**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**ИНСТИТУТ НЕПРЕРЫВНОГО ОБРАЗОВАНИЯ**

**МНОГОПРОФИЛЬНЫЙ КОЛЛЕДЖ**

Программа подготовки специалистов среднего звена по специальности

09.02.07 Информационные системы и программирование

**КУРСОВОЙ ПРОЕКТ**

**на тему**

**Разработка программного модуля для вычисления наибольшего общего делителя**

Автор студент группы 22КИП1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Щукин Н. А. ы

*(подпись, дата) (фамилия, инициалы)*

Руководитель

Преподаватель СПО \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Пузренков А. Н. ы

*(уч. ст., должность) (подпись, дата) (фамилия, инициалы)*

Пенза 2025

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**ФГБОУ ВО «ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**ИНСТИТУТ НЕПРЕРЫВНОГО ОБРАЗОВАНИЯ**

**МНОГОПРОФИЛЬНЫЙ КОЛЛЕДЖ**

|  |  |
| --- | --- |
|  | **«УТВЕРЖДАЮ»**  Зав. отделением ИСиП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /О.В. Волкова/  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_\_ г. |

**ЗАДАНИЕ  
на** **курсовой проект**

|  |  |  |
| --- | --- | --- |
| Студент | Щукин Никита Алексеевич | |
| Группа | 22КИП1 | |
| Специальность | | 09.02.07 Информационные системы и программирование |
| Квалификация | | программист |

**Тема курсового проекта**

Разработка программного модуля для вычисления наибольшего общего делителя

**Техническое задание на курсовой проект**

Разработка программного модуля для вычисления наибольшего общего делителя (НОД) двух целых чисел представляет собой задачу создания надежного и эффективного решения для нахождения НОД с использованием алгоритма Евклида. Программа должна обеспечивать корректную обработку целочисленных входных данных, включая положительные и отрицательные числа, а также нулевые значения. Основные функции включают ввод и корректировку исходных чисел, наглядный вывод результатов с возможностью отображения промежуточных вычислений. Реализация на языке Java обеспечит кроссплатформенность решения, а модульная архитектура позволит в дальнейшем расширять функционал. Особое внимание уделяется качественному документированию кода Разрабатываемый модуль предназначен как для учебных целей при изучении алгоритмов теории чисел, так и для практического применения в вычислительных задачах, требующих нахождения наибольшего общего делителя.

**Содержание основной части курсового проекта**

1. Пояснительная записка:

Введение

1 Общий раздел

1.1 Анализ предметной области

1.2 Формирование требований к программному продукту

1.3 Обоснование выбора программно-аппаратных средств

2 Специальный раздел

2.1 Разработка структуры программного продукта

2.2 Описание разработанного приложения

2.3 Документирование программного продукта

Заключение

Список использованной литературы

Приложения

**Календарный график работ по выполнению курсового проекта**

|  |  |  |
| --- | --- | --- |
| Наименование этапов работы | Срок предоставления | Отметка об исполнении |
|
| Подбор литературы, ее изучение и проработка |  |  |
| Составление библиографии по основным источникам |  |  |
| Составление плана КП и согласование его с руководителем |  |  |
| Разработка и представление проекта на проверку |  |  |
| Разработка и представление на проверку пояснительной записки |  |  |
| Переработка (доработка) КП в соответствии с замечаниями |  |  |
| Предоставление КП в электронном и бумажном виде |  |  |
| Разработка тезисов доклада для защиты КП |  |  |
| Ознакомление с отзывом |  |  |
| Завершение подготовки к защите с учетом отзыва |  |  |

Дата выдачи задания на курсовой проект \_\_\_\_\_\_\_\_\_\_\_\_

Срок сдачи курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_

Руководитель курсового проекта

Пузренков Александр Николаевич

*(фамилия, имя, отчество)*

Преподаватель СПО

*(должность, ученая степень, ученое звание)*

*(подпись руководителя)*

Задание получил(а)

*(подпись студента) (ФИО студента)*

**Содержание основной части курсового проекта**

**Реферат**

Пояснительная записка 18 листов, 1 рисунок, 2 приложения, 5 источников.

