Лабораторная работа № 3.

Объектно-ориентированный подход. Классы и объекты

Цель работы: дать первые представления о программировании в объектно-ориентированном стиле. научится создавать классы объектов

Теоретическое обоснование

Идея классов отражает строение объектов реального мира - каждый предмет или процесс обладает набором характеристик или отличительных черт, иными словами, свойствами и поведением.

Классы – это типы данных, определенные в конкретной программе. Определение класса включает в себя описание, из каких составных частей или атрибутов он состоит и какие операции определены для класса.

Имена классов, **их методов** и **атрибутов** составляются из английских слов, описывающих их смысл, при этом, если слов несколько, они пишутся слитно. Имена классов начинаются с *заглавной буквы*, если название состоит из нескольких слов, каждое слово начинается с заглавной буквы, остальные маленькие.

```
// пример класса
class Complex
{
public:
int real;    // вещественная часть
int imaginary;     // мнимая часть
void add(Complex x); // прибавить комплексное число
      }
```

Приведенный выше пример – упрощенное определение класса *Complex*, представляющее комплексное число. Комплексное число состоит из вещественной части – целого числа *real* и мнимой части, которая представлена целым числом *imaginary*. *real* и *imaginary* это атрибуты класса. Для класса *Complex* определена одна операция или метод – *add*. Определив класс, мы можем создать переменную типа Complex.

Complex number;

Имея объект, мы можем установить значения атрибутов объекта:

number.real=1;

number.imaginary=2;

Операция «.» обозначает обращение к атрибуту объекта. Создав еще один объект класса Complex, можно прибавить его к первому:

Complex num2;

number.add(num2);

Состояние объекта характеризуется перечнем (обычно неизменным) всех свойств данного объекта и текущими (обычно изменяемыми) значениями каждого из этих свойств. Тот факт, что всякий объект имеет состояние, означает, что всякий объект занимает определенное пространство (физически или в памяти компьютера).

К числу свойств относятся присущие объекту или приобретаемые им характеристики, черты, качества или способности, делающие данный объект самим собой. Эти свойства принято называть атрибутами класса. Атрибуты содержатся внутри класса, поэтому они скрыты от других классов. В связи с этим иногда требуется указать, какие классы имеют право читать и изменять атрибуты. Это свойство называется видимостью атрибута.

У атрибутов и операций, в зависимости от их назначения и требований доступности, определяют следующие значения этого параметра:

public (открытый). В этом разделе размещают атрибуты, доступные всем остальным классам. Любой класс может просмотреть или изменить их значением.

private (закрытый). Такой атрибут не виден никаким другим классам, кроме дружественных.

protected (защищенный). Атрибуты этого раздела доступны только самому классу, его потомкам и друзьям (friend).

Можно задавать несколько секций private и public, порядок их следования значения не имеет.

Видимостью элементов класса можно также управлять с помощью ключевых слов struct и class. Если при описании класса используется слово struct, то все поля и методы по умолчанию будут общедоступными (public). Если при описании класса используется слово class, то по умолчанию все методы и поля класса будут скрытыми (private).

Свойства атрибутов класса: могут иметь любой тип, кроме типа этого же класса (но могут быть указателями на этот класс); могут быть описаны с модификатором const, при этом они инициализируются только один раз (с помощью конструктора) и не могут изменяться.

Инициализация атрибутов при описании не допускается.

Если тело метода определено внутри класса, он является встроенным (inline). Как правило, встроенными делают короткие методы. Если внутри класса записано только объявление (заголовок) метода, сам метод должен быть определен в другом месте программы с помощью операции доступа к области видимости (::).

Пример 1. Написать программу для учета успеваемости студентов.

1. Создадим заголовочный файл **students.h**, в котором будет находиться класс Students.

