Федеральное государственное автономное

образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | Институт космических и информационных технологий | | институт | | Кафедра программной инженерии | | кафедра | |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №4**

|  |
| --- |
| Аппликативы и алгоритмы |
| тема |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | К. В. Богданов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-16/2Б, 032153882 |  |  |  | Д. А. Скоробогатов |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2024

**СОДЕРЖАНИЕ**

[1 Цель 3](#_Toc157723631)

[2 Задачи 3](#_Toc157723632)

[3 Ход выполнения 4](#_Toc157723633)

[3.1 Написание скрипта 4](#_Toc157723634)

[4 Выводы 7](#_Toc157723635)

[Список использованных источников 8](#_Toc157723636)

# Цель

Изучить возможности классов BigInt в Scala и класса BigInteger в Java. Используя эти классы, реализовать на Scala алгоритм RSA и вспомогательные функции для чисел любой размерности и оформить их в отдельный пакет.

Написать с использованием данного пакета утилиту (по выбору: либо для командной строки, либо GUI c использованием Spring), позволяющую решать следующие задачи:

* генерировать диапазон простых чисел и позволять их выбирать из списка
* генерировать ключи RSA, основанные на простых числах любой размерности
* разделять открытый и секретный ключи RSA (хранить их раздельно)
* вводить (или вставлять из буфера) сообщение для шифрования
* шифровать сообщение открытым ключом
* отображать зашифрованное сообщение в виде текста или сохранять зашифрованное сообщение в файле для возможности его передачи
* прочитать зашифрованное сообщение из файла при его получении
* расшифровывать полученное сообщение секретным ключом

# Задачи

* Имплементировать алгоритм RSA в Scala;
* Написать скрипт для своего варианта;
* Написать отчёт.

# Ход выполнения

## Написание скрипта

Листинг 1 – Исходный код программы

package controllers

import javafx.beans.value.{ChangeListener, ObservableValue}

import javafx.fxml.{FXML, Initializable}

import javafx.scene.input.{Clipboard, ClipboardContent}

import javafx.scene.{control => jfxsc}

import rsa.RSA

import scalafx.scene.control.Alert

import scalafx.scene.control.Alert.AlertType

import utils.StringManipulation

import java.net.URL

import java.nio.charset.StandardCharsets

import java.util.ResourceBundle

class GeneratePrimesController extends Initializable{

var rsa: RSA = \_

@FXML

var minPrime: jfxsc.TextField = \_

@FXML

var maxPrime: jfxsc.TextField = \_

@FXML

var PQList: jfxsc.ListView[BigInt] = \_

@FXML

var PField: jfxsc.TextField = \_

@FXML

var QField: jfxsc.TextField = \_

@FXML

var EField: jfxsc.TextField = \_

@FXML

var NField: jfxsc.TextField = \_

@FXML

var DField: jfxsc.TextField = \_

@FXML

var inputField: jfxsc.TextArea = \_

@FXML

var outputField: jfxsc.TextArea = \_

@FXML

def generatePrimes(): Unit = {

try {

val minPrimeInt = BigInt(minPrime.getText)

Продолжение листинга 1

val maxPrimeInt = BigInt(maxPrime.getText.toInt)

if (minPrimeInt > maxPrimeInt) throw new ArithmeticException("Min > Max!")

PQList.setItems(RSA.primes(minPrimeInt, maxPrimeInt))

} catch {

case \_: ArithmeticException => new Alert(AlertType.Error, "Min > Max!").showAndWait()

case \_ => new Alert(AlertType.Error, "Min or Max not Int!").showAndWait()

}

}

@FXML

def calculateRSA(): Unit = {

try {

val p = BigInt(PField.getText)

val q = BigInt(QField.getText)

rsa = new RSA(p, q)

EField.setText(rsa.publicKey.toString())

NField.setText(rsa.n.toString())

DField.setText(rsa.privateKey.toString())