Ключевые слова:

НОД — это наибольшее натуральное число, которое делит два или более числа без остатка.

Алгоритм RSA — это один из первых и наиболее широко используемых асимметричных криптографических алгоритмов, применяемый для шифрования данных и цифровой подписи. Назван по первым буквам фамилий его создателей: **R**ivest, **S**hamir и **A**dleman (1977 год).

Предметной областью разработки является разработка консольного приложения для вычисления наибольшего общего делителя. Цель работы – разработать консольное приложение на языке Java, которое эффективно находит НОД с использованием алгоритма Евклида.

В результате работы реализовано приложение на Java с контролем пользовательского ввода и точными вычислениями, а также выполнено тестирование и оформление документации.

При разработке использовалась среда программирования IntelliJ IDEA.

**Содержание**

[Введение 3](#_Toc197175875)

[1 Общий раздел 5](#_Toc197175876)

[1.1 Анализ предметной области 5](#_Toc197175877)

[1.2 Формирование требований к программному продукту 7](#_Toc197175880)

[1.3 Обоснование выбора программно-аппаратных средств 8](#_Toc197175887)

[2 Специальный раздел 10](#_Toc197175890)

[2.1 Разработка структуры программного продукта 10](#_Toc197175891)

[2.2 Описание разработанного приложения 10](#_Toc197175897)

[2.3 Документирование программного продукта 11](#_Toc197175900)

[Заключение 12](#_Toc197175902)

[Список используемых источников 14](#_Toc197175903)

[Приложение А 15](#_Toc197175904)

[Приложение Б 17](#_Toc197175905)

Введение

В условиях стремительного развития вычислительных технологий и роста требований к эффективности алгоритмов особую значимость приобретают задачи, связанные с математическими вычислениями. Одной из таких задач является нахождение наибольшего общего делителя (НОД) двух чисел. Эта операция широко применяется в криптографии, алгоритмах сжатия данных, теории чисел, а также в различных инженерных и научных расчетах.

Алгоритмы вычисления НОД лежат в основе многих современных вычислительных систем. Например, в криптографии НОД используется в алгоритме RSA для генерации ключей, а в компьютерной алгебре — для упрощения дробей. Кроме того, эффективные методы нахождения НОД важны в высокопроизводительных вычислениях, где скорость и точность играют ключевую роль. Программное приложение, реализующее данную функциональность, может быть полезным как для студентов, изучающих математику, так и для профессионалов, работающих в научных и инженерных областях.

Цель курсовой работы - разработать программный модуль на языке Java для вычисления наибольшего общего делителя (НОД) двух целых чисел с использованием алгоритма Евклида. Разработанное решение найдет применение как в образовательном процессе при изучении теории чисел, так и в практических областях - криптографии и компьютерной алгебре, где требуются эффективные методы работы с целыми числами. В результате выполнения курсовой работы планируется получить готовое программное решение, которое можно будет использовать как в образовательных целях, так и в практической деятельности.

Задачи курсовой работы:

* Изучить теоретические основы вычисления наибольшего общего делителя (НОД) и его применение.
* Провести сравнительный анализ алгоритмов вычисления НОД.
* Разработать архитектуру программного приложения на языке Java, обеспечивающую высокую производительность и точность вычислений.
* Реализовать программное приложение, которое будет вычислять НОД.

1 Общий раздел

* 1. Анализ предметной области

В рамках поставленной задачи требуется исследовать наибольший общий делитель (НОД) и фундаментальное значение НОД в теории чисел. НОД двух целых чисел — это наибольшее положительное целое число, которое делит каждое из данных чисел без остатка.

Метод перебора делителей:

Перебор делителей — алгоритм тестирования простоты числа путем полного перебора всех возможных потенциальных делителей.

