Московский авиационный институт
(национальный исследовательский университет)

Институт информационных технологий и прикладной математики

Кафедра вычислительной математике и программирования

Лабораторная работа №1 по курсу «Объектно-ориентированное программирование »

Студент: В.П. Будникова

Группа: М8-107Б-19

Москва, 2020

Лабораторная работа №1

Постановка задачи:

Вариант 13

Создать класс Long для работы с целыми беззнаковыми числами из 64 бит. Число должно быть представлено двумя полями unsigned int. Должны быть реализованы арифметические операции, присутствующие в C++, и сравнения.

Ссылка на репозиторий на GitHub: https://github.com/Ler-B/OOP

Решение задачи:

На вход поступают 64-знаковые беззнаковые числа. констуктор класса лонг преобразует число типа unsignet long long в число класса Long состоящее из двух полей типа unsigned int.

Класс Long и его методы:

```
namespace Long
{
    class Long {
    private:
        unsigned int field1;
        unsigned int field2;
    public:
        Long();
        Long(const unsigned long long a);
        Long(const unsigned int a, const unsigned int b);
        ~Long();
        const Long operator=(const unsigned long long a);
        void Print();
        const Long operator+(const Long &b);
        bool operator>(const Long &b) const;
        bool operator<(const Long &b) const;
        bool operator==(const Long &b) const;
        const Long operator-(const Long &b);
        const Long Multiplication_without_longlong(const Long &b);
//Умножение, без использования класса insigned long long
        const Long operator*(const Long &b);
```

```
        const Long Division_without_longlong(const Long &b); //
Целочисленное деление, без использования класса insigned long long
        const Long operator/(const Long &b);
        const Long RemainderOfDivision_without_longlong(const Long
&b); //Нахождение остатка от деления, без использования класса
insigned long long
        const Long operator%(const Long &b);
        friend std::ostream& operator<<(std::ostream &os, const
Long& b);
    };
```

Преобразование входного числа:

```
Long::Long() {
        field1 = 0;
        field2 = 0;
    }

    Long::Long(const unsigned long long a) {
        field1 = a >> 32;
        field2 = a & 0xffffffff;
    }

    Long::Long(const unsigned int a, const unsigned int b) {
        field1 = a;
        field2 = b;
    }
const Long Long::operator=(const unsigned long long a) {
        Long num(a);
        return num;
    }
```

При объявлении числа типа Long его поля становятся равными 0. Если в аргументы будет подано одно число типа unsigned long long, то оно преобразуется в два поля типа unsigned int, поле один получается путем сдвига числа на 32 бита(получаем "верхние" 32 бита числа), второе число получается путем применения & 0xffffffff, следовательно первые 32 бита числа обнуляются и остаются последние 32 бита. Если используется оператор "=", то число аналогично преобразуется. Если

в аргументы будут поданы два числа типа unsigned int, то их значения присвоются соответствующим полям.

Сумма, разность:

```
const Long Long::operator+(const Long &b) {
        Long rez;
        rez.field2 = field2 + b.field2;
        if (((((field2 >> 1) + (b.field2 >> 1)) + ((field2 &
1)&(b.field2 & 1))) >> 31) == 1) {
            rez.field1 = field1 + b.field1 + 1;
        } else {
            rez.field1 = field1 + b.field1;
        }
        return rez;
    }

const Long Long::operator-(const Long &b) {
        Long rez;
        if (field1 < b.field1) {
            std::cout << "Ошибка! Получается отрицательное
число.";
            return rez;
        }
        if (field1 == b.field1 && field2 < b.field2) {
            std::cout << "Ошибка! Получается отрицательное
число.";
            return rez;
        }

        if (field2 > b.field2) {
            rez.field2 = field2 - b.field2;
            rez.field1 = field1 - b.field1;
        }
        if (field2 < b.field2) {
            rez.field1 = field1 - b.field1 - 1;
            unsigned int s = b.field2 - field2;
            int count = 0;
            while ((s & 1) != 1) {
                s = s >> 1;
                count++;
```

```
            }
            rez.field2 = (((((0xffffffff) >> count << count) -
(b.field2 - field2)) >> count) + 1) << count;
        }
        return rez;
    }
```

Сумма и разность чисел считается с помощью побитовых операций и сдвигов. При выполнении разности, если число, из которого вычитают меньше вычитаемого, выдается ошибка, так как числа типа Long беззнаковые.

