**Отчет ПО ЛАБОРАТОРНЫМ РАБОТАМ**

**По информатике**

Лабораторная работа №5.

Библиотека шаблонов STL

(наименование лабораторной работы в соответствии с учебным планом)

***Цель и задачи работы:***

Изучение стандартной библиотеки шаблонов (STL) языка С++, программирование и отладка программ формирования и обработки контейнеров, комбинации контейнеров.

***Задание №1***

Много файловый проект ENUM.

Необходимо реализовать систему хранения и обработки информации по индивидуальному заданию.

График движения самолетов.

***Листинг программы:***

1) C++

(Main)

#include "Header.h"

int main() {

main2();

}

(Header)

#pragma once

void main2();

(Source)

#include <iostream>

#include <vector>

#include <string>

#include <map>

#include "Header.h"

using namespace std;

enum class Plane {

CREATE\_PLANE,

PLANES\_FOR\_TOWN,

TOWNS\_FOR\_PLANE,

PLANES

};

void CREATE\_PLANE(int num, int N, multimap<int, string>& plane)

{

string TOWN;

int j;

for (j = 0; j < N; j++)

{

cin >> TOWN;

plane.emplace(make\_pair(num, TOWN));

}

}

void PLANES\_TOWN(multimap<int, string>& plane, string TOWN, vector <int> x)

{

cout << TOWN << " : ";

for (int i = 0; i < x.size(); i++)

{

auto range = plane.equal\_range(x[i]);

for (auto j = range.first; j != range.second; j++)

{

if (j->second == TOWN)

{

cout << j->first << ' ';

}

}

}

cout << endl;

}

void TOWNS\_PLANE(int num, string TOWN, multimap<int, string>& plane)

{

auto range = plane.equal\_range(num);

cout << "PLANE " << num << ": ";

for (auto i = range.first; i != range.second; i++)

{

cout << i->second << ' ';

}

cout << endl;

}

void PLANES(int num, string TOWN, multimap <int, string> plane, vector <int> x)

{

for (int i = 0; i < x.size(); i++)

{

auto range = plane.equal\_range(x[i]);

cout << "PLANE " << x[i] << ": ";

for (auto i = range.first; i != range.second; i++)

{

cout << i->second << ' ';

}

cout << endl;

}

}

void main2()

{

multimap <int, string> train;

vector <int> list;

int number, num, N;

string TOWN, slovo;

num = -1;

while (1) {

cout << "\nFeature List:\n1.CREATE\_PLANE plane number of towns town1 town2 ... \n2.PLANES\_FOR\_TOWN town \n3.TOWNS\_FOR\_PLANE plane \n4.PLANES \n5.EXIT \n\n";

cin >> slovo;

if (slovo == "CREATE\_PLANE") {

number = 0;

}

if (slovo == "PLANE\_FOR\_TOWN") {

number = 1;

}

if (slovo == "TOWNS\_FOR\_PLANE") {

number = 2;

}

if (slovo == "PLANES") {

number = 3;

}

if (slovo == "EXIT") break;

switch (Plane(number)) {

case Plane::CREATE\_PLANE: {

cin >> num >> N;

CREATE\_PLANE(num, N, train);

list.push\_back(num);

break;

}

case Plane::PLANES\_FOR\_TOWN: {

cin >> TOWN;

PLANES\_TOWN(train, TOWN, list);

break;

}

case Plane::TOWNS\_FOR\_PLANE: {

cin >> num;

TOWNS\_PLANE(num, TOWN, train);

break;

}

case Plane::PLANES: {

if (num < 0) {

cout << "\nPlanes is absent";

break;

}

else {

PLANES(num, TOWN, train, list);

break;

}

}

default: break;

}

}

}

2) Python

import enum

from collections import defaultdict

class TYPE(enum.Enum):

CREATE\_PLANE = 'CREATE\_PLANE'

PLANES\_FOR\_TOWN = 'PLANES\_FOR\_TOWN'

TOWNS\_FOR\_PLANE = 'TOWNS\_FOR\_PLANE'

PLANES = 'PLANES'

planes=defaultdict(list)

towns=[]

while(1):

command=input()

if(TYPE(command) == TYPE.CREATE\_PLANE):

n\_plane = input()

N = int(input())

for i in range(N):

city=input()

planes[n\_plane].append(city)

if(TYPE(command) == TYPE.PLANES\_FOR\_TOWN):

town = input()

for key, value in planes.items():

if town in value:

print(key)

if(TYPE(command) == TYPE.TOWNS\_FOR\_PLANE):

n\_plane=input()

for key, value in planes.items():

if n\_plane in key:

print(value)

for i in range(len(value)):

print(value[i])

for k, v in planes.items():

if (value[i] in v) and (n\_plane!=k):

