**Examen #1**

**Integrantes del grupo:**

**1. Alfredo Jiménez #116518**

**2. Coral S. Schmidt Montilla #148830**

**3. Jorge Zapata #135565**

**4. Aleysha M. Rivera Cores #108408**

**PARTE 1:**

Haga una clase cuyos objetos representen, cada uno, un libro. El único atributo que tendrá un objeto libro es la cantidad de páginas del libro. También, la clase tiene que incluir los siguientes servicios para sus objetos: • determinar si dos libros son iguales

* cambiar la cantidad de páginas del libro
* obtener la cantidad de páginas del libro
* mostrar la información del libro

**Libro.h (declaración de la clase):**

#pragma once

// preprocessor directives #include <iostream> using namespace std;

class Libro { private:

int cantidadPaginas;

public:

// Constructor para inicializar la cantidad de páginas

Libro(int paginas);

// Determinar si dos libros son iguales bool sonIguales(const Libro& otroLibro) const;

// Cambiar la cantidad de páginas del libro void cambiarPaginas(int nuevasPaginas);

// Obtener la cantidad de páginas del libro int obtenerPaginas() const;

// Mostrar la información del libro

void mostrarInformacion() const;

};

**Libro.cpp (definición de los métodos):**

#include <iostream>

// Constructor para inicializar la cantidad de páginas Libro::Libro(int paginas) { this->cantidadPaginas = paginas;

}

// Determinar si dos libros son iguales

bool Libro::sonIguales(const Libro& otroLibro) const { return this->cantidadPaginas == otroLibro.cantidadPaginas;

}

// Cambiar la cantidad de páginas del libro void Libro::cambiarPaginas(int nuevasPaginas) { this->cantidadPaginas = nuevasPaginas;

}

// Obtener la cantidad de páginas del libro int Libro::obtenerPaginas() const { return this->cantidadPaginas;

}

// Mostrar la información del libro void Libro::mostrarInformacion() const {

cout << "Libro con " << this->cantidadPaginas << " páginas." << endl;

}

**PARTE 2:**

Haga una clase cuyos objetos representen, cada uno, una caja de libros. Tambien, la clase tiene que incluir los siguientes servicios para sus objetos:

* mostrar lo que tiene la caja
* obtener la cantidad de paginas de todos los libros guardados en la caja
* determinar si dos cajas son iguales

**CajaDeLibros.h (declaración de la clase):**

#pragma once

#include <vector>

#include "Libro.h"

class CajaDeLibros {

private:

vector<Libro> libros;

public:

CajaDeLibros();

// Agregar un libro a la caja

void agregarLibro(const Libro& libro);

// Mostrar los libros en la caja

void mostrarContenido()const;

// Obtener la cantidad total de páginas en la caja

int obtenerCantidadTotalDePaginas()const;

// Determinar si dos cajas son iguales

bool sonIguales(const CajaDeLibros& otraCaja) const;

};

**CajaDeLibros.cpp (definición de los métodos):**

#include "CajaDeLibros.h"

CajaDeLibros::CajaDeLibros() {}

void CajaDeLibros::agregarLibro(const Libro& libro) {

this->libros.push\_back(libro);

}

void CajaDeLibros::mostrarContenido() const {

for (const Libro& libro : this->libros) {

libro.mostrarInformacion();

}

}

int CajaDeLibros::obtenerCantidadTotalDePaginas() const {

int totalPaginas = 0;

for (const Libro& libro : this->libros) {

totalPaginas += libro.obtenerPaginas();

}

return totalPaginas;

}

bool CajaDeLibros::sonIguales(const CajaDeLibros& otraCaja) const {

// Dos cajas son iguales si tienen la misma cantidad de libros y los libros son iguales en orden

if (this->libros.size() != otraCaja.libros.size()) {

return false;

}

for (size\_t i = 0; i < this->libros.size(); i++) {

if (!this->libros[i].sonIguales(otraCaja.libros[i])) {

return false;

}

}

return true;

}

Use un vector pq CajaDe Libros va a terminar siendo un array y el vector nos permite guardar espacio sin nesesitar especificar cuantos espacios de memoria temenos que guardar de antemano so its better for memory management