|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное АВТОНОМНОЕ образовательное учреждение высшего образования  «Национальный исследовательский ядерный университет «МИФИ» |
| **Новоуральский технологический институт –**  филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»  **(НТИ НИЯУ МИФИ)** |

**Колледж НТИ**

Цикловая методическая комиссия информационных технологий

ОТЧЕТ №11

по практическому занятию на тему

**«ПРОГРАММНАЯ РЕАЛИЗАЦИЯ КРИПТОГРАФИЧЕСКИХ АЛГОРИТМОВ»**

ПМ.05 «Разработка программного обеспечения компьютерных сетей»

МДК.05.01 «Защита информации в КС»

Специальность СПО 09.02.03

«Программирование в компьютерных системах»

очная форма обучения

на базе основного общего образования

|  |  |  |
| --- | --- | --- |
| Выполнил  студент группы КПР–47 Д  Егорушкин И.А. | \_\_\_\_\_\_\_\_\_26.01.2021\_\_\_\_\_\_\_\_  дата | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись |
| Проверил  преподаватель  Горницкая И.И. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  дата | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись |

Новоуральск 2020

**Цель работы:** Анализ рисков информационной безопасности

**Оборудование:**

AMD Ryzen 5 3550U

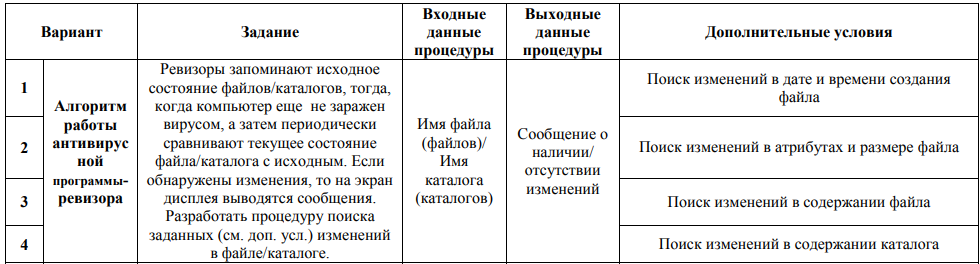
ОЗУ 8 Гб

**Программное обеспечение:**

Windows 10 Professional 64 бит;

**Ход работы:**

**Задание**



**Алгоритм работы программы**

Пользователь вводит файл или каталог после чего все файлы заносятся в хранилище, которое сверяет каждые 10 секунд файлы.

В случаи повреждения или измены файла будет предложено переместить в карантин.

Поиск ищет совпадения в хранилище файлов.