Сравнения:

```
bool Long::operator>(const Long &b) const {
        if (field1 > b.field1) {
            return true;
        }
        if (field1 == b.field1 && field2 > b.field2) {
            return true;
        }
        return false;
    }

    bool Long::operator<(const Long &b) const {
        if (field1 < b.field1) {
            return true;
        }
        if (field1 == b.field1 && field2 < b.field2) {
            return true;
        }
        return false;
    }

    bool Long::operator==(const Long &b) const {
        if (field1 == b.field1 && field2 == b.field2) {
            return true;
        }
        return false;
    }
```

Сравнения производятся путем сравнений полей соответствующих чисел.

Умножение, Целочисленное деление, Остаток от деления:

```
const Long Long::Multiplication_without_longlong(const Long &b) {
        Long rez;
        if (field1 != 0 && b.field1 != 0) {
            std::cout << "Ошибка! Переполнение.";
            return rez;
        }
        for (unsigned int i = 0; i < field2; ++i) {
            rez = rez + b;
            if ((rez.field1 & 0xffffffff) == 0xffffffff) {
                std::cout << "Ошибка! Переполнение.";
                return rez;
            }
        }
        return rez;

    }

const Long Long::operator*(const Long &b) {
        Long rez;
        unsigned long long a1 = field1;
        unsigned long long a2 = b.field1;
        a1 = (a1 << 32) + field2;
        a2 = (a2 << 32) + b.field2;
        a1 *= a2;
        rez.field1 = a1 >> 32;
        rez.field2 = a1 & 0xffffffff;
        return rez;
    }

    const Long Long::Division_without_longlong(const Long &b) {
        Long rez;
        Long temp;
        const Long temp2(1);
        temp.field1 = field1;
        temp.field2 = field2;
        if (b > temp) {
            return rez;
```

```
        }
        if (rez == b) {
            std::cout << "Ошибка! Делить на ноль нельзя.";
            return rez;
        }
        while ((b < temp) || (b == temp)) {
            temp = temp - b;
            rez = rez + temp2;
        }
        return rez;
    }

    const Long Long::operator/(const Long &b) {
        Long rez;
        unsigned long long a1 = field1;
        unsigned long long a2 = b.field1;
        a1 = (a1 << 32) + field2;
        a2 = (a2 << 32) + b.field2;
        if (a1 < a2) {
            return rez;
        }
        if (rez == b) {
            std::cout << "Ошибка! Делить на ноль нельзя.";
            return rez;
        }
        a1 /= a2;
        rez.field1 = a1 >> 32;
        rez.field2 = a1 & 0xffffffff;
        return rez;
    }

    const Long Long::RemainderOfDivision_without_longlong(const
Long &b) {
        Long rez;
        rez.field1 = field1;
        rez.field2 = field2;
        if (rez < b) {
            rez.field1 = field1;
            rez.field2 = field2;
        }
```

```
    if (b.field1 == 0 && b.field2 == 0) {
        std::cout << "Ошибка! Делить на ноль нельзя.";
        return rez;
    }
    while ((b < rez) || (b == rez)) {
        rez = rez - b;
    }
    return rez;
}

const Long Long::operator%(const Long &b) {
    Long rez;
    unsigned long long a1 = field1;
    unsigned long long a2 = b.field1;
    a1 = (a1 << 32) + field2;
    a2 = (a2 << 32) + b.field2;
    if (a1 < a2) {
        rez.field1 = field1;
        rez.field2 = field2;
    }
    if (rez == b) {
        std::cout << "Ошибка! Делить на ноль нельзя.";
        return rez;
    }
    a1 %= a2;
    rez.field1 = a1 >> 32;
    rez.field2 = a1 & 0xffffffff;
    return rez;
}
```

Операторы умножение, деления и нахождения остатка от деления реализуются через преобразования двух полей в число типа unsigned long long, путем применения побитовых операций. Возможно также умножение, деление и нахождение остатка от деления буз использования типа unsigned long long, они реализованы методами: Multiplication_without_longlong, Division_without_longlong, RemainderOfDivision_without_longlong, там используются циклы, что увеличивает время работы программы при больших числах. Также

при делении производится проверка деления на ноль, при таком случае, выдается соответсвующее сообщение.

