

ФГАОУВО «Южно-Уральский государственный университет (НИУ)»

Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

## ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

по дисциплине «Объектно-ориентированное программирование»

Автор работы

студент группы ЕТ-212

\_\_\_\_\_ Б.А. Мухутдинов

\_\_\_\_\_ 2022 г.

Работа зачтена с оценкой

\_\_\_\_\_

\_\_\_\_\_ А.В. Лут

\_\_\_\_\_ 2022 г.

Челябинск, 2022

# 1 Постановка задачи

## I. Реализовать класс

14. Длина в милях, ярдах, футах, дюймах Length

1 миля=1760 ярдов

1 ярд=3 фута=36 дюймов

Конструктор: `Length(m[,y[,f[,i]]])`

Операции:

`x+y, x-y`, (увеличить/уменьшить длину на соответствующую длину)

`x+=y, x-=y`,

`z*x, x/z`, (увеличить/уменьшить длину в `z` раз)

`x*=z, x/=z`,

`x==y, x!=y, x<y, x>y, x<=y, x>=y`,

`!x` (`x` равна нулю)

где `x, y` - длины, `z` - целое число  $\geq 1$

вывод, ввод в виде `2m 5y 2f 4i`

Методы:

`int getMile(); [0..∞]`

`int getYard(); [0..1759]`

`int getFoot(); [0..2]`

`int getInch(); [0..11]`

Операции (если есть в задании) `=`, `[]`, `+=`, `-=`, `*=`, `/=`, префиксные `++`, `--` определять как методы.

Для ввода переопределить `>>`, для вывода - `<<`. Формат ввода-вывода объектов делать так, как указано в задании.

Запись `[текст]` означает, что `текст` может отсутствовать, например, конструктор `ИмяКласса(a[,b[,c]])` может быть вызван с 1, 2 или 3 аргументами.

Пример:

```
Vector a(0,0),b(1.5,0.3);
```

```
cout<<"Введите вектор a:"<<endl;
```

```
cin>>a;
```

```
//нужно ввести (1.2,3.2) вместе со скобками и запятой
```

```
cout<<a<<" + "<<b<<" = "<<(a+b)<<endl;
```

```
// печатается (1.2,3.2) + (1.5,0.3) = (2.7,3.5)
```

II. Реализовать `main` с тестами (создание объектов и выполнение действий с ними)

III. Написать отчет

- Постановка задачи
- Описание интерфейса класса (`class {}` и комментарии ко всем полям, методам и функциям)
- Описание тестов для проверки классов (`main` с комментариями, какие действия выполнялись, полученные результаты)
- Листинг реализации класса (реализация методов и функций, отступы, без комментариев)

## 2 Описание интерфейса класса

```
class Length{

private:

    int inch;//переменная для хранения длины, все хранится в
    дюймах

public:

    //конструктор

    Length();

    Length(int);

    Length(int, int);

    Length(int, int, int);

    Length(int, int, int, int);

    //деструктор

    ~Length(){};

    void set_inch(int);//сеттер для получения доступа извне

    //геттеры

    int getMile();

    int getYard();

    int getFoot();

    int getInch();

    //перегрузка операторов

    friend Length operator+(const Length &, const Length &);

    friend Length operator-(const Length &, const Length &);

    friend Length operator*(int, const Length &);

    friend Length operator/(const Length &, int);

    Length &operator+=(const Length &);

    Length &operator-=(const Length &);

    Length &operator*=(int);

    Length &operator/=(int);

    friend bool operator==(Length, Length);

    friend bool operator!=(Length, Length);
```

```

friend bool operator<(Length, Length);
friend bool operator>(Length, Length);
friend bool operator<=(Length, Length);
friend bool operator>=(Length, Length);
friend bool operator!(Length);
friend istream& operator>>(istream&, Length&);
friend ostream& operator<<(ostream&, Length&);
};

```

### 3 Описание тестов для проверки классов

```

int main(){
    Length a(0);
    Length b(0);
    Length c(0);
    cout << "a: ";
    cin >> a;
    cout << "b: ";
    cin >> b;
    if(!c)
        cout << "c равно 0" << endl;
    else
        cout << "c не равно 0" << endl;
    cout << "a: " << a << endl;
    cout << "b: " << b << endl;
    c = a + b;
    cout << "c: " << c << endl;
    if(a >= b)
        c = a - b;
    else
        c = b - a;
    cout << "c: " << c << endl;
}

