# 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» Тема: Консольное приложение для вычислений над квадратной матрицей, заданной комплексными числами.

|  |  |
| --- | --- |
| Студенты гр. 1361 | Горбунова Д.А.  Кравцов И.Ю. |
| Преподаватель | Егоров С.С. |

Санкт-Петербург 2024

# ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ

Тема работы: консольное приложение, реализующее функции перечисленные в описании работы №1, но на множестве комплексных чисел.

Исходные данные:

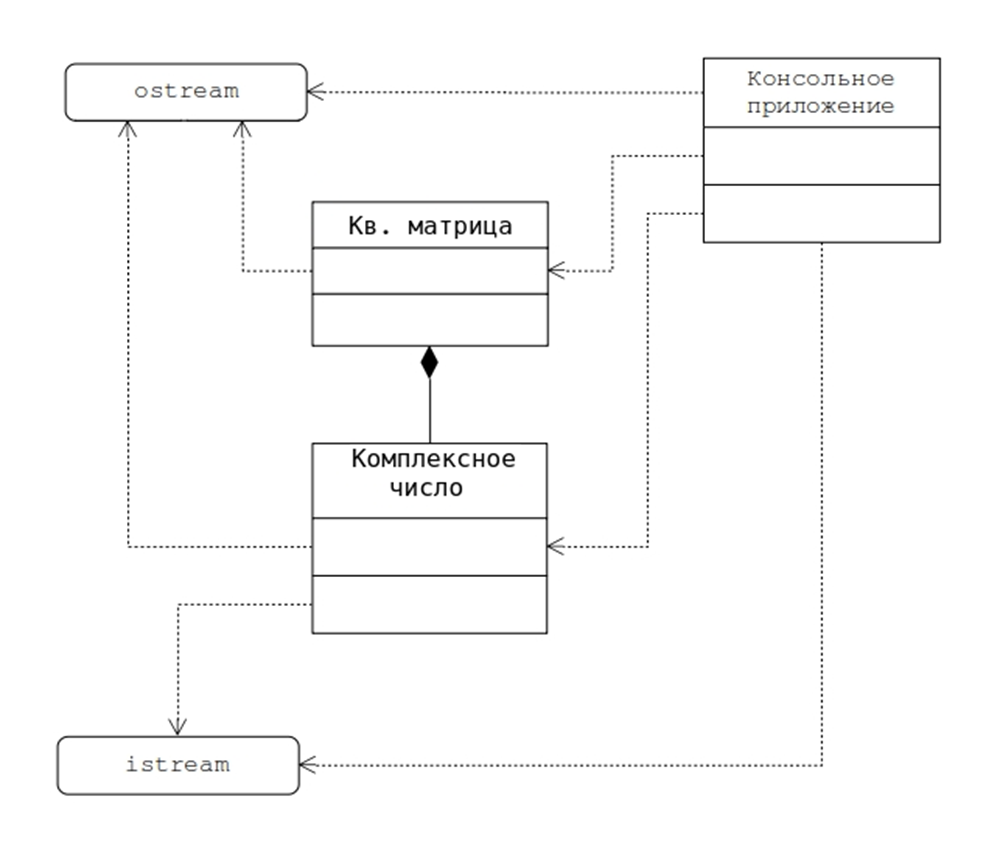


Рис.1 – Диаграмма классов

Описание работы:

Создать консольное приложение, реализующее функции перечисленные в описании работы №1, но на множестве комплексных чисел.

Приложение должно включать основной модуль, модуль «application», модуль «matrix» и модуль «complex. 3

Для этого в проект лабораторной работы №1 следует добавить модуль с собственным (не брать из ООБ!) описанием и реализацией класса комплексных чисел TComplex. Класс TComplex должен быть встроен в проект согласно диаграмме классов на рис.1. При этом основной модуль, модуль «application» и модуль «matrix» не должны изменяться.

В классе TComplex 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 протестировать на них программу. Оформить отчет, сделать выводы по работе.

# СОДЕРЖАНИЕ

|  |  |  |
| --- | --- | --- |
| 1. | Спецификация разработанных классов | 6 |
| 2. | Диаграмма классов | 9 |
| 3. | Описание контрольного примера | 10 |
| 4. | Работа программы на контрольных примерах | 12 |
|  | Выводы по выполненной работе | 20 |

1. **СПЕЦИФИКАЦИЯ РАЗРАБОТАННЫХ КЛАССОВ**
   1. Класс SquareMatrix(«Квадратная матрица»): Атрибуты класса:
      1. int m\_rows, m\_cols

Область видимости: private.

Описание: хранение размерности матрицы.

Стандартное значение: 3.

