Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Лабораторная работа № 3

по дисциплине: «Теория псевдослучайных генераторов»

на тему: «Генератор Фибоначчи псевдослучайных чисел на регистрах сдвига с линейной обратной связью»

Выполнил

обучающийся гр. ВКБ41

Якушевский Сергей Сергеевич

Проверила:

Ст. пр. Артамонова Е.А.

Лабораторная работа № 3

«Генератор Фибоначчи псевдослучайных чисел на регистрах сдвига с линейной обратной связью»

Вариант № 28

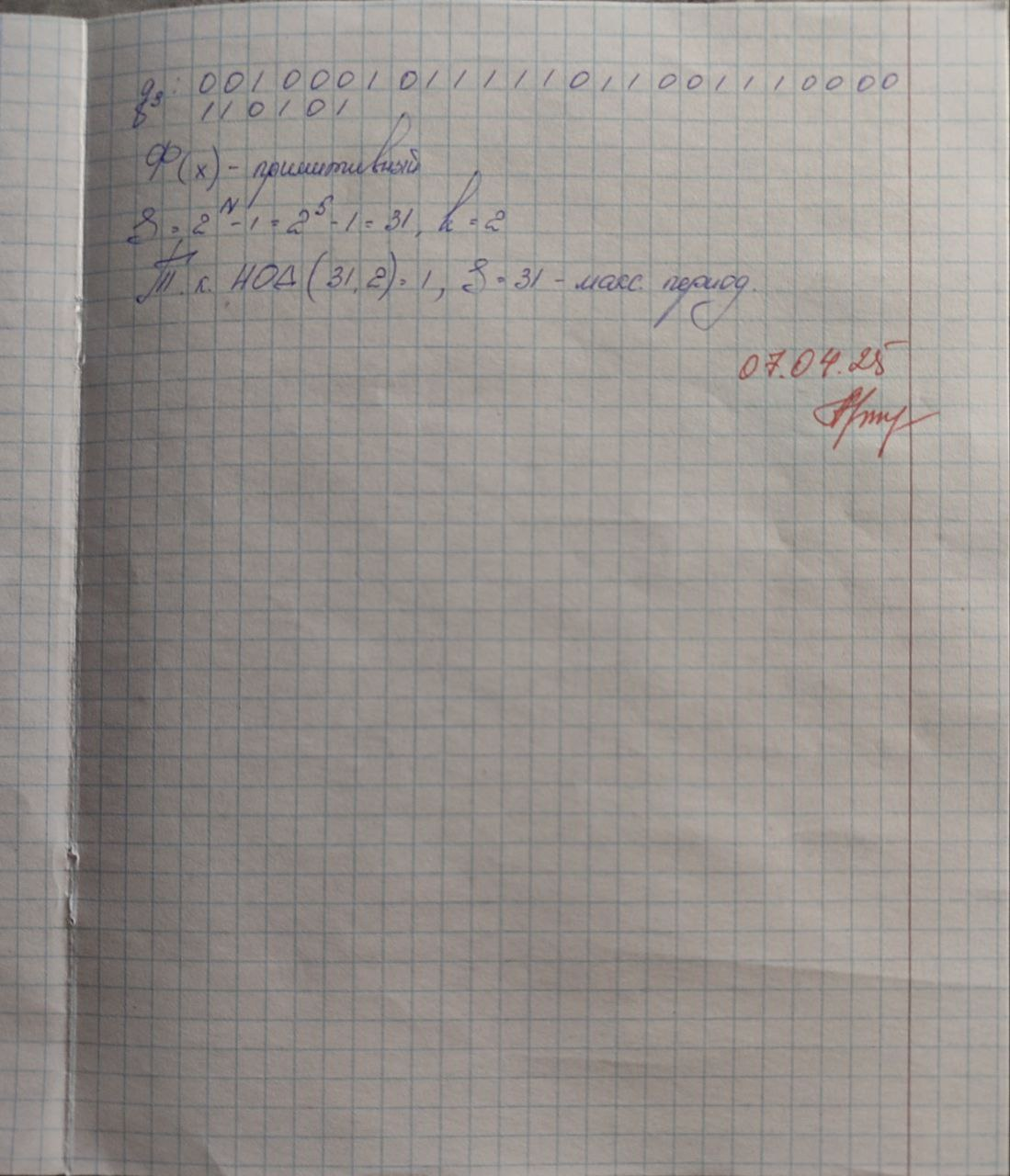
Задание.

Построить псевдослучайную двоичную М – последовательность генератором Фибоначчи. Примитивный многочлен и разряд сдвига выбрать согласно варианту из таблицы. Начальное состояние выбрать произвольно. Выводить все промежуточные результаты. Определить максимальный период счетом и по формуле, сравнить полученные результаты. Письменно составить схему и диаграмму построения двоичной М – последовательности. Полученную двоично-десятичную последовательность перевести в десятичную. Последовательность сохранить для дальнейшего исследования.

