

Министерство науки и образования РФ
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
«Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)
Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Отчёт
по лабораторной работе № 1
на тему:
“Множества”
по дисциплине “Алгоритмы и структуры данных”
Вариант 19

Выполнили студенты гр. 4306:
Табаков А.В.,
Сыромятников М.А.
Принял: Колинко П.Г.

Цель

Получить практические навыки работы с логическими операциями над множествами.

Задание

Множество содержащие все цифры, общие для A и B, а также все цифры являющиеся общими для C и D.

Универсум шестнадцатеричные цифры (0123456789ABCDEF).

Способы задания множества

1. Последовательность символов
2. Список
3. Машинное слово
4. Массив битов

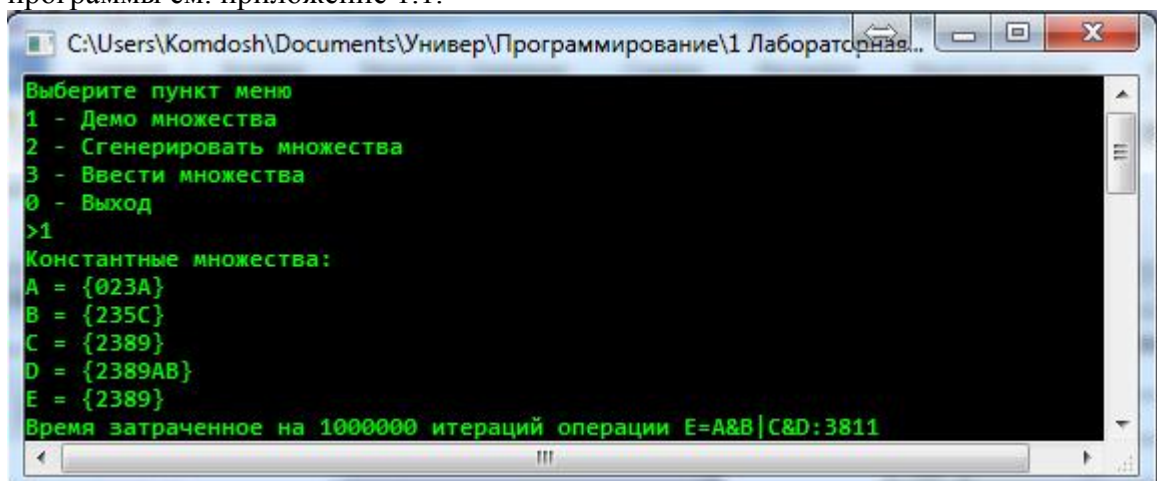
Контрольные примеры

Контрольные примеры представлены в таблице 1.

Таблица. 1. Контрольные примеры

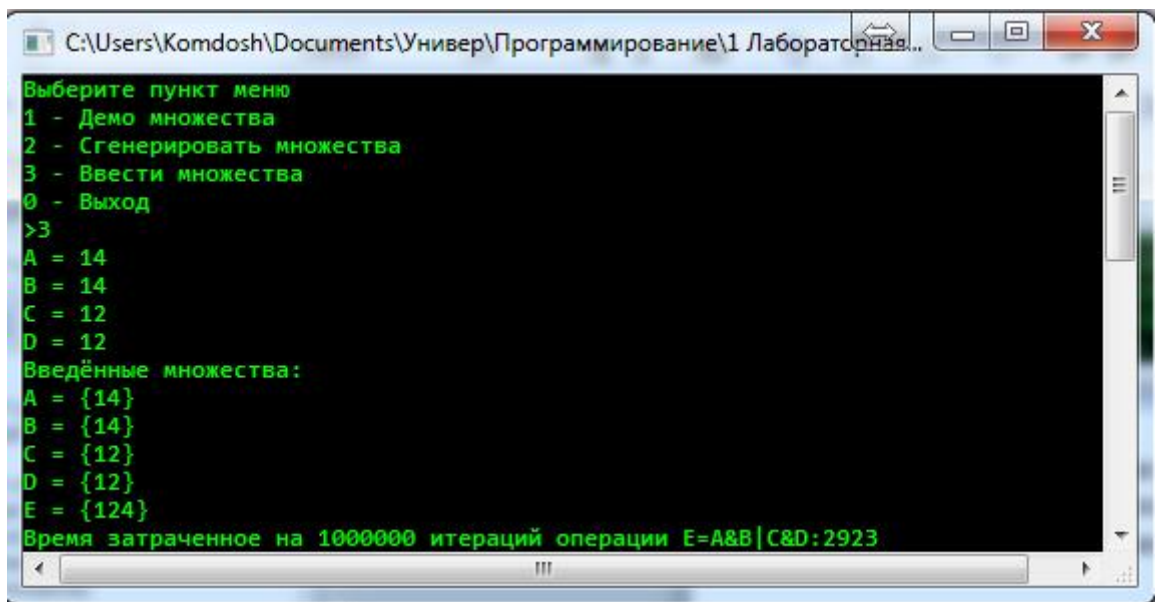
№	Исходные множества				Результат
	A	B	C	D	E
1	0A23	23C5	2389	2389AB	2389
2	14	14	12	12	124
3	0	01	85	12	0
4	23A	24567CD	12ADEF	01348B	12
5	468ACE	123469CE	0239AC	168ABCDE	46ACE

1. Демонстрация работы программы с контрольным примером номер 1 из таблицы контрольных примеров. Способ представления: последовательность символов. Код программы см. приложение 1.1.



