Оглавление

Введение

По мере развития и модернизации компьютерных систем и программного обеспечения увеличивается объем и уязвимость хранимых в них данных. Наибольшую опасность в результате риска заражения компьютерными вирусами представляет возможность искажения, удаления, копирования важной информации. Одним из методов борьбы с вирусами является антивирусные программы.

Также развитие глобальных сетей ведет к быстрому увеличению количества не только пользователей, но и атак на компьютеры, подключенные к Интернету. Именно поэтому при подключении к Интернету корпоративной или локальной сети важно не забыть позаботиться об обеспечении ее информационной безопасности. Часть задач по отражению наиболее возможных угроз для внутренних сетей могут решать межсетевые экраны.

Межсетевой экран пропускает через себя весь трафик, принимая относительно каждого проходящего пакета решение: дать ему возможность пройти или нет. В последнее время тема использования межсетевых экранов становится актуальной, т.к. большее число людей и организаций становится жертвами компьютерных взломщиков. И, тем не менее, количество пользователей в сети не уменьшается, а наоборот растет с огромной прогрессией.

Основной целью является проверка наличия и работоспособности средств безопасности персонального компьютера (ПК), таких как межсетевой экран и антивирусная программа. Также осуществить проверку подключения к сети Интернет, реализовать интерфейс ПО и разработать алгоритмы модулей.

1. Разработка алгоритма, интерфейса и программная реализация.

На первом этапе требуется разработать интерфейс приложения. Данный интерфейс должен быть интуитивно понятен, а также реализовывать все функциональные возможности. Приложение выполнено в виде окна с соответствующими элементами управления.

На листинге 1.1 представлена программная реализация интерфейса.

Листинг 1.1 – Создание интерфейса.

```
sg.theme('LightGreen5')
layout = [  [sg.Button('Проверка подключения к интернету'), sg.Output(key =
1,)],
            [sg.Button('Проверка наличия межсетевого экрана'), sg.Output(key =
2)],
            [sg.Button('Проверка работоспособности межсетевого экрана'),
sg.Output(key = 3)],
            [sg.Button('Проверка наличия антивируса'), sg.Output(key = 4)],
            [sg.Button('Проверка работоспособности антивируса'),
sg.Output(key = 5)],
            [sg.Text('Вывод результата тестирования ')],
            [sg.Output(key = 6,size=(88, 20))],
            [sg.Button('Вывести результаты тестирования'),sg.Button('Сохранить
результаты тестирования'),sg.Button('Выход')]
            ]
window = sg.Window('Проверка работоспособности средств безопасности ПК',
layout)
```

В представленном выше программном коде с помощью соответствующих элементов осуществляется создание оконного интерфейса: выбирается цветовая тема приложения, создание кнопок и полей для вывода данных. Так же создается переменная строкового типа result, куда в дальнейшем будет осуществляться запись результатов.

На рисунке 1.1 представлен вид оконного интерфейса.

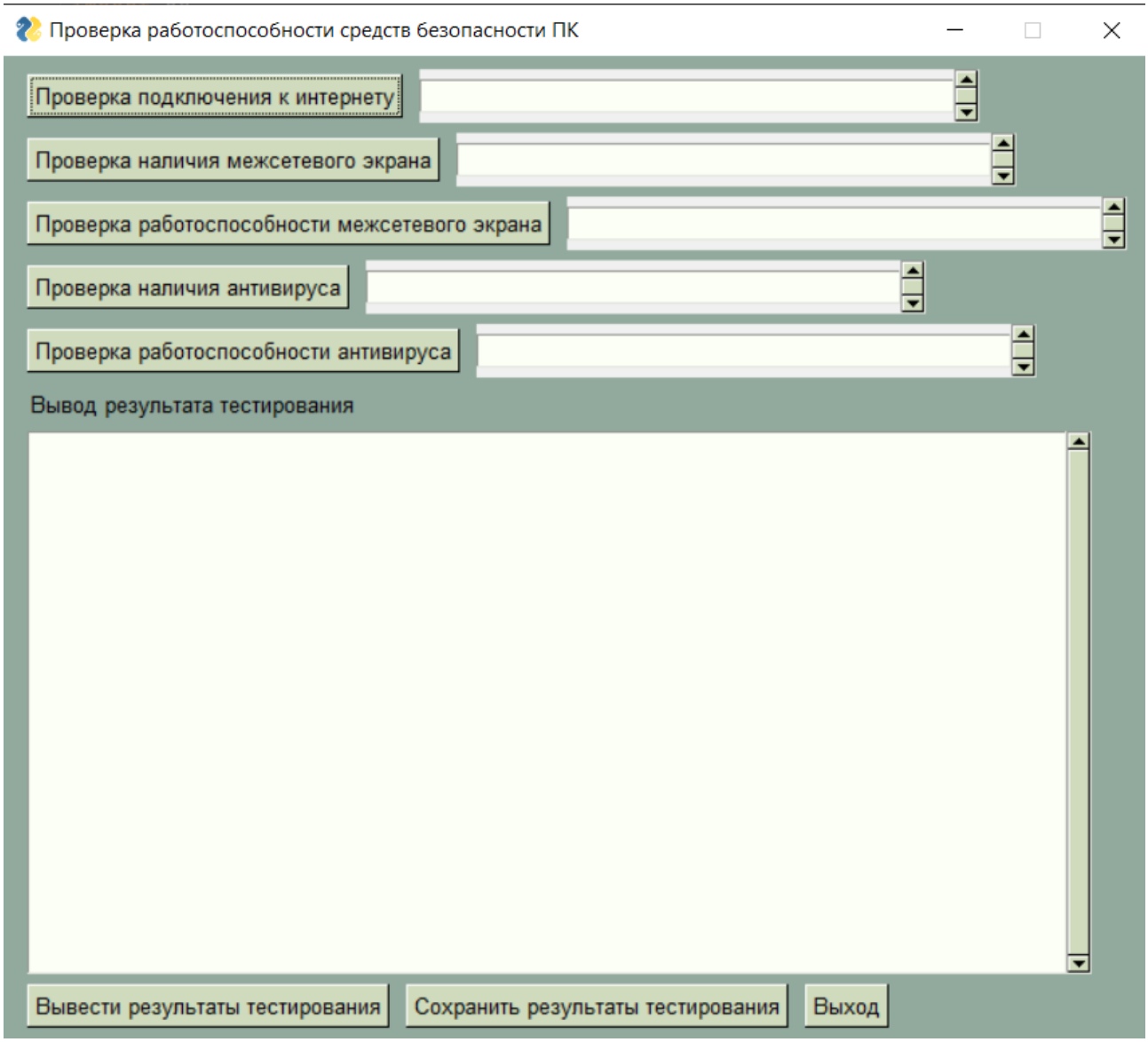

Рисунок 1.1. Интерфейс программы

1.1.Модуль проверки наличия соединения с интернетом.

