**EXAMEN** **PARCIAL**

**NOMBRE DEL CURSO: ANÁLISIS DE ALGORITMOS Y ESTRATEGIAS DE PROGRAMACIÓN**

**SEMESTRE 2020-II**

**Duración: 90 min. Calificación:**

**Docente:** Chicana Aspajo Jorge

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***ESTUDIANTE*** |  | | | |
| ***CARRERA*** | Ingeniería De Sistemas Computacionales | ***FECHA: / /*** | ***CLASE*** | **6238** |

**INDICACIONES:**

1. Sea cuidadoso con su ortografía y redacción, el cual formará parte de su calificación. Escriba con letra clara y legible.

2. Lea bien la pregunta o enunciado antes de responder. Administre su tiempo eficazmente.

3. Sea breve y objetivo en su respuesta. No se califica por extensión, sino por calidad de respuesta.

4. No puede utilizar apuntes, notas de clase, diapositivas o libros. A menos que el docente del curso se lo permita.

Para todos los casos se evaluará claridad, orden y limpieza en el código.

1. Se tiene una estructura Artículo con los siguientes campos: int idArticulo, string nombre, int cantidad, int idCategoria, y una colección de 10 registros, se desea ubicar y mostrar el artículo (o los artículos) con la menor cantidad; aplique fuerza bruta.

Categoría (1. Fajas; 2. Correas; 3. Rodamientos) (5 puntos).

1. Indicar los valores que toman las variables: n, num, ite y contar en cada iteración, además se debe mostrar (salida) los valores que estas van tomando.

¿Cuántos y cuáles son los contadores que utiliza el algoritmo? 3: contar, ite, num

¿Cuántas y cuáles son las variables que utiliza el algoritmo? 1: n

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **n** | **num** | **contar** | **ite** | **salida** | | 12 | 2 | 1 | 1 |  | | 6 | 2 | 2 | 2 |  | | 3 | 2 | 3 | 3 | 2 ^ 3 | | 3 | 3 | 3 | 4 |  | | 1 | 3 | 1 | 5 | 3 ^ 1 | | 1 | 4 | 1 | 6 |  | |

(5 puntos).

1. Con la estructura de la pregunta 1, se desea mostrar los artículos ordenados, según su nombre, en forma descendente y el usuario podrá buscar un artículo según su nombre, aplique búsqueda binaria recursiva. (5 puntos).
2. En la colección de la pregunta 1, encontrar el artículo (o los artículos), de la categoría rodamientos, con mayor cantidad de rodamientos y la cantidad total de rodamientos; aplicando el algoritmo divide y vencerás (5 puntos).

Resolución general:

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <conio.h>

#include <sstream>

#include <string>

using namespace std;

#define OPC\_GENERAR '1'

#define OPC\_MOSTRAR '2'

#define OPC\_PREGUNTA1 '3'

#define OPC\_ORDERNAR\_DESC '4'

#define OPC\_BUSCAR\_NOMBRE '5'

#define OPC\_MAYOR\_ROD '6'

#define OPC\_CANT\_ROD '7'

#define OPC\_SALIR '9'

enum Categoria { Fajas = 1, Correas = 2, Rodamientos = 3 };

struct Articulo{

int idArticulo;

string nombre;

int cantidad;

Categoria idCategoria;

};

char menu();

int generarLista();

void mostrarLista(Articulo lista[],int inicial,int final);

int menorCantidad(Articulo arr[], int n);

int mayorCantidad(Articulo arr[], int inicio, int fin);

int sumar(Articulo lista[],int n);

Articulo\* crearArregloCantidad(Articulo arr[], int n,int numero,int &cantidadElementos);

Articulo\* crearArregloCategoria(Articulo lista[], int n,Categoria cat,int &cantidadElementos);

void ordenarSELDESC(Articulo lista[],int n);

int busquedaBinREC(Articulo lista[],int inicial,int final,string vb);

string valorBusqueda(string texto);

Articulo\* lista;

int main(int argc, char\*\* argv) {

int n;

char op = 'A';

while (op != '9'){

op = menu();

switch (op){

case OPC\_GENERAR :{

cout << endl << "generar" << endl;

n = generarLista();

break;

}

case OPC\_MOSTRAR:{

cout << endl << "mostrar" << endl;

mostrarLista(lista,0, n);

break;

}

case OPC\_PREGUNTA1:{

int menor = menorCantidad(lista,n);

int cElementosMenores = 0;

Articulo\* listaMenores = crearArregloCantidad(lista,n,menor,cElementosMenores);

cout << endl << endl << "Numero cantidad menor : " << menor << endl;

cout << "Articulos menores :" << endl;

mostrarLista(listaMenores,0, cElementosMenores);

break;

}

case OPC\_ORDERNAR\_DESC:{

ordenarSELDESC(lista,n);

break;

}

case OPC\_BUSCAR\_NOMBRE:{

string vbuscar = valorBusqueda("Ingrese nombre a buscar :");

int idx = busquedaBinREC(lista,0,n-1,vbuscar);

if(idx >= 0){

cout << "Articulo "<< vbuscar << " encontrado" << endl;