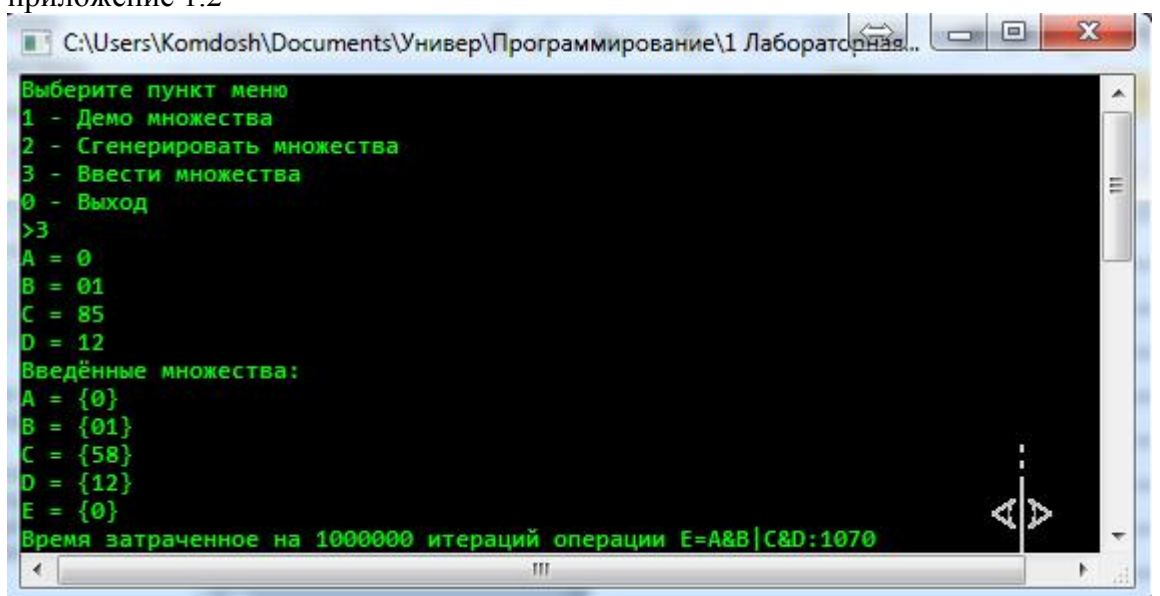
```
Выберите пункт меню
1 - Демо множества
2 - Сгенерировать множества
3 - Ввести множества
0 - Выход
>1
Константные множества:
A = {023A}
B = {235C}
C = {2389}
D = {2389AB}
E = {2389}
Время затраченное на 1000000 итераций операции E=A&B|C&D:3811
```

2. Демонстрация работы программы с контрольным примером номер 2 из таблицы контрольных примеров. Способ представления: последовательность символов. Код программы см. приложение 1.1



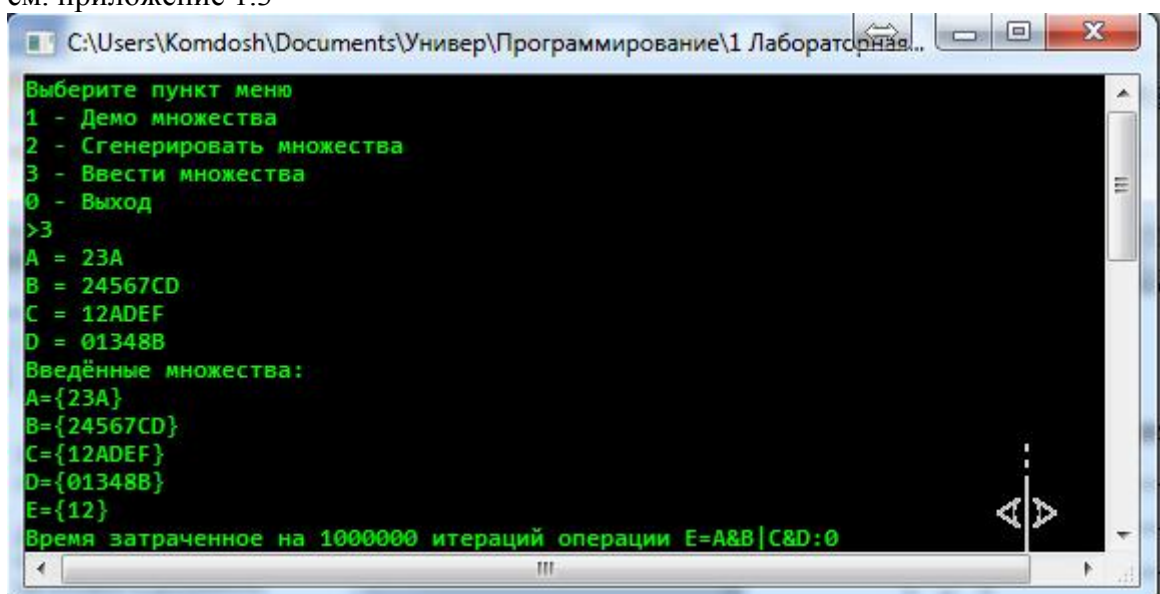
```
C:\Users\Komdosh\Documents\Универ\Программирование\1 Лабораторная...
Выберите пункт меню
1 - Демо множества
2 - Сгенерировать множества
3 - Ввести множества
0 - Выход
>3
A = 14
B = 14
C = 12
D = 12
Введённые множества:
A = {14}
B = {14}
C = {12}
D = {12}
E = {124}
Время затраченное на 1000000 итераций операции E=A&B|C&D:2923
```