Для проверки интернет соединения осуществляется обращение на web-сайт. При подключение к соответствующей странице происходит ошибка – у данного персонального компьютера отсутствует подключение к интернету и выдается соответствующее сообщение. Если подключение осуществляется, то у данного компьютера есть подключение к интернету, и выдается сообщение «Есть подключение». На рисунке 1.2 изображена блок-схема данного алгоритма.

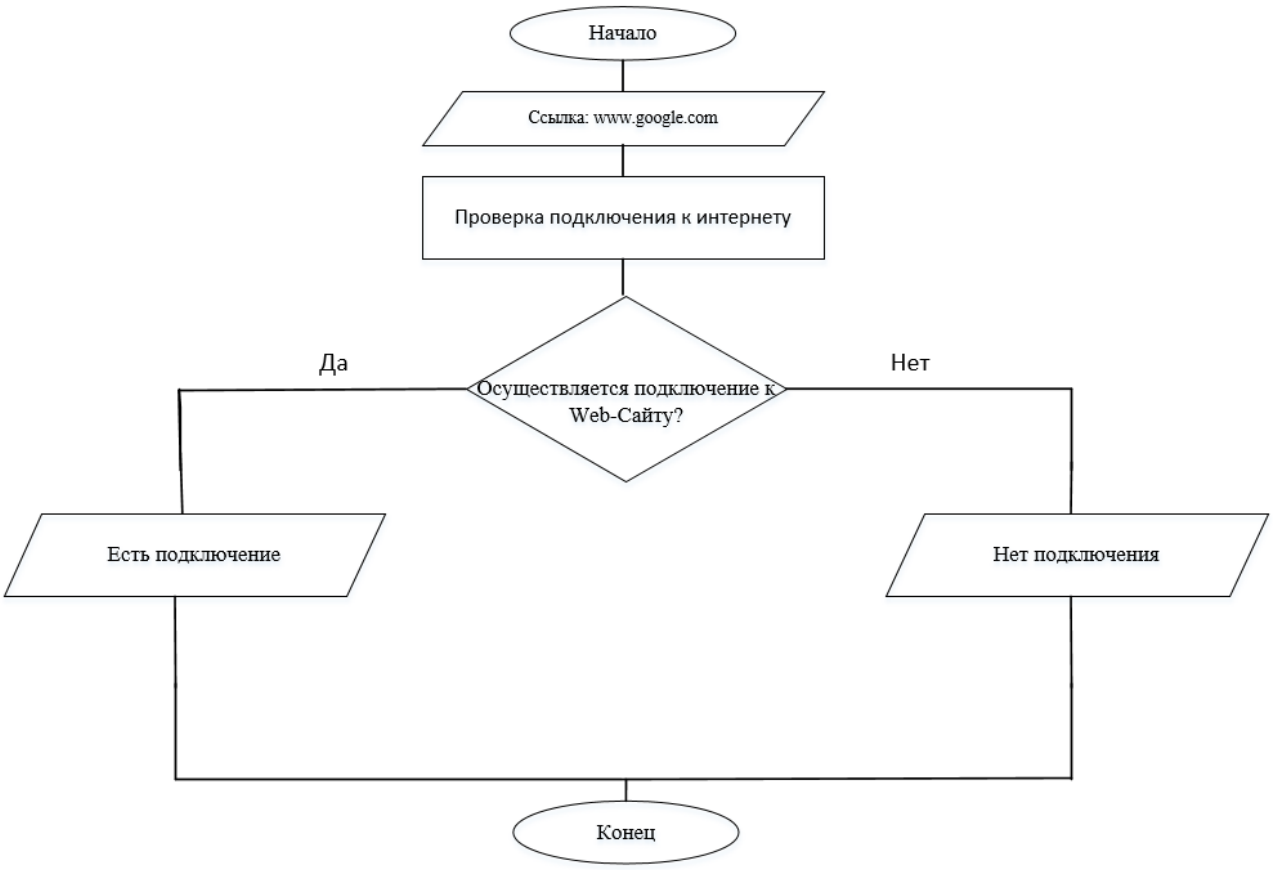

Рисунок 1.2. Блок-схема алгоритма проверки подключения к интернету

На листинге 1.2 программная реализация данного модуля.

Листинг 1.2. Проверка наличия соединения с интернетом.

```
if event == 'Проверка подключения к интернету':
        try:
            socket.gethostbyaddr('www.google.com')
        except socket.gaierror:
            window[1].update('Нет подключения')
            result = '1.Данный компьютер не подключен к интернету\n'
        else:
            window[1].update('Есть подключение')
            result = '1.Данный компьютер подключен к интернету\n
```

При нажатие на соответствующую кнопку осуществляется получение IP адреса с помощью модуля socket.gethostbyaddr(). Если не удалось получить IP адрес, то это является ошибкой и происходит соответствующая запись в Output элемент – «Нет подключения». Если IP адрес получен, то выполняется оператор else и соответствующая запись в Output элемент – «Есть подключение». Также соответствующие записи занесены в переменную result.

На рисунке 1.3 продемонстрирован скриншот с результатом.