* + 1. vector<vector<number>> m\_values

Область видимости: private.

Описание: вектор, содержащий вектора-строки матрицы. Стандартное значение: Единичная матрица.

Чтобы предотвратить несанкционированный доступ, атрибуты класса ограничены доступом и могут быть использованы только внутри модуля, где определен класс квадратной матрицы.

Методы класса:

1. void generate\_1()

Область видимости: public.

Назначение: заполняет единичную матрицу.

1. void print\_screen ()

Область видимости: public.

Назначение: выводит текущую матрицу.

1. SquareMatrix transpose ()

Область видимости: public.

Назначение: транспонирует текущую матрицу.

1. double determinant ()const

Область видимости: public.

Назначение: вычисляет определитель текущей матрицы.

1. int rows () const

Область видимости: public.

Назначение: возвращает количество строк в текущей матрице.

1. int cols () const

Область видимости: public.

Назначение: возвращает количество столбцов в текущей матрице.

1. int rank ()

Область видимости: public.

Назначение: возвращает ранг текущей матрицы.

1. double determinantHelper (const std:: vector<std:: vector<double>>& matrix) const

Область видимости: private.

Назначение: вычисляет определитель матрицы меньшего размера.

Метод determinantHelper имеет тип доступа private т.к. для его работы необходим доступ к приватным полям класса. Остальные методы класса являются общедоступными т.к. задействуются в других модулях и используют публичные поля для работы.

* 1. Класс ConsoleApplication(«Консольное приложение»):

Атрибуты класса: Не имеется.

Методы класса:

* + 1. int exec ()

Область видимости: public.

Назначение: запускает работу приложения.

* + 1. int menu ()

Область видимости: private.

Назначение: предоставляет выбор методов для пользователя.

Метод menu имеет закрытый тип доступа т. к. используется исключительно в модуле Application.

* 1. Класс TComplex(«Комплексные числа»):

Атрибуты класса:

1. double real;

Область видимости: private.

Описание: действительная часть комплексного числа.

Стандартное значение: real = 0.

1. double imaginary;

Область видимости: private.

Описание: мнимая часть комплексного числа.

Стандартное значение: imaginary = 0.

Для защиты от неконтролируемого доступа, атрибуты класса имеют закрытый тип доступа, они используются только внутри модуля, где описан класс комплексного числа.

Методы класса:

1. TComplex ();

Область видимости: public.

Назначение: конструктор комплексного числа по умолчанию.

1. TComplex(const int&);

Область видимости: public.

Назначение: конструктор комплексного числа по заданному числу. Нужен для создания пользовательской матрицы квадратного типа.

1. ~TComplex ();

Область видимости: public.

Назначение: Деструктор комплексного числа.

1. ostream& operator<< (ostream& os, TComplex c);

Область видимости: public.

Назначение: вывод на экран комплексного числа. Возвращает указатель на поток вывода.

1. istream& operator>> (istream& is, TComplex& c);

Область видимости: public.

Назначение: ввод на экран комплексного числа. Возвращает указатель на поток ввода.

1. TComplex abs (const TComplex& t);

Область видимости: public.

Назначение: вычисление модуля комплексного числа. Возвращает вещественное значения модуля комплексного числа.

1. TComplex operator\*(int n, const TComplex& c);

Область видимости: public.

Назначение: умножение целого числа на матрицу комплексных чисел. Возвращает матрицу комплексных чисел

1. double abs () const;

Область видимости: public.

Назначение: нужен для вычисления модуля сиюминутного комплексного числа. Вспомогательная функция. Возвращает вещественное значения модуля комплексного числа.

1. TComplex operator\*(TComplex) const;

Область видимости: public.

Назначение: перемножение матриц над комплексными числами. Возвращает матрицу результата.

1. TComplex operator\*(int n) const;

Область видимости: public.

Назначение: умножение целого числа на комплексное число. Возвращает преобразованное комплексное число.

1. TComplex operator- (const TComplex& other) const;

Область видимости: public.

Назначение: возвращает разность двух комплексных чисел.

1. TComplex operator+ (const TComplex& other) const;

Область видимости: public.

Назначение: возвращает сумму двух комплексных чисел.

1. TComplex operator/ (TComplex& other);

Область видимости: public.

Назначение: возвращает частное двух комплексных чисел.

1. bool operator> (const TComplex& other);

Область видимости: public.

Назначение: возвращает true если комплексное число больше, иначе false.

1. bool operator== (TComplex);

Область видимости: public.

Назначение: возвращает true если комплексные числа равны, иначе false.

1. TComplex operator/= ( TComplex& other);

Область видимости: public.

