Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

**Знакомство с языком программирования Java и средой разработки Eclipse**

Лабораторная работа №1

по курсу «Разработка программных систем»

Вариант 3

Выполнил студент группы ИВТб-31 /Категов А. Д./ Проверил преподаватель /Чистяков Г.А./

Киров 2024

Задание:

Реализовать класс SmallInteger для работы с целыми числами, не превосходящими по абсолютному значению 104 . Класс должен содержать следующие публичные методы:

− сложения, вычитания, умножения;

− целочисленного деления и определения остатка от деления.

Сигнатура методов должна иметь вид «public SmallInteger operation(SmallInteger arg)». Кроме того, класс должен иметь не менее двух конструкторов. В случае выхода промежуточных или окончательного результата за границы диапазона представления должна инициироваться исключительная ситуация, расширяющая базовую иерархию.

1. Описание назначения и функциональных возможностей консольного приложения

Консольное приложение предназначено для работы с целыми числами в диапазоне от -10,000 до 10,000, используя класс SmallInteger, который предоставляет возможности выполнения основных арифметических операций: сложение, вычитание, умножение, целочисленное деление и вычисление остатка от деления.

Приложение позволяет пользователю:

1. Ввести два целых числа в диапазоне, допустимом для типа SmallInteger;
2. Выбрать одну из предложенных операций;
3. Получить результат выбранной операции или сообщение об ошибке, если результат выходит за пределы допустимого диапазона;
4. Завершить работу приложения по выбору пользователя.
5. Программная реализация

Листинг программной реализации приведен в приложении А.

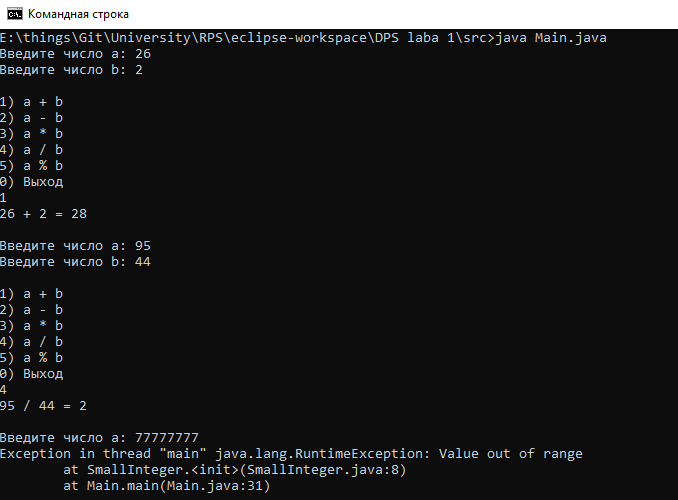


Рисунок 1 – Экранная форма

**Вывод**: в ходе данной лабораторной работы были изучены основные конструкции языка программирования Java, структура программы, стандартные средства ввода/вывода, изучен основной функционал IDE Eclipse. На основе полученных знаний был разработан класс для работы с целыми числами, не превосходящими по абсолютному значению 104 и написано консольное приложение. Производя проверку результатов действия программы, можно прийти к заключению о верности её работы. На основании всего сказанного выше можно сделать вывод о том, что цель достигнута, необходимые знания получены, задание выполнено верно, что доказывается полученными результатами.

Приложение А

(обязательное)

Листинг программы

**import** java.util.\*; // for Scanner

// Main class of the application, containing the entry point

**public** **class** Main {

// Method to print the menu with available operations

**public** **static** **void** printMenu() {

System.***out***.println("1) a + b");

System.***out***.println("2) a - b");

System.***out***.println("3) a \* b");

System.***out***.println("4) a / b");

System.***out***.println("5) a % b");

System.***out***.println("0) Выход");

}

// Method for user input with validation to ensure an integer is entered

**public** **static** **int** input(Scanner s) {

**while** (s.hasNextInt() == **false**) { // Continues to prompt the user until a valid integer is entered

System.***out***.print("Введите целое число: ");

s.next(); // Skip the invalid input

}

**return** s.nextInt(); // Return the entered integer

}

// Entry point of the application

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***); // Create a Scanner object for reading input