Руководство по использованию программы:
На вод программе подаются пары беззнаковых числел типа unsigned long long, до окончания ввода числа считываются, в программе производятся применения операций к числам и результат выдается на экран.
Файлы программы: classlong.hpp(содержит описание класса), classlong.cpp(содержит реализацию методов класса), main.cpp(основной код - применение операций), makefile(собирает и компилирует программу), test_01.txt, test_02.txt, test_03.txt
Чтобы запустить программу требуется использовать make. При вводе make run тестовые файлы автоматически подадутся программе на вход.

Код программы:

main.cpp:

```
//Будникова Валерия М8О-207Б-19
//Создать класс Long для работы с целыми беззнаковыми числами из 64 бит. Число должно быть представлено двумя полями unsigned int.
//Должны быть реализованы арифметические операции, присутствующие в C++, и сравнения.

#include "classlong.hpp"
#include <iostream>

int main() {
    unsigned long long a;
    unsigned long long b;
    while(std::cin >> a >> b) {
        Long::Long a3(a), b3(b);
        std::cout << "СУММА:" << std::endl;
        Long::Long rez(a3 + b3);
        std::cout << a3 << " + " << b3 << " = " << rez << std::endl;
        std::cout << "РАЗНОСТЬ:" << std::endl;
```

```
        rez = a3 - b3;
        std::cout << a3 << " - " << b3 << " = " << rez <<
std::endl;
        std::cout << "ПРОИЗВЕДЕНИЕ:" << std::endl;
        rez = a3 * b3;
        std::cout << a3 << " * " << b3 << " = " << rez <<
std::endl;
        std::cout << "Подсчет произведения, без использования
unsigned long long:" << a3.Multiplication_without_longlong(b3) <<
std:: endl;
        std::cout << "ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:" << std::endl;
        rez = a3 / b3;
        std::cout << a3 << " / " << b3 << " = " << rez <<
std::endl;
        std::cout << "Подсчет деления, без использования unsigned
long long:" << a3.Division_without_longlong(b3) << std:: endl;
        std::cout << "ОСТАТОК ОТ ДЕЛЕНИЯ:" << std::endl;
        rez = a3 % b3;
        std::cout << a3 << " % " << b3 << " = " << rez <<
std::endl;
        std::cout << "Подсчет ост. от деления, без использования
unsigned long long:" <<
a3.RemainderOfDivision_without_longlong(b3) << std:: endl;
        std::cout << "ОТНОШЕНИЕ ЧИСЕЛ:" << std::endl;
        if (a3 > b3) { std::cout << a3 << " > " << b3 <<
std::endl; }
        if (a3 < b3) { std::cout << a3 << " < " << b3 <<
std::endl; }
        if (a3 == b3) { std::cout << a3 << " = " << b3 <<
std::endl; }
        std::cout << std::endl;
    }
}
```

classlong.hpp:

```
#pragma once
#include <iostream>

namespace Long
{
    class Long {
    private:
        unsigned int field1;
        unsigned int field2;
    public:
        Long();
        Long(const unsigned long long a);
        Long(const unsigned int a, const unsigned int b);
        ~Long();
        const Long operator=(const unsigned long long a);
        void Print();
        const Long operator+(const Long &b);
        bool operator>(const Long &b) const;
        bool operator<(const Long &b) const;
        bool operator==(const Long &b) const;
        const Long operator-(const Long &b);
        const Long Multiplication_without_longlong(const Long &b);
//Умножение, без использования класса insigned long long
        const Long operator*(const Long &b);
        const Long Division_without_longlong(const Long &b); //
Целочисленное деление, без использования класса insigned long long
        const Long operator/(const Long &b);
        const Long RemainderOfDivision_without_longlong(const Long
&b); //Нахождение остатка от деления, без использования класса
insigned long long
        const Long operator%(const Long &b);
        friend std::ostream& operator<<(std::ostream &os, const
Long& b);
    };

}
```

