**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра Информационных систем**

**ОТЧЕТ**

**по практической работе №2**

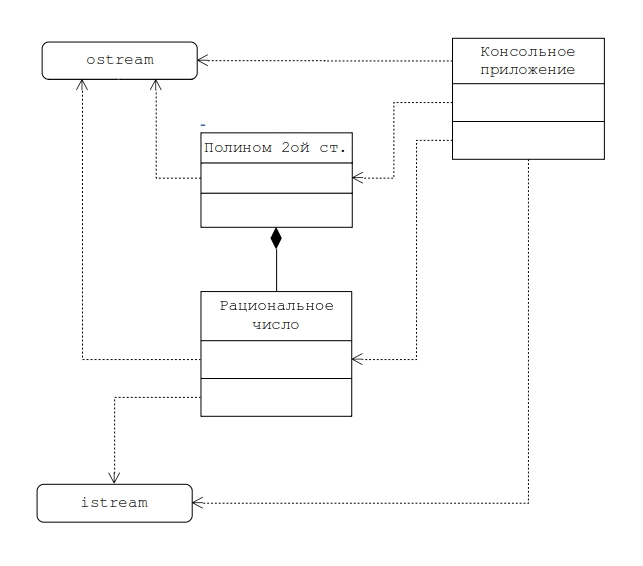
**по дисциплине «Объектно-ориентированное программирование»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8374 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Пихтовников К.С. |
| Студент гр. 8374 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Подсекин Г.С. |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Егоров С.С. |

Санкт-Петербург

2021

**Задание на практическую работу**

****

*Рис.1. Диаграмма классов работы №2*

Создать консольное приложение, реализующее функции перечисленные в описании работы №1 (вычисление корней, вычисление значения, представление полинома в классической и канонических формах) на множестве рациональных чисел.

Приложение должно включать основной модуль, модуль «application», модуль «polinom» и модуль «rational».

Для этого в проект лабораторной работы №1 следует добавить модуль с описанием и реализацией класса рациональных чисел TRational. Класс TRational должен быть встроен в проект согласно диаграмме классов на рис.1. При этом основной модуль, модуль «application» и модуль «polinom» не должны изменяться.

В классе **TRational** следует определить только те члены класса и спецификации, которые необходимы для совместимости модулей проекта и реализации отношений, приведенных в ДК объектной модели.

Реализовать и отладить программу, удовлетворяющую сформулированным требованиям и заявленной цели. Разработать контрольные примеры и оттестировать на них программу. Оформить отчет, сделать выводы по работе.

**Спецификация классов**

Класс Tapplication

Предназначен для выполнения функций ввода коэффициентов полинома, значения аргумента, инициализации процесса вычисления корней, инициализации процесса вычисления и вывода полинома в классической и канонической формах.

|  |  |
| --- | --- |
| Метод/атрибут | Описание |
| Метод exec() | Формальных параметров нет, тип void, область видимости-public.  В этом методе идет вызов функции menu(), задается конкретное действие, которое пожелал сделать пользователь и результат выводится на экран. |
| Метод menu() | Формальных параметров нет, тип возвращаемого значения-int, область видимости-private.  Вывод на экран необходимого меню, с помощью которого пользователь взаимодействует с программой. |
| Метод Tapplication() | Конструктор класса |

*Таблица 1. Класс Tapplication*

Класс Tpolinom

Методы и атрибуты данного класса необходимы для выполнения цели разрабатываемой программы (например, получение значений коэффициентов полинома, вычисление дискриминанта, вывод уравнения на экран).