} catch {

case \_: Exception => new Alert(AlertType.Error, "P or Q not Int!").showAndWait()

}

}

@FXML

def encode(): Unit = {

if (rsa == null) {

new Alert(AlertType.Error, "E, N and D not generate!!!").showAndWait()

return

}

val string = inputField.getText

val splitString = StringManipulation.splitStringToBytes(string, 20)

val encodeString = rsa.encrypt(splitString)

outputField.setText(encodeString.map(\_.toInt).mkString(","))

}

@FXML

def decode(): Unit = {

val bigIntArray = inputField.getText.split(",").map(BigInt(\_))

val decodeBytes = rsa.decrypt(bigIntArray)

outputField.setText(new String(decodeBytes, StandardCharsets.UTF\_8))

}

override def initialize(url: URL, resourceBundle: ResourceBundle): Unit = {

PQList.getSelectionModel.selectedItemProperty().addListener(new ChangeListener[BigInt] {

override def changed(observableValue: ObservableValue[\_ <: BigInt], t: BigInt, t1: BigInt): Unit = {

val currentPrime = PQList.getSelectionModel.getSelectedItem

if (currentPrime == null) return

val clipboard = Clipboard.getSystemClipboard

val content = new ClipboardContent()

content.putString(currentPrime.toString())

Окончание листинга 1

clipboard.setContent(content)

}

})

}

}

Пример выполнения скрипта представлен на рисунке 1.

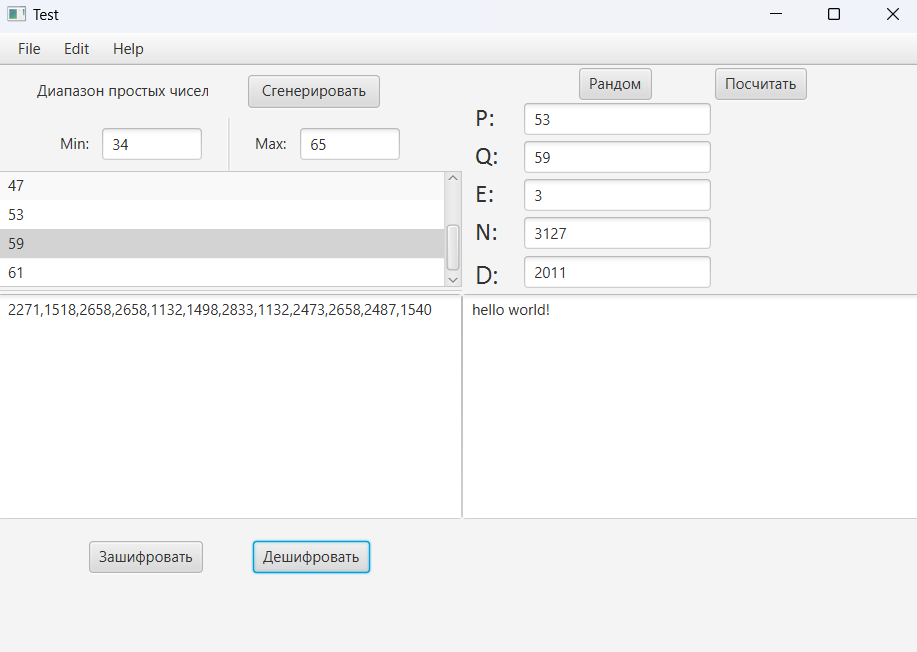


Рисунок 1 – Пример выполнения скрипта

# Выводы

В ходе выполнения практической работы:

* Имплементировать алгоритм RSA в Scala;
* Написать скрипт для своего варианта;
* Написан отчёт.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СТО 7.5-07-2021 Система менеджмента качества. Общие требования построению, изложению и оформлению документов учебной деятельности. – Взамен СТО 4.2-07-2014 ; введ. 07.12.2021. – Красноярск : СФУ 2021. – 61 с.