**Листинг программы**

**package org.example;**

**import java.io.File;**

**import java.io.IOException;**

**import java.util.\*;**

**import javafx.application.Platform;**

**import javafx.collections.FXCollections;**

**import javafx.collections.ObservableList;**

**import javafx.fxml.FXML;**

**import javafx.scene.control.Button;**

**import javafx.scene.control.Label;**

**import javafx.scene.control.TableColumn;**

**import javafx.scene.control.TextField;**

**import javafx.scene.control.cell.PropertyValueFactory;**

**import javafx.scene.input.MouseEvent;**

**import javafx.scene.layout.VBox;**

**import javafx.stage.DirectoryChooser;**

**import javafx.stage.FileChooser;**

**import org.apache.commons.io.FileUtils;**

**public class PrimaryController {**

**@FXML**

**public Button fileInput;**

**@FXML**

**public TextField inputFile;**

**public CustomTableView<Table> table;**

**public static Set<SaveFile> saveFiles = new TreeSet<>(Comparator.comparing(SaveFile::getUrl));**

**public VBox root;**

**public Button packageInput;**

**private static final int tikConst = 10;**

**public static int tik = tikConst;**

**public static Thread thread;**

**public Label chekTime;**

**@FXML**

**private void initialize() throws IOException {**

**table = new CustomTableView<Table>();**

**TableColumn<Table, String> colName = new TableColumn<Table, String>("Имя");**

**TableColumn<Table, String> colUrl = new TableColumn<Table, String>("Url");**

**TableColumn<Table, String> colStat = new TableColumn<Table, String>("Статус");**

**TableColumn<Table, String> tableStringTableColumn = crateTable("Дата создания", "timeCrate", 100);**

**TableColumn<Table, String> tableStringSize = crateTable("Дата создания", "size", 100);**

**TableColumn<Table, String> colDataEdit = new TableColumn<Table, String>("Дата изменения");**

**colName.setCellValueFactory(new PropertyValueFactory<>("name"));**

**colUrl.setCellValueFactory(new PropertyValueFactory<>("url"));**

**colStat.setCellValueFactory(new PropertyValueFactory<>("circle"));**

**colDataEdit.setCellValueFactory(new PropertyValueFactory<>("time"));**

**colStat.setMinWidth(100);**

**colUrl.setMinWidth(100);**

**colDataEdit.setMinWidth(100);**

**colStat.setMaxWidth(100);**

**colName.setStyle(" -fx-alignment: CENTER");**

**colUrl.setStyle(" -fx-alignment: CENTER");**

**colStat.setStyle(" -fx-alignment: CENTER");**

**colDataEdit.setStyle(" -fx-alignment: CENTER");**

**colName.setEditable(false);**

**colUrl.setEditable(false);**

**colStat.setEditable(false);**

**colDataEdit.setEditable(false);**

**table.getColumns().addAll(colName, colUrl, tableStringTableColumn, colDataEdit, tableStringSize, colStat);**

**root.getChildren().add(root.getChildren().size() - 2, table);**

**try {**

**thread.stop();**

**} catch (Exception e) {**

**}**

**thread = new Thread(new Runnable() {**

**@Override**

**public void run() {**

**while (true) {**

**Platform.runLater(new Runnable() {**

**@Override**

**public void run() {**

**chekTime.setText("Проверка через:" + tik);**

**}**

**});**

**if (tik <= 1) {**

**check();**

**tik = tikConst;**

**}**

**try {**

**tikAdd();**

**} catch (InterruptedException e) {**

**e.printStackTrace();**

**}**

**}**

**}**

**});**

**thread.setDaemon(true);**

**thread.start();**

**check();**

**}**

**private void tikAdd() throws InterruptedException {**

**tik--;**

**Thread.sleep(1000);**

**}**

**private TableColumn<Table, String> crateTable(String s, String name, int w) {**

**TableColumn<Table, String> col = new TableColumn<Table, String>(s);**

**col.setCellValueFactory(new PropertyValueFactory<>(name));**

**col.setStyle(" -fx-alignment: CENTER");**

**col.setEditable(false);**

**return col;**

**}**

**@FXML**

**public void failGet(MouseEvent mouseEvent) {**

**FileChooser fileChooser = new FileChooser();**

**File file = fileChooser.showOpenDialog(App.getStage());**

**if (file != null) {**

**inputFile.setText(file.getPath());**

**}**

**}**

**public void packageGet(MouseEvent mouseEvent) {**

**DirectoryChooser directoryChooser = new DirectoryChooser();**

**File file = directoryChooser.showDialog(App.getStage());**

**if (file != null) {**

**inputFile.setText(file.getPath());**

**}**

**}**

**@FXML**

**public void addFile(MouseEvent mouseEvent) throws IOException {**

**new Thread(new Runnable() {**

**@Override**

**public void run() {**

**File file = new File(inputFile.getText());**

**if (!file.isDirectory()) {**

**try {**

**saveFiles.add(new SaveFile(file, file.getPath(), FileUtils.readFileToByteArray(file)));**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**ObservableList<Table> list = FXCollections.observableArrayList();**

**saveFiles.forEach(saveFile -> list.add(new Table(saveFile.getName().getName(), saveFile.getName())));**

**table.setItems(list);**

**} else {**

**processFilesFromFolder(file).forEach(file1 -> {**

**try {**

**saveFiles.add(new SaveFile(file1, file1.getPath(), FileUtils.readFileToByteArray(file1)));**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**ObservableList<Table> list = FXCollections.observableArrayList();**