Обычно перебор делителей заключается в переборе всех целых (как вариант: простых) чисел от 2 до квадратного корня из тестируемого числа n и в вычислении остатка от деления n на каждое из этих чисел. Если остаток от деления на некоторое число m равен нулю, то m является делителем n — в этом случае n объявляется составным, и алгоритм заканчивает работу. При достижении квадратного корня из n и невозможности сократить n ни на одно из меньших чисел, n объявляется простым.

Почему можно остановить перебор делителей на корне из n? Предположим, у n есть некоторый делитель p. Тогда n/p так же будет его делителем, и, очевидно, один из этих делителей не превосходит корня из n.

В практических задачах алгоритм применяется редко ввиду большой асимптотической сложности, однако его применение оправдано в случае, если проверяемые числа относительно невелики, так как данный алгоритм довольно легко реализуем.

Бинарный алгоритм (алгоритм Стейна):

Бинарный алгоритм вычисления НОД находит наибольший общий делитель, а именно НОД двух целых чисел. В эффективности данный алгоритм превосходит метод Евклида, что связано с использованием сдвигов, то есть операций деления на степень 2-ки, в нашем случае на 2.

Компьютеру проще поделить (умножить) на 2, 4, 8 и т.д., чем на какое-либо другое число. Но в тоже время бинарный алгоритм уступает алгоритму Евклида в простоте реализации.

Алгоритм:

1. НОД(0, n) = n; НОД(m, 0) = m; НОД(m, m) = m; НОД(n, n) = n;
2. НОД(1, n) = 1; НОД(m, 1) = 1;
3. Если m, n чётные, то НОД(m, n) = 2\*НОД(m/2, n/2);
4. Если m чётное, n нечётное, то НОД(m, n) = НОД(m/2, n);
5. Если n чётное, m нечётное, то НОД(m, n) = НОД(m, n/2);
6. Если m, n нечётные и n > m, то НОД(m, n) = НОД(m, (n-m)/2);
7. Если m, n нечётные и n < m, то НОД(m, n) = НОД((m-n)/2, n);

Алгоритм Евклида:

Алгоритм Евклида - один из первых алгоритмов в истории, использовался ещё в Древней Греции, и дошёл до наших дней. В изначальном виде метод назывался “взаимным вычитанием”, так как заключался в поочерёдном вычитании меньшего числа из большего, пока одно из них не станет равным 0. Сегодня чаще всего вместо вычитания используется взятие остатка от деления, но суть алгоритма сохранилась.

Алгоритм нахождения НОД делением

1. Большее число делим на меньшее.
2. Если делится без остатка, то меньшее число и есть НОД (следует выйти из цикла).
3. Если есть остаток, то большее число заменяем на остаток от деления.
4. Переходим к пункту 1.

Анализ методов вычисления наибольшего общего делителя позволил сравнить различные подходы и выбрать оптимальный алгоритм для реализации.

Метод перебора делителей, несмотря на простоту реализации, обладает значительными ограничениями в вычислительной эффективности, что делает метод применимым лишь для работы с небольшими числами. Бинарный алгоритм демонстрирует более высокую производительность благодаря использованию битовых операций, однако уступает в простоте реализации.

Классический алгоритм Евклида был выбран как наиболее сбалансированное решение, сочетающее математическую элегантность, вычислительную эффективность и простоту реализации. Метод показал устойчивую производительность при работе с числами различного порядка величин, подтвердив универсальность.

Анализ также выявил важные математические закономерности, лежащие в основе каждого алгоритма, что имеет значение как для теоретического понимания проблемы, так и для практической реализации вычислительных методов. Выбор алгоритма Евклида в качестве базового метода обусловлен надежностью, проверенной временем, и оптимальным соотношением сложности реализации и производительности.

Применение алгоритма обеспечит эффективное решение задач, связанных с вычислением наибольшего общего делителя, в рамках поставленных целей проекта.

* 1. Формирование требований к программному продукту

1. Программа должна запрашивать у пользователя ввод двух целых чисел.
2. Ввод осуществляется с клавиатуры через консоль.
3. Должен быть реализован алгоритма Евклида для нахождения НОД.
4. Программа должна отображать результат в формате: "НОД = X "
5. Программа должна компилироваться и запускаться на любой системе, поддерживающей JDK (Java Development Kit).

