Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

**Тема: Классы и объекты. Инкапсуляция.**

**Вариант 5**

Дисциплина: «Основы алгоритмизации и программирования»

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Батин Владислав Владимирович

Проверил

Доцент кафедры ИТАС

Полякова Ольга Андреевна

2023 год

**Постановка задачи**

Поля first и second – положительные целые числа(минуты и секунды). Реализовать метод minutes() – приведение времени в минуты.

**Код**

**KL.1.main.cpp**

#include <iostream>

#include "fraction.h"

#include <Windows.h>

using namespace std;

fraction make\_fraction(int F, int S)

{

fraction t;

t.Init(F, S);

return t;

}

void main()

{

// Определение переменных А и В

cout << endl << "\_\_\_\_\_Main\_\_\_\_\_" << endl;

fraction A;

fraction B;

A.Init(3, 40);

B.Read();

A.Show();

B.Show();

cout << endl;

// Вывод приведенного времени в минуты

cout << "Time A : " << A.Minutes() << endl;

cout << "Time B : " << B.Minutes() <<endl;

// Указатели

cout << endl << "\_\_\_Pointers\_\_\_" << endl;

fraction\* C = new fraction;

C->Init(0, 0);

C->Show();

C->Read();

C->Minutes();

cout << "Time C : " << C->Minutes() << endl;;

// Массивы

cout << endl << "\_\_\_\_Arrays\_\_\_\_" << endl;

fraction M[2];

for (int i = 0; i < 2; i++)

{

M[i].Read();

}

for (int i = 0; i < 2; i++)

{

cout << "Time number " << i + 1 << " = " << M[i].Minutes() << endl;

}

// Динамические массивы

cout << endl << "\_Dynamic arrays\_" << endl;

int n;

cout << "N : "; cin >> n;

fraction\* DM = new fraction[n];

for (int i = 0; i < n; i++)

{

DM[i].Read();

}

for (int i = 0; i < n; i++)

{

cout << "Time number " << i + 1 << " = " << DM[i].Minutes() << endl;

}

delete DM;

// Вызов функции make\_fraction

cout << endl << "\_\_make\_fraction\_\_" << endl;

int x, y;

cout << "First >>"; cin >> x;

cout << "Second >>"; cin >> y;

// Формирование переменной(Т) с помощью функции make\_fraction

fraction T = make\_fraction(x, y);

cout <<"Time T = " << T.Minutes();

}

**Fraction.cpp**

#include <iostream>

#include "fraction.h"

using namespace std;

// метод инициализации полей структуры

void fraction::Init(int F, int S)

{

first = F;

second = S;

}

// метод для чтения значений полей

void fraction::Read()

{

cout << endl << "First >> "; cin >> first;

cout << "Second >> "; cin >> second;

}

// метод для вывода значений полей

void fraction::Show()

{

cout << endl << "First = " << first;

cout << endl << "Second = " << second;

cout << endl;

}

// метод для приведения времени в минуты

int fraction::Minutes()

{

return (60 \* first + second);

}

**fraction.h**

struct fraction

{

int first;

int second;

void Init(int, int); // метод инициализации полей

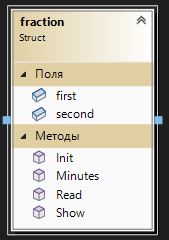
void Read(); // метод для чтения значений полей

void Show(); // метод для вывода значений полей

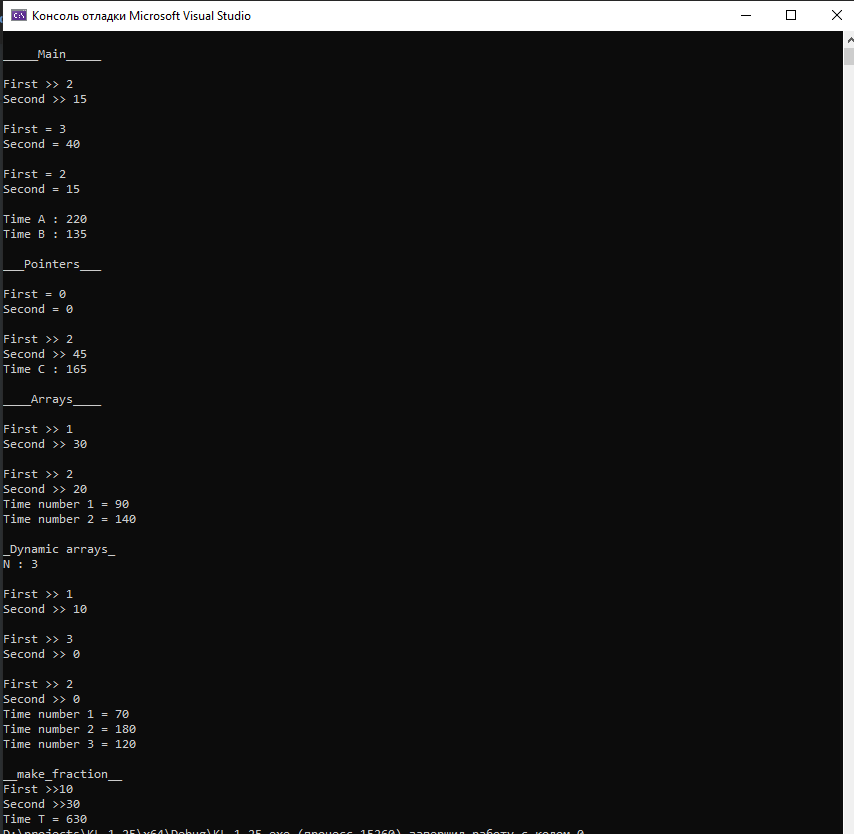
int Minutes(); // приведение времени в минуты

};

**Диаграмма класса**



**Результат работы программы**

****

Ответы на контрольные вопросы

1) Это абстрактный тип данных. Он сочетает в себе два функционала:

1. структура, в которой можно хранить различные типы данных: массивы, переменные, функции.

2. возможность пользоваться объектно-ориентированным программированием

2)В языке программирования объект – это переменная типа «класс». Класс описывает данные и методы , которые будут использоваться объектом этого класса. Каждый класс описывает логически завершенную единицу программы.

3)По́ле кла́сса или атрибу́т в объектно-ориентированном программировании — переменная, связанная с классом или объектом.

4)Методы.

5)Чтобы предотвратить изменения «жизненно важных» переменных и методов от других пользователей.

6)Для доступа к методам и полям через main().

7)Для работы с этими полями и методами внутри класса.

8)Private.

9)Public.

10)Public, для комфортной работы из любой функции.

11)С помощью конструкторов копирования, с параметрами, без параметров, с помощью сеттеров с помощью обычной инициализации в функции main.

12)С помощью методов класса show()(простой cout), либо с использованием геттеров(возвращает значение).

13)S.name.

14)S.name.

15)С помощью сеттера void SetName(VolName){name=VolNAme;} из функции main(){s.SetName};

16)С помощью сеттера void SetName(VolName){name=VolNAme;} из функции main(){s.SetName};

17)S.name;