|  |  |
| --- | --- |
| Метод/атрибут | Описание |
| Атрибут number a, b, c | область видимости – private, хранит значение коэффициентов. По умолчанию коэффициенты равны: a=1, b=2, c=1. |
| Атрибут EPrintMode printMode | область видимости – private, содержит вид уравнения для печати, который выбрал пользователь |
| Метод TPolinom(number,number,number) | Конструктор класса |
| Метод Void setPrintMethod(EPrintMode ePrintMethod) | Тип формального параметра - EPrintMode, область видимости public.  Метод устанавливает вид полинома, в котором его необходимо вывести (классический или канонический) |
| Методы getA(), getB(), getC() | Формальных параметров нет, тип number, область видимости public.  Предназначены для получения коэффициентов a, b, c. |
| Метод Int QuantityOfRoots() | Формальных параметров нет, тип возвращаемого значения - int, область видимости public.  Возвращает количество корней полинома |
| Метод number getValue(number x) | Тип формального параметра - number, тип возвращаемого значения - number, область видимости public.  Метод вычисляет и возвращает значение полинома. |
| Метод number Discriminant() | Формальных параметров нет, тип возвращаемого значения - number, область видимости private.  Возвращает значение дискриминанта |
| Метод Bool RootsInteger (number\*ArrayOfRoots, int quantityRoots, number a, number b, number c) | Тип возвращаемого значения - bool, область видимости private. Типы формальных параметров: указатель на массив с корнями (number\*), int количество корней, number коэффициенты a, b, c.  Метод позволяет определить, являются ли корни уравнения целыми числами. |
| Метод number \*Roots() | Формальных параметров нет, тип возвращаемого значения - number, область видимости public.  Возвращает указатель на массив с корнями. |
| Метод ostream& operator << (ostream& os, TPolinom& p) | Тип возвращаемого значения – ostream object, область видимости – public.  Данный метод выводит на экран полином в классической или канонической форме. |

*Таблица 2. Класс Tpolinom*

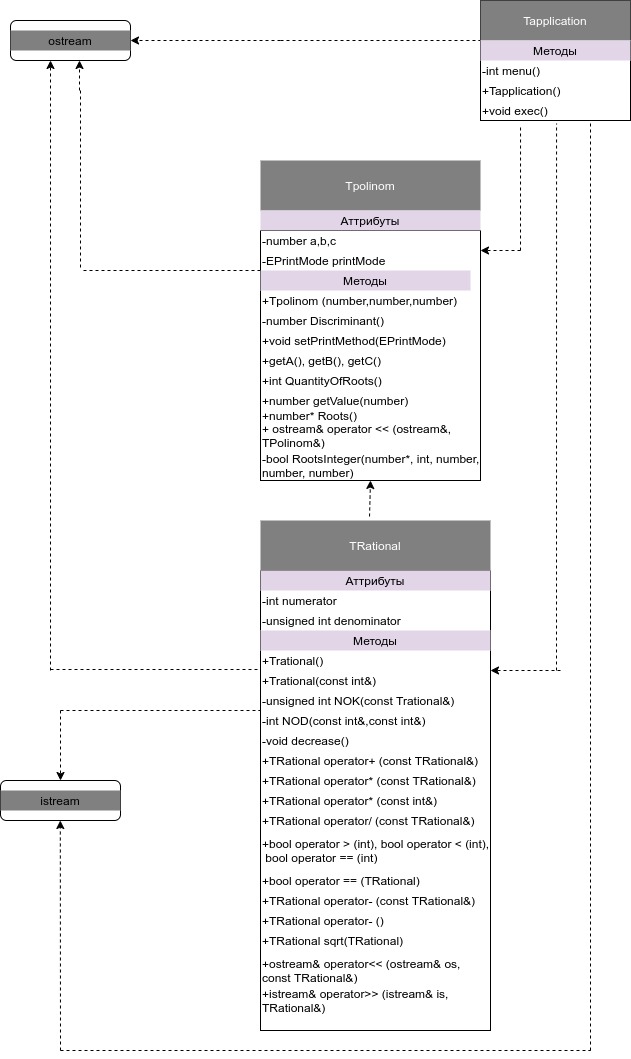
Класс TRational

Методы и атрибуты данного класса необходимы для вычисления корней, значений, представления полинома в классической и канонических формах на множестве рациональных чисел.