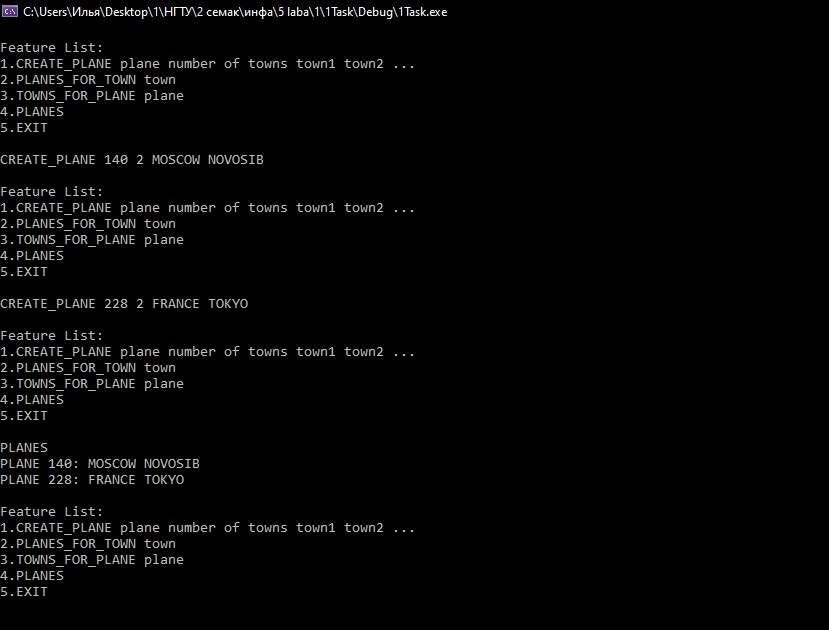
print(k)

if(TYPE(command) == TYPE.PLANES):

for key, value in planes.items():

print(key, value)

***Пример работы программы:***



*(рис. 1 – пример работы программы на C++)*

**Задание №2.**

Комбинация контейнеров. Реализовать автоматизированную систему: У каждого студента есть расписание занятий. Посещать занятия необходимо в конкретный день определенного месяца.

***Листинг программы:***

1) C++

#include <iostream>

#include <vector>

#include <string>

#include <map>

using namespace std;

enum class Schedule

{

СLASS = 1,

NEXT,

VIEW

};

multimap <int, string> plan;

Schedule type;

void Class(int m[], int N, multimap <int, string>& s) {

int i;

string yrok;

cin >> i;

cin >> yrok;

if (i <= m[N]) s.emplace(make\_pair(i, yrok));

else exit(0);

}

int Next(int m[], int N, multimap <int, string>& s) {

int t;

N++;

if (N > 12) {

N = N % 12;

}

for (int i = m[N] + 1; i < 32; i++) {

t = s.count(i);

if (t != 0) {

for (int j = 0; j < t; j++) {

string t = s.find(i)->second;

s.emplace(make\_pair(m[N] - 1, t));

s.erase(i);

}

}

}

return N;

}

void View(int m[], int N, multimap <int, string>& s)

{

int i, y;

cin >> i;

auto range = s.equal\_range(i);

if (i <= m[N]) {

y = s.count(i);

if (y != 0) {

cout << "In " << i << " day " << y << " classes in university:" << endl;

for (auto j = range.first; j != range.second; j++) {

cout << j->second;

if (y > 1 && (j++) != range.second) cout << ", ";

}

cout << endl;

}

else {

cout << "In " << i << " day we all free" << endl;

}

}

else {

exit(0);

}

}

int main() {

multimap <int, string> s;

int K, N = 1, type;

int m[13] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

cout << "Enter the number of teams: ";

cin >> K;

cout << endl;

cout << "\nFeature List:\n1)CLASS date dicipline \n2)NEXT \n3)VIEW date \n4)EXIT\n\n";

for (int i = 0; i < K; K--) {

string slovo;

cin >> slovo;

type = 0;

if (slovo == "CLASS") type = 1;

if (slovo == "NEXT") type = 2;

if (slovo == "VIEW") type = 3;

if (slovo == "EXIT") return 0;

switch (Schedule(type)) {

case Schedule::СLASS: {

Class(m, N, s);

break;

}

case Schedule::NEXT: {

Next(m, N, s);

break;

}

case Schedule::VIEW: {

View(m, N, s);

break;

}

}

}

return 0;

}

2) Python

import enum

from collections import defaultdict

class TYPE(enum.Enum):

CLASS='CLASS'

NEXT='NEXT'

VIEW='VIEW'

day=defaultdict(list)

N=int(input())

month=[-1, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

N\_month=1;

while(N>0):

command=input()

if(TYPE(command)==TYPE.CLASS):

i=int(input())

s=input()

if(i<=month[N\_month]):

day[i].append(s)

else:

print("We can't set this day")

if(TYPE(command)==TYPE.NEXT):

N\_month=N\_month+1

if(N\_month>12):

N\_month=N\_month-12

i=month[N\_month]+1

while(i<=31):

k=0

for key,value in day.items():

if(key==i):

k=value

break

if k:

day[month[N\_month]-1].append(k)

day.pop(i)

i=i+1

if(TYPE(command)==TYPE.VIEW):

i=int(input())

if(i<=month[N\_month]):

for k, v in day.items():

if (k==i):

print('In '+str(i)+' day '+str(len(v))+' classes in university')

for key, value in day.items():

if(key==i):

print(value)

if i in day:

continue

else:

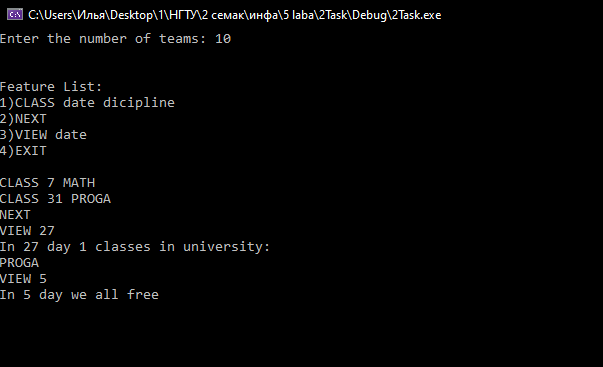
print("In "+str(i)+" day we all free")

else:

print("We can't see this day")

N=N-1

***Пример работы программы:***



*(рис. 2 – пример работы программы на C++)*