Для создания заголовочного файла в Обозревателе решений выберите папку Файлы заголовков (рисунок 1)

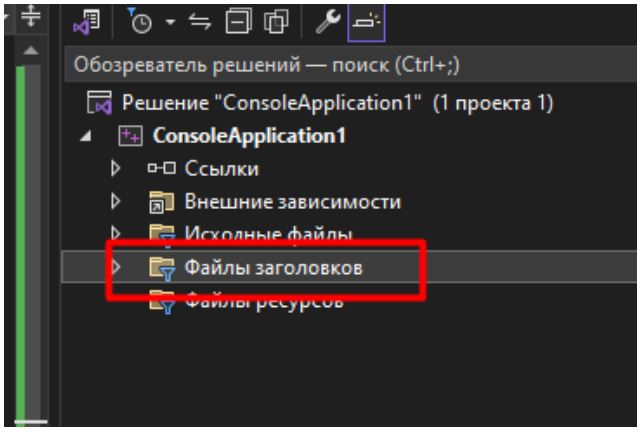

Рисунок 1

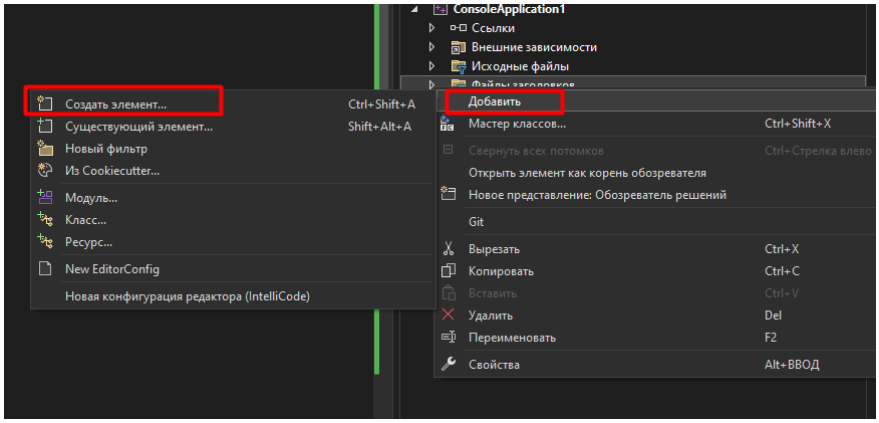

Рисунок 2

Далее выберите в контекстном меню, вызванное правой кнопкой мыши команду

Добавить/Создать элемент (рисунок 2). Далее выберите Файл заголовка (рисунок 3)

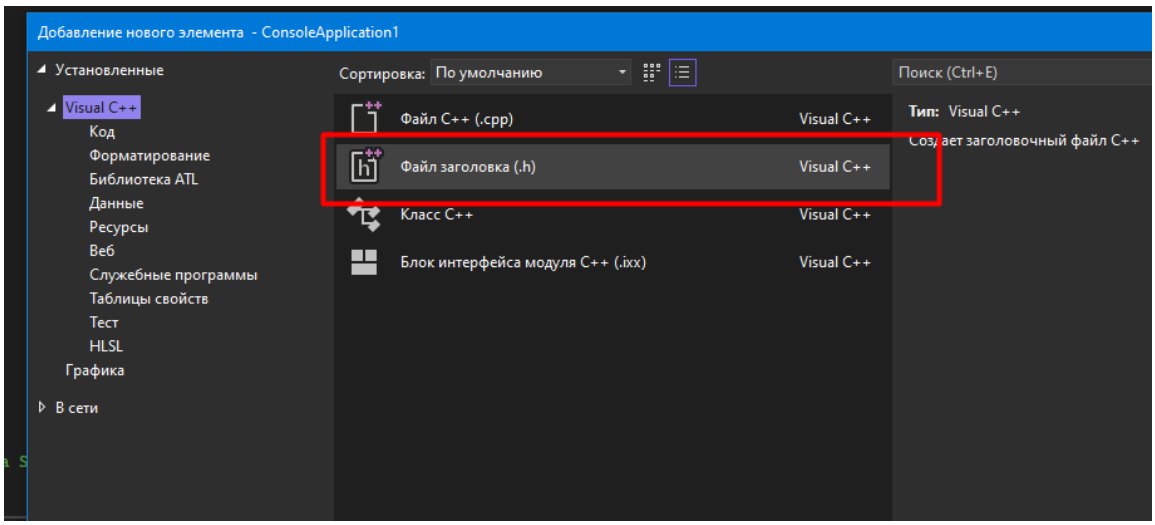

Рисунок 3

В новом файле и создадим класс Students и его методы.

```
#include <string>

class Students {
public:
    // Установка имени студента
    void set_name(std::string student_name)
    {
        name = student_name;
    }
    // Получение имени студента
    std::string get_name()
    {
        return name;
    }
    // Установка фамилии студента
    void set_last_name(std::string student_last_name)
    {
        last_name = student_last_name;
    }
    // Получение фамилии студента
    std::string get_last_name()
    {
        return last_name;
    }
    // Установка промежуточных оценок
    void set_scores(int student_scores[])
    {
        for (int i = 0; i < 5; ++i) {
            scores[i] = student_scores[i];
        }
    }
    // Установка среднего балла
    void set_average_ball(float ball)
    {
        average_ball = ball;
    }
    // Получение среднего балла
    float get_average_ball()
    {
        return average_ball;
    }
```

```
private:
    // Промежуточные оценки
    int scores[5];
    // Средний балл
    float average_ball;
    // Имя
    std::string name;
    // Фамилия
    std::string last_name;
      };
```