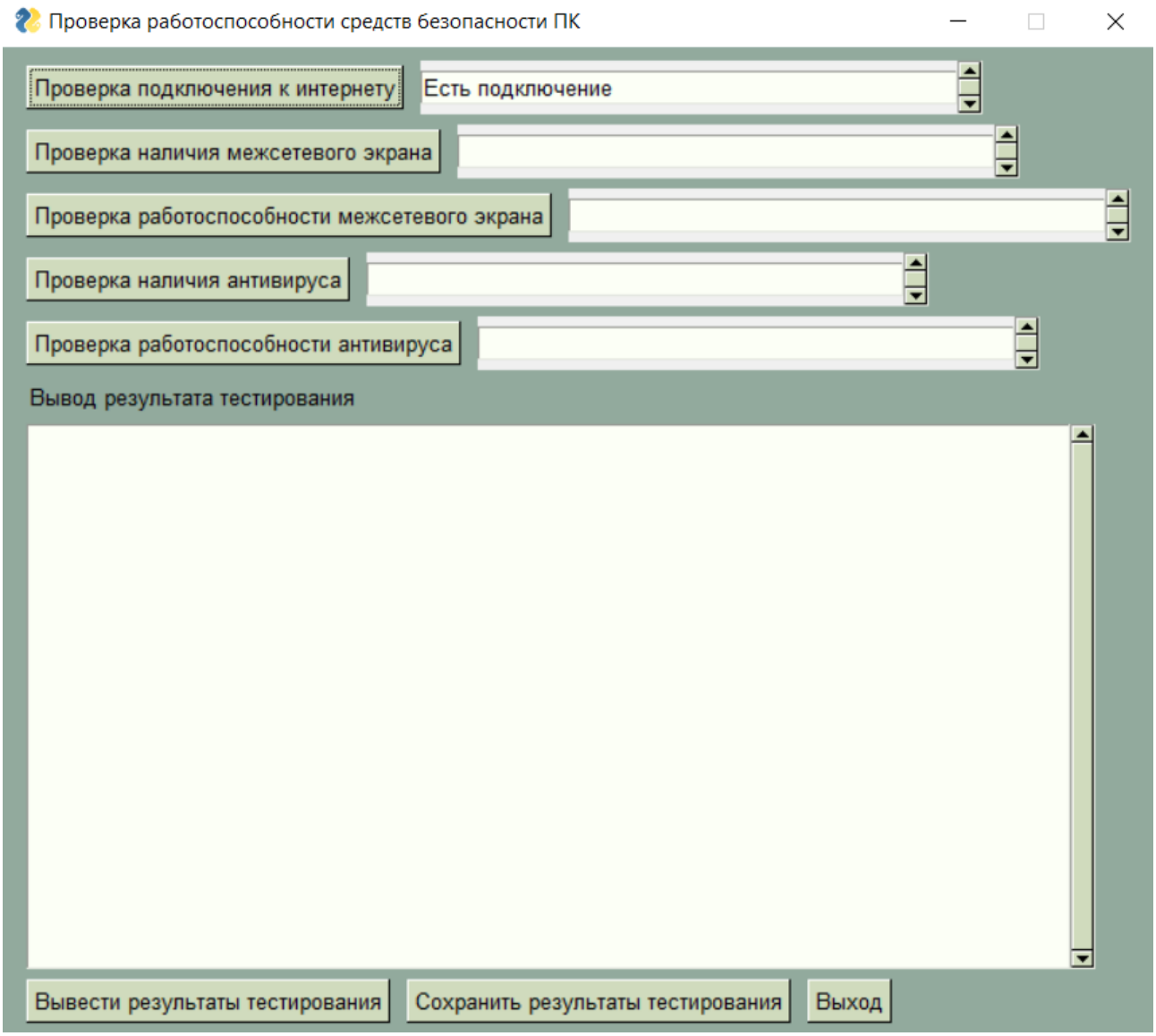

Рисунок 1.3. Проверка подключения к интернету

1.2. Модуль проверки наличия установленного межсетевого экрана.

В данном модуле проверка установленного межсетевого экрана осуществляется с помощью проверки наличия исполняемого файла фаервола. При наличие требующего файла – межсетевой экран установлен, иначе делаем вывод – межсетевой экран не установлен, также происходит вывод соответствующих сообщений. На рисунке 1.4 представлена блок-схема данного алгоритма.

