**МИНОБРНАУКИ РОССИИ**
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**
**Кафедра Вычислительной техники**

# ОТЧЕТ

**по Лабораторной работе №2**

**по дисциплине «Алгоритмы и структуры данных»**

**Тема: Множество как объект**

Студент гр. 1306 _____ Пестерев В.А.

Преподаватель _____ Колинько П.Г.

Санкт-Петербург

2022

**Введение**

Цель работы: исследование эффекта от использования классов

**Задание (вариант №27)**

Преобразовать программу, созданную в ходе выполнения Лабораторной работы №1, так, чтобы множества были объектами некоторого класса, а операции над ними — функциями членами этого класса.

**Результаты эксперимента**

В качестве контрольных примеров приведены тесты на случайно сгенерированных множествах с различной средней длиной, использованных в Лабораторной работе №1.

Исходные множества и результаты их обработки представлены на рисунках 1-5.

```
A = [f e 9 8 6 4 1 ]
B = [f d c b 8 6 ]
C = [d c b a 9 7 4 2 0 ]
D = [e b a 9 8 3 1 0 ]
E = [1 4 6 8 9 b e f ]
time: 1797
```

Рисунок 1 — Вариант «списки»

```
A(8) = [034789ab]
B(10) = [01245789cf]
C(9) = [345678abf]
D(8) = [0489bcdf]
E(9) = [034789abf]
time: 1690
```

Рисунок 2 — Вариант «массивы символов»

```
A(8) = [034789ab]
B(10) = [01245789cf]
C(9) = [345678abf]
D(8) = [0489bcdf]
E(9) = [034789abf]
time: 214
```

Рисунок 3 — Вариант «машинные слова»

```
Создано A(3) = [3ef]
Создано B(6) = [1589ef]
Создано C(7) = [4578bde]
Создано D(12) = [012345678bcf]
Создано пустое E(0) = []
A(3) = [3ef]
B(6) = [1589ef]
C(7) = [4578bde]
D(12) = [012345678bcf]
Создано F(6) = [1589ef] из B(6) = [49954]
Вычислено F(6) = [1589ef] = C & B
Вычислено F(3) = [58e] = F & C
Создано F(3) = [58e] из F(3) = [16672]
Удалено F(3) = [58e]
Создано G(3) = [58e] из F(3) = [16672]
Вычислено G(3) = [58e] = D & F
Вычислено G(2) = [58] = G & D
Создано G(2) = [58] из G(2) = [288]
Удалено G(2) = [58]
Создано H(3) = [3ef] из A(3) = [49160]
Вычислено H(3) = [3ef] = G | A
Вычислено H(5) = [358ef] = H | G
Создано I(5) = [358ef] из H(5) = [49448]
Удалено H(5) = [358ef]
E = I(5) = [358ef]
Удалено I(5) = [358ef]
Удалено G(2) = [58]
Удалено F(3) = [58e]
E(5) = [358ef]
Удалено E(5) = [358ef]
Удалено D(12) = [012345678bcf]
Удалено C(7) = [4578bde]
Удалено B(6) = [1589ef]
Удалено A(3) = [3ef]
```