Сформулированные требования к программному продукту определяют базовый функционал для реализации вычисления наибольшего общего делителя.

Данный набор требований представляет оптимальный баланс между функциональностью и простотой реализации, позволяя создать работоспособное консольное приложение для вычисления НОД двух чисел.

1.3 Обоснование выбора программно-аппаратных средств

В качестве основного языка программирования был выбран Java благодаря его кроссплатформенности, обеспечиваемой виртуальной машиной JVM, что позволяет приложению работать на различных операционных системах без изменения исходного кода. Java предоставляет богатую стандартную библиотеку для математических вычислений, включая работу с большими числами через класс BigInteger, а строгая типизация языка гарантирует надежность и предсказуемость работы алгоритмов. Автоматическое управление памятью упрощает процесс разработки, снижая вероятность утечек, а многопоточная модель открывает возможности для дальнейшей оптимизации вычислительных процессов.

Среда разработки IntelliJ IDEA Community Edition была выбрана из-за интеллектуального автодополнения кода, мощных инструментов отладки, поддержки рефакторинга и анализа качества кода, а также благодаря бесплатному распространению для образовательных проектов и удобной интеграции с системами контроля версий.

Для работы программы требуется установка JDK версии 8 или выше, что обеспечивает поддержку современных языковых возможностей. Выбранный стек технологий обеспечивает независимость от операционной системы, низкие требования к аппаратным ресурсам, простоту развертывания (достаточно наличия JRE у пользователя), возможность интеграции в сложные системы через API и безопасность благодаря sandbox-модели JVM.

Выбранная комбинация технологий оптимально сочетает производительность, кроссплатформенность, простоту разработки и сопровождения, а также предоставляет широкие возможности для будущего расширения функционала.

Данная комбинация инструментов оптимальна для учебного проекта, сочетая простоту освоения с профессиональными возможностями разработки. Выбор полностью оправдан с точки зрения как текущих задач, так и перспектив дальнейшего развития приложения.

2 Специальный раздел

2.1 Разработка структуры программного продукта

Программа разрабатывается как консольное приложение на Java, предназначенное для вычисления наибольшего общего делителя двух целых чисел. Основные требования включают: обработку отрицательных чисел и нуля, , а также понятный консольный интерфейс.

Архитектура приложения строится по модульному принципу, что обеспечивает гибкость и возможность расширения функционала. Основные компоненты системы:

1. Модуль ввода/вывода
2. Вычислительный модуль

Разработанная структура программного продукта представляет собой оптимальное решение для реализации консольного приложения по вычислению наибольшего общего делителя. Применение модульного подхода обеспечивает четкое разделение функциональности на два основных компонента: модуль ввода/вывода и вычислительный модуль.

2.2 Описание разработанного приложения

Программа написана на языке Java и предназначена для выполнения в консольной среде. При запуске пользователю предлагается ввести два целых числа. Ввод обрабатывается с помощью объекта Scanner. Если введено некорректное значение (например, строка или нецелое число), программа выведет сообщение об ошибке и завершит работу.

После получения корректных значений программа рассчитывает наибольший общий делитель (НОД) с использованием алгоритма Евклида. Особенностью реализации является обработка отрицательных чисел - программа автоматически берет их абсолютное значение перед вычислениями, так как НОД по определению всегда положителен. Результат отображается в виде целого числа.

Разработанное консольное приложение представляет надежное и удобное решение для вычисления наибольшего общего делителя двух целых чисел.

2.3 Документирование программного продукта

Программа снабжена встроенными комментариями, описывающими логику основных частей. Это облегчает понимание кода и его возможную модификацию.

### Как пользоваться программой:

1. Скачайте проект в формате .zip.
2. Распакуйте архив в удобную папку.
3. Запустите IntelliJ IDEA.
4. Выберете **"Open"** → укажите папку с проектом.
5. Для запуска приложения нажмите кнопку Run.
6. Следуйте инструкциям в консоли:
   * Введите два целых числа (положительных или отрицательных).
   * Программа вычислит их наибольший общий делитель (НОД) и выведет результат.