|  |  |
| --- | --- |
| Метод/атрибут | Описание |
| Атрибут int numerator | область видимости - private.  Является числителем дроби (целое число) |
| Атрибут unsigned int denominator | область видимости - private.  Является знаменателем дроби (целое число, большее нуля) |
| Метод TRational() | Конструктор класса |
| Метод TRational(const int&) | Конструктор класса |
| Метод unsigned int NOK(const TRational&) | Тип формального параметра - const TRationals&, тип возвращаемого значения - unsigned int, область видимости private.  Предназначен для вычисления наименьшего общего кратного. |
| Метод int NOD(const int&,const int&) | Тип формального параметра - const int&, тип возвращаемого значения - int, область видимости private.  Предназначен для вычисления наибольшего общего кратного. |
| Метод void decrease() | Формальных параметров нет, область видимости private.  Метод позволяет привести дробь к несократимому виду |
| Метод TRational operator+ (const TRational&) | Тип формального параметра - const TRationals&, тип возвращаемого значения - объект класса TRational,  область видимости public.  Перегрузка оператора сложения |
| Метод TRational operator\* (const TRational&) | Тип формального параметра - const TRational&,  тип возвращаемого значения - объект класса TRational,  область видимости public.  Перегрузка оператора умножения |
| Метод TRational operator\* (const int&) | Тип формального параметра - const int&, тип возвращаемого значения - объект класса TRational, область видимости public.  Перегрузка оператора умножения на число |
| Метод TRational operator/ (const TRational&) | Тип формального параметра - const TRational&, тип возвращаемого значения - объект класса TRational, область видимости public.  Перегрузка оператора деления |
| Методы:  bool operator > (int)  bool operator < (int)  bool operator == (int) | Тип формального параметра - int, тип возвращаемого значения - bool, область видимости public.  Перегрузка операторов сравнения (для целых чисел) |
| Метод bool operator == (TRational) | Тип формального параметра - TRational, тип возвращаемого значения - bool, область видимости public.  Перегрузка оператора сравнения (проверка сравнения дробей) |
| Метод TRational operator- (const TRational&) | Тип формального параметра - const TRational&,  тип возвращаемого значения - объект класса TRational,  область видимости public.  Перегрузка бинарного минуса |
| Метод TRational operator- () | Формальных параметров нет,  тип возвращаемого значения - объект класса TRational,  область видимости public.  Перегрузка унарного минуса |
| Метод friend TRational sqrt(TRational) | Тип формального параметра - TRational, тип возвращаемого значения - объект класса TRational, область видимости public.  Перегрузка оператора, который вычисляет квадратный корень |
| Метод friend ostream& operator<< (ostream& os, const TRational&) | Тип формальных параметров- ostream& os, const TRational&, тип возвращаемого значения - ostream object, область видимости – public.  Перегрузка оператора вывода |
| Метод friend istream& operator>> (istream& is, TRational&) | Тип формальных параметров - istream& is, TRational&, тип возвращаемого значения – istream object, область видимости – public.  Перегрузка оператора ввода |

*Таблица 3. Класс TRational*

**Диаграмма классов**



*Рис.2. Реализация диаграммы классов работы №2*

|  |  |
| --- | --- |
| Символ | Значение |
| + | public - открытый доступ |
| - | private - только из операций того же класса |
| # | protected - только из операций этого же класса и классов, создаваемых на его основе |

*Таблица 4. Обозначение аттрибутов и методов класса*

**Описание контрольного примера с исходными и ожидаемыми (расчетными) данными**

**Пример 1:**

Исходные данные:

Коэффициенты:

a = ; b = ; c = -;

Ожидаемые данные:

x1=-2, x2=0.5

p(1/6)==-

Классический вид:

Канонический вид:

**Пример 2:**

Исходные данные:

Коэффициенты:

a=, b=, c=

Ожидаемые данные:

x1=x2=-0.5

p(1/3)==

Классический вид:

Канонический вид:

**Пример 3:**

Исходные данные:

Коэффициенты:

a = ; b = ; c = ;

Ожидаемые данные:

