#include <iostream>

#include <vector>

#include <thread>

#include <future>

**using namespace** std;

**void** sumar(vector<**int**> arreglo1,**int**& suma){

**for**(**int** i:arreglo1){

suma=suma+i;

}

};

**class** Contenedor{

**public**:

vector<**int**>\* elementos;

Contenedor():elementos(**new** vector<**int**>){};

**void** addElemento(**int** elemento){

elementos->push\_back(elemento);

};

**void** particionarColeccion(Contenedor& contenedor1,Contenedor contenedor2){

**for** (**int** i:\*elementos){

**if**((i%2)==0){

contenedor1.addElemento(i);

} **else**{

contenedor2.addElemento(i);

}

}

};

};

**template** <**typename** T>

**class** Conjunto

{

**private**:

vector<T>\* elementos;

**public**:

Conjunto():elementos(**new** vector<T>){};

**void** adicionarElemento(T elemento){

**bool** is = **false**;

**for** (T element:\*elementos){

**if**(element == elemento){

is = **true**;

}

}

**if**(!is){

elementos->push\_back(elemento);

}

}

**void** mostrarElementos(){

**for** (T elemento:\*elementos){

cout<<elemento<<" ";

}

}

Conjunto realizarUnion(Conjunto conjunto){

Conjunto Union ;

**for** (T elemento:\*elementos){

Union.adicionarElemento(elemento);

}

**for** (T elemento:\*(conjunto.elementos)){

Union.adicionarElemento(elemento);

}

**return** Union;

}

Conjunto realizarInterseccion(Conjunto conjunto){

Conjunto Interseccion;

**for** (T elementoA:\*elementos){

**for** (T elementoB:\*(conjunto.elementos)){

**if** (elementoA==elementoB){

Interseccion.adicionarElemento(elementoA);

}

}

}

**return** Interseccion;

}

Conjunto realizarDiferencia(Conjunto conjunto){

Conjunto Diferencia;

**for** (T elementoA:\*elementos){

**bool** esta = **false**;

**for** (T elementoB:\*(conjunto.elementos)){

**if**(elementoA == elementoB){

esta = **true**;

}

}

**if** (esta==**false**){

Diferencia.adicionarElemento(elementoA);

}

}

**return** Diferencia;

}

Conjunto realizarDiferenciaSimetrica(Conjunto conjunto){

Conjunto DiferenciaSimetrica;

Conjunto AmenosB = **this**->realizarDiferencia(conjunto);

Conjunto BmenosA = conjunto.realizarDiferencia(\***this**);

DiferenciaSimetrica = AmenosB.realizarUnion(BmenosA);

**return** DiferenciaSimetrica;

}

Conjunto realizarComplemento(Conjunto conjunto){

conjunto = conjunto.realizarDiferencia(\***this**);

**return** conjunto;

}

};

**int** main() {

**char** char1='a',char2='b',char3='c',char4='f',char5='a',char6='b',char7='c',char8='e';

Conjunto<**char**> A;

Conjunto<**char**> B;

Conjunto<**char**> C;

Conjunto<**char**> D;

Conjunto<**char**> E;

Conjunto<**char**> F;

Conjunto<**char**> G;

Conjunto<**char**> Universal;

A.adicionarElemento(char1);

A.adicionarElemento(char2);

A.adicionarElemento(char3);

A.adicionarElemento(char4);

std::cout<<"Conjunto A: ";

A.mostrarElementos();

B.adicionarElemento(char5);

B.adicionarElemento(char6);

B.adicionarElemento(char7);

B.adicionarElemento(char8);

std::cout<<"\nConjunto B: ";

B.mostrarElementos();

Universal = A.realizarUnion(B);

std::cout<<"\nConjunto Universal: ";

Universal.mostrarElementos();

C = A.realizarUnion(B);

std::cout<<"\nUnion: ";

C.mostrarElementos();

D = A.realizarInterseccion(B);

std::cout<<"\nInterseccion: ";

D.mostrarElementos();

E = A.realizarDiferencia(B);

std::cout<<"\nDiferencia A-B: ";

E.mostrarElementos();

F = A.realizarDiferenciaSimetrica(B);

std::cout<<"\nDiferencia Simetrica: ";

F.mostrarElementos();

G = A.realizarComplemento(B);

std::cout<<"\nComplemento de A: ";

G.mostrarElementos();

srand(time(**nullptr**));

Contenedor particion1;

Contenedor particion2;

Contenedor coleccion;

**int** suma=0;

**for** (**int** i =0;i<2000;i++){

**int** aleatoria=1+rand()%100;

coleccion.addElemento(aleatoria);

}

coleccion.particionarColeccion(particion1,particion2);

thread t1{sumar,\*(particion1.elementos),ref(suma)};

thread t2{sumar,\*(particion2.elementos),ref(suma)};

t1.join();

t2.join();

cout<<"\nLa suma total de los hilos es: "<<suma;

**return** 0;

}