Министерство образования и науки РФ

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра Информационные технологии и автоматизированные системы

Дискретная математика

Лабораторная работа № 1

Тема: «Множества»

Выполнил: студент группы РИС-24-3б

Караваев А.А

г. Пермь – 2025

# Основная часть

1. Создать 3 множества без повторяющихся чисел;
2. Создать условия введения чисел (только положительные или отрицательные, четность, кратность n)
3. Создать методы работы со множествами (объединение, пересечение, разность, симметрическая разность, дополнение)

Описание методов:

1. Класс Set

class Set {

private:

vector<int> elements;

elements — контейнер для хранения уникальных элементов множества.

1. Метод contains

bool contains(int value) const {

for (int el : elements) {

if (el == value) return true;

}

return false;

}

Проверяет наличие элемента value в множестве. Используется при добавлении, чтобы избежать дубликатов.

1. Метод add

void add(int value) {

if (!contains(value)) {

elements.push\_back(value);

print();

}

}

void add(int value) {

if (!contains(value)) {

elements.push\_back(value);

print();

}

}

Добавляет элемент в множество, если его там ещё нет. После добавления выводит текущее состояние множества.

1. Метод manual

void manual() {

while (elements.size() < 10) {

int value;

cout << "Введите число, которое хотите добавить в множество: ";

cin >> value;

add(value);

}

}

Позволяет пользователю ввести 10 уникальных чисел вручную. После каждого успешного добавления выводит множество.

1. Метод random

void random(function<bool(int)> condition = nullptr) {

while (elements.size() < 10) {

int value = -50 + rand() % 101; // случайное число от -50 до 50

if (!condition || condition(value)) {

add(value);

}

}

}

Заполняет множество случайными числами в диапазоне [-50, 50]. Если передана функция condition, добавляет только те числа, которые удовлетворяют условию. Позволяет гибко фильтровать случайные значения.

1. Метод term

void term() {

cout << "Выберите условие для заполнения множества рандомными числами:\n"

<< "1: Чётные числа\n"

<< "2: Нечётные числа\n"

<< "3: Положительные числа (включая 0)\n"

<< "4: Отрицательные числа\n"

<< "5: Числа, кратные n (введите число)\n";

int choice;

cin >> choice;

switch (choice) {

case 1:

random([](int x) { return x % 2 == 0; });

break;

case 2:

random([](int x) { return x % 2 != 0; });

break;

case 3:

random([](int x) { return x >= 0; });

break;

case 4:

random([](int x) { return x < 0; });

break;

case 5: {

int n;

cout << "Введите число n: ";

cin >> n;

random([n](int x) { return n != 0 && x % n == 0; });

break;

}

default:

cout << "Неверный выбор!" << endl;

break;

}

}

Позволяет пользователю выбрать условие для генерации случайных чисел. В зависимости от выбора вызывает random с соответствующей лямбда-функцией. При выборе пункта 5 запрашивает число n и генерирует числа, кратные n.

1. Метод print

void print() const {

cout << "Множество: ";

for (int el : elements) {

cout << el << " ";

}

cout << endl;

}

Выводит все элементы множества в одну строку.

1. Объединение (+)

Set operator+(const Set& lhs, const Set& rhs) {

Set result = lhs;

for (int el : rhs.elements) {

if (!result.contains(el)) {

result.elements.push\_back(el);

}

}

return result;

}

Возвращает объединение двух множеств — все элементы из обоих без повторов.

1. Пересечение (^)

Set operator^(const Set& lhs, const Set& rhs) {

Set result;

for (int el : lhs.elements) {

if (rhs.contains(el)) {

result.elements.push\_back(el);

}

}

return result;

}

Возвращает множество элементов, которые присутствуют в обоих множествах.

1. Разность (/)

Set operator/(const Set& lhs, const Set& rhs) {

Set result;

for (int el : lhs.elements) {

if (!rhs.contains(el)) {

result.elements.push\_back(el);

}

}

return result;

}

Возвращает элементы из lhs, которых нет в rhs.

1. Симметричная разность (%)

Set operator%(const Set& lhs, const Set& rhs) {

return (lhs / rhs) + (rhs / lhs);

}

Возвращает элементы, которые есть в одном из множеств, но не в обоих.