**int** key; // Variable to store the user's choice of operation

**do** {

System.***out***.print("Введите число a: ");

SmallInteger a = **new** SmallInteger(*input*(scan));

System.***out***.print("Введите число b: ");

SmallInteger b = **new** SmallInteger(*input*(scan));

System.***out***.println();

*printMenu*();

key = *input*(scan);

**switch** (key) {

**case** 1: // Perform addition and display the result.

System.***out***.println(a.getValue() + " + " + b.getValue() + " = " + a.add(b) + "\n");

**break**;

**case** 2: // Perform subtraction and display the result

System.***out***.println(a.getValue() + " - " + b.getValue() + " = " + a.sub(b) + "\n");

**break**;

**case** 3: // Perform multiplication and display the result

System.***out***.println(a.getValue() + " \* " + b.getValue() + " = " + a.mul(b) + "\n");

**break**;

**case** 4: // Perform integer division and display the result

System.***out***.println(a.getValue() + " / " + b.getValue() + " = " + a.integerDivision(b) + "\n");

**break**;

**case** 5: // Perform remainder division and display the result

System.***out***.println(a.getValue() + " % " + b.getValue() + " = " + a.remainderDivision(b) + "\n");

**break**;

**default**:

**break**;

}

} **while** (key != 0); // Continue until the user chooses to exit (0)

scan.close();

}

}

// Class for handling integers within the range [-10000, 10000]

**public** **class** SmallInteger {

**private** **int** value; // Stores the value of the number

**private** **static** **final** **int** ***MAX\_VALUE*** = 10000; // Maximum allowable value

**private** **static** **final** **int** ***MIN\_VALUE*** = -10000; // Minimum allowable value

// Primary constructor that accepts an integer and checks if it is within the valid range

**public** SmallInteger(**int** value) {

**if** (value > ***MAX\_VALUE*** || value < ***MIN\_VALUE***) {

**throw** **new** RuntimeException("Value out of range");

}

**this**.value = value;

}

// Copy constructor that creates a new object based on an existing one

**public** SmallInteger(SmallInteger other) {

**this**(other.value); // Calls the primary constructor

}

// Method to add two SmallInteger objects

**public** SmallInteger add(SmallInteger arg) {

**int** result = **this**.value + arg.value;

**if** (result > ***MAX\_VALUE*** || result < ***MIN\_VALUE***) {

**throw** **new** RuntimeException("Result out of range");

}

**return** **new** SmallInteger(result);

}

// Method to subtract one SmallInteger object from another

**public** SmallInteger sub(SmallInteger arg) {

**int** result = **this**.value - arg.value;

**if** (result > ***MAX\_VALUE*** || result < ***MIN\_VALUE***) {

**throw** **new** RuntimeException("Result out of range");

}

**return** **new** SmallInteger(result);

}

// Method to multiply two SmallInteger objects

**public** SmallInteger mul(SmallInteger arg) {

**int** result = **this**.value \* arg.value;

**if** (result > ***MAX\_VALUE*** || result < ***MIN\_VALUE***) {

**throw** **new** RuntimeException("Result out of range");

}

**return** **new** SmallInteger(result);

}

// Method for integer division of one SmallInteger object by another

**public** SmallInteger integerDivision(SmallInteger arg) {

**if** (arg.value == 0) {

**throw** **new** ArithmeticException("Division by zero");

}

**int** result = **this**.value / arg.value;

**return** **new** SmallInteger(result);

}

// Method to find the remainder of the division of one SmallInteger object by another

**public** SmallInteger remainderDivision(SmallInteger arg) {

**if** (arg.value == 0) {

**throw** **new** ArithmeticException("Division by zero");

}

**int** result = **this**.value % arg.value;

**return** **new** SmallInteger(result);

}

// Method to get the current value

**public** **int** getValue() {

**return** **this**.value;

}

}