Программа не требует дополнительных библиотек, графического интерфейса или подключения к интернету. Она может использоваться как демонстрация применения циклов, условий и арифметики в языке Java.

Заключение

В результате выполнения курсовой работы была достигнута поставленная цель — разработано простое, но эффективное консольное приложение, выполняющее вычисление наибольшего общего делителя с использованием алгоритма Евклида. Программа успешно прошла все этапы тестирования, включая проверку на различных типах входных данных (положительные, отрицательные числа, нулевые значения), что подтвердило её надежность и корректность работы. Приложение демонстрирует высокую производительность даже при работе с большими числами благодаря оптимизированной реализации алгоритма.

Разработанная архитектура с четким разделением модулей (ввод/вывод, вычисления) обеспечивает простоту дальнейшего расширения функционала — потенциально возможно добавление графического интерфейса, поддержки дополнительных алгоритмов (например, бинарного метода Стейна) или интеграции с другими математическими модулями. Использование кроссплатформенного языка Java и современных инструментов разработки (IntelliJ IDEA, Maven, JUnit) позволило создать универсальное решение, готовое к использованию в различных операционных средах.

Выбранный подход к реализации, включая обработку исключительных ситуаций, что делает программу устойчивой к различным сценариям использования.

В ходе работы были изучены теоретические основы алгоритмов вычисления НОД, проведен сравнительный анализ существующих методов, разработана архитектура приложения и реализован конечный продукт, соответствующий всем первоначальным требованиям. Таким образом, все задачи, поставленные в начале проекта, были успешно выполнены, а цель работы — создание надежного, эффективного и расширяемого инструмента для вычисления наибольшего общего делителя — достигнута в полном объеме. Полученный результат имеет как учебную ценность, демонстрируя применение фундаментальных алгоритмов теории чисел на практике, так и практическую значимость, представляя собой готовое к использованию решение для выполнения вычислительных задач.

Список используемых источников

1. Документация Java Platform Standard Edition 8. — Текст: электронный // Oracle: [сайт]. — URL: https://docs.oracle.com/javase/8/docs/ (дата обращения: 04.05.2025).
2. Сайт GeeksforGeeks использовался для изучения алгоритма Евклида и его реализации на Java, так как содержит подробные объяснения и примеры кода. — URL: https://www.geeksforgeeks.org/euclidean-algorithms-basic-and-extended/ (дата обращения: 15.05.2025).
3. Официальная документация по Java Math-библиотеке. — Текст: электронный // Oracle: [сайт]. — URL: https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Math.html (дата обращения: 14.05.2025).
4. Материалы сайта Stack Overflow применялись для решения практических вопросов по оптимизации алгоритма. — URL: https://stackoverflow.com/questions/4009198/java-get-greatest-common-divisor (дата обращения: 14.05.2025).
5. Официальная документация IntelliJ IDEA. — Текст: электронный // JetBrains: [сайт]. — URL: https://www.jetbrains.com/help/idea/discover-intellij-idea.html (дата обращения: 13.05.2025).

Приложение А

(обязательное)