```

```

c = 10*a;

cout << "c: " << c << endl;

a /= 10;

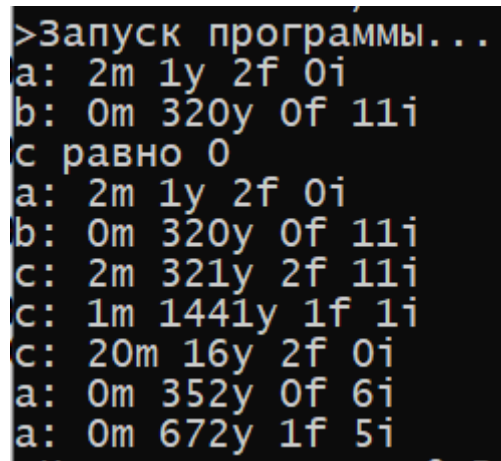
cout << "a: " << a << endl;

a += b;

cout << "a: " << a << endl;

}

```



```

>Запуск программы...
a: 2m 1y 2f 0i
b: 0m 320y 0f 11i
c равно 0
a: 2m 1y 2f 0i
b: 0m 320y 0f 11i
c: 2m 321y 2f 11i
c: 1m 1441y 1f 1i
c: 20m 16y 2f 0i
a: 0m 352y 0f 6i
a: 0m 672y 1f 5i

```

Рис. 1 Результат тестов

#### 4 Листинг реализации класса

```

Length::Length(){inch = 0;}

Length::Length(int i){inch = i;}

Length::Length(int f, int i){inch = f*12+i;}

Length::Length(int y, int f, int i){inch = y*36+f*12+i;}

Length::Length(int m, int y, int f, int i){inch =
m*1760*36+y*36+f*12+i;}

int Length::getMile(){return inch/(1760*36);}

int Length::getYard(){return (inch/(3*12))%1760;}

int Length::getFoot(){return (inch/12)%3;}

int Length::getInch(){return inch%12;}

Length operator+(const Length &L1, const Length &L2){return
Length(L1.inch + L2.inch);}

Length operator-(const Length &L1, const Length &L2){return
(L1.inch - L2.inch>=0)?Length(L1.inch - L2.inch): Length(0);}

```

```

Length operator*(int z, const Length &L1){return Length(L1.inch
* z);}

Length operator/(const Length &L1, int z){return Length(L1.inch
/ z);}

bool operator==(Length L1, Length L2){
    if(L1.inch == L2.inch) return true;
    return false;
}

bool operator!=(Length L1, Length L2){
    if(L1.inch != L2.inch) return true;
    return false;
}

bool operator<(Length L1, Length L2){
    if(L1.inch < L2.inch) return true;
    return false;
}

bool operator>(Length L1, Length L2){
    if(L1.inch > L2.inch) return true;
    return false;
}

bool operator<=(Length L1, Length L2){
    if(L1.inch <= L2.inch) return true;
    return false;
}

bool operator>=(Length L1, Length L2){
    if(L1.inch >= L2.inch) return true;
    return false;
}

bool operator!(Length L){
    if(L.inch == 0) return true;
    return false;
}

```

```

Length &Length::operator+=(const Length &L2){
    this->inch += L2.inch;
    return *this;
}

Length &Length::operator-=(const Length &L2){
    this->inch -= L2.inch;
    return *this;
}

Length &Length::operator*=(int z){
    this->inch *= z;
    return *this;
}

Length &Length::operator/=(int z){
    this->inch /= z;
    return *this;
}

void Length::set_inch(int i){
    inch = i;
}

istream& operator>>(istream &i, Length &L){
    int m, y, f, inch;
    i >> m;
    i.ignore(1);
    i >> y;
    i.ignore(1);
    i >> f;
    i.ignore(1);
    i >> inch;
    i.ignore(1);
    L.set_inch(m*1760*36+y*36+f*12+inch);
    return i;
}

```

```
}  
  
ostream& operator<<(ostream &o, Length L) {  
    return o<<L.getMile()<<"m "<<L.getYard()<<"y  
    "<<L.getFoot()<<"f "<<L.getInch()<<"i";  
}
```