Рисунок 4 — Отслеживание вызовов функций-членов в варианте «машинные слова»

```
Создано A(8) = [034789ab]
Создано B(10) = [01245789cf]
Создано C(9) = [345678abf]
Создано D(8) = [0489bcdf]
Создано пустое E(0) = []
A(8) = [034789ab]
B(10) = [01245789cf]
C(9) = [345678abf]
D(8) = [0489bcdf]
Создано F(10) = [01245789cf] из B(10) = [01245789cf]
Создано G(10) = [01245789cf] из F(10) = [01245789cf]
Вычислено F(5) = [4578f] = G & C
Удалено G(10) = [01245789cf]
Вычислено F(5) = [4578f] = C & B
Создано G(5) = [4578f] из F(5) = [4578f]
Удалено F(5) = [4578f]
Создано H(5) = [4578f] из G(5) = [4578f]
Создано I(5) = [4578f] из H(5) = [4578f]
Вычислено H(3) = [48f] = I & D
Удалено I(5) = [4578f]
Вычислено H(3) = [48f] = D & G
Создано I(3) = [48f] из H(3) = [48f]
Удалено H(3) = [48f]
Создано J(8) = [034789ab] из A(8) = [034789ab]
Выполнена сортировка J(9) = [034789abf]
Вычислено J(9) = [034789abf] = J | I
Вычислено J(9) = [034789abf] = I | A
Создано K(9) = [034789abf] из J(9) = [034789abf]
Удалено J(9) = [034789abf]
E = K(9) = [034789abf]
Удалено K(9) = [034789abf]
Удалено I(3) = [48f]
Удалено G(5) = [4578f]
E(9) = [034789abf]
Удалено E(9) = [034789abf]
Удалено D(8) = [0489bcdf]
Удалено C(9) = [345678abf]
Удалено B(10) = [01245789cf]
Удалено A(8) = [034789ab]
```

Рисунок 5 — Отслеживание вызовов функций-членов в варианте «массивы символов»

Сравнение результатов измерений времени:

*Таблица 1: Результаты измерений времени без использования классов*

| Средняя мощность множеств | Время обработки | | | |
|---|---|---|---|---|
| | Массивы симсволов | Списки | Массивы битов | Машин. слова |
| 8,75 | 498 | 2419 | 48 | 2 |
| 7,5 | 331 | 1780 | 44 | 2 |
| 7 | 281 | 1559 | 42 | 3 |

*Таблица 2: Результаты измерений времени с использованием классов*

| Средняя мощность множеств | Время обработки | | | |
|---|---|---|---|---|
| | Массивы символов | Списки | Массивы битов | Машин. слова |
| 8,75 | 1667 | 2589 | 1195 | 207 |
| 7,5 | 1425 | 1797 | 1194 | 202 |
| 7 | 1328 | 1480 | 1196 | 210 |

**Вывод**

Основываясь на результатах измерений времени, можно сделать следующие выводы:

1. Для массивов символов, массивов битов и машинных слов время обработки в разы увеличилось, однако машинные слова все еще занимают первое место по скорости, массивы битов — второе, а массивы символов — третье;
2. Зависимость времени обработки массивов битов от средней мощности множеств больше не проявляется, так как теперь выполняются все условные операции, независимо от расположения;
3. Время обработки списков не изменилось;
4. Наибольшие изменения во времени обработки произошли в варианте «машинные слова», увеличилось примерно в 100 раз.

Таким образом, наиболее целесообразно применять классы при представлении множеств в виде списков, потому что время обработки не меняется, а написание кода становится проще.

В случае же с машинными словами применение классов оказалось нецелесообразно, так как время обработки увеличилось примерно в 100 раз, а все разработанные операции изначально были доступны для машинных слов.

**Список литературы**

Колинько П. Г. Алгоритмы и структуры данных. Часть 1. Пользовательские структуры данных: Конспект лекций. Вып. 2201. — СПб.: СПбГЭТУ «ЛЭТИ», 2021. — 565с.

Колинько П. Г. Пользовательские структуры данных: Методические указания по дисциплине «Алгоритмы и структуры данных, часть 1». — СПб.: СПбГЭТУ «ЛЭТИ», 2022 — 64 с. (вып.2209)

Шилдт Г. C++ шаг за шагом. Пер. С англ. М.: — Экон Паблишерс, 2009. — 640с.: ил.

Страуструп Б. Язык программирования C++. Специальное издание. Пер. с англ. — М.: Изд-во Бином, 2015 — 1136 с.: ил.