Мы не можем напрямую обращаться к закрытым данными класса. Работать с этими данными можно только посредством методов этого класса. Поэтому в этом примере мы используем функцию get_average_ball() для получения средней оценки студента, и set_average_ball() для выставления этой оценки.

Функция set_average_ball() принимает средний балл в качестве параметра и присваивает его значение закрытой переменной average_ball. Функция get_average_ball() просто возвращает значение этой переменной.

Функция set_name() сохраняет имя студента в переменной name, а get_name() возвращает значение этой переменной.

Принцип работы функций set_last_name() и get_last_name() аналогичен.

Функция set_scores() принимает массив с промежуточными оценками и сохраняет их в приватную переменную int scores.

Теперь создайте файл **main.cpp** со следующим содержимым.

```
#include <iostream>
#include <fstream>
#include "students.h"

using namespace std;


int main()
{
    // Создание объекта класса Student
    Students student;

    std::string name;
    std::string last_name;

    // Ввод имени с клавиатуры
    std::cout << "Name: ";
    getline(std::cin, name);

    // Ввод фамилии
    std::cout << "Last name: ";
    getline(std::cin, last_name);

    // Сохранение имени и фамилии в объект класса Students
    student.set_name(name);
```

```
    student.set_last_name(last_name);

    // Оценки
    int scores[5];
    // Сумма всех оценок
    int sum = 0;

    // Ввод промежуточных оценок
    for (int i = 0; i < 5; ++i) {
        std::cout << "Score " << i + 1 << ": ";
        std::cin >> scores[i];
        // суммирование
        sum += scores[i];
    }

    // Сохраняем промежуточные оценки в объект класса Student
    student.set_scores(scores);
    // Считаем средний балл
    float average_ball = sum / 5.0;
    // Сохраняем средний балл в объект класса Students
    student.set_average_ball(average_ball);
    // Выводим данные по студенту
    std::cout << "Average ball for " << student.get_name() << " "
        << student.get_last_name() << " is "
        << student.get_average_ball() << std::endl;

    return 0;
}
```

В самом начале программы создается объект класса Students. Дело в том, что сам класс является только описанием его объекта. Класс Students является описанием любого из студентов, у которого есть имя, фамилия и возможность получения оценок.

Объект класса Students характеризует конкретного студента. Если мы захотим выставить оценки всем ученикам в группе, то будем создавать новый объект для каждого из них. Использование классов очень хорошо подходит для описания объектов реального мира.

После создания объекта student, мы вводим с клавиатуры фамилию, имя и промежуточные оценки для конкретного ученика. Пускай это будет Вася Пупкин, у которого есть пять оценок за семестр - две тройки, две четверки и одна пятерка.

Введенные данные мы передаем **set**-функциям, которые присваивают их закрытым переменным класса. После того, как были введены промежуточные оценки, мы высчитываем средний балл на основе этих оценок, а затем сохраняем это значение в закрытом свойстве average_ball, с помощью функции set_average_ball().

Следующим шагом будет отделение данных от логики.

Вынесем реализацию всех методов класса в отдельный файл **students.cpp** (рисунок 4)