}else{

cout << "Articulo "<< vbuscar << " no encontrado" << endl;

}

break;

}

case OPC\_MAYOR\_ROD:{

int cantidad = 0;

Articulo\* arr =crearArregloCategoria(lista,n,(Categoria)3,cantidad);

int mayor = mayorCantidad(arr,0,cantidad - 1);

cout << endl <<"Elementos de la categoria Rodamientos :" << endl;

mostrarLista(arr,0,cantidad);

cout << endl << "La mayor cantidad es :" << mayor << endl;

break;

}

case OPC\_CANT\_ROD:{

int cantidad = 0;

Articulo\* arr =crearArregloCategoria(lista,n,(Categoria)3,cantidad);

int suma = sumar(arr,cantidad);

cout << endl <<"Elementos de la categoria Rodamientos :" << endl;

mostrarLista(arr,0,cantidad);

cout << endl << "La suma es :" << suma << endl;

break;

}

case OPC\_SALIR:{

cout << endl << "salir" << endl;

break;

}

default:{

cout << endl << "No valido" << endl;

break;

}

}

system("pause");

}

return 0;

}

int generarLista(){

int ne;

cout << "Nro de elementos:";

cin >> ne;

lista = new Articulo[ne];

srand(time(0));

for (int i=0;i<ne;i++){

Articulo p;

p.idArticulo = (i+1);

std::ostringstream ss;

ss << (i+1);

p.nombre = "articulo " + ss.str();

p.cantidad = rand() % (ne \* 2) + 1;

p.idCategoria = (Categoria) (rand() % 3 + 1);

lista[i] = p;

}

return ne;

}

void mostrarLista(Articulo lista[],int inicial,int final){

for(int i=inicial;i<final;i++){

Articulo p = lista[i];

cout<<p.idArticulo << " " << p.nombre << " " << p.cantidad << " " << p.idCategoria <<endl;

}

}

int menorCantidad(Articulo arr[], int n)

{

int min = arr[0].cantidad;

for(int i = 1; i < n; i++)

{

if (arr[i].cantidad < min)

min = arr[i].cantidad;

}

return min;

}

Articulo\* crearArregloCantidad(Articulo lista[], int n,int numero,int &cantidadElementos){

int ne = 0;

for(int i = 0; i < n; i++)

{

if (lista[i].cantidad == numero)

ne++;

}

cantidadElementos = ne;

Articulo\* arr = new Articulo[ne];

int cont = 0;

for(int i = 0; i < n; i++)

{

if (lista[i].cantidad == numero)

{

arr[cont] = lista[i];

cont++;

}

}

return arr;

}

Articulo\* crearArregloCategoria(Articulo lista[], int n,Categoria cat,int &cantidadElementos){

int ne = 0;

for(int i = 0; i < n; i++)

{

if (lista[i].idCategoria == cat)

ne++;

}

cantidadElementos = ne;

Articulo\* arr = new Articulo[ne];

int cont = 0;

for(int i = 0; i < n; i++)

{

if (lista[i].idCategoria == cat)

{

arr[cont] = lista[i];

cont++;

}

}

return arr;

}

void ordenarSELDESC(Articulo lista[],int n){

int idx;

for (int i=0;i<n-1;i++){

idx = i;

for (int j=i+1;j<n;j++){

if (lista[j].nombre.compare(lista[idx].nombre) > 0){

idx = j;

}

}

Articulo aux = lista[i];

lista[i] = lista[idx];

lista[idx] = aux;

}

}

int busquedaBinREC(Articulo lista[],int inicial,int final,string vb){

int medio;

if(inicial <= final){

medio = (inicial + final) / 2;

if(lista[medio].nombre.compare(vb) == 0){

return medio;

}else if(lista[medio].nombre.compare(vb) > 0){

final = medio - 1;

}else if(lista[medio].nombre.compare(vb) < 0){

inicial = medio + 1;

}

return busquedaBinREC(lista,inicial,final,vb);

}

return -1;

}

string valorBusqueda(string texto){

string vb;

cout<< endl << texto <<endl;

fflush(stdin);

getline (cin,vb);

return vb;

}

int mayorCantidad(Articulo arr[], int inicio, int fin){

if(inicio == fin) return lista[inicio].cantidad;

int medio = (inicio + fin) / 2;

int izq = mayorCantidad(lista,inicio, medio);

int der = mayorCantidad(lista,medio + 1, fin);

return max(izq,der);

}

int sumar(Articulo lista[],int n){

if (n <= 0)

return 0;

return (sumar(lista, n - 1) + lista[n - 1].cantidad);

}

char menu(){

fflush(stdin);

setlocale(LC\_ALL,"spanish");

system("cls");

cout << "Menu principal " << endl;

cout << "[1] Generar lista " << endl;

cout << "[2] Mostrar lista " << endl;

cout << "[3] PREGUNTA 1 " << endl;

cout << "[4] Ordenar SEL DESC" << endl;

cout << "[5] Buscar nombre " << endl;

cout << "[6] Mayor rodamientos" << endl;

cout << "[7] Cantidad rodamientos" << endl;

cout << "[9] Salir " << endl;

cout << "Seleccione opcion:";

return toupper(getche());

}