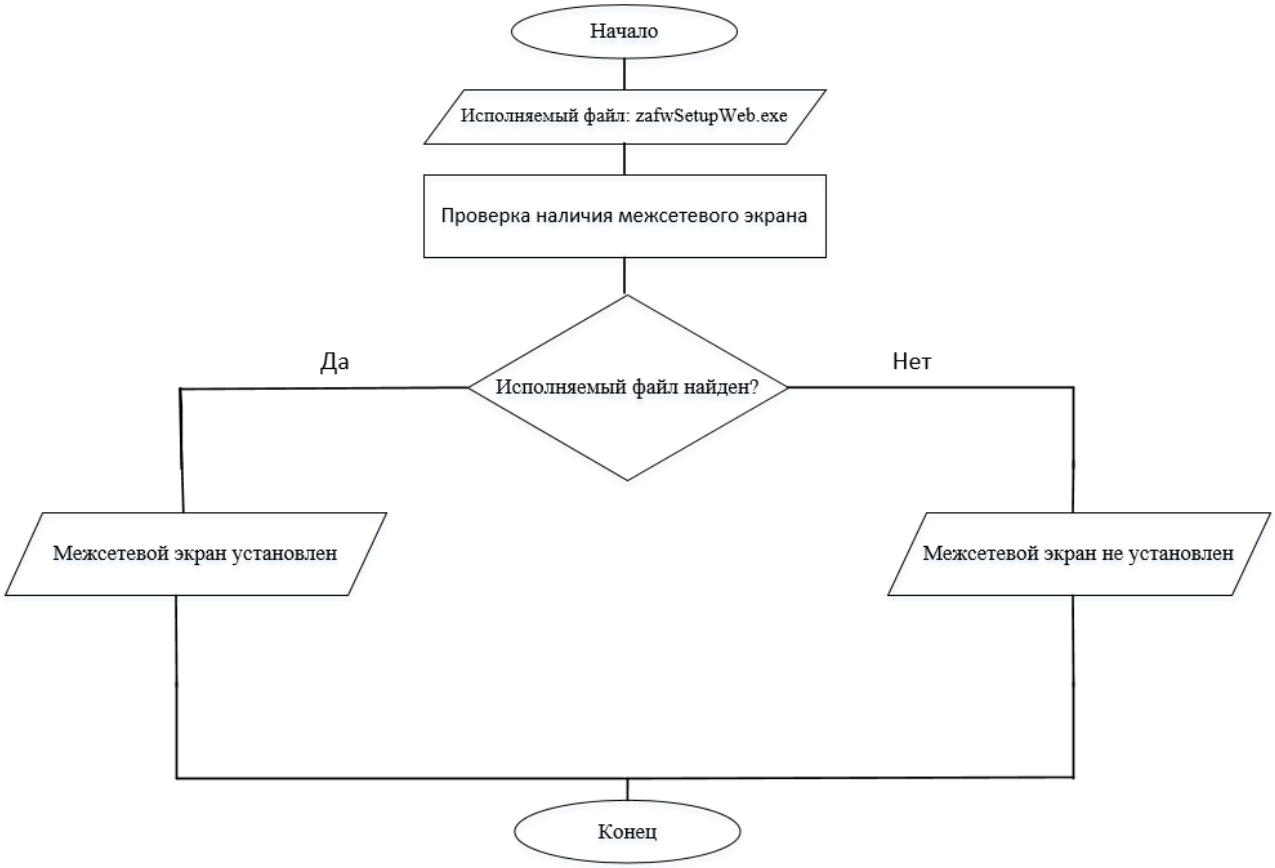

Рисунок 1.4. Блок-схема алгоритма проверки наличия межсетевого экрана

Ниже продемонстрирован листинг 1.3 с программной реализацией данного модуля.

Листинг 1.3. Проверка наличия установленного межсетевого экрана.

```
if event == 'Проверка наличия межсетевого экрана':
    test_path = r"C:\Users\Darina\zafwSetupWeb.exe"

    if os.path.exists(test_path):
        window[2].update('Межсетевой экран установлен')
        result = result + '2.На данном компьютере межсетевой экран
установлен\n'
    else:
        window[2].update('Межсетевой экран не установлен')
        result = result + '2.На данном компьютере межсетевой экран не
установлен\n'
```

При нажатие на соответствующую кнопку осуществляется выполнение конструкции if else. В условие прописывается адрес с местонахождением межсетевого экрана, если данный файл найден происходит запись в Output

элемент – «Межсетевой экран установлен», иначе – «Межсетевой экран не установлен». Также соответствующие записи занесены в переменную result.

На рисунке 1.5 продемонстрирован скриншот с результатом.