3. Демонстрация работы программы с контрольным примером номер 3 из таблицы контрольных примеров. Способ представления: список. Код программы см. приложение 1.2



```
C:\Users\Komdosh\Documents\Универ\Программирование\1 Лабораторная...
Выберите пункт меню
1 - Демо множества
2 - Сгенерировать множества
3 - Ввести множества
0 - Выход
>3
A = 0
B = 01
C = 85
D = 12
Введённые множества:
A = {0}
B = {01}
C = {58}
D = {12}
E = {0}
Время затраченное на 1000000 итераций операции E=A&B|C&D:1070
```

4. Демонстрация работы программы с контрольным примером номер 1 из таблицы контрольных примеров. Способ представления: машинное слово. Код программы см. приложение 1.3



```
C:\Users\Komdosh\Documents\Универ\Программирование\1 Лабораторная...
Выберите пункт меню
1 - Демо множества
2 - Сгенерировать множества
3 - Ввести множества
0 - Выход
>3
A = 23A
B = 24567CD
C = 12ADEF
D = 01348B
Введённые множества:
A={23A}
B={24567CD}
C={12ADEF}
D={01348B}
E={12}
Время затраченное на 1000000 итераций операции E=A&B|C&D:0
```

5. Демонстрация работы программы с контрольным примером номер 1 из таблицы контрольных примеров. Способ представления: массив битов. Код программы см. приложение 1.4

```

C:\Users\Komdosh\Documents\ЛэштхЁ\ЁюуЁрьшЁютрэш\1 трсюЁретоЁ..
Выберите пункт меню
1 - Демо множества
2 - Сгенерировать множества
3 - Ввести множества
0 - Выход
>3
A = 468ACE
B = 123469CE
C = 0239AC
D = 168ABCDE
Введённые множества:
A={...4.6.8.A.C.E.}
B={.1234.6..9..C.E.}
C={0.23.....9A.C...}
D={.1....6.8.ABCDE.}
E={...4.6...A.C.E.}
Средняя мощность множеств: 6
Время затраченное на 1000000 итераций операции E=A&B|C&D:252

```

Временная сложность

Временная сложность представлена в таблице 2.

Таблица. 2. Временная сложность

Способ представления	Ожидаемая	Фактическая
Последовательность	$O(n^2)$	$O(n^2)$
Список	$O(n^2)$	$O(n^2)$
Машинное слово	$O(1)$	$O(1)$
Массив битов	$O(1)$	$O(1)$

Результаты измерения времени обработки

Результаты измерения времени обработки представлены в таблице 3.

Таблица. 3. Результаты измерения времени обработки

Способ представления	Количество тиков	Количество повторов цикла	Зависимость от количества в множестве
Последовательность	4321-9451	1000000	есть
Список	3246-14277		есть
Машинное слово	2-3		нет
Массив битов	600-650		нет

Вывод: Машинное слово самый быстрый из способов формирования множества, т.к. данный способ не зависит от количества элементов в множестве.

Классы и объекты

Использование классов и объектов представлено в приложении 1.4.

Вывод: Использование классов и объектов облегчают понимание программы. Дают возможность защиты переменных от несанкционированного изменения.

Результаты решения задачи

При выполнении программы были получены результаты, совпадающие со значениями, приведенными в таблице 1. Ошибок не обнаружено.

Вывод

При выполнении лабораторной работы были получены практические навыки работы с логическими операциями над множествами на языке программирования «C/C++».

Список используемых источников

- Алгоритмы и структуры данных: методические указания к лабораторным работам, практическим занятиям и курсовому проектированию. Федеральный образовательный стандарт / сост.: П.Г. Колинко. - СПб.: Изд-во СПбГЭТУ "ЛЭТИ", 2014. - 63 с.
- Освой C++ самостоятельно за 21 день. Сиддхартха Рао. 688 стр., с ил.; ISBN 978-5-8459-1825-3; 7 издание.
- <http://stackoverflow.com> – Сайт вопросов и ответов по программированию.
- <http://cyberforum.ru> – Форум программистов и сисадминов.