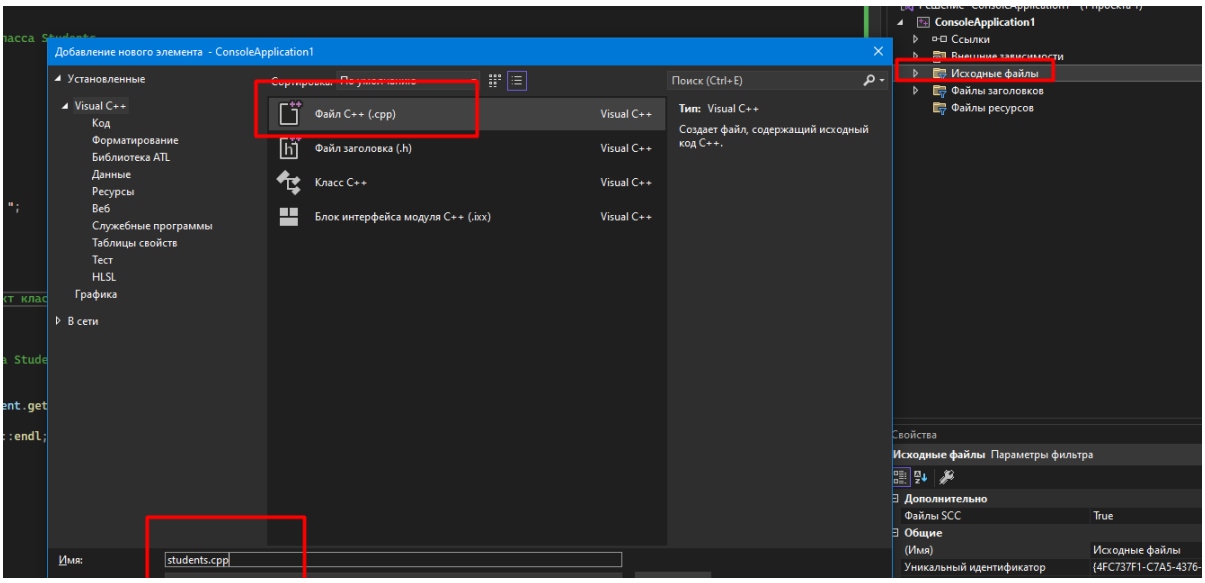

Рисунок 4

```cpp
#include <string>
#include "students.h"

// Установка имени студента
void Students::set_name(std::string student_name)
{
    Students::name = student_name;
}

// Получение имени студента
std::string Students::get_name()
{
    return Students::name;
}

// Установка фамилии студента
void Students::set_last_name(std::string student_last_name)
{
    Students::last_name = student_last_name;
}

// Получение фамилии студента
std::string Students::get_last_name()
{
    return Students::last_name;
}

// Установка промежуточных оценок
void Students::set_scores(int scores[])
{
    for (int i = 0; i < 5; ++i) {
        Students::scores[i] = scores[i];
    }
}

// Установка среднего балла
void Students::set_average_ball(float ball)
{
    Students::average_ball = ball;
}

// Получение среднего балла
```

```
float Students::get_average_ball()
{
    return Students::average_ball;
                                                }
```

В заголовочном файле **students.h** оставим только прототипы этих методов.

```
#pragma once /* Защита от двойного подключения заголовочного файла */
#include <string>

class Students {
public:
    // Установка имени студента
    void set_name(std::string);
    // Получение имени студента
    std::string get_name();
    // Установка фамилии студента
    void set_last_name(std::string);
    // Получение фамилии студента
    std::string get_last_name();
    // Установка промежуточных оценок
    void set_scores(int[]);
    // Установка среднего балла
    void set_average_ball(float);
    // Получение среднего балла
    float get_average_ball();

private:
    // Промежуточные оценки
    int scores[5];
    // Средний балл
    float average_ball;
    // Имя
    std::string name;
    // Фамилия
    std::string last_name;
      };
```

Такой подход называется **абстракцией данных** - одного из фундаментальных принципов объектно-ориентированного программирования. К примеру, если кто-то другой захочет использовать этот класс в своем коде, ему не обязательно знать, как именно высчитывается средний балл. Он просто будет использовать функцию calculate_average_ball(), не вникая в алгоритм ее работы, подобно тому как мы используем такие методы как cin, cout и не задумываемся как они работают.

Над крупными проектами обычно работает несколько программистов. Каждый из них занимается написанием определенной части продукта. В таких масштабах кода, одному человеку практически нереально запомнить, как работает каждая из внутренних функций проекта. В нашей программе, мы используем оператор потокового вывода cout, не задумываясь о том, как он реализован на низком уровне. Кроме того, отделение данных от логики является хорошим тоном программирования.

Известно, что каждый класс в C++ использует свое пространство имен. Это

сделано для того, чтобы избежать конфликтов при именовании переменных и функций. В файле students.cpp используем оператор принадлежности :: перед именем каждой функции. Это делается для того, чтобы указать компилятору, что эти функции принадлежат классу Students.