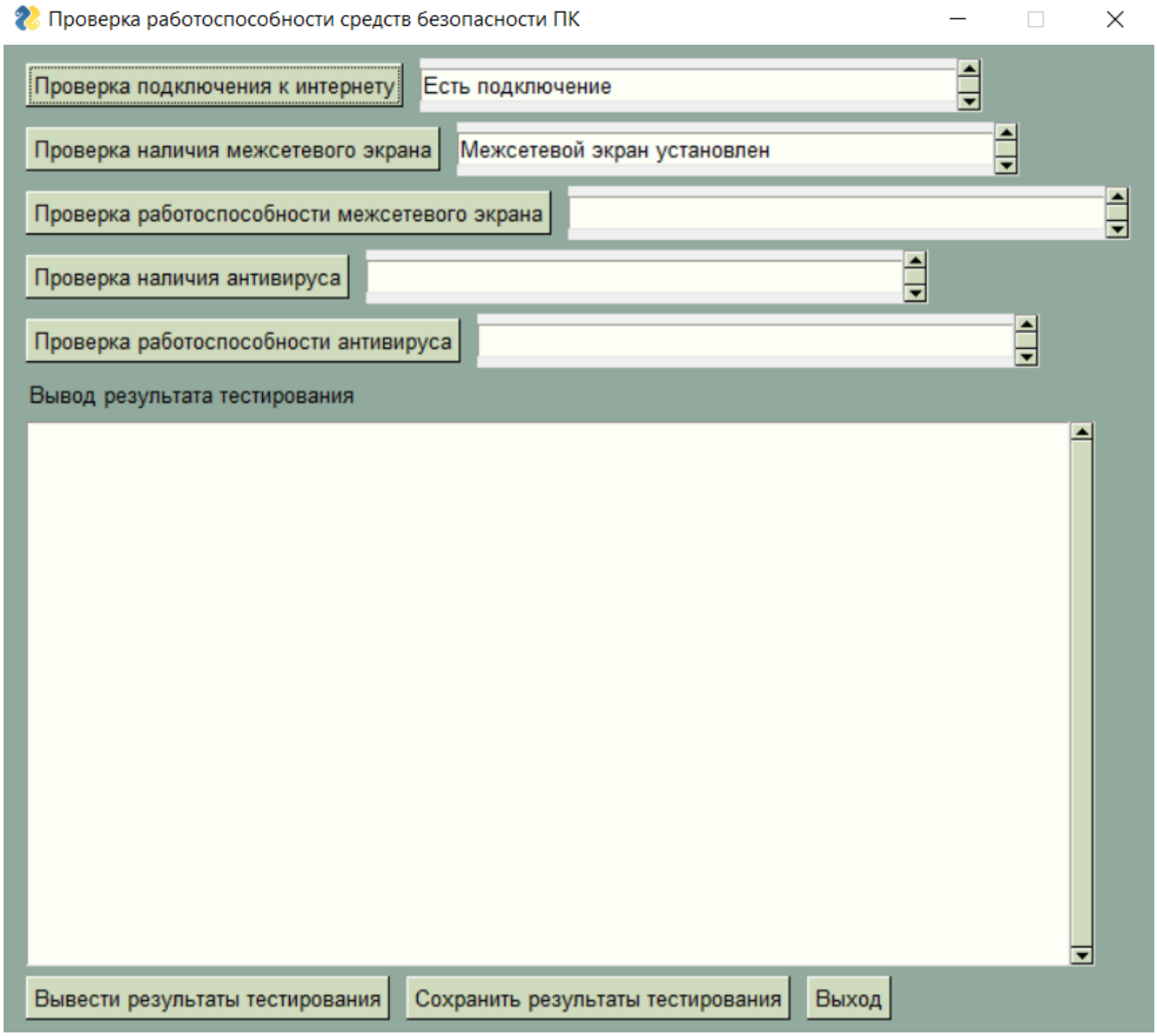

Рисунок 1.5. Проверка наличия межсетевого экрана

1.3. Модуль проверки работоспособности межсетевого экрана.

В данном модуле для проверки работоспособности межсетевого экрана определяется результат ответа. Код состояния равный 200 информирует об успешном статусе выполнения запроса. Межсетевой экран (МЭ) работает не верно, если код ответа отличается. Если код ответа совпадает – МЭ работает верно. На рисунке 1.6 представлена блок-схема данного алгоритма.

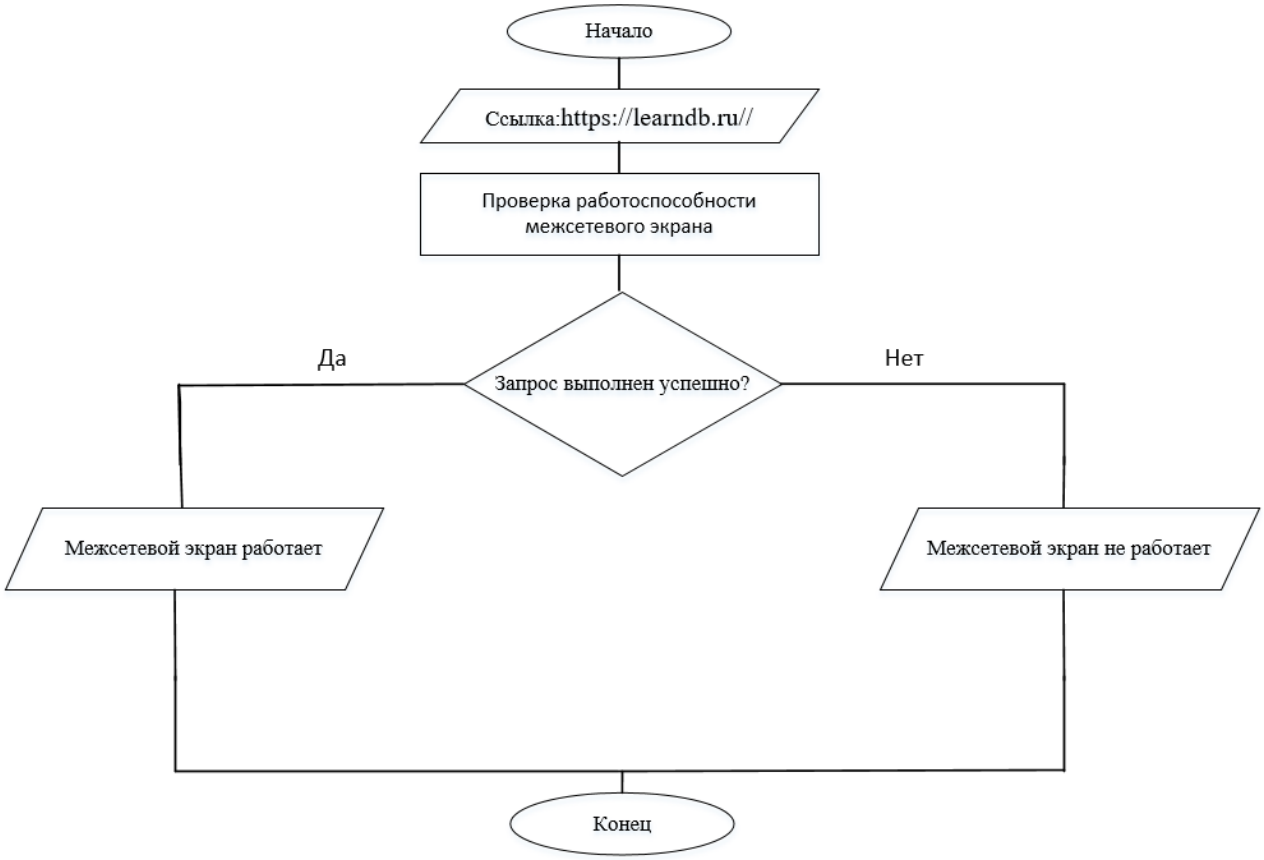

Рисунок 1.6. Блок-схема алгоритма проверки работоспособности межсетевого экрана

На листинге 1.4 представлена программная реализация данного модуля.

Листинг 1.4. Проверка работоспособности межсетевого экрана.

```
if event == 'Проверка работоспособности межсетевого экрана':
    req = Request("https://learndb.ru//")
    response = urlopen(req).status
    if response == 200:
        window[3].update('Межсетевой экран работает')
        result = result + '3.На данном компьютере межсетевой экран
работает верно\n'
    else:
        window[3].update('Межсетевой экран не работает')
        result = result + '3.На данном компьютере межсетевой экран не
работает\n'
```

При нажатие на соответствующую кнопку осуществляется переход на указанный адрес. Если код ответа равен 200 при переходе на данный ресурс, то производится соответствующая запись в Output элемент – «Межсетевой экран

работает», иначе – «Межсетевой экран не работает». Также соответствующие записи занесены в переменную result.

На рисунке 1.7 продемонстрирован скриншот с результатом.