Приложение 1

Листинги программ

1.1. Способ представления: последовательность символов

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <time.h>

using namespace std;

enum{u=16, timer=1000000};

//*****
//Прототипы
int menu();
string inputStr(char);
string strOr(string, string);
string strAnd(string, string);
string genStr();
string delReply(string);
//*****
//Основная функция
int main(int argc, char** argv)
{
    srand(time(NULL));
    setlocale(0, ".1251");
    string A,B,C,D,E;
    int pause;
    do
    {
        switch(pause = menu())
        {
            case 1:
                A=delReply("0A23"); B=delReply("23C5"); C=delReply("2389"); D=delReply("2389AB");
                cout<<"Константные множества:"<<endl;
                break;
            case 2:
                A=genStr(); B=genStr(); C=genStr(); D=genStr();
                cout<<"Сгенерированные множества:"<<endl;
                break;
            case 3:
                A=delReply(inputStr('A')); B=delReply(inputStr('B')); C=delReply(inputStr('C'));
                D=delReply(inputStr('D'));
                cout<<"Введённые множества:"<<endl;
                break;
            case 0:
                cout<<"До новых встреч!"<<endl;
                break;
            default:
                cout<<"Такого пункта не существует, повторите ввод!"<<endl;
        }
        if(pause)
        {
            clock_t timeStart=clock();
            for(int i=0; i< timer; ++i)
                E=delReply(strOr(strAnd(A, B), strAnd(C,D)));
            clock_t timeEnd=clock();
            cout<<"A = {"<<A<<"}"<<endl; cout<<"B = {"<<B<<"}"<<endl;
            cout<<"C = {"<<C<<"}"<<endl; cout<<"D = {"<<D<<"}"<<endl; cout<<"E = {"<<E<<"}"<<endl;
            cout<<"Время затраченное на "<<timer<<" итераций операции E=A&B|C&D:" \
```

```

        <<(timeEnd-timeStart)<<endl;
    }
}
while(pause);
return 0;
}
//*****
//Функции
int menu()
{
    int point;
    do{
        cin.clear();
        cin.sync();
        cout << "Выберите пункт меню" << endl;
        cout << "1 - Демо множества" << endl;
        cout << "2 - Сгенерировать множества" << endl;
        cout << "3 - Ввести множества" << endl;
        cout << "0 - Выход" << endl;
        cout << ">";
        cin >> point;
        if(cin.fail())
            cout<<"Что-то пошло не так, выберите пункт меню повторно"<<endl;
    }
    while(cin.fail());
    return point;
}
//*****
string inputStr(char name)
{
    string str;
    string const universum("0123456789ABCDEF");
    int mBool, j;
    do
    {
        mBool=1;
        cout<<name<<" = ";
        cin>>str;
        transform(str.begin(), str.end(), str.begin(), ::toupper);
        for(int i=0; i<str.length() && mBool; ++i)
            for(j=0, mBool=0; j<u && !mBool; ++j)
                if(str[i]==universum[j])
                    mBool=1;
        if(!mBool)
            cout<<"Вы ввели недопустимое множество, повторите ввод!"<<endl;
    }while(!mBool);
    return str;
}
//*****
string strOr(string str1, string str2)
{
    return str1+str2;
}
//*****
string strAnd(string str1, string str2)
{
    string str;
    for(int i=0; i<str1.length(); ++i)
        for(int j=0; j<str2.length(); ++j)
            if(str1[i]==str2[j])
                str+=str1[i];
}

```

```

    return str;
}
//*****
string genStr()
{
    string str;
    string const universum("0123456789ABCDEF");
    for(int i=0; i<u; ++i)
        if(rand()%3)
            str+=universum[i];
    return str;
}
//*****
string delReply(string str)
{
    string result;
    string const universum("0123456789ABCDEF");
    int bits[u]={0};
    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
                bits[j]=1;
    for(int i=0; i<u; ++i)
        if(bits[i])
            result+=universum[i];
    return result;
}

```

1.2.Способ представления: списки

```

#include <iostream>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <time.h>

using namespace std;

enum {u=16, timer=1000000};

struct sLIST
{
    char x;
    sLIST *next;
    ~sLIST() {if(next)delete next;}
};
typedef sLIST* LIST;
//*****
//Прототипы
int menu();
string inputStr(char);
string genStr();
string delReply(string);
LIST delReplyInList(LIST);
LIST initialEl(LIST, string);
LIST process(LIST, LIST, LIST);
void output(LIST, const char);
LIST del(LIST el);
LIST inverseList(LIST list);
//*****
//Основная функция
int main(int argc, char** argv)
{

```



```

srand(time(NULL));
setlocale(0, ".1251");
LIST A=NULL,B=NULL,C=NULL,D=NULL,E=NULL;
int pause;
do
{
    switch(pause = menu())
    {
        case 1:
            A = initialEl(A, delReply("0A23")); B = initialEl(B, delReply("23C5"));
            C = initialEl(C, delReply("2389")); D = initialEl(D, delReply("2389AB"));
            cout<<"Константные множества:"<<endl;
            break;
        case 2:
            A = initialEl(A, genStr()); B = initialEl(B, genStr()); C = initialEl(C, genStr());
            D = initialEl(D, genStr());
            cout<<"Сгенерированные множества:"<<endl;
            break;
        case 3:
            A = initialEl(A, delReply(inputStr('A'))); B = initialEl(B, delReply(inputStr('B')));
            C = initialEl(C, delReply(inputStr('C'))); D = initialEl(D, delReply(inputStr('D')));
            cout<<"Введённые множества:"<<endl;
            break;
        case 0:
            cout<<"До новых встреч!"<<endl;
            break;
        default:
            cout<<"Такого пункта не существует, повторите ввод!"<<endl;
    }
    if(pause)
    {
        clock_t timeStart=clock();
        for(int i=0; i< timer; ++i)
        {
            E = process(A, B, E);
            E = process(C, D, E);
            E = delReplyInList(E);
        }
        clock_t timeEnd=clock();
        output(A, 'A'); output(B, 'B'); output(C, 'C'); output(D, 'D'); output(E, 'E');
        A=del(A); B=del(B); C=del(C); D=del(D); E=del(E);
        cout<<"Время затраченное на "<<timer<<" итераций операции E=A&B|C&D:" \
            <<(timeEnd-timeStart)<<endl;
    }
}
while(pause);