**saveFiles.forEach(saveFile -> list.add(new Table(saveFile.getName().getName(), saveFile.getName())));**

**table.setItems(list);**

**});**

**}**

**}**

**}).start();**

**}**

**public Set<File> processFilesFromFolder(File folder) {**

**Set<File> files = new HashSet<>();**

**File[] folderEntries = folder.listFiles();**

**for (File entry : folderEntries) {**

**if (entry.isDirectory()) {**

**processFilesFromFolder(entry);**

**continue;**

**}**

**files.add(entry);**

**}**

**return files;**

**}**

**public void check(MouseEvent mouseEvent) {**

**check();**

**}**

**private void check() {**

**ObservableList<Table> list = FXCollections.observableArrayList();**

**saveFiles.forEach(**

**saveFile -> {**

**try {**

**saveFile.setChek(Arrays.equals(FileUtils.readFileToByteArray(new File(saveFile.Url)), saveFile.bytes));**

**Table table = new Table(saveFile.name.getName(), saveFile.getName());**

**if (!saveFile.isChek()) {**

**table.setCircle("Не корректен");**

**} else {**

**table.setCircle("Ok");**

**}**

**list.add(table);**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**);**

**Platform.runLater(new Runnable() {**

**@Override**

**public void run() {**

**table.setItems(list);**

**}**

**});**

**}**

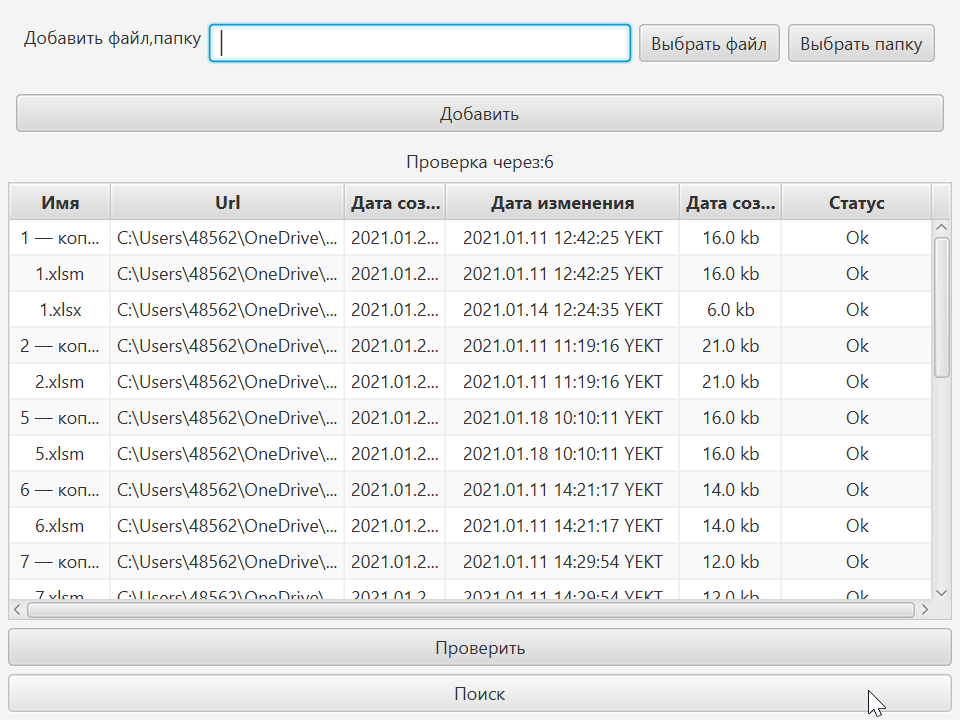
**public void search(MouseEvent mouseEvent) throws IOException {**

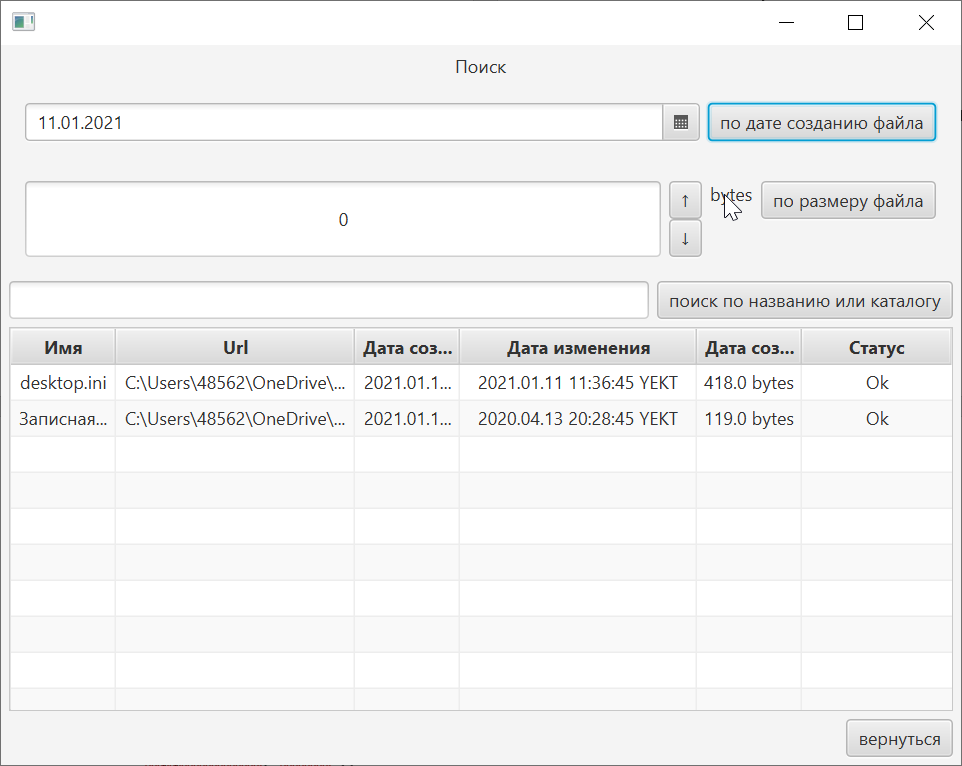
**App.setRoot("serch");**

**}**

**}**

**Пример работы программы**





Вывод : Было изучено сравнение файлов и их поиск.