Создание объекта через указатель

При создании объекта, лучше не копировать память для него, а выделять ее в куче с помощью указателя. И освобождать ее после того, как закончили работу с объектом. Реализуем это в нашей программе, немного изменив содержимое файла **main.cpp**.

```
#include <iostream>
#include "students.h"

int main()
{
    // Выделение памяти для объекта Students
    Students* student = new Students;

    std::string name;
    std::string last_name;

    // Ввод имени с клавиатуры
    std::cout << "Name: ";
    getline(std::cin, name);

    // Ввод фамилии
    std::cout << "Last name: ";
    getline(std::cin, last_name);

    // Сохранение имени и фамилии в объект класса Students
    student->set_name(name);
    student->set_last_name(last_name);

    // Оценки
    int scores[5];
    // Сумма всех оценок
    int sum = 0;

    // Ввод промежуточных оценок
    for (int i = 0; i < 5; ++i) {
        std::cout << "Score " << i + 1 << ": ";
        std::cin >> scores[i];
        // суммирование
        sum += scores[i];
    }
    // Сохраняем промежуточные оценки в объект класса Student
    student->set_scores(scores);

    // Считаем средний балл
    float average_ball = sum / 5.0;
    // Сохраняем средний балл в объект класса Students
    student->set_average_ball(average_ball);
    // Выводим данные по студенту
    std::cout << "Average ball for " << student->get_name() << " "
        << student->get_last_name() << " is "
        << student->get_average_ball() << std::endl;
    // Удаление объекта student из памяти
    delete student;
    return 0;
```

```
}
```

При создании статического объекта, для доступа к его методам и свойствам, используют операция прямого обращения — «.» (символ точки). Если же память для объекта выделяется посредством указателя, то для доступа к его методам и свойствам используется оператор косвенного обращения — «->».

Конструктор и деструктор класса

Конструктор класса - это специальная функция, которая автоматически вызывается сразу после создания объекта этого класса. Он не имеет типа возвращаемого значения и должен называться также, как класс, в котором он находится. По умолчанию, заполним двойками массив с промежуточными оценками студента.

```
class Students {
public:
    // Конструктор класса Students
    Students(int default_score)
    {
        for (int i = 0; i < 5; ++i) {
            scores[i] = default_score;
        }
    }

private:
    int scores[5];
};

int main()
{
    // Передаем двойку в конструктор
    Students* student = new Students(2);
    return 0;
      }
```

Мы можем исправить двойки, если ученик будет хорошо себя вести, и вовремя сдавать домашние задания.

Деструктор класса вызывается при уничтожении объекта. Имя деструктора аналогично имени конструктора, только в начале ставится знак тильды ~. Деструктор не имеет входных параметров.

```
#include <iostream>

class Students {
public:
    // Деструктор
    ~Students()
    {
        std::cout << "Memory has been cleaned. Good bye." << std::endl;
    }
};

int main()
```

```
{
    Students* student = new Students;
    // Уничтожение объекта
    delete student;
    return 0;
      }
```

Методика выполнения практической части

Задание 1. Создание данных типа «класс»

Базовый уровень

Задание: для всех вариантов задач создать класс с указанными двумя полями (Поле 1, Поле 2) и тремя методами:

- конструктор для инициализации объекта;
- функция формирования строки с информацией об объекте;
- функция обработки значений полей по индивидуальному варианту.

В основной программе вводить значения полей каждого объекта из компонентов Edit и выводить результаты в компонент Memo. Индивидуальные варианты заданий приведены в таблице 1.

Таблица 1 – Варианты индивидуальных заданий

№ вар.	Поле 1	Поле 2	Функция обработки полей
1	Номинал купюры (1, 2, 5, 10 и т.д.)	Количество купюр	Вычислить сумму купюр
2	Номинал монеты (1, 2, 5, 10 и т.д.)	Количество монет	Вычислить сумму монет
3	Цена товара	Количество единиц товара	Вычислить общую стоимость товара
4	Калорийность 100г продукта	Вес продукта в граммах	Вычислить общую калорийность продукта
5	Вещественное число – левая граница диапазона	Вещественное число – правая граница диапазона	Квадрат длины диапазона
6	Количество минут	Количество секунд	Вычислить общее количество секунд
7	Количество часов	Количество минут	Вычислить общее количество минут