1. Оператор дополнения (~)

Set operator~() const {

Set result;

for (int i = -50; i <= 50; ++i) {

if (!contains(i)) {

result.elements.push\_back(i);

}

}

return result;

}

Возвращает множество всех чисел в диапазоне [-50; 50], отсутствующих в текущем множестве. Используется как универсальное множество фиксированного диапазона.

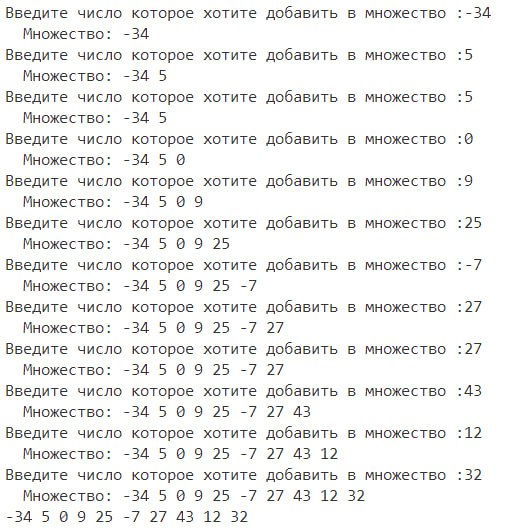


Рис. 1 Заполнение вводом.

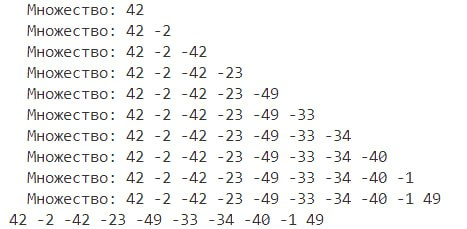


Рис. 2 Заполнение случайными числами.

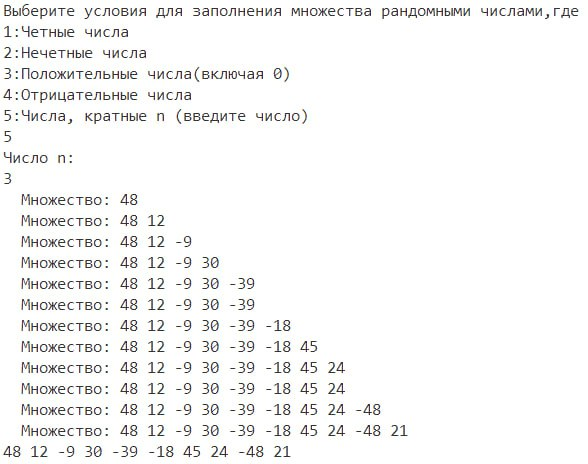


Рис. 3 Заполнение случайными числами по определенному критерию.

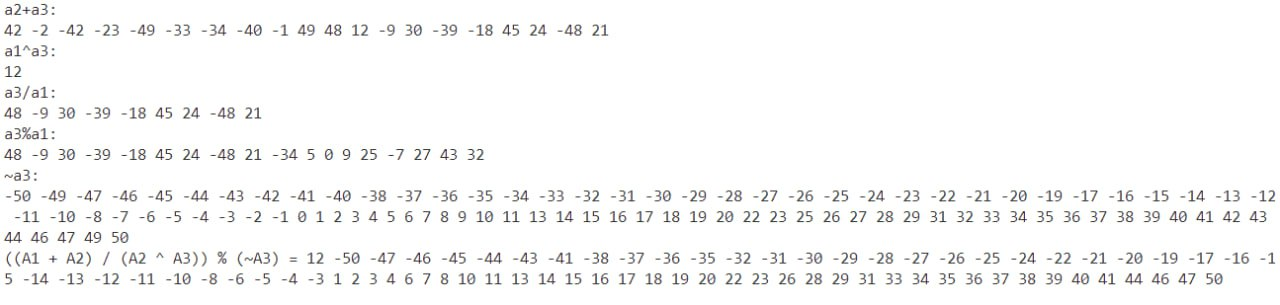


Рис. 4 объединение, пересечение, разность, симметрическая разность, дополнение, формула

Ссылка на github: https://github.com/Prefix008/Diskretka\_labs

.