classlong.cpp:

```
#include "classlong.hpp"

namespace Long
{
    Long::Long() {
        field1 = 0;
        field2 = 0;
    }

    Long::Long(const unsigned long long a) {
        field1 = a >> 32;
        field2 = a & 0xffffffff;
    }

    Long::Long(const unsigned int a, const unsigned int b) {
        field1 = a;
        field2 = b;
    }

    Long::~Long() {}

    const Long Long::operator=(const unsigned long long a) {
        Long num(a);
        return num;
    }

    void Long::Print(){
        unsigned long long rez = field1;
        rez = (rez << 32) + field2;
        std::cout << rez << std::endl;
    }

    const Long Long::operator+(const Long &b) {
        Long rez;
        rez.field2 = field2 + b.field2;
        if (((((field2 >> 1) + (b.field2 >> 1)) + ((field2 &
1)&(b.field2 & 1))) >> 31) == 1) {
            rez.field1 = field1 + b.field1 + 1;
```

```
        } else {
            rez.field1 = field1 + b.field1;
        }
        return rez;
    }

    bool Long::operator>(const Long &b) const {
        if (field1 > b.field1) {
            return true;
        }
        if (field1 == b.field1 && field2 > b.field2) {
            return true;
        }
        return false;
    }

    bool Long::operator<(const Long &b) const {
        if (field1 < b.field1) {
            return true;
        }
        if (field1 == b.field1 && field2 < b.field2) {
            return true;
        }
        return false;
    }

    bool Long::operator==(const Long &b) const {
        if (field1 == b.field1 && field2 == b.field2) {
            return true;
        }
        return false;
    }

    const Long Long::operator-(const Long &b) {
        Long rez;
        if (field1 < b.field1) {
            std::cout << "Ошибка! Получается отрицательное
число.";
            return rez;
        }
```

```
        if (field1 == b.field1 && field2 < b.field2) {
            std::cout << "Ошибка! Получается отрицательное
число.";
            return rez;
        }

        if (field2 > b.field2) {
            rez.field2 = field2 - b.field2;
            rez.field1 = field1 - b.field1;
        }
        if (field2 < b.field2) {
            rez.field1 = field1 - b.field1 - 1;
            unsigned int s = b.field2 - field2;
            int count = 0;
            while ((s & 1) != 1) {
                s = s >> 1;
                count++;
            }
            rez.field2 = (((((0xffffffff) >> count << count) -
(b.field2 - field2)) >> count) + 1) << count;
        }
        return rez;
    }

    const Long Long::Multiplication_without_longlong(const Long
&b) {
        Long rez;
        if (field1 != 0 && b.field1 != 0) {
            std::cout << "Ошибка! Переполнение.";
            return rez;
        }
        for (unsigned int i = 0; i < field2; ++i) {
            rez = rez + b;
            if ((rez.field1 & 0xffffffff) == 0xffffffff) {
                std::cout << "Ошибка! Переполнение.";
                return rez;
            }
        }
        return rez;
```

```
}

const Long Long::operator*(const Long &b) {
    Long rez;
    unsigned long long a1 = field1;
    unsigned long long a2 = b.field1;
    a1 = (a1 << 32) + field2;
    a2 = (a2 << 32) + b.field2;
    a1 *= a2;
    rez.field1 = a1 >> 32;
    rez.field2 = a1 & 0xffffffff;
    return rez;
}

const Long Long::Division_without_longlong(const Long &b) {
    Long rez;
    Long temp;
    const Long temp2(1);
    temp.field1 = field1;
    temp.field2 = field2;
    if (b > temp) {
        return rez;
    }
    if (rez == b) {
        std::cout << "Ошибка! Делить на ноль нельзя.";
        return rez;
    }
    while ((b < temp) || (b == temp)) {
        temp = temp - b;
        rez = rez + temp2;
    }
    return rez;
}

const Long Long::operator/(const Long &b) {
    Long rez;
    unsigned long long a1 = field1;
    unsigned long long a2 = b.field1;
    a1 = (a1 << 32) + field2;
    a2 = (a2 << 32) + b.field2;
```

```
        if (a1 < a2) {
            return rez;
        }
        if (rez == b) {
            std::cout << "Ошибка! Делить на ноль нельзя.";
            return rez;
        }
        a1 /= a2;
        rez.field1 = a1 >> 32;
        rez.field2 = a1 & 0xffffffff;
        return rez;
    }

    const Long Long::RemainderOfDivision_without_longlong(const
Long &b) {
        Long rez;
        rez.field1 = field1;
        rez.field2 = field2;
        if (rez < b) {
            rez.field1 = field1;
            rez.field2 = field2;
        }
        if (b.field1 == 0 && b.field2 == 0) {
            std::cout << "Ошибка! Делить на ноль нельзя.";
            return rez;
        }
        while ((b < rez) || (b == rez)) {
            rez = rez - b;
        }
        return rez;
    }

    const Long Long::operator%(const Long &b) {
        Long rez;
        unsigned long long a1 = field1;
        unsigned long long a2 = b.field1;
        a1 = (a1 << 32) + field2;
        a2 = (a2 << 32) + b.field2;
        if (a1 < a2) {
            rez.field1 = field1;
```

```
            rez.field2 = field2;
        }
        if (rez == b) {
            std::cout << "Ошибка! Делить на ноль нельзя.";
            return rez;
        }
        a1 %= a2;
        rez.field1 = a1 >> 32;
        rez.field2 = a1 & 0xffffffff;
        return rez;
    }

    std::ostream& operator<<(std::ostream &os, const Long& b) {
        unsigned long long a = b.field1;
        a = (a << 32) + b.field2;
        os << a;
        return os;
    }

}
```