8	Вещественное число – первый катет прямоугольного треугольника	Вещественное число – второй катет прямоугольного треугольника	Вычислить площадь прямоугольного треугольника
9	Вещественное число – скорость движения (м/сек)	Целое число – время движения в минутах	Вычислить пройденное расстояние (в метрах)
10	Вещественное число – первый катет прямоугольного треугольника	Вещественное число – второй катет прямоугольного треугольника	Вычислить длину гипотенузы прямоугольного треугольника
11	Целое число – нижнее основание трапеции	Целое число – верхнее основание трапеции	Вычислить полусумму оснований трапеции
12	Вещественное число – первый катет прямоугольного треугольника	Вещественное число – второй катет прямоугольного треугольника	Вычислить тангенс угла α, противолежащего второмукатету прямоугольного треугольника
13	Вещественное число	Вещественное число	Вычислить полуразность чисел
14	Вещественное число	Вещественное число	Вычислить корень квадратный из произведения чисел
15	Целое число – x	Целое число – y	Вычислить целую часть от деления x на y
16	Целое число – x	Целое число – y	Вычислить квадрат меньшего из чисел
17	Целое число – x	Целое число – y	Вычислить куб большего из чисел
18	Продолжительность телефонного разго-вора в минутах	Стоимость одной минуты разговора	Вычислить общую стоимость разговора

19	Координата точки на плоскости (по горизонтали)	Координата точкина плоскости (по вертикали)	Определить периметр прямоугольника, ограниченного координатами точки и осями О*x* и О*y*
20	Вещественное число – *a*	Вещественное число – *b*	Вычислить разность квадратов чисел $a^2 - b^2$
21	Вещественное число – *a*	Вещественное число – *b*	Вычислить сумму квадратов чисел $a^2 + b^2$
22	Координата точки на плоскости (по горизонтали) – *x*1	Координата точкина плоскости (по вертикали) – *y*1	Определить площадь прямоугольника, ограни-ченного координатами точки и осями О*x* и О*y*
23	Координата точки на плоскости (по горизонтали) – *x*1	Координата точкина плоскости (по вертикали) – *y*1	Вычислить расстояние от точки до начала координат
24	Количество часов работы	Тариф оплаты за час работы	Общая стоимость работы
25	Радиус окружности	Угол в радианах	Вычислить длину дуги
26	Радиус окружности основания	Высота цилиндра	Вычислить площадь поверхности цилиндра
27	Радиус окружности основания конуса	Высота конуса	Вычислить объем конуса
28	Напряжение (в Вольтах)	Сопротивление (в Омах)	Вычислить значение тока (в Амперах)

29	Ток в амперах	Сопротивление резистора R1(вОмах)	Вычислить мощность на участке электрической цепи(в Ваттах)

Средний уровень

***Задание*:** создать класс с полями, указанными в индивидуальном задании (табл. 11.2, столб 2).

Реализовать в классе методы:

- конструктор по умолчанию;
- конструктор перезагрузки с параметрами;
- деструктор для освобождения памяти (с сообщением об уничтожении объекта);
- функции обработки данных (1 и 2), указанные в индивидуальном задании (табл. 11.2, столбцы 3 и 4);
- функцию формирования строки информации об объекте.

Создать проект для демонстрации работы: сформировать объекты со значениями-константами и с введенными значениями полей объекта из компонентов Edit. Выводить результаты в компонент Memo.

Таблица 11.2 – Варианты индивидуальных заданий.

№ вар.	Класс-родитель и его поля	Функция-метод 1 обработки данных	Функция-метод 2 обработки данных
1	Дата (три числа): день, месяц, год	Определить, является ли год высокосным (кратным 4)	Увеличить дату на 5 дней
2	Дата (три числа): день, месяц, год	Увеличить год на 1	Уменьшить дату на 2 дня
3	Дата (три числа): день, месяц, год	Определить, совпадают ли номер месяца и число дня	Увеличить дату на один месяц
4	Время (три числа): часы, минуты, секунды	Вычислить количество секунд в указанном времени	Увеличить время на 5 секунд
5	Время (три числа): часы, минуты, секунды	Вычислить количество полных минут в указанном времени	Уменьшить время на 10 минут