Назначение: возвращает матрицу равную частному этой же матрицы и числа.

# ДИАГРАММА КЛАССОВ

Диаграмма классов представлена на рисунке 2. На ней обозначены атрибуты классов и их методы. Знаками «+» обозначены методы, атрибуты, имеющие общедоступный тип доступа; соответственно «-» – приватный. Также через двоеточие указаны типы возвращаемых значений, где они есть.

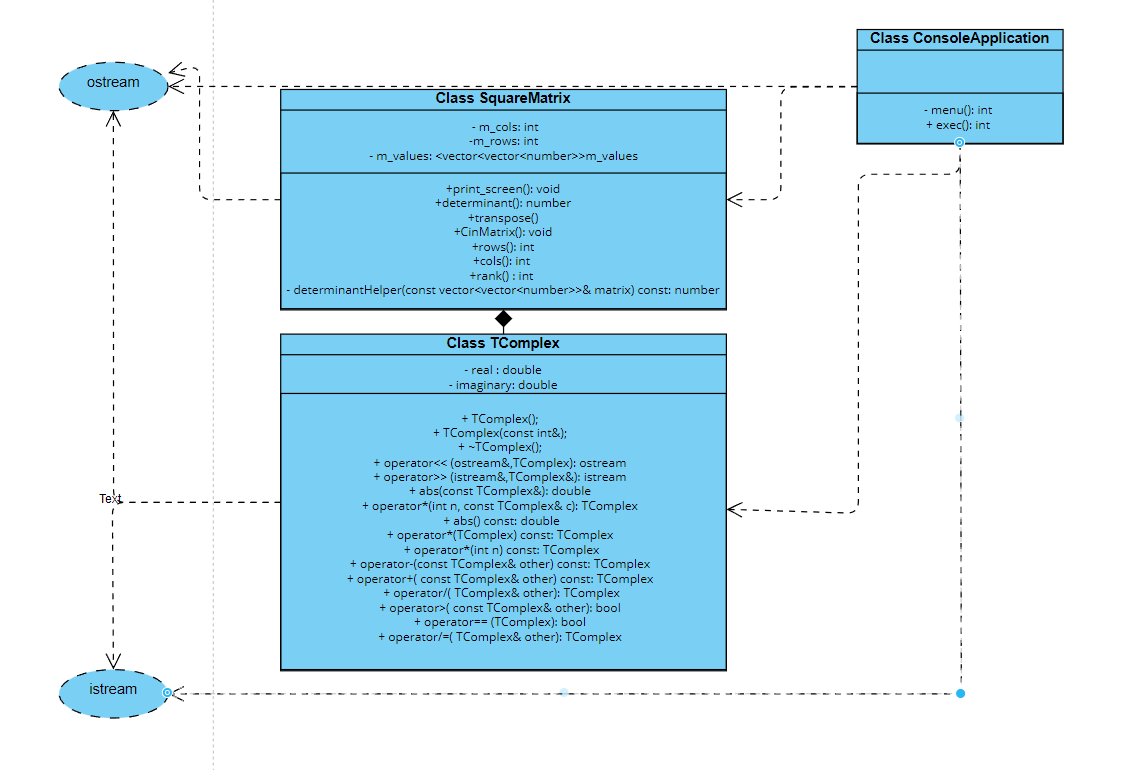


Рисунок 2 – Диаграмма классов

# ОПИСАНИЕ КОНТРОЛЬНОГО ПРИМЕРА

Посчитаем ранг для данной матрицы.

Приведем матрицу к верхнетреугольной матрицей:

Разделим первую строку на (3+8i).

Умножаем первую строку на (2+8i).

Вычитаем первую строку из второй и записываем результат во вторую строку.

Восстановим первую строку матрицы до изначального состояния.

Таким образом количество линейно независимых строк =2. Следовательно ранг матрицы равен 2.

Посчитаем определитель для данной матрицы.

Воспользуемся преобразованием которые проведены выше:

Определитель считается по следующей формуле:

# РАБОТА ПРОГРАММЫ НА КОНТРОЛЬНЫХ ПРИМЕРАХ

При запуске приложения, программа выводит на экран доступные варианты работы с матрицей (рисунок 3).

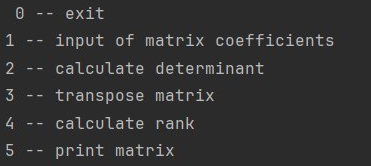


Рисунок 3 – Меню

Далее производим ввод пользовательской матрицы с клавиатуры.