return 0;
}
//*****
//Функции
int menu()
{
    int point;
    do{
        cin.clear();
        cin.sync();
        cout << "Выберите пункт меню" << endl;
        cout << "1 - Демо множества" << endl;
        cout << "2 - Сгенерировать множества" << endl;
        cout << "3 - Ввести множества" << endl;

```

```

    cout << "0 - Выход" << endl;
    cout << ">";
    cin >> point;
    if(cin.fail())
        cout<<"Что-то пошло не так, выберите пункт меню повторно"<<endl;
    }
    while(cin.fail());
    return point;
}
//*****
string inputStr(char name)
{
    string str;
    string const universum("0123456789ABCDEF");
    int mBool, j;
    do
    {
        mBool=1;
        cout<<name<<" = ";
        cin>>str;
        transform(str.begin(), str.end(), str.begin(), ::toupper);
        for(int i=0; i<str.length() && mBool; ++i)
            for(j=0, mBool=0; j<u && !mBool; ++j)
                if(str[i]==universum[j])
                    mBool=1;
        if(!mBool)
            cout<<"Вы ввели недопустимое множество, повторите ввод!"<<endl;
    }while(!mBool);
    return str;
}
//*****
string genStr()
{
    string str;
    string const universum("0123456789ABCDEF");
    for(int i=0; i<u; ++i)
        if(rand()%3)
            str+=universum[i];
    return str;
}
//*****
string delReply(string str)
{
    string result;
    string const universum("0123456789ABCDEF");
    int bits[u]={0};
    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
                bits[j]=1;
    for(int i=0; i<u; ++i)
        if(bits[i])
            result+=universum[i];
    return result;
}
//*****
LIST delReplyInList(LIST list)
{
    LIST result;
    string const universum("0123456789ABCDEF");
    string str;

```

```

int bits[u]={0};
for(; list; list=list->next)
    for(int j=0; j<u; ++j)
        if(list->x==universum[j] && !bits[j])
            bits[j]=1;
for(int i=0; i<u; ++i)
    if(bits[i])
        str+=universum[i];
return initialEl(result, str);
}
//*****
void output(LIST el, const char c)
{
    if(el)
    {
        cout << c << " = {";
        for(;el; el=el->next)
            cout << el->x;
        cout << "}" << endl;
    }
}
//*****
LIST initialEl(LIST el, string M)
{
    el = new sLIST;
    LIST startList = el;
    if(M[0])
    {
        el->x = M[0];
        for(int i=1; M[i]; ++i)
        {
            LIST B = new sLIST;
            B->x = M[i];
            el->next = B;
            el = el->next;
        }
        el->next = NULL;
    }
    return startList;
}
//*****
LIST del(LIST el)
{
    LIST t;
    while(!el)
    {
        t = el;
        el = el->next;
        delete t;
    }
    return NULL;
}
//*****
LIST process(LIST oSource, LIST sSource, LIST E)
{
    LIST sTemp=sSource;
    for(; oSource; oSource = oSource->next)
        for(sSource=sTemp; sSource; sSource=sSource->next)
            if((oSource->x)==(sSource->x))
                {

```

```

        LIST el = new sLIST;
        el->x = oSource->x;
        el->next=E;
        E=el;
    }
    return E;
}
//*****
LIST inverseList(LIST list)
{
    LIST ptr=NULL, tmp;
    while (list)
    {
        tmp=list->next;
        list->next=ptr;
        ptr=list;
        list=tmp;
    }
    return ptr;
}

```

1.3.Способ представления: машинное слово

```

#include <iostream>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <time.h>

using namespace std;

enum {u=16, timer=1000000};

//*****
//Прототипы
int menu();
unsigned short int inputStr(char);
string delReply(string);
void outputByUniversum(unsigned short int, char);
unsigned short int myRand();
//*****
//Основная функция
int main(int argc, char** argv)
{
    srand(time(NULL));
    setlocale(0, ".1251");
    unsigned short int A,B,C,D,E;
    int pause;
    do
    {
        switch(pause = menu())
        {
            case 1:
                A=1036; B=12532; C=2352134; D=2331; //23A 24567CD 12ADEF 01348B
                cout<<"Константные множества:"<<endl;
                break;
            case 2:
                A=myRand(); B=myRand(); C=myRand(); D=myRand();
                cout<<"Сгенерированные множества:"<<endl;
                break;
            case 3:
                A=inputStr('A'); B=inputStr('B'); C=inputStr('C'); D=inputStr('D');
                cout<<"Введённые множества:"<<endl;

```