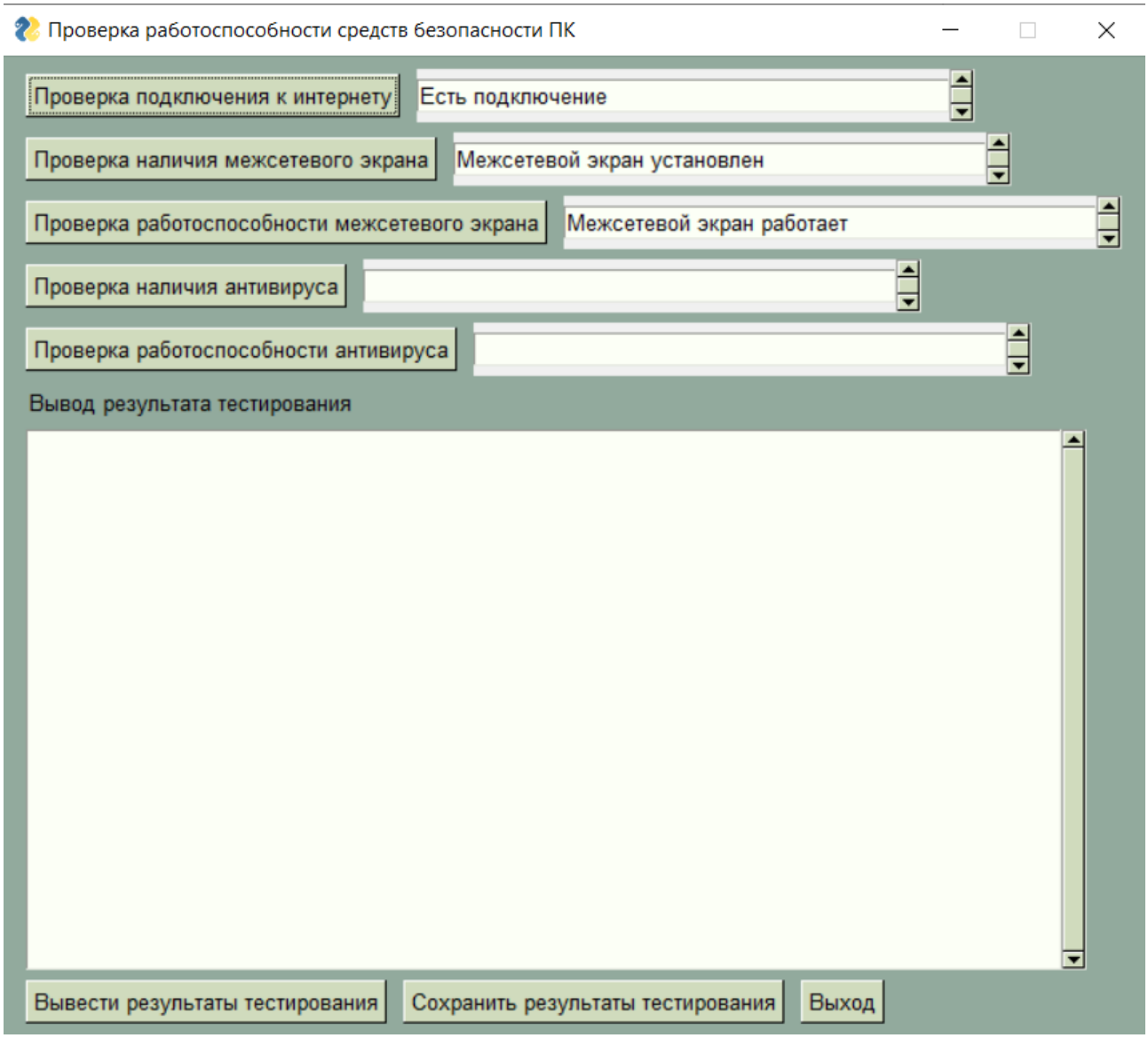

Рисунок 1.7. Проверка работоспособности межсетевого экрана

1.4. Модуль проверки наличия установленного антивируса.

В данном модуле для проверки наличия установленного антивируса осуществляется поиск определенной программы по реестру. С помощью ключа проверяется наличие на компьютере KasperskyLab, далее делаются соответствующие выводы. Программа найдена – «Антивирус установлен», иначе «Антивирус не установлен». В реестре найден антивирус – антивирус

установлен, иначе антивирус не установлен. На рисунке 1.8 представлена блок-схема данного алгоритма.

