Realice un programa para almacenar el No. Cliente, Nombre del cliente, estado (1-activo, 0-inactivo) y los montos de ciertos depósitos en una cuenta. El programa debe contener:

1. Una función **LeerDatos** que preguntara al usuario Si desea ingresar un cliente, en caso afirmativo le solicitará al usuario introducirlo, esto se repetirá hasta que el usuario indique que No requiere introducir más clientes.
2. Una función **colaMayores** que obtiene en una cola con los depósitos que fueron mayores a una cantidad especificada por el usuario. Debe mostrar dicha cola con la información completa del cliente.
3. Otra función **montoProm** que calcule el monto promedio de deposito.

Recuerde utilizar únicamente las operaciones del TAD Cola, No utilizar arreglos como auxiliares para el análisis de lo que se le pide.

TAD Cola:

Main.c:

#include<stdio.h>

#include "cola.h"

#include<stdlib.h>

void manejaMsg(int);

void agregarDato(COLA);

COLA agrupaColas(COLA, COLA, COLA, COLA);

COLA mezclaCola(COLA, COLA);

void mostrarCola(COLA);

void main(){

    COLA C1,C2,C3,C4,C5;

    C1 = crearCola();

    C2 = crearCola();

    C3 = crearCola();

    C4 = crearCola();

    C5 = crearCola();

    printf("Datos C1\n");

    agregarDato(C1);

    printf("\nDatos C2\n");

    agregarDato(C2);

    printf("\nDatos C3\n");

    agregarDato(C3);

    printf("\nDatos C4\n");

    agregarDato(C4);

    C5 = agrupaColas(C1,C2,C3,C4);

    printf("\nC1:\n");

    mostrarCola(C1);

    printf("\nC2:\n");

    mostrarCola(C2);

    printf("\nC3:\n");

    mostrarCola(C3);

    printf("\nC4:\n");

    mostrarCola(C4);

    printf("\nC5:\n");

    mostrarCola(C5);

    liberarCOLA(C1);

    liberarCOLA(C2);

    liberarCOLA(C3);

    liberarCOLA(C4);

    liberarCOLA(C5);

}

void agregarDato(COLA C){

    int opcion;

    int aux = 0;

    do

    {

        printf("Agrega el elemento a la cola:\n");

        scanf("%d",&aux);

        encolar(C,aux);

        printf("\nAgregar otro elemeto a la cola:\nNo -0-\t Si-1-\n\n");

        scanf("%d",&opcion);

    } while (opcion != 0);

}

COLA agrupaColas(COLA C1, COLA C2, COLA C3, COLA C4){

    COLA C5 = mezclaCola(C1,C2);

    C5 = mezclaCola(C5,C3);

    C5 = mezclaCola(C5,C4);

    return C5;

}

COLA mezclaCola(COLA C1, COLA C2){

    COLA C3 = crearCola();

    Cola temp1 = \*C1;

    Cola temp2 = \*C2;

    int n;

    while(!es\_vaciaCola(&temp1)&&!es\_vaciaCola(&temp2)){

        encolar(C3, desencolar(&temp1));

        encolar(C3, desencolar(&temp2));

    }

    while(!es\_vaciaCola(&temp1))

            encolar(C3, desencolar(&temp1));

    while(!es\_vaciaCola(&temp2))

            encolar(C3, desencolar(&temp2));

    return C3;

}

void mostrarCola(COLA C){

    Cola temp = \*C;

    int n;

    while(!es\_vaciaCola(&temp))

        printf("%d ",desencolar(&temp));

    printf("\n");

}

void manejaMsg(int msg){

    char \* mensajes[] = {"No hay memoria disponible . . .","Se ha liberado la memoria . . .  ",

    "La Pila esta llena . . .","La Pila esta vacia . . .","Cola Vacia","Cola LLena","Cola liberada"};

    printf("%s\n", mensajes[msg] );

}

Cola.c:

#include "cola.h"

#include<stdio.h>

#include<stdlib.h>

COLA crearCola (){

    COLA C;

    C = (COLA)malloc(sizeof(Cola));

    if(C== NULL ){

        manejaMsg(0);

        exit(0);

    }

    C -> primero = 0;

    C -> ultimo = 0;

    C -> tam = 0;

    return C;

}

int es\_vaciaCola(COLA C){

    if( C -> tam ==0)

        return TRUE;

    else

        return FALSE;

}

int sig (int i){

    int p;

    p = (i + 1) % TAMCOLA;

    return p;

}

void encolar (COLA C, int x){

    if( C -> tam == TAMCOLA){

        manejaMsg(3);

    }

    C->cola[C->ultimo] = x;

    C->ultimo = sig(C->ultimo);

    C->tam = C->tam+1;

}

int desencolar (COLA C){

    int v;

    if(es\_vaciaCola(C)==TRUE){

        manejaMsg(2);

        exit(0);

    }

    v = C->cola[C->primero];

    C->primero = sig(C->primero);

    C->tam=C->tam-1;

    return v;

}

int primero(COLA C){

    int v;

    if( es\_vaciaCola(C)==TRUE){

        manejaMsg(2);

        exit(0);

    }

    v = C->cola[C->primero];

    return v;

}

void liberarCOLA(COLA C){

    free(C);

    manejaMsg(1);

}

Cola.h:

#ifndef \_cola\_

#define \_cola\_

#define TAMCOLA 100

#define TRUE 1

#define FALSE 0

typedef struct{

int cola [TAMCOLA];

int primero, ultimo;

int tam;

}Cola;

typedef Cola \* COLA;

COLA crearCola ();

int es\_vaciaCola(COLA C);

void encolar (COLA C, int x);

int desencolar (COLA C);

int primero(COLA C);

void liberarCOLA(COLA C);

void manejaMsg(int msg);

#endif

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente Texto

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente