

Московский Авиационный Институт  
(Национальный Исследовательский Университет)

Кафедра 806 «Вычислительная информатика и программирование»  
Факультет: «Информационные технологии и прикладная математика»

Лабораторная работа  
Дисциплина: «Объектно-ориентированное программирование»  
II семестр  
Задание 1: «Простые классы»

Москва, 2019

1.Тема: Простые классы

1.Цель работы: Изучение системы сборки на языке C++, изучение систем контроля версии. Изучение основ работы с классами в C++.

1.Задание (вариант 7):

Создать класс для работы с 128-битовыми строками. Битовая строка должна быть представлена двумя полями типа `uint64_t`. Должны быть реализованы все традиционные операции для работы с битами: `and`, `or`, `xor`, `not`. Реализовать сдвиг влево и сдвиг вправо на заданное количество битов. Реализовать операцию вычисления количества единичных битов, операцию сравнения по количеству единичных битов и операцию проверки включения.

1.Адрес репозитория на GitHub [https://github.com/DragonKeker/oop\\_exercise\\_01](https://github.com/DragonKeker/oop_exercise_01)

1.Код программы на C++

logika.h

```
#ifndef D_LOGIKA_H_
#define D_LOGIKA_H_ 1
#include <iostream>

struct logika {

    logika _and (const logika& obj2) const;
    logika _or(const logika& obj2) const;
    logika _xor(const logika& obj2) const;
    logika _not() const;
    logika _shiftLeft(const int k) const;
    logika _shiftRight(const int k) const;
    int _count1() const;
    void _sravnenie(const logika& obj2) const;
    bool _vkluch(const logika& obj2) const;

public:
    uint64_t field1;
    uint64_t field2;
    static void read(std::istream&, logika&);
    void write(std::ostream& );
};

#endif // D_LOGIKA_H_#pragma once
```

logika.cpp

```
#include "logika.h"

void logika::read(std::istream& is, logika& obj) {
    is >> obj.field1 >> obj.field2;
}
void logika::write(std::ostream& os) {
    os << field1 << " " << field2<<"\n";
}
logika logika::_and(const logika& obj2) const
{
```

```

        logika result;
        result.field1 = field1 & obj2.field1;
        result.field2 = field2 & obj2.field2;
        return(result);
};
logika logika::_or(const logika& obj2)const
{
    logika result;
    result.field1 = field1 | obj2.field1;
    result.field2 = field2 | obj2.field2;
    return(result);
};
logika logika::_xor(const logika& obj2)const
{
    logika result;
    result.field1 = field1 ^ obj2.field1;
    result.field2 = field2 ^ obj2.field2;
    return(result);
};
logika logika::_not()const
{
    logika result;
    result.field1 = ~field1;
    result.field2 = ~field2;
    return(result);
};
logika logika::_shiftLeft(const int k)const
{
    logika result;
    uint64_t a = field1, b = field2;
    unsigned long long pow63 = 1;
    for (int i = 0; i < 63; i++) {
        pow63 *= 2;
    }

    for (int i = 0; i < k; i++) {
        a = a << 1;
        if (b >= pow63) {
            a += 1;
        }
        b = b << 1;
    }
    result.field1 = a;
    result.field2 = b;
    return (result);
};
logika logika::_shiftRight(const int k)const
{
    logika result;
    uint64_t a=field1, b=field2;

    unsigned long long pow63 = 1;
    for (int i = 0; i < 63; i++) {
        pow63 *= 2;
    }
    for (int i = 0; i < k; i++) {
        b = b >> 1;
        if (a % 2 == 1) {
            b += pow63;
        }
        a = a >> 1;
    }

    result.field1 = a;

```

```

        result.field2 = b;
        return (result);
    };
    int logika::_count1()const
    {
        uint64_t a = field1;
        uint64_t b = field2;
        int t = 0;
        int sum = 0;
        while (a != 0) {
            t += 1; a &= a - 1;
        }
        sum = sum + t;

        t = 0;
        while (b != 0) {
            t += 1; b &= b - 1;
        }
        sum = sum + t;

        return(sum);
    };

    void logika::_sravnenie(const logika& obj2)const
    {
        if (this->_count1() > obj2._count1())
        {
            std::cout << "Количество единиц в первом числе больше" << std::endl;
        }
        else if (this->_count1() < obj2._count1())
        {
            std::cout << "Количество единиц во втором числе больше" << std::endl;
        }
        else
        {
            std::cout << "Количество единиц в числах равно" << std::endl;
        }
    }; //делаем лабы

    bool logika::_vkluch(const logika& obj2)const {
        logika result;
        logika obj;
        if ((obj2.field1 == this->_and(obj2).field1) && (obj2.field2 == this->_and(
obj2).field2))
            return (1);
        else
            return 0;
    };
};

```

main.cpp

```

#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include "logika.h"

int main() {
    setlocale(LC_ALL, "RUSSIAN");
    int action;
    int k;

    logika obj1;
    logika obj2;
    printf("Введите данные\n");
}

```

```

obj1.read(std::cin, obj1);
obj2.read(std::cin, obj2);
obj1._and( obj2).write(std::cout);
obj1._or( obj2).write(std::cout);
obj1._xor( obj2).write(std::cout);
obj1._not().write(std::cout);
printf("Введите : k ");
scanf_s("%d", &k);
obj1._shiftLeft(k).write(std::cout);
printf("Введите : k ");
scanf_s("%d", &k);
obj1._shiftRight(k).write(std::cout);
std::cout << "Всего единиц в первой строке: " << (obj1._count1()) << "\n";
obj1._sravnenie( obj2);
if (obj1._vkluch( obj2))
    std::cout << "Включает" << std::endl;
else
    std::cout << "Не включает" << std::endl;
}

```

### 1.Набор testcases

1 тест

Ожидаемое действие

Ввод

7 0

1 1

Ввод k = 1

Ввод k = 1

Ожидаемый результат

1 0

7 1

6 1

18446744073709551608 18446744073709551615

Введите : k 1

14 0

Введите : k 1

3 9223372036854775808

Всего единиц в первой строке: 3

Количество единиц в первом числе больше

Не включает

2 тест

test\_02

Ввод

1 1

1 1

Ввод  $k = 2$

Ввод  $k = 2$  1 1

1 1

0 0

18446744073709551614 18446744073709551614

Введите :  $k$  2

4 4

Введите :  $k$  2

0 4611686018427387904

Всего единиц в первой строке: 2

Количество единиц в числах равно

Включает

3 тест test\_03 Ввод

1 7

5 5

Ввод  $k = 3$

Ввод k = 11 5

5 7

4 2

18446744073709551614 18446744073709551608

Введите : k 3

8 56

Введите : k 1

0 9223372036854775811

Всего единиц в первой строке: 4

Количество единиц в числах равно

Не включает

### 1.Результаты выполнения тестов

Введите данные

7 0

1 1

1 0

7 1

6 1

18446744073709551608 18446744073709551615

Введите : k 1

14 0

Введите : k 1

3 9223372036854775808

Всего едениц в первой строке: 3

Количество единиц в первом числе больше

Не включает

C:\Users\Андрей\source\repos\Совершенная 1 лаба\Debug\Совершенная 1 лаба.exe (процесс 17360) завершает работу с кодом 0.

Чтобы автоматически закрывать консоль при остановке отладки, установите параметр "Сервис" -> "Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".

Чтобы закрыть это окно, нажмите любую клавишу...

Введите данные

1 1

1 1

1 1

1 1

0 0

18446744073709551614 18446744073709551614

Введите : k 2

4 4

Введите : k 2  
0 4611686018427387904  
Всего едениц в первой строке: 2  
Количество единиц в числах равно  
Включает

C:\Users\Андрей\source\repos\Совершенная 1 лаба\Debug\Совершенная 1 лаба.exe (процесс 18128)  
завершает работу с кодом 0.  
Чтобы автоматически закрывать консоль при остановке отладки, установите параметр "Сервис" ->  
"Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".  
Чтобы закрыть это окно, нажмите любую клавишу...

Введите данные  
1 7  
5 5  
1 5  
5 7  
4 2  
18446744073709551614 18446744073709551608  
Введите : k 3  
8 56  
Введите : k 1  
0 9223372036854775811  
Всего едениц в первой строке: 4  
Количество единиц в числах равно  
Не включает

C:\Users\Андрей\source\repos\Совершенная 1 лаба\Debug\Совершенная 1 лаба.exe (процесс 16044)  
завершает работу с кодом 0.  
Чтобы автоматически закрывать консоль при остановке отладки, установите параметр "Сервис"->  
"Параметры" -> "Отладка"-> "Автоматически закрыть консоль при остановке отладки".  
Чтобы закрыть это окно, нажмите любую клавишу...

## **1.Объяснение результатов работы программы - вывод**

Были заданы и описаны методы и свойства класса.  
Классы, описывают методы и свойства объектов, позволяют работать с этими объектами,  
не вдаваясь в подробности их реализации, что является примером абстракции данных.  
Данная работа позволила вспомнить особенности побитовых операций в языке с++, а  
также показала удобство абстракции, реализованное с помощью класса.