**ПРИЛОЖЕНИЕ A**

Код программы:  
**#include <iostream>**

**#include <vector>**

**#include <cstdlib>**

**#include <ctime>**

**#include <functional>**

**using namespace std;**

**class set**

**{**

**vector <int> a;**

**public:**

**void manual() {**

**while (a.size()<10) {**

**int f;**

**cout << "Введите число которое хотите добавить в множество :";**

**cin >> f;**

**del(f);**

**}**

**};**

**void random(function<bool(int)> func=nullptr){**

**while (a.size()<10) {**

**int f;**

**f=-50+rand()%101;**

**if(func==nullptr||(func(f))){**

**del(f);**

**}**

**}**

**}**

**void term() {**

**int k;**

**cout<<"Выберите условия для заполнения множества рандомными числами,где"<<endl**

**<<"1:Четные числа"<<endl<<"2:Нечетные числа"<<endl<<"3:Положительные числа(включая 0)"<<endl<<"4:Отрицательные числа"<<endl<<"5:Числа, кратные n (введите число)"<<endl;**

**cin>>k;**

**switch (k)**

**{**

**case 1:**

**random([](int x){return x%2==0;});**

**break;**

**case 2:**

**random([](int x){return x%2!=0;});**

**break;**

**case 3:**

**random([](int x){return x>=0;});**

**break;**

**case 4:**

**random([](int x){return x<0;});**

**break;**

**case 5:**

**int n;**

**cout<<"Чиcло n:"<<endl;**

**cin>>n;**

**random([n](int x){return x%n==0;});**

**break;**

**default:**

**cout << "Неверный выбор!" << endl;**

**break;**

**}**

**};**

**void del(int f) {**

**bool fl=true;**

**for (int i = 0; i < a.size(); i++) {**

**if (f == a[i]) {**

**fl = false;**

**}**

**}**

**if (fl) {**

**a.push\_back(f);**

**}**

**cout << " Множество: ";**

**print();**

**}**

**void print() {**

**for (int i = 0; i <a.size() ; i++) {**

**cout << a[i]<<" ";**

**}**

**cout << endl;**

**}**

**friend set operator+(const set&,const set&);**

**friend set operator^(const set&,const set&);**

**friend set operator/(const set&,const set&);**

**friend set operator%(const set&,const set&);**

**set operator~(){**

**set temp;**

**for(int i=-50;i<51;i++){**

**bool exists = false;**

**for(int j:a){**

**if(j==i){**

**exists=true;**

**break;**

**}**

**}**

**if (!exists){**

**temp.a.push\_back(i);**

**}**

**}**

**return temp;**

**}**

**};**

**set operator+(const set& aa,const set& bb){**

**set temp=aa;**

**for (int item : bb.a) {**

**bool exists = false;**

**for (int existing\_item : aa.a) {**

**if (item == existing\_item) {**

**exists = true;**

**break;**

**}**

**}**

**if (!exists) {**

**temp.a.push\_back(item);**

**}**

**}**

**return temp;**

**}**

**set operator^(const set& aa,const set& bb){**

**set temp;**

**for (int item : bb.a) {**

**bool exists = false;**

**for (int existing\_item : aa.a) {**

**if (item == existing\_item) {**

**exists = true;**

**break;**

**}**

**}**

**if (exists) {**

**temp.a.push\_back(item);**

**}**

**}**

**return temp;**

**}**

**set operator/(const set& aa,const set& bb){**

**set temp;**

**for (int item : aa.a) {**

**bool exists = false;**

**for (int existing\_item : bb.a) {**

**if (item == existing\_item) {**

**exists = true;**

**break;**

**}**

**}**

**if (!exists) {**

**temp.a.push\_back(item);**

**}**

**}**

**return temp;**

**}**

**set operator%(const set& aa,const set& bb){**

**set temp;**

**temp=(aa/bb)+(bb/aa);**

**return temp;**

**}**

**int main()**

**{**

**srand(time(NULL));**

**set a1;**

**a1.manual();**

**a1.print();**

**set a2;**

**a2.random();**

**a2.print();**

**set a3;**

**a3.term();**

**a3.print();**

**cout<<"a2+a3:"<<endl;**

**(a2+a3).print();**

**cout<<"a1^a3:"<<endl;**

**(a1^a3).print();**

**cout<<"a3/a1:"<<endl;**

**(a3/a1).print();**

**cout<<"a3%a1:"<<endl;**

**(a3%a1).print();**

**cout<<"~a3:"<<endl;**

**(~a3).print();**

**cout << "((A1 + A2) / (A2 ^ A3)) % (~A3) = ";**

**(((a1 + a2) / (a2 ^ a3)) % (~a3)).print();**

**}**