#include<iostream>

using namespace std;

template<class T, int max>

class Kolekcija

{

T\* \_elementi;

int \_trenutnoElemenata;

public:

//Implementirati potrebne funkcije za rad sa kolekcijom.

};

ostream& operator<<(ostream& cout, const Predmet& p)

{

cout << "Predmet: " << p.\_naziv << endl;

cout << "Godina studija: " << p.\_godinaStudija << endl;

return cout;

}

class Prisustvo

{

char\* \_brojIndeksa;

int \_brojSati;

public:

//Osnovne funkcije za rad s klasom

Prisustvo(char\* brojIndeksa, int brojSati) : \_brojSati(brojSati)

{

int size = strlen(brojIndeksa) + 1;

\_brojIndeksa = new char[size];

strcpy\_s(\_brojIndeksa, size, brojIndeksa);

}

~Prisustvo()

{

delete[] \_brojIndeksa;

\_brojIndeksa = nullptr;

}

//Preklopljeni operator + i += za uvećavanje broja sati prisustva za vrijednost proslijeđenog parametra

};

class Nastava

{

char \_datum[12];

char \_tipNastave; //P ili V

int \_odrzanoSati;

Prisustvo\* \_prisutni;

int \_brojStudenata;

public:

//Konstruktor sa osnovnim atributima klase

//Destruktor

//Preklopljeni operator+= za dodavanje objekata tipa Prisustvo u niz \_prisutni.

//Nije moguće dodati više puta prisustvo za istog studenta (preklopiti operator== u klasi Prisustvo).

//Elementi u svakom momentu treba da budu sortirani po broju indeksa studenta (preklopiti operator < u klasi Prisustvo).

//Preklopljeni operator-= za uklanjanje prisustva za studenta sa proslijedjenim brojem indeksa. Nakon uklanjanja,

//potrebno je sačuvati redoslijed dodavanja, odnosno podaci ostaju sortirani po broju indeksa.

//Preklopljeni operator [] koji na osnovu broja indeksa provjerava da li je

//određeni student prisustvovao nastavi i vraća njegove podatke o prisustvu.

//Preklopljeni operator () koji pronalazi studenta sa proslijeđenim brojem indeksa i uvećava mu broj sati prisustva

//za vrijednost drugog parametra. Ako je studentu već evidentiran odrzani broj sati nastave prekinuti funkciju.

//Preklopljeni operator + koji zbraja dva objekta tipa Nastava na način da kreira novi objekat na osnovu lijevog operanda

//i dodaje mu prisutne studente desnog operanda. Funkciju napustiti ukoliko objekti koji učestvuju u operaciji sabiranja

//nisu istog tipa nastave. Spriječiti dodavanje prisustva za istog studenta više puta.

//Operator implementirati kao globalnu funkciju.

};

class Predmet

{

char\* \_naziv;

int \_godinaStudija;

Kolekcija<Nastava, 90> \_odrzanaNastava;

public:

Predmet(char\* naziv, int godinaStudija) : \_godinaStudija(godinaStudija)

{

int size = strlen(naziv) + 1;

\_naziv = new char[size];

strcpy\_s(\_naziv, size, naziv);

}

~Predmet()

{

delete[] \_naziv;

\_naziv = nullptr;

}

//Funkciju za evidenciju održane nastave na predmetu.

/\*Funkciju koja na osnovu datuma i tipa održane nastave dodaje prisustvo studentu sa proslijeđenim brojem indeksa i brojem sati.

Voditi računa da broj sati prisustva ne može biti veći od broja sati održane nastave.\*/

/\*Funkciju koja na osnovu datuma i tipa održane nastave uklanja prisustvo studentu sa proslijeđenim brojem indeksa.\*/

/\*Funkciju koja na osnovu broja indeksa studenta vraća sve podatke o njegovom prisustvu na nastavi\*/

/\*Funkciju koja na osnovu tipa nastave (P ili V) vrši ispis prisustva za sve studente (sumarno) u datom formatu:

IB140001 30/40 75%, gdje je prva kolona "broj indeksa", druga kolona "broj sati prisustva / ukupan broj sati održane nastave" i

treca kolona "procenat prisustva studenta na nastavi".\*/

friend ostream& operator<<(ostream&, const Predmet&);

};

int main()

{

//Testirati sve dostupne funkcionalnosti klasa Prisustvo i Nastava.

return 0;

}