```

        break;
    case 0:
        cout<<"До новых встреч!"<<endl;
        break;
    default:
        cout<<"Такого пункта не существует, повторите ввод!"<<endl;
    }
    if(pause)
    {
        clock_t timeStart=clock();
        for(int i=0; i< timer; ++i)
            E=A&B|C&D;
        clock_t timeEnd=clock();
        outputByUniversum(A, 'A');
        outputByUniversum(B, 'B');
        outputByUniversum(C, 'C');
        outputByUniversum(D, 'D');
        outputByUniversum(E, 'E');
        cout<<"Время затраченное на "<<timer<<" итераций операции E=A&B|C&D:" \
            <<(timeEnd-timeStart)<<endl;
    }
}
while(pause);
return 0;
}
//*****
//Функции
int menu()
{
    int point;
    do{
        cin.clear();
        cin.sync();
        cout << "Выберите пункт меню" << endl;
        cout << "1 - Демо множества" << endl;
        cout << "2 - Сгенерировать множества" << endl;
        cout << "3 - Ввести множества" << endl;
        cout << "0 - Выход" << endl;
        cout << ">";
        cin >> point;
        if(cin.fail())
            cout<<"Что-то пошло не так, выберите пункт меню повторно"<<endl;
    }
    while(cin.fail());
    return point;
}
//*****
unsigned short int inputStr(char name)
{
    string str;
    string const universum("0123456789ABCDEF");
    int mBool, j;
    unsigned short int twoBytes=0;
    do
    {
        mBool=1;
        cout<<name<<" = ";
        cin>>str;
        transform(str.begin(), str.end(), str.begin(), ::toupper);
        for(int i=0; i<str.length() && mBool; ++i)
            for(j=0, mBool=0; j<u && !mBool; ++j)

```

```

        if(str[i]==universum[j])
            mBool=1;
    if(!mBool)
        cout<<"Вы ввели недопустимое множество, повторите ввод!"<<endl;
    }while(!mBool);
    str = delReply(str);

    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j])
                twoBytes+=1<<j;
    return twoBytes;
}
//*****
string delReply(string str)
{
    string result;
    string const universum("0123456789ABCDEF");
    int bits[u]={0};
    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
                bits[j]=1;
    for(int i=0; i<u; ++i)
        if(bits[i])
            result+=universum[i];
    return result;
}
//*****
void outputByUniversum(unsigned short int num, char name)
{
    cout << name << "="<< "{";
    string const universum("0123456789ABCDEF");
    for (int i=0; i<u; ++i)
        if (num >> i & 0x1)
            cout << universum[i];
    cout<<"}"<<endl;
}
//*****
unsigned short int myRand()
{
    if(RAND_MAX<32768)
        return (rand()*3)%65536;
    else
        return rand()%65536;
}

```

Приложение 2

Листинги программ с классами

2.1. Способ представления: массив битов с использованием классов

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <time.h>

using namespace std;

enum{u=16, timer=1000000};

class MySet
{
private:
    int bits[u], power;
    string str;
    char name;
public:
    MySet(char iName = 'T'):name(iName),power(0){}

    //перегружаем операторы
    MySet operator | (const MySet &) const;
    MySet operator & (const MySet &) const;
    MySet operator = (const MySet &);
    //геттеры
    void getInf();
    string getMas(){return str;};
    char getName(){return name;};
    int getPower(){return power;};
    //сеттеры
    void setName(char iName){name=iName;};
    void setBits(int* iBits){for(int i=0; i<u; ++i) bits[i]=iBits[i];}
    void setStr(string st){str=st; power=str.length();}
    //доп. функции
    void strToBits(string);
    void genSet();
    ~MySet(){}
};

//*****
//Перегрузка операторов MySet
MySet MySet::operator & (const MySet & rightExp) const
{
    MySet temp;
    for (int i = 0; i < u; i++)
        if (bits[i] && rightExp.bits[i]) temp.bits[i] = 1;
    return temp;
}
//*****
MySet MySet::operator | (const MySet & rightExp) const
{
    MySet temp;
    for(int i = 0; i<u; ++i)
        if(bits[i] || rightExp.bits[i]) temp.bits[i] = 1;
    return temp;
}
//*****
MySet MySet::operator = (const MySet & rightExp)
{

```

```

    if (this != &rightExp)
    {
        power=0;
        for(int i=0; i<u; ++i)
        {
            bits[i]=rightExp.bits[i];
            if(bits[i]) ++power;
        }
        name = 'E';
    }
    return *this;
}
//*****
//Функции-члены MySet
void MySet::getInf()
{
    cout << name << "=";
    string const universum("0123456789ABCDEF");
    for (int i=0; i<u; ++i)
        if (bits[i])
            cout << universum[i];
        else
            cout<< ".";
    cout<<"}"<<endl;
}
//*****
void MySet::genSet()
{
    power=0;
    for (int i = 0; i < u; ++i) {bits[i]=0;}
    for (int i = 0; i < u; ++i)
        if (rand () % 3)
        {
            bits[i]=1;
            ++power;
        }
}
//*****
void MySet::strToBits(string str)
{
    power=0;
    for (int i = 0; i < u; ++i) bits[i]=0;
    string const universum("0123456789ABCDEF");
    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
            {
                bits[j]=1;
                ++power;
            }
}
//*****
//Прототипы функций
int menu();
string inputStr(char);
//*****
//Основная функция
int main(int argc, char** argv)
{
    srand(time(NULL));