## Приложение

Исходные текст программы main.cpp:

```
//============================================================================
// Name        : lab2.cpp
// Author      : Pesterev
// Version     :
// Description : Исследование возможностей обработки множеств как объектов
//============================================================================

#include <iostream>
#include <stdlib.h>
#include <ctime>
#include <iomanip>

using namespace std;
#include "array.h"
//#include "list.h"
//#include "bits_arr.h"
//#include "bits.h"

int main() {
	locale::global(locale("Russian"));
	srand(time(nullptr));

	int iterate = 1000000;

	//El::isOutput = true;
	Set::isOutput = true;

	Set A("034789ab"), B("01245789cf"), C("345678abf"), D("0489bcdf"), E;
	//Set A("14689ef"), B("68bcdf"), C("02479abcd"), D("01389abe"), E;
	//Set A("3ef"), B("1589ef"), C("4578bde"), D("012345678bcf"), E;

	clock_t t;

	/*
	A.fillRand();
	B.fillRand();
	C.fillRand();
	D.fillRand();
	 */
	A.show();
	B.show();
	C.show();
	D.show();

	t = clock();
	for(int i = 0; i < (A.isOutput ? 1 : iterate); i++) E = (A | (B & C &
D));
	t = clock() - t;

	E.show();

	if(!A.isOutput) cout <<"time: "<< t<< endl;

	return 0;
}
```

Исходный текст заголовочного файла array.h:

```
/*
 *  array.h
 *
 *  Created on: 9 окт. 2022 г.
 *      Author: Pesterev
 */

#include <cstdlib>
#include <random>
#include <cstring>

#pragma once
using namespace std;

class Set {
private:
	static int N, cnt;
	static char univers[];
	int n;
	char S, *A;
public:
	static bool isOutput;

	Set();
	Set(const Set&);
	Set(char *B);

	Set& operator &=(const Set &B);
	Set& operator |=(const Set &B);

	Set operator |(const Set&) const;
	Set operator &(const Set&) const;
	Set& operator =(const Set &B);

	~Set() {
		if(isOutput) cout << "Удалено " << S << "(" << dec << n << ") = ["
<< A << "]" << endl;
		delete[] A;
	}

	void show();
	void fillRand();
	void alphSort(const Set &B);

};
int Set::N = 16;
int Set::cnt = 0;
char Set::univers[] = "0123456789abcdef";
bool Set::isOutput = false;

void Set::show() {
	cout << S <<"(" << n << ")" << " = [" << A << "]" << endl;
}

void Set::fillRand() {
	n = 0;
	for (int i = 0; i < N; ++i)
		if (rand() % 2)
			A[n++] = univers[i];

	A[n] = 0;
```

```
	cout << "Заполнено случайными числами " << S <<"(" << n << ")" << " = ["