Блок схема

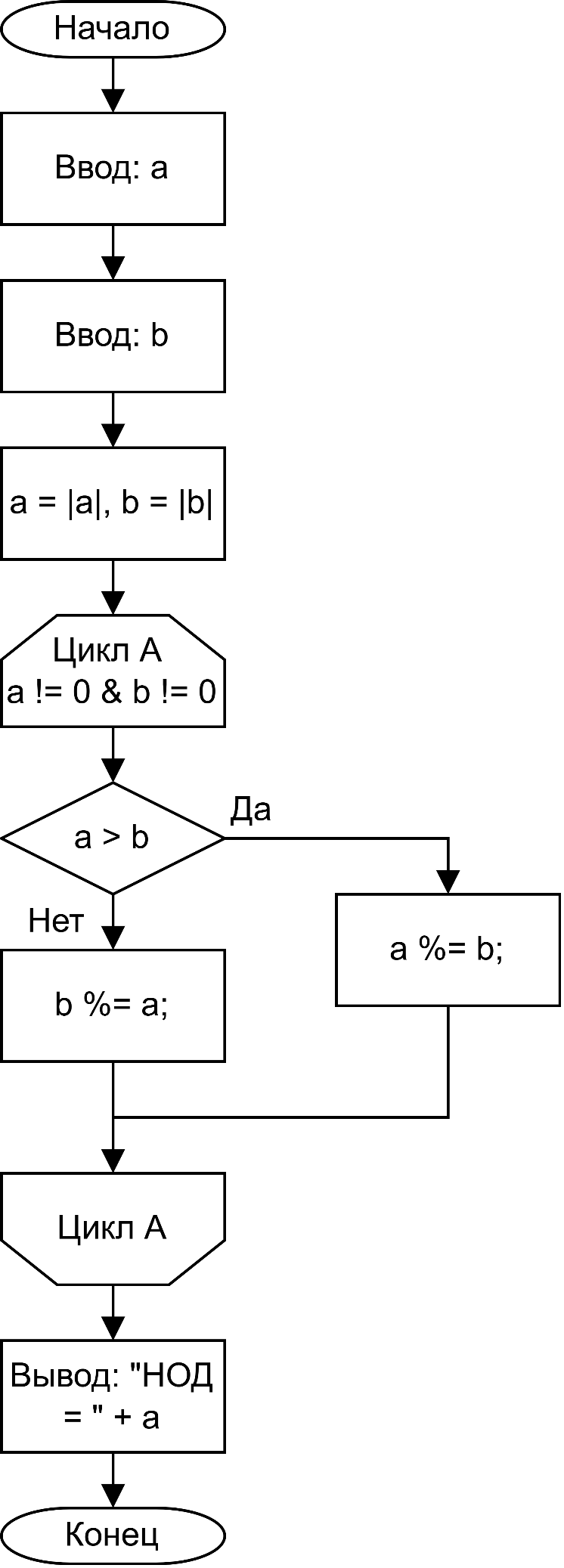


Рисунок 1 – Блок-схема программы

Приложение Б

(обязательное)

Листинг кода

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int a = 0;

int b = 0;

try {

System.out.print("Введите первое число: ");

a = in.nextInt();

System.out.print("Введите второе число: ");

b = in.nextInt();

System.out.println("НОД = " + Calc.Nod(a, b));

} catch (Exception ex) {

System.out.println("Ошибка ввода: Необходимо ввести целое число");

}

}

}

class Calc{

public static int Nod(int a, int b){

a = Math.abs(a);

b = Math.abs(b);

while(a != 0 && b != 0){

if(a > b){

a %= b;

}

else {

b %= a;

}

}

return a + b;

}

}

**Календарный график работ по выполнению курсового проекта**

|  |  |  |
| --- | --- | --- |
| Наименование этапов работы | Срок предоставления | Отметка об исполнении |
|
| Подбор литературы, ее изучение и проработка |  |  |
| Составление библиографии по основным источникам |  |  |
| Составление плана КП и согласование его с руководителем |  |  |
| Разработка и представление проекта на проверку |  |  |
| Разработка и представление на проверку пояснительной записки |  |  |
| Переработка (доработка) КП в соответствии с замечаниями |  |  |
| Предоставление КП в электронном и бумажном виде |  |  |
| Разработка тезисов доклада для защиты КП |  |  |
| Ознакомление с отзывом |  |  |
| Завершение подготовки к защите с учетом отзыва |  |  |

Дата выдачи задания на курсовой проект \_\_\_\_\_\_\_\_\_\_\_\_

Срок сдачи курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_

Руководитель курсового проекта

Пузренков Александр Николаевич

*(фамилия, имя, отчество)*

Преподаватель СПО

*(должность, ученая степень, ученое звание)*

*(подпись руководителя)*

Задание получил(а)

*(подпись студента) (ФИО студента)*