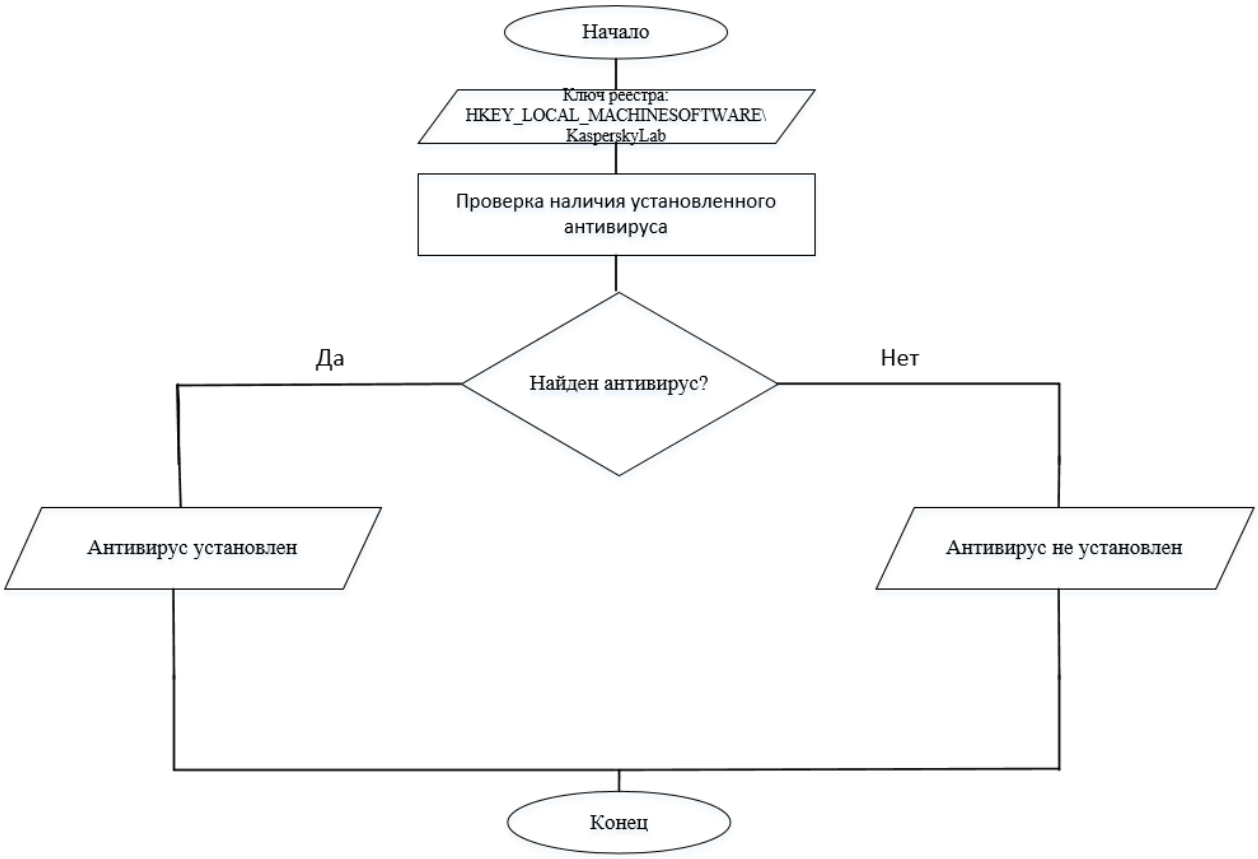

Рисунок 1.8. Блок-схема алгоритма проверки наличия установленного антивируса

На листинге 1.5 представлена программная реализация модуля.

Листинг 1.5. Проверка наличия установленного антивируса.

```
if event == 'Проверка наличия антивируса':
    try:
        aReg = ConnectRegistry(None, HKEY_LOCAL_MACHINE)
        aKey = OpenKey(aReg, r"SOFTWARE\KasperskyLab")
    except IOError as e:
        window[4].update("Антивирус не установлен")
        result = result + '4.На данном компьютере антивирус не установлен\n'
    else:
            window[4].update("Антивирус установлен")
            result = result + '4.На данном компьютере антивирус установлен\n'
```

При нажатие на соответствующую кнопку отправляется запрос в реестр при помощи модулей ConnectRegistry и OpenKey. Если программа не найдена

производится запись в Output элемент – «Антивирус не установлен», иначе – «Антивирус установлен». Также соответствующие записи занесены в переменную result.

На рисунке 1.9 представлен результат работы модуля.

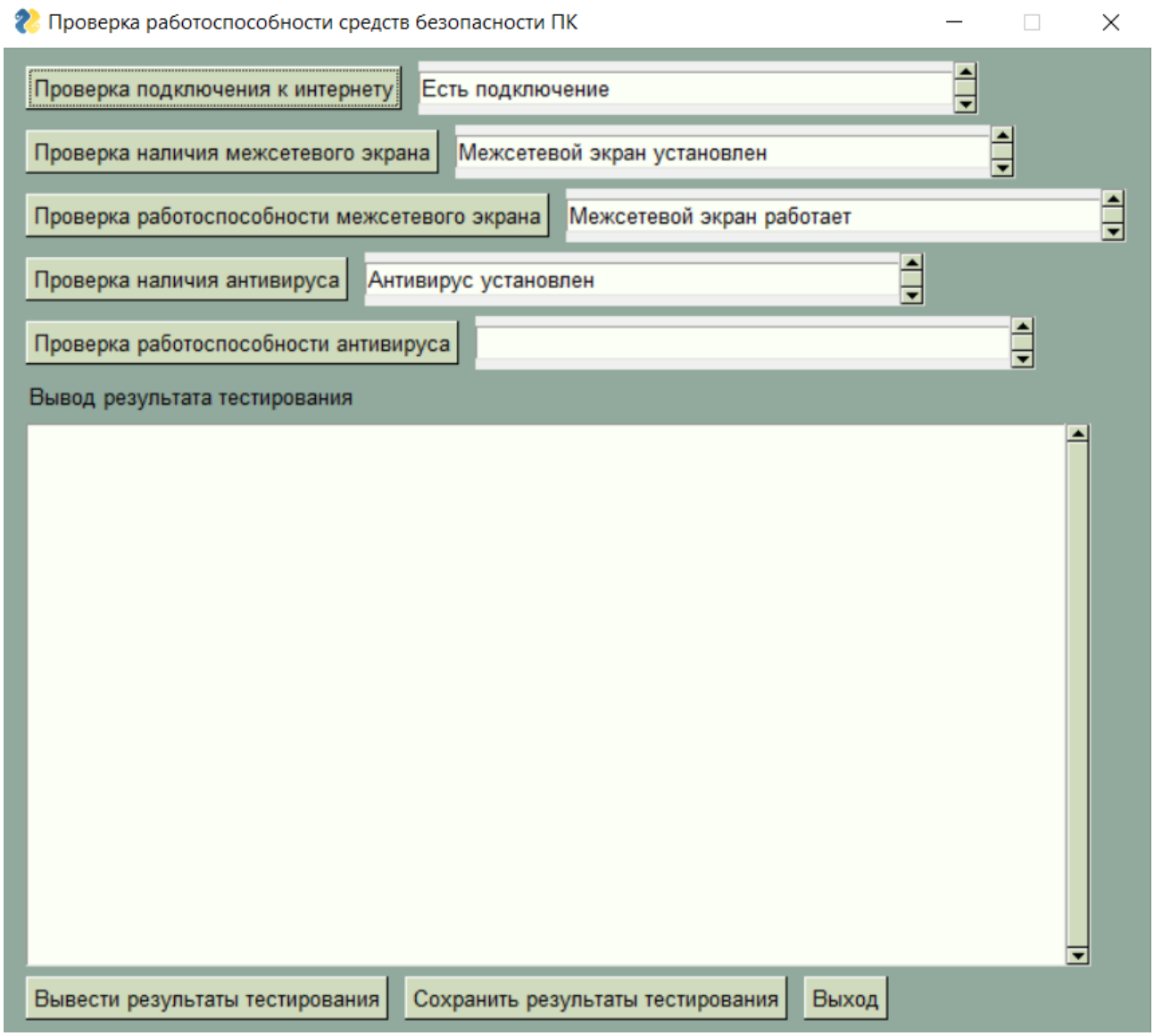

Рисунок 1.9. Проверка наличия антивируса

1.5.Модуль проверки работоспособности антивирусного ПО.