Корней нет

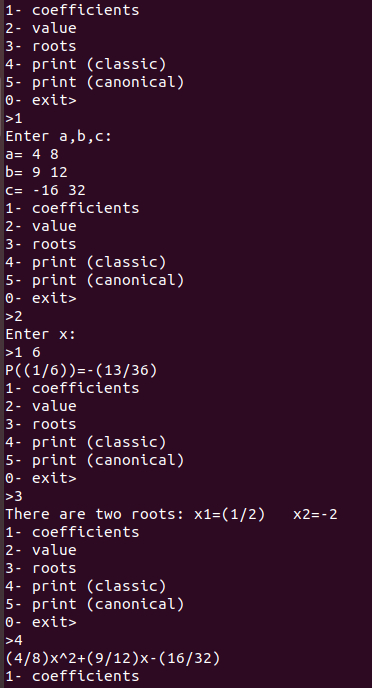
p()==

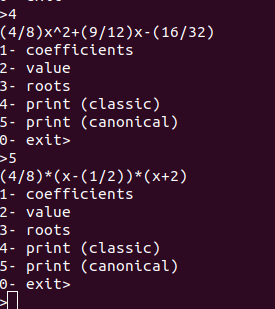
Классический вид:

Канонический вид: полином не имеет корней, поэтому его невозможно вывести в канонической форме.

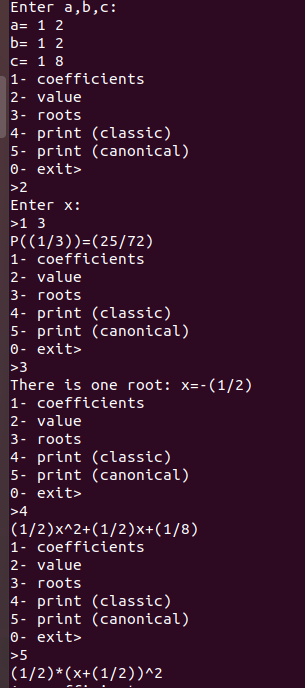
**Скриншоты программы на контрольных примерах**

**Пример 1:**

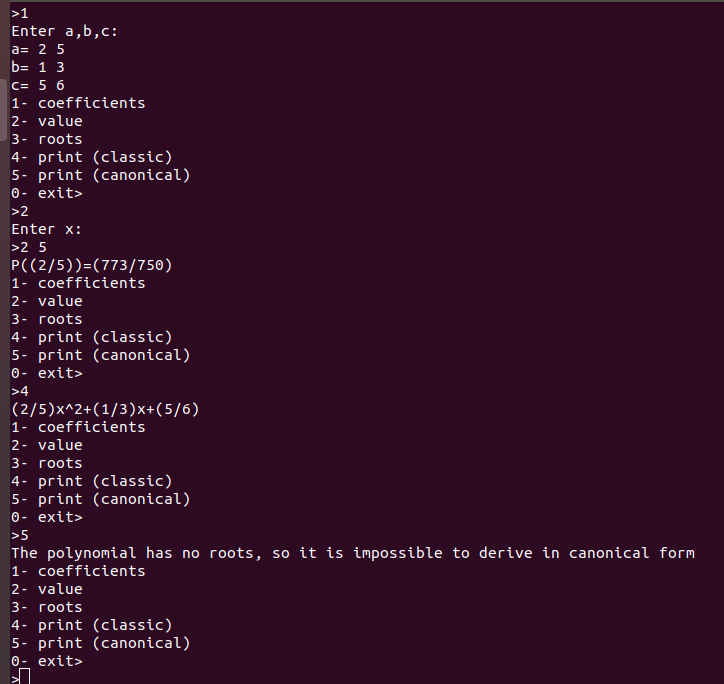




**Пример 2:**



**Пример 3:**

****

**Вывод**

В ходе данной лабораторной работы было создано консольное приложение согласно представленной на рис.1 диаграмме классов. Модули «application», «polinom» и основной модуль (функция main) были задействованы из 1 практической работы и не подвергались изменениям. Также был введен новый модуль «rational», который позволяет решать квадратные уравнения над полем рациональных чисел.

Помимо этого, была создана диаграмма классов (рис.2) и произведена отладка работы программы. Разработаны контрольные примеры с исходными и ожидаемыми данными, которые затем были протестированы в созданном консольном приложении. Все результаты совпали.