<< A << "]" << endl;
}

void Set::alphSort(const Set &B){
	char tmp;
	for (int i = 0; B.A[i]; ++i) {
			for (int j = i + 1; B.A[j]; ++j) {
				if (B.A[i] > B.A[j]) {
					tmp = B.A[i];
					B.A[i] = B.A[j];
					B.A[j] = tmp;
				}
			}
		}

	if(isOutput) cout << "Выполнена сортировка " << B.S <<"(" << B.n << ")"
<< " = [" << B.A << "]" << endl;
}

Set::Set(char *B) : n(0), S('A' + cnt++), A(new char[N + 1])  {
	for (int i = 0; B[i]; ++i)
		A[n++] = B[i];

	A[n] = 0;
	if(isOutput) cout <<"Создано " << S <<"(" << n << ")" << " = [" << A <<
"]" << endl;
}

Set::Set() :
		n(0), S('A' + cnt++), A(new char[N + 1]) {
	A[0] = 0;

	if(isOutput) cout <<"Создано пустое " << S <<"(" << n << ")" << " = ["
<< A << "]" << endl;
}

Set::Set(const Set &B) :
		S('A' + cnt++), n(B.n), A(new char[N + 1]) {
	char *dst(A), *src(B.A);
	while (*dst++ = *src++);
	if(isOutput) cout <<"Создано " << S <<"(" << n << ")" << " = [" << A <<
"] из " << B.S <<"(" << B.n << ")" << " = [" << B.A << "]" << endl;
}

Set& Set::operator &=(const Set &B) {
	Set C(*this);
	n = 0;
	for (int i = 0; i < C.n; ++i) {
		for (int j = 0; j < B.n; j++) {
			if (C.A[i] == B.A[j])
				A[n++] = C.A[i];
		}
	}
	A[n] = 0;

	if(isOutput) cout << "Вычислено " << S <<"(" << n << ")" << " = [" << A
<< "] = " << C.S <<" & " << B.S << endl;
	cnt--;
	return *this;
}

Set Set::operator &(const Set &B) const {
```

```
	Set C(*this);
	C &= B;
	if(isOutput) cout << "Вычислено " << C.S <<"(" << C.n << ")" << " = ["
<< C.A << "] = " << B.S <<" & "<< S << endl;
	return C;
}

Set& Set::operator |=(const Set &B) {
	bool f;
	for (int i = 0; i < B.n; i++) {
		f = true;
		for (int j = 0; j < n; ++j)
			if (B.A[i] == A[j])
				f = false;
		if (f)
			A[n++] = B.A[i];
	}
	A[n] = 0;

	alphSort(*this);

	if(isOutput) cout << "Вычислено " << S <<"(" << n << ")" << " = [" << A
<< "] = " << S <<" | " << B.S << endl;
	return *this;
}

Set Set::operator |(const Set &B) const {
	Set C(*this);
	C |= B;
	if(isOutput) cout << "Вычислено " << C.S <<"(" << C.n << ")" << " = ["
<< C.A << "] = " << B.S <<" | "<< S << endl;
	return C;

}

Set& Set::operator =(const Set &B) {
	if (this != &B) {
		char *dst(A), *src(B.A);
		n = B.n;
		while (*dst++ = *src++)
			;
	}

	if(isOutput) cout << S << " = " << B.S <<"("<<n<<") = ["<< A << "]"<<
endl;
	return *this;
}
```

Исходный текст заголовочного файла list.h:

```
/*
 *  list.h
 *
 *  Created on: 10 окт. 2022 г.
 *      Author: Pesterev
 */

#pragma once
using namespace std;

class El {
    char e;
    El* next;
    static const int maxmup = 100000000;
    static El mem[maxmup]; //Свободная память для элементов спис-ков
    static int mup, mup0;
public:
    static bool isOutput;
    El() : e('!'), next(nullptr) { }
    El(char e, El* n = nullptr) : e(e), next(n) { if(isOutput) cout << "+" <<
e; }
    ~El() {
        if (this) { //Проверка обязательна
            if (next) { delete next; }
            if(isOutput) cout << "-" << e;
            e = '*';
        }
        else if(isOutput) cout << "<Пусто!>";
    }
    static void* operator new(size_t) {
        return (mup < maxmup ? &mem[mup++] : nullptr);
    }
    static void operator delete(void*, size_t) { }
    static void mark() { mup0 = mup; }    //Фиксировать состояние памяти
    static void release() { mup = mup0; }  //Сбросить до фиксированного
    static void clear() { mup = 0; }    //Очистить память полностью
    friend class Set;
    friend std::ostream& operator << (std::ostream& o, El& S);
    friend static void memOut();
};
std::ostream& operator << (std::ostream& o, El& S)
{
    for (El* p = &S; p; p = p->next) o << p->e;
    return o;
}

bool El::isOutput = false;
El El::mem[El::maxmup];  //"Свободная память"
int El::mup = 0, El::mup0 = 0;

void memOut()  //Вывод использованного содержимого "свободной памяти"
{
    cout << "\n\nПамять элементов списков (всего - " << dec << El::mup << ")\
n";
    for (int i = 0; i < El::mup; ++i) cout << El::mem[i].e;
}


class Set
{
private:
```

```
    static int N, cnt;      //Порядковый номер множества
    int n;    //Мощность множества
    char S;    //Тег
    El* A;    //Список элементов
    static char univers[];

public:
    static bool isOutput;

    Set();    //Пустое множество
    Set(char *B);
    Set(const Set&);
    Set& operator = (const Set&);
    ~Set() {
        if(isOutput) cout << "Удалено " << S << "(" << std::dec << n << ") =
[" << *A << "]";
        A->El::~El();
        if(isOutput) cout << std::endl;
    }

    void show();
    void alphSort(const Set &B);
    int power() { return n; }
    void swap(Set& other) { std::swap(S, other.S); std::swap(n, other.n);
std::swap(A, other.A); }
    Set& operator |= (const Set&);
    Set& operator &= (const Set&);
    Set operator | (const Set&) const;
    Set operator & (const Set&) const;
};

int Set::N = 16;
int Set::cnt = 0;
char Set::univers[] = "0123456789abcdef";
bool Set::isOutput = false;

Set::Set() : n(0), S('A' + cnt++), A(nullptr)
{
    if(isOutput) cout << "-> Создано " << S << "(" << n << ") = [" << *A <<
"] \n";
}

Set::Set(char *B) : n(0), S('A' + cnt++){
      A = nullptr;
      for (int i = 0; B[i]; ++i)
            {
            A = new El(B[i], A);
            ++n;
            }

      if(isOutput) cout << "-> Создано " << S << "(" << n << ") = [" << *A <<
"] \n";
}

Set::Set(const Set& B) : n(B.n), S('A' + cnt++), A(nullptr)
{
    for (El* p = B.A; p; p = p->next) A = new El(p->e, A);
    if(isOutput) cout << "-> Создано " << S << "(" << n << ") = [" << *A <<
"] из " << B.S << std::endl;
}

void Set::alphSort(const Set &B){
      char tmp;
```

```
      for (El *p = B.A; p; p = p->next) {
                        for (El *q = p->next; q; q = q->next) {
                              if (p->e > q->e) {
                                    tmp = p->e;
                                    p->e = q->e;
                                    q->e = tmp;
                              }
                        }
                  }

      if(isOutput) cout << "Выполнена сортировка " << B.S <<"(" << B.n << ")"
<< " = [" << *B.A << "]" << endl;
}

Set& Set::operator &= (const Set& B)
{
    Set C;
    for (El* i = A; i; i = i->next)
    {
        for (El* j = B.A; j; j = j->next)
            if (i->e == j->e)
                C.A = new El(i->e, C.A), ++C.n;
    }
    swap(C);
    if(isOutput) cout << "; Получено " << S << "(" << n << ") = [" << *A <<
"] = " << C.S << "&" << B.S << endl;
    return *this;
}
Set Set::operator & (const Set& B) const
{
    Set C(*this);
    if(isOutput) cout << "Вычисление " << C.S << " & " << B.S << endl;
    return C &= B;
}

Set& Set::operator |= (const Set& B)
{
    Set C(*this);
    for (El* i = B.A; i; i = i->next)
    {
        bool f = true;
        for (El* j = A; f && j; j = j->next)
            f = f && (i->e != j->e);
        if (f)
            C.A = new El(i->e, C.A), ++C.n;
    }
    swap(C);
    alphSort(*this);
    if(isOutput) cout << "; Получено " << S << "(" << n << ") = [" << *A <<
"] = " << C.S << "|" << B.S << endl;
    return *this;
}
Set Set::operator | (const Set& B) const
{
    Set C(*this);
    if(isOutput) cout << "Вычисление " << C.S << " | " << B.S << endl;
    return C |= B;
}

Set& Set::operator = (const Set& B)
{
    if (this != &B)
    {
```

```
        if(isOutput) cout << "\nУдалено " << S << "(" << n << ") = [" << *A
<< "]";
        delete A;
        A = nullptr;
        n = 0;
        for (El* p = B.A; p; p = p->next)
            A = new El(p->e, A), ++n;

    }
    if(isOutput) cout << "; Создано " << S << "(" << n << ") = [" << *A << "]
из " << B.S << std::endl;
    return *this;
}

void Set::show()
{
    cout << S <<  " = [";
    for (El* p = A; p; p = p->next) cout << p->e << " ";
    cout << "]\n";
}
```

Исходный текст заголовочного файла bits_arr.h:

```
/*
 * bits_arr.h
 *
 *  Created on: 10 окт. 2022 г.
 *      Author: vecto
 */

#pragma once
using namespace std;

class Set {
private:
	static int N, cnt;
	static char univers[];
	int n;
	char S;
	bool *A;

public:
	static bool isOutput;

	Set();
	Set(char *B);
	Set(const Set&);

	Set& operator &=(const Set &B);
	Set& operator |=(const Set &B);

	Set operator |(const Set&) const;
	Set operator &(const Set&) const;
	Set& operator =(const Set &B);

	~Set() {
		if (isOutput) {
			cout << "Удалено " << S << "(" << n << ") = [";
			printA();
			cout << "]" << endl;
		}
		delete[] A;
	}

	void show();
	void fillRand();
	void printA();
};

int Set::N = 16;
int Set::cnt = 0;
char Set::univers[] = "0123456789abcdef";
bool Set::isOutput = false;

void Set::show() {
	cout << S << "(" << n << ") = [";
	printA();
	cout <<']'<<endl;
}

void Set::fillRand() {
	n = 0;
	for (int i = 0; i < N; ++i)
		if(A[i] = rand()%2) ++n;
```

```
	if (isOutput) {
		cout << "Заполнено случайными числами " << S << "(" << n << ")"
				<< " = [";
		printA();
		cout << "]" << endl;
	}

}

void Set::printA(){
	for(int i =0; i<N; ++i) if(A[i]) cout << univers[i];
}

Set::Set(char *B) : n(0), S('A' + cnt++), A(new bool[N]{})  {
	for (int j = 0; j < N; ++j)
		for (int i = 0; B[i]; ++i)
			if(univers[j] == B[i]){
				A[j] = true;
				++n;
			}

	if (isOutput) {
		cout << "Создано " << S << "(" << n << ")" << " = [";
		printA();
		cout << "]" << endl;
	}
}

Set::Set() :
		n(0), S('A' + cnt++), A(new bool[N]{}){
	if (isOutput) {
		cout << "Создано пустое " << S << "(" << n << ")" << " = [";
		printA();
		cout << "]" << endl;
	}
}

Set::Set(const Set &B) :
		S('A' + cnt++), n(B.n), A(new bool[N]{}) {
	bool *dst(A), *src(B.A);
	for (int i = 0; i < N; ++i) (*dst++ = *src++);
	if (isOutput) {
		cout << "Создано " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] из " << B.S << "(" << B.n << ")" << " = [" << B.A <<
"]"
				<< endl;
	}
}

Set& Set::operator &=(const Set &B) {
	n=0;
	for(int i = 0; i<N; ++i){
		if(A[i]&=B.A[i]) ++n;;
	}

	if (isOutput) {
		cout << "Вычислено " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] = " << S << " & " << B.S << endl;
	}
		cnt--;
```

```
	return *this;
}

Set Set::operator &(const Set &B) const {
	Set C(*this);
	if (isOutput) {
		cout << "Вычислено " << C.S << "(" << C.n << ")" << " = [";
		C.printA();
		cout << "] = " << B.S << " & " << S << endl;
	}
	return (C &= B);
}

Set& Set::operator |=(const Set &B) {
	n=0;
	for (int i = 0; i < N; ++i) {
		if(A[i] |= B.A[i]) ++n;
	}
	if (isOutput) {
		cout << "Вычислено " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] = " << S << " | " << B.S << endl;
	}
	return *this;
}

Set Set::operator |(const Set &B) const {
	Set C(*this);
	if (isOutput) {
		cout << "Вычислено " << C.S << "(" << C.n << ")" << " = [";
		C.printA();
		cout << "] = " << B.S << " | " << S << endl;
	}
	return (C |= B);

}

Set& Set::operator =(const Set &B) {
	if (this != &B) {
		bool *dst(A), *src(B.A);
		n = B.n;
		for (int i = 0; i < N; ++i) (*dst++ = *src++);
	}
	if (isOutput) {
		cout << S << " = " << B.S << "(" << n << ") = [";
		printA();
		cout << "]" << endl;
	}
	return *this;
}
```

Исходный текст заголовочного файла bits.h:

```
/*
 * bits.h
 *
 *  Created on: 20 окт. 2022 г.
 *      Author: Pesterev
 */

#pragma once
using namespace std;

class Set {
private:
	static int N, cnt;
	static char univers[];
	int n;
	char S;
	unsigned short A;

public:
	static bool isOutput;

	Set();
	Set(char *B);
	Set(const Set&);

	Set& operator &=(const Set &B);
	Set& operator |=(const Set &B);

	Set operator |(const Set&) const;
	Set operator &(const Set&) const;
	Set& operator =(const Set &B);

	~Set() {
		if (isOutput) {
			cout << "Удалено " << S << "(" << n << ") = [";
			printA();
			cout << "]" << endl;
		}
	}

	void show();
	void fillRand();
	void printA();
};

int Set::N = 16;
int Set::cnt = 0;
char Set::univers[] = "0123456789abcdef";
bool Set::isOutput = false;

void Set::printA() {
	for (int i = 0; i < N; ++i)
		if (A & (1 << i))
			cout << univers[i];
}

void Set::show() {
	cout << S << "(" << n << ") = [";
	printA();
	cout << ']' << endl;
}
```

```
void Set::fillRand() {
	n = 0;
	A = rand() % UCHAR_MAX;
	for (int i = 0; i < N; ++i)
		if (A & (1 << i))
			++n;
}

Set::Set(char *B) :
		n(0), S('A' + cnt++), A(0) {
	for (int j = 0; j < N; ++j)
		for (int i = 0; B[i]; ++i)
			if (univers[j] == B[i]) {
				A += 1 << j;
				++n;
			}

	if (isOutput) {
		cout << "Создано " << S << "(" << n << ")" << " = [";
		printA();
		cout << "]" << endl;
	}
}

Set::Set() :
		n(0), S('A' + cnt++), A(0) {

	if (isOutput) {
		cout << "Создано пустое " << S << "(" << n << ")" << " = [";
		printA();
		cout << "]" << endl;
	}
}

Set::Set(const Set &B) :
		S('A' + cnt++), n(B.n), A(0) {
	A = B.A;
	if (isOutput) {
		cout << "Создано " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] из " << B.S << "(" << B.n << ")" << " = [" << B.A <<
"]"
			<< endl;
	}
}

Set& Set::operator &=(const Set &B) {
	A &= B.A;

	n = 0;
	for (int i = 0; i < N; ++i)
		if (A & (1 << i))
			++n;

	if (isOutput) {
		cout << "Вычислено " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] = " << S << " & " << B.S << endl;
	}
	cnt--;

	return *this;
```

```
}

Set Set::operator &(const Set &B) const {
	Set C(*this);
	if (isOutput) {
		cout << "Вычислено " << C.S << "(" << C.n << ")" << " = [";
		C.printA();
		cout << "] = " << B.S << " & " << S << endl;
	}
	return (C &= B);
}

Set& Set::operator |=(const Set &B) {
	A |= B.A;

	n = 0;
	for (int i = 0; i < N; ++i)
		if (A & (1 << i))
			++n;

	if (isOutput) {
		cout << "Вычислено " << S << "(" << n << ")" << " = [";
		printA();
		cout << "] = " << S << " | " << B.S << endl;
	}
	return *this;
}

Set Set::operator |(const Set &B) const {
	Set C(*this);
	if (isOutput) {
		cout << "Вычислено " << C.S << "(" << C.n << ")" << " = [";
		C.printA();
		cout << "] = " << B.S << " | " << S << endl;
	}
	return (C |= B);

}

Set& Set::operator =(const Set &B) {
	if (this != &B) {
		n = B.n;
		A = B.A;
	}
	if (isOutput) {
		cout << S << " = " << B.S << "(" << n << ") = [";
		printA();
		cout << "]" << endl;
	}
	return *this;
}
/* BITS_H_ */
```