```



```

setlocale(0, ".1251");
int pause;
MySet A('A'), B('B'), C('C'), D('D'), E('E');
do
{
    switch(pause = menu())
    {
        case 1:
            A.strToBits("0A23"); B.strToBits("23C5"); C.strToBits("2389"); D.strToBits("2389AB");
            cout<<"Константные множества:"<<endl;
            break;
        case 2:
            A.genSet(); B.genSet(); C.genSet(); D.genSet();
            cout<<"Сгенерированные множества:"<<endl;
            break;
        case 3:
            A.strToBits(inputStr(A.getName())); B.strToBits(inputStr(B.getName()));
            C.strToBits(inputStr(C.getName())); D.strToBits(inputStr(D.getName()));
            cout<<"Введённые множества:"<<endl;
            break;
        case 0:
            cout<<"До новых встреч!"<<endl;
            break;
        default:
            cout<<"Такого пункта не существует, повторите ввод!"<<endl;
    }
    if(pause)
    {
        clock_t timeStart=clock();
        for(int i=0; i< timer; ++i)
            E=A&B|C&D;
        clock_t timeEnd=clock();
        A.getInf(); B.getInf(); C.getInf(); D.getInf(); E.getInf();
        cout<<"Средняя мощность множеств:
            "<<(A.getPower()+B.getPower()+C.getPower()+D.getPower()+E.getPower())/5<<endl;
        cout<<"Время затраченное на "<<timer<<" итераций операции E=A&B|C&D:"
            <<(timeEnd-timeStart)<<endl;
    }
}
while(pause);
return 0;
}
//*****
//Функции
int menu()
{
    int point;
    do{
        cin.clear();
        cin.sync();
        cout << "Выберите пункт меню" << endl;
        cout << "1 - Демо множества" << endl;
        cout << "2 - Сгенерировать множества" << endl;
        cout << "3 - Ввести множества" << endl;
        cout << "0 - Выход" << endl;
        cout << ">";
        cin >> point;
        if(cin.fail())
            cout<<"Что-то пошло не так, выберите пункт меню повторно"<<endl;
    }
    while(cin.fail());
}

```

```

    return point;
}
//*****
string inputStr(char name)
{
    string str;
    string const universum("0123456789ABCDEF");
    int mBool, j;
    do
    {
        mBool=1;
        cout<<name<<" = ";
        cin>>str;
        transform(str.begin(), str.end(), str.begin(), ::toupper);
        for(int i=0; i<str.length() && mBool; ++i)
            for(j=0, mBool=0; j<u && !mBool; ++j)
                if(str[i]==universum[j])
                    mBool=1;
        if(!mBool)
            cout<<"Вы ввели недопустимое множество, повторите ввод!"<<endl;
    }while(!mBool);
    return str;
}

```

2.2. Способ представления: строка с использованием классов

```

#include <iostream>
#include <cstdlib>
#include <cstring>
#include <algorithm>
#include <time.h>
#include <string>

using namespace std;

enum{u=16, timer=1000};

class MySet
{
private:
    int bits[u], power;
    string str;
    char name;
public:
    MySet(char iName = 'T'):name(iName),power(0){}

    //перегружаем операторы
    MySet operator | (const MySet &) const;
    MySet operator & (const MySet &) const;
    MySet operator = (const MySet &);
    //геттеры
    void getInf();
    string getMas(){return str;};
    char getName(){return name;};
    int getPower(){return power;};
    //сеттеры
    void setName(char iName){name=iName;};
    void setBits(int* iBits){for(int i=0; i<u; ++i) bits[i]=iBits[i];};
    void setStr(string st){str=st; power=str.length();};
    //доп. функции
    void strToBits(string);
}

```

```

        void genSet();
        ~MySet() {}
};

MySet MySet::operator & (const MySet & rightExp) const
{
    MySet temp;
    for(int i=0; i<rightExp.str.length(); ++i)
        for(int j=0; j<str.length(); ++j)
            if(rightExp.str[i]==str[j])
                temp.str+=str[j];
    return temp;
}
//*****

MySet MySet::operator | (const MySet & rightExp) const
{
    MySet temp;
    temp.setStr(str+rightExp.str);
    return temp;
}
//*****

MySet MySet::operator = (const MySet & rightExp)
{
    if (this != &rightExp)
    {
        power=0;
        str=rightExp.str;
    }
    return *this;
}
//*****
//Функции-члены MySet
void MySet::getInf()
{
    cout << name << "="<<endl;
    cout<<str;
    cout<<"}"<<endl;
}
//*****

void MySet::genSet()
{
    power=0;
    string const universum("0123456789ABCDEF");
    for(int i=0; i<u; ++i)
        if(rand()%3)
            {str+=universum[i]; power++;}
}
//*****
//Перегрузка операторов MySet
MySet MySet::operator & (const MySet & rightExp) const
{
    MySet temp;
    for (int i = 0; i < u; ++i)
        temp.bits[i] = (bits[i] && rightExp.bits[i]) ;
    return temp;
}
//*****

MySet MySet::operator | (const MySet & rightExp) const
{
    MySet temp;
    for(int i = 0; i<u; ++i)