№ вар.	Класс-родитель и его поля	Функция-метод 1 обработки данных	Функция-метод 2 обработки данных
6	Время (три числа): часы, минуты, секунды	Определить количество минут до полуночи (24:00:00)	Увеличить время 100 минут
7	Координаты изображения прямоугольника: $x1$, $y1$, $x2$, $y2$	Вычислить площадь прямоугольника в пикселях	Изобразить прямоугольник на форме (Image) с толщиной линии 2 пикселя
8	Координаты изображения прямоугольника: $x1$, $y1$, $x2$, $y2$	Вычислить длину диагонали прямоугольника в пикселях	Изобразить прямоугольник и его диагональ на форме (Image)
9	Координаты изображения прямоугольника: $x1$, $y1$, $x2$, $y2$	Определить, является ли прямоугольник квадратом?	Изобразить прямоугольник на форме (Image), закрашенный зеленым цветом
10	Правильная дробь: числитель, знаменатель	Выразить значение дроби в процентах	Найти сумму цифр значения знаменателя
11	Комплексное число: действительная ($a1$) и мнимая ($b1$) части числа	Вычислить модуль комплексного числа	Найти комплексное число, обратное заданному
12	Комплексное число: действительная и мнимая часть числа	Вычислить произведение комплексного числа на число, вводимое пользователем	Вычислить аргумент комплексного числа в градусах
13	Книга: название, количество страниц, цена	Вычислить среднюю стоимость одной страницы	Увеличить цену книги в два раза, если название начинается со слова «Программирование»
14	Книга: название, автор, год издания	Вычислить, сколько лет книге	Количество дней, прошедших после года издания книги
15	Работник: фамилия, оклад, год поступления на работу	Вычислить стаж работы работника на данном предприятии	Сколько дней прошло после года поступления на работу
16	Работник: фамилия, оклад, год рождения	Вычислить возраст работника	Сколько календарных дней до исполнения работнику 50 лет

№ вар.	Класс-родитель и его поля	Функция-метод 1 обработки данных	Функция-метод 2 обработки данных
17	Вектор на плоскости: координаты вектора на плоскости ($x1$, $y1$, $x2$, $y2$)	Вычислить длину вектора	Изобразить линию вектора на форме (Image) с толщиной линии 2 пикселя
18	Вектор на плоскости: координаты вектора на плоскости ($x1$, $y1$, $x2$, $y2$)	Вычислить координаты середины вектора	Равен ли угол наклона вектора 45 градусов?
19	Вектор на плоскости: координаты вектора на плоскости ($x1$, $y1$, $x2$, $y2$)	Вычислить координаты вектора, удвоенной длины	Вычислить площадь прямоугольного треугольника, образованного вектором и прямыми, параллельн. осям Оx, Оy.
20	Цилиндр: диаметр основания, высота	Вычислить объем цилиндра	Изобразить круг заданного диаметра на форме (Image), закрашенный красным цветом.
21	Параллелепипед: длины сторон	Вычислить объем параллелепипеда	Вычислить длину наибольшей диагонали параллелепипеда
22	Параллелепипед: длины сторон	Вычислить площадь поверхности	Вычислить сумму длин всех ребер параллелепипеда.
23	Четыре целых числа: a, b ,c, d	Вычислить среднее арифметическое чисел	Определить максимальное из чисел
24	Три вещественных числа x, y, z	Вычислить среднее геометрическое чисел	Определите, сколько цифр содержит сумма заданных трех чисел.
25	Товар: наименование, цена, год выпуска	Определить, сколько лет назад был выпущен товар	Увеличить цену товара на 20%, если в наименовании товара есть слово «TV».
26	Товар: наименование, цена в гривне, изготовитель	Пересчитать цену товара в долларах	Увеличить цену товара в долларах, если название товара содержит слово «Toyota».
27	Координаты изображения эллипса: $x1$, $y1$, $x2$, $y2$	Определить, является ли эллипс окружностью?	Изобразить эллипс на форме (Image) зеленым цветом .
28	Книга: название, количество стра-	Увеличить количество страниц на 10	Уменьшить цену в два раза, если количество

№ вар.	Класс-родитель и его поля	Функция-метод 1 обработки данных	Функция-метод 2 обработки данных
	ниц, цена		страниц больше 100 (после увеличения)
29	Комната: длина, ширина, высота (в метрах)	Площадь стен (вместе с окнами и дверьми)	Площадь стен без окна (размер 2×15 м) и двери (размер 2 ×8 м).
30	Работник: фамилия, должность, оклад	Увеличить оклад на 15% (каждому работнику)	Работникам, у которых фамилия начинается с сочетания букв «Иван», присвоить должность «инженер».