3; break;
    case 20: pos=4; break;
    case 50: pos=5; break;
    case 100:pos=6; break;
    default: printf("valor de cedula inválido\n");
  }//end switch
  for(i=0; i<quant; i++){
    //aloca a cedula na memória
```

```
    novo = ??????
    //mudar o valor da cedula para o valor recebido
    novo->valor = ??????
    //adiciona como primeiro elemento da pilha[pos]
    novo->prox = ??????
    pilha[pos] = ??????
  }//end for
}//end abastecePilha
```

c) A função imprimeSaldo, percorre todas as pilhas imprimindo e somando as cédulas de cada pilha. Complete a função **imprimeSaldo()** abaixo e depois faça a chamada no programa principal.

```
void imprimeSaldo(){
  noCedula *atual;
  int i, soma=0;
  //para cada pilha
  for(i=0; i<7; i++){
    atual = pilha[i];
    //percorre cada pilha
    while(atual!=NULL){
      //acumulando a soma
      soma += ??????
      //proximo elemento
      atual = ??????
    }//end while
  }//end for
  printf("saldo=%d\n", soma);
}//end imprimeSaldo
```

d) Complete a **função void imprimeSomaCedulas()** abaixo que deve imprimir na tela a soma dos valores das cédulas de 1, 2, 5, 10, 20, 50 e 100 contidas na pilha. Adicione ao final do programa principal (main) uma chamada para essa função.

```
void imprimeSomaCedulas(){
  noCedula *atual;
  int i, soma=0;
  //para cada pilha
  for(i=0;i<7; i++){
    atual = pilha[i];
    soma = 0;
    //percorre cada pilha
    while(atual!=NULL){
      //acumulando a soma
      soma += ??????
      //proximo elemento
      atual = ??????
    }//end while
    printf("saldo em notas de %d = %d\n", valorNota[i], soma);
  }//end for
  printf("\n");
}//end imprimeSomaCedulas
```

e) A função saqueDisponível dada abaixo, recebe como parâmetro um valor que deverá ser sacado das pilhas. Essa função deverá retornar verdadeiro se o saque foi realizado com sucesso, ou falso, caso contrário. Complete a **função bool saqueDisponivel(int valor)** abaixo:

```
bool saqueDisponivel(int valor){
  int i;
  noCedula *atual;
  for(i=6; i >=0; i--){
    atual = pilha[i]; //inicio de pilha
    //enquanto tem elementos na pilha e
    //enquanto eh possivel dividir o valor pelo valor das cedulas
    while((atual!=NULL) && ((valor / valorNota[i])  >= 1)){
     //decrementa o valor
     valor -= ??????
     //proximo elemento da pilha
     atual = ??????
```

```
    }//end while
  }//end for
  if(valor == 0){
     return true;
  }else{
     return false;
  }//end else
}//end  saqueDisponivel
```

f) Complete a função **bool saque(int valor)** que recebe um valor e remove da pilha as cédulas que totalizam o valor passado para a função.

```
bool saque(int valor){
  int i;
  noCedula *atual;
  if(saqueDisponivel(valor)==true){
    for(i=6; i >=0; i--){
      //enquanto tem elementos na pilha e
      //enquanto eh possível dividir o valor pelo valor das cedulas
      while((pilha[i]!=NULL)&&((valor / valorNota[i])  >= 1)){
        //decrementa o valor
        valor -= ??????
        //pegando o elemento a ser removido da pilha
        atual = ??????
        //mundando o novo primeiro da pilha
        pilha[i] = ??????
        //apagando atual
        ??????;
      }//end while
    }//end for
    return true;
  }else{
     return false;
  }//end else
}//end saque
```

g) Faça uma função chamada **destroiPilha** para liberar toda a memória alocada no programa. A função não retorna nada (void) e não recebe parâmetros.

h) Faça uma função **main** contendo um menu com as seguintes opções:

1- **Abastecer Pilha**: a primeira vez a pilha deve ser abastecida com os seguintes valores: 80 notas de R$100; 70 notas de R$50; 150 notas de R$20; 200 notas de R$10; 30 notas de R$5; 55 notas de R$2; 100 notas de R$1.
2- **Imprimir Saldo**: realizar a chamada para a função imprimeSaldo, onde deverão ser mostrado o saldo de cédulas de cada uma das pilhas.
3- **Saque**: realizar a leitura pelo teclado do valor a ser sacado. Dever ser verificado através da função saqueDisponivel, se é possível realizar o saque ou não. Caso tenha o valor solicitado deve ser impresso uma mensagem, como abaixo. Imprima a soma das cédulas novamente.

```
if(saqueDisponivel(valor) == true){
  printf("saque disponivel");
}
```

4- **Sair**: sair da execução do programa. Chamar a função **destroiPilha** para liberar toda a memória alocada.