```

```

        if(bits[i] || rightExp.bits[i]) temp.bits[i] = 1;
    return temp;
}
//*****
MySet MySet::operator = (const MySet & rightExp)
{
    if (this != &rightExp)
    {
        power=0;
        for(int i=0; i<u; ++i)
        {
            bits[i]=rightExp.bits[i];
            if(bits[i]) ++power;
        }

    }
    return *this;
}
//*****
//Функции-члены MySet
void MySet::getInf()
{
    cout << name << "=";
    string const universum("0123456789ABCDEF");
    for (int i=0; i<u; ++i)
        if (bits[i])
            cout << universum[i];
        else
            cout<< ".";
    cout<<"}"<<endl;
}
//*****
void MySet::genSet()
{
    power=0;
    for (int i = 0; i < u; ++i) {bits[i]=0;}
    for (int i = 0; i < u; ++i)
        if (rand () % 3)
        {
            bits[i]=1;
            ++power;
        }
}
//*****
void MySet::strToBits(string str)
{
    power=0;
    for (int i = 0; i < u; ++i) bits[i]=0;
    string const universum("0123456789ABCDEF");
    for(unsigned int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
            {
                bits[j]=1;
                ++power;
            }
}

}*/
//*****
//Прототипы функций
int menu();

```

```

string inputStr(char);
string delReply(string str);
//*****
//Основная функция
int main(int argc, char* argv[])
{

    srand(time(NULL));
    setlocale(0, ".1251");
    int pause;
    MySet A('A'), B('B'), C('C'), D('D'), E('E');
    do
    {
        switch(pause = menu())
        {
            case 1:
                A.setStr("0A23"); B.setStr("23C5"); C.setStr("2389"); D.setStr("2389AB");
                cout<<"Константные множества:"<<endl;
                break;
            case 2:
                A.genSet(); B.genSet(); C.genSet(); D.genSet();
                cout<<"Сгенерированные множества:"<<endl;
                break;
            case 3:
                A.setStr(inputStr(A.getName())); B.setStr(inputStr(B.getName())); C.setStr(inputStr(C.getName()));
                D.setStr(inputStr(D.getName()));
                cout<<"Введённые множества:"<<endl;
                break;
            case 0:
                cout<<"До новых встреч!"<<endl;
                break;
            default:
                cout<<"Такого пункта не существует, повторите ввод!"<<endl;
        }
        if(pause)
        {
            clock_t timeStart=clock();
            for(int i=0; i< timer; ++i)
                E=(A&B)|(C&D);
            clock_t timeEnd=clock();
            E.setStr(delReply(E.getMas()));
            A.getInf(); B.getInf(); C.getInf(); D.getInf(); E.getInf();
            cout<<"Средняя мощность множеств:
                "<<(A.getPower()+B.getPower()+C.getPower()+D.getPower()+E.getPower())/5<<endl;
            cout<<"Время затраченное на "<<timer<<" итераций операции E=A&B|C&D:"
                <<(timeEnd-timeStart)<<endl;
        }
    }
    while(pause);
    return 0;
}
//*****
//Функции
int menu()
{
    int point;
    do{
        cin.clear();
        cin.sync();
        cout << "Выберите пункт меню" << endl;
        cout << "1 - Демо множества" << endl;

```

```

    cout << "2 - Сгенерировать множества" << endl;
    cout << "3 - Ввести множества" << endl;
    cout << "0 - Выход" << endl;
    cout << ">";
    cin >> point;
    if(cin.fail())
        cout<<"Что-то пошло не так, выберите пункт меню повторно"<<endl;
    }
    while(cin.fail());
    return point;
}
//*****
string inputStr(char name)
{
    string str;
    string const universum("0123456789ABCDEF");
    int mBool, j;
    do
    {
        mBool=1;
        cout<<name<<" = ";
        cin>>str;
        transform(str.begin(), str.end(), str.begin(), ::toupper);
        for(int i=0; i<str.length() && mBool; ++i)
            for(j=0, mBool=0; j<u && !mBool; ++j)
                if(str[i]==universum[j])
                    mBool=1;
        if(!mBool)
            cout<<"Вы ввели недопустимое множество, повторите ввод!"<<endl;
    }while(!mBool);
    return str;
}
//*****
string delReply(string str)
{
    string result;
    string const universum("0123456789ABCDEF");
    int bits[u]={0};
    for(int i=0; i<str.length(); ++i)
        for(int j=0; j<u; ++j)
            if(str[i]==universum[j] && !bits[j])
                bits[j]=1;
    for(int i=0; i<u; ++i)
        if(bits[i])
            result+=universum[i];
    return result;
}

```