makefile:

```
CC=g++
CFLAGS=-std=c++14 -pedantic
OUTPUT=oop_exercise_01

all:
    $(CC) $(CFLAGS) classlong.cpp main.cpp -o $(OUTPUT)
run:
    ./oop_exercise_01 < test_01.txt
    ./oop_exercise_01 < test_02.txt
    ./oop_exercise_01 < test_03.txt
```

test_01.txt:
0 0
0 975
test_02.txt:
12345678 456
123 3456789087
34532432 1
1 1
test_03.txt:
24 2
45 56
34 56

Результаты тестов:

```
Lera:OOP valeriabudnikova$ make run
./oop_exercise_01 < test_01.txt
СУММА:
0 + 0 = 0
РАЗНОСТЬ:
0 - 0 = 0
ПРОИЗВЕДЕНИЕ:
0 * 0 = 0
Подсчет произведения, без использования unsigned long long:0
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
Ошибка! Делить на ноль нельзя.0 / 0 = 0
Подсчет деления, без использования unsigned long long:Ошибка! Делить на ноль нельзя.0
ОСТАТОК ОТ ДЕЛЕНИЯ:
Ошибка! Делить на ноль нельзя.0 % 0 = 0
Подсчет ост. от деления, без использования unsigned long long:Ошибка! Делить на ноль
нельзя.0
ОТНОШЕНИЕ ЧИСЕЛ:
0 = 0

СУММА:
0 + 975 = 975
РАЗНОСТЬ:
Ошибка! Получается отрицательное число.0 - 975 = 0
ПРОИЗВЕДЕНИЕ:
0 * 975 = 0
Подсчет произведения, без использования unsigned long long:0
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
0 / 975 = 0
Подсчет деления, без использования unsigned long long:0
ОСТАТОК ОТ ДЕЛЕНИЯ:
0 % 975 = 0
Подсчет ост. от деления, без использования unsigned long long:0
ОТНОШЕНИЕ ЧИСЕЛ:
0 < 975

./oop_exercise_01 < test_02.txt
```

```
СУММА:
12345678 + 456 = 12346134
РАЗНОСТЬ:
12345678 - 456 = 12345222
ПРОИЗВЕДЕНИЕ:
12345678 * 456 = 5629629168
Подсчет произведения, без использования unsigned long long:5629629168
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
12345678 / 456 = 27073
Подсчет деления, без использования unsigned long long:27073
ОСТАТОК ОТ ДЕЛЕНИЯ:
12345678 % 456 = 390
Подсчет ост. от деления, без использования unsigned long long:390
ОТНОШЕНИЕ ЧИСЕЛ:
12345678 > 456

СУММА:
123 + 3456789087 = 3456789210
РАЗНОСТЬ:
Ошибка! Получается отрицательное число.123 - 3456789087 = 0
ПРОИЗВЕДЕНИЕ:
123 * 3456789087 = 425185057701
Подсчет произведения, без использования unsigned long long:425185057701
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
123 / 3456789087 = 0
Подсчет деления, без использования unsigned long long:0
ОСТАТОК ОТ ДЕЛЕНИЯ:
123 % 3456789087 = 123
Подсчет ост. от деления, без использования unsigned long long:123
ОТНОШЕНИЕ ЧИСЕЛ:
123 < 3456789087

СУММА:
34532432 + 1 = 34532433
РАЗНОСТЬ:
34532432 - 1 = 34532431
ПРОИЗВЕДЕНИЕ:
34532432 * 1 = 34532432
Подсчет произведения, без использования unsigned long long:34532432
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