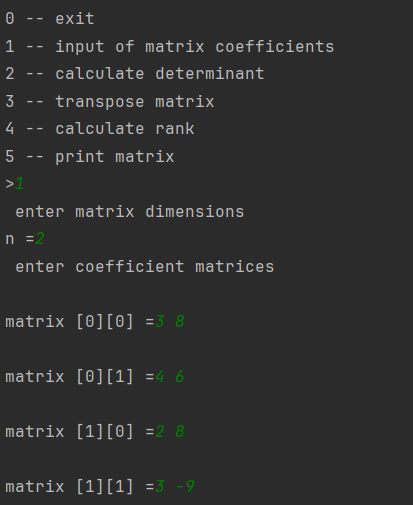


Рисунок 4 – ввод коэффициентов матрицы

После того мы определили матрицу, ее можно вывести (рисунок 5).

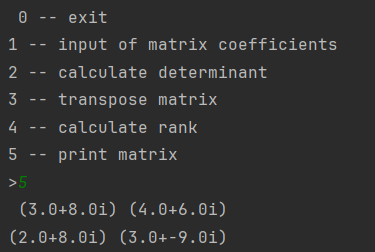


Рисунок 5 – Вывод матрицы.

После того, как у нас матрица задана, мы производим над ней вычисления. Сначала вычислим транспонированную матрицу (рисунок 6).

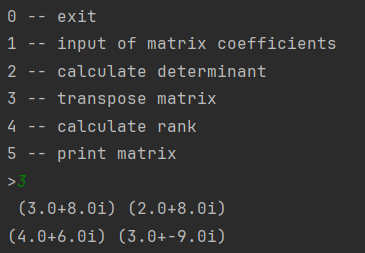


Рисунок 6 – Транспонирование матрицы.

Далее вычисляем ранг матрицы (рисунок 7).

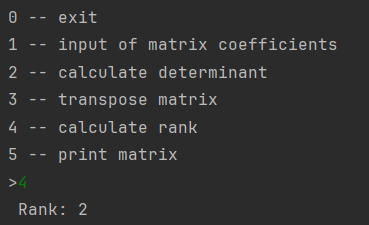


Рисунок 7 – Ранг матрицы.

Если же выбрать в меню пункт под номером 2, то программа вычислит определитель матрицы (рисунок 8).

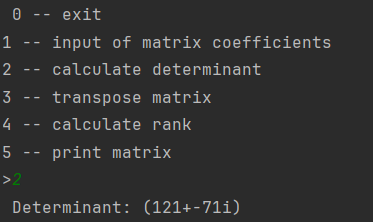


Рисунок 8 – Определитель матрицы.

После всех нужных вычислений, с помощью пункта под номером 0, мы можем завершить работу программы (рисунок 9).

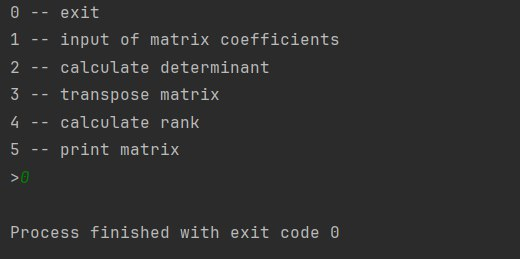


Рисунок 9 – Завершение работы программы.

# ВЫВОДЫ ПО ВЫПОЛНЕННОЙ РАБОТЕ

В ходе выполнения лабораторной работы было создано консольное приложение, реализующее функции для матриц на множестве комплексных чисел. Приложение было построено на основе модульной структуры, включающей основной модуль, модуль «application», модуль «matrix» и дополнительный модуль «complex», содержащий класс TComplex.

Класс TComplex был разработан с учетом требований к совместимости с остальными модулями проекта и реализации отношений, приведенных в диаграмме классов. В классе TComplex были определены только необходимые члены класса и спецификации, что позволило обеспечить гибкость и модульность приложения.

В процессе разработки и отладки программы были учтены все требования и цели, заявленные в описании работы. Были разработаны контрольные примеры, на которых была протестирована программа, что позволило убедиться в ее корректной работе.

В ходе работы были выявлены и устранены некоторые технические проблемы, связанные с обработкой ввода данных и выводом результатов. Также были исправлены некоторые ошибки, допущенные в предыдущей работе (в частности, две размерности одной матрицы). Это позволило улучшить качество программы и сделать ее более надежной и удобной для пользователя.

В результате работы была получена работающая программа, способная выполнять заданные функции на множестве комплексных чисел. Программа была тщательно протестирована, что подтвердило ее корректность и эффективность.

В дальнейшем планируется расширить функциональность программы, добавив новые функции и улучшив интерфейс пользователя. Также предполагается исследовать возможности оптимизации производительности программы и улучшения ее масштабируемости.