## Ручная реализация.

## 





## Входные данные.

## 

## Результат.



## Листинг программы.

import dstu.csae.index.Index;  
import dstu.csae.polynomial.Polynomial;  
import lombok.NonNull;  
import dstu.csae.topg.register.Register;  
  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.List;  
import java.util.Scanner;  
import java.util.stream.Collectors;  
  
public class Main {  
 public static final String *RETRY* = "Повторите попытку.";  
 public static final String *LINE\_SEPARATOR* = File.*separator*;  
 public static final String *FILE\_PATH* = System.*getProperty*("user.home") +  
 *LINE\_SEPARATOR* + "Desktop" + *LINE\_SEPARATOR* + "Fibonacci.txt";  
  
 public static void main(String[] args) {  
 final String OUTPUT\_SEPARATOR = " -> ";  
 final String FILE\_SEPARATOR = "";  
 Register register = null;  
 while(register == null){  
 try{  
 int k = *inputK*();  
 Polynomial polynomial = *inputPrimitivePolynomial*();  
 int[] startState = *inputStartState*();  
 register = new Register(polynomial, startState, k);  
 }catch (Exception ex){  
 System.*out*.println(ex.getMessage());  
 }  
 }  
 System.*out*.println(register);  
 System.*out*.println("Значения регистра:");  
 System.*out*.println(*printStates*(register, OUTPUT\_SEPARATOR));  
 System.*out*.println("Период генератора: " + register.getPeriod());  
 System.*out*.println("S = 2" + Index.*toSuperscript*("n") + " - 1 = " + register.getS());  
 System.*out*.println("Период генератора " + (register.getS() == register.getPeriod() ? "" : "не ") + "максимальный.");  
 try(FileWriter print = new FileWriter(*FILE\_PATH*)) {  
 print.write(*printStates*(register, FILE\_SEPARATOR));  
 System.*out*.println("Значения сохранены в файл: " + *FILE\_PATH*);  
 } catch (IOException e) {  
 System.*out*.println(e.getMessage());  
 }  
 }  
  
 public static int inputK(){  
 final String QUERY = "Введите значение сдвига: ";  
 Scanner scanner = new Scanner(System.*in*);  
 int shift = -1;  
 while(shift < 1){  
 System.*out*.print(QUERY);  
 try{  
 shift = Integer.*parseInt*(scanner.nextLine());  
 if(shift < 1){  
 System.*out*.println(*RETRY*);  
 }  
 }catch (Exception ex){  
 System.*out*.println(*RETRY*);  
 }  
 }  
 return shift;  
 }  
  
 public static int[] inputStartState(){  
 final String QUERY = "Введите начальные значения ячеек регистра через пробел: ";  
 Scanner scanner = new Scanner(System.*in*);  
 int[] startState = null;  
 while (startState == null){  
 System.*out*.print(QUERY);  
 try{  
 startState = Arrays.*stream*(scanner.nextLine().split("\\s"))  
 .mapToInt(Integer::*parseInt*)  
 .toArray();  
 }catch (Exception ex){  
 System.*out*.println(*RETRY*);  
 }  
 }  
 return startState;  
 }  
  
 public static Polynomial inputPrimitivePolynomial(){  
 final String QUERY = "Введите значение коэффициентов примитивного полинома через пробел: ";  
 Scanner scanner = new Scanner(System.*in*);  
 int[] coefficients = null;  
 while (coefficients == null){  
 System.*out*.print(QUERY);  
 try{  
 int[] coefs = Arrays.*stream*(scanner.nextLine().split("\\s"))  
 .mapToInt(Integer::*parseInt*)  
 .toArray();  
 if (new Polynomial(coefs).getDegree() < 2) {  
 System.*out*.println(*RETRY*);  
 continue;  
 }  
 coefficients = coefs;  
 }catch (Exception ex){  
 System.*out*.println(ex.getMessage());  
 System.*out*.println(*RETRY*);  
 }  
 }  
 return new Polynomial(coefficients);  
 }  
  
 public static String printStates(@NonNull Register register,  
 @NonNull String separator){  
 List<String> out = new ArrayList<>();  
 int[] startState = register.getStartPosition();  
 int cellCount = startState.length;  
 int period = 1;  
 int[] current = register.next(cellCount);  
 String state = *stateToString*(current);  
 out.add(state);  
 current = register.next(cellCount);  
 period ++;  
 while (!Arrays.*equals*(current, startState)){  
 state = *stateToString*(current);  
 out.add(state);  
 current = register.next(cellCount);  
 period ++;  
 }  
 state = *stateToString*(current);  
 out.add("" + Integer.*parseInt*(state, 2));  
 return String.*join*("", out);  
 }  
  
 private static String stateToString(int[] state){  
 //return String.valueOf(state[state.length - 1]);  
 return Arrays.*stream*(state)  
 .mapToObj(Integer::*toBinaryString*)  
 .collect(Collectors.*joining*(""));  
 }  
}

## Вывод по работе.

В лабораторной работе были изучены и применены на практике методологические знания в принципе работы, построении и использовании генератора Фибоначчи на основе регистра сдвига с линейной обратной связью. Исходя из варианта, были определены следующие данные для примера:

Данный генератор обладает периодом последовательности в 31 значение, что соответствует его максимальному периоду.