34532432 / 1 = 34532432
Подсчет деления, без использования unsigned long long:34532432
ОСТАТОК ОТ ДЕЛЕНИЯ:
34532432 % 1 = 0
Подсчет ост. от деления, без использования unsigned long long:0
ОТНОШЕНИЕ ЧИСЕЛ:
34532432 > 1

СУММА:
1 + 1 = 2
РАЗНОСТЬ:
1 - 1 = 0
ПРОИЗВЕДЕНИЕ:
1 * 1 = 1
Подсчет произведения, без использования unsigned long long:1
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
1 / 1 = 1
Подсчет деления, без использования unsigned long long:1
ОСТАТОК ОТ ДЕЛЕНИЯ:
1 % 1 = 0
```

```
Подсчет ост. от деления, без использования unsigned long long:0
ОТНОШЕНИЕ ЧИСЕЛ:
1 = 1

./oop_exercise_01 < test_03.txt
СУММА:
24 + 2 = 26
РАЗНОСТЬ:
24 - 2 = 22
ПРОИЗВЕДЕНИЕ:
24 * 2 = 48
Подсчет произведения, без использования unsigned long long:48
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
24 / 2 = 12
Подсчет деления, без использования unsigned long long:12
ОСТАТОК ОТ ДЕЛЕНИЯ:
24 % 2 = 0
Подсчет ост. от деления, без использования unsigned long long:0
ОТНОШЕНИЕ ЧИСЕЛ:
24 > 2

СУММА:
45 + 56 = 101
РАЗНОСТЬ:
Ошибка! Получается отрицательное число.45 - 56 = 0
ПРОИЗВЕДЕНИЕ:
45 * 56 = 2520
Подсчет произведения, без использования unsigned long long:2520
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
45 / 56 = 0
Подсчет деления, без использования unsigned long long:0
ОСТАТОК ОТ ДЕЛЕНИЯ:
45 % 56 = 45
Подсчет ост. от деления, без использования unsigned long long:45
ОТНОШЕНИЕ ЧИСЕЛ:
45 < 56

СУММА:
34 + 56 = 90
РАЗНОСТЬ:
Ошибка! Получается отрицательное число.34 - 56 = 0
ПРОИЗВЕДЕНИЕ:
34 * 56 = 1904
Подсчет произведения, без использования unsigned long long:1904
ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ:
34 / 56 = 0
Подсчет деления, без использования unsigned long long:0
ОСТАТОК ОТ ДЕЛЕНИЯ:
34 % 56 = 34
Подсчет ост. от деления, без использования unsigned long long:34
ОТНОШЕНИЕ ЧИСЕЛ:
34 < 56
```

Вывод

При выполнении лабораторной работы я научилась работе с классами на языке C++, создавать объекты класса, реализовывать их методы, перегружать операторы, в том числе оператор вывода. При реализации методов умножения, деления и нахождения остатка от деления без использования типа unsigned long long, я столкнулась с проблемой большого времени выполнения программы, при вводе больших чисел, поэтому я отдельно перегрузила операторы(с использованием unsigned long long) и реализовала методы, не использующие unsigned long long.

Литература

https://ru.wikipedia.org/wiki/Класс_(программирование)
https://ru.wikipedia.org/wiki/Метод_(программирование)
http://cppstudio.com/post/439/