Для проверки работоспособности антивирусного ПО создается текстовый документ с последовательностью символов при которой антивирус не разрешает открыть файл. Если данный документ удается открыть – антивирус работает не верно, в обратном случае – антивирус работает верно. Выдается

соответствующее сообщение. На рисунке 1.10 представлена блок-схема данного модуля.

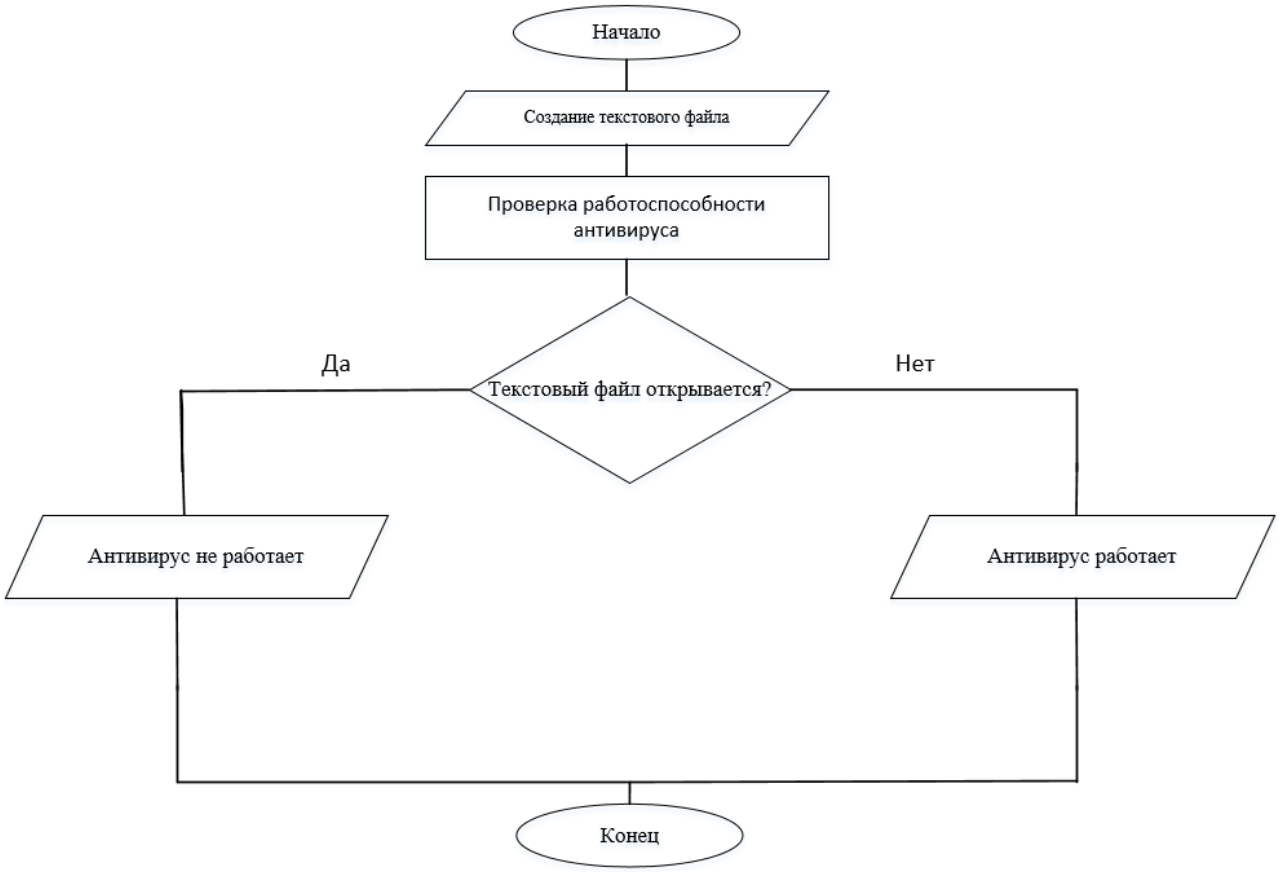

Рисунок 1.10. Блок-схема алгоритма проверки работоспособности антивирусного ПО

На листинге 1.6 представлен код программы реализующий данный модуль.

Листинг 1.6. Проверка работоспособности антивирусного ПО.

```
if event == 'Проверка работоспособности антивируса':
    try:
        my_file = open("antiv.txt", "w+")
        my_file.write("X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-
TEST-FILE!$H+H*")
        my_file.readline('antiv.txt')
    except IOError as e:
        window[5].update("Антивирус работает")
        result = result + '5.На данном компьютере антивирус работает верно\n'
    else:
        window[5].update("Антивирус не работает")
        result = result + '5.На данном компьютере антивирус работает
неисправно\n'
```

При нажатие на соответствующую кнопку создается текстовой файл antiv.txt с помощью команды open(), далее осуществляется запись последовательности с помощью команды write(), следующий шаг открытие файла – команда readline(). Если файл не удается открыть производится запись в Output элемент – «Антивирус работает», иначе – «Антивирус не работает». Также соответствующие записи занесены в переменную result.

На рисунке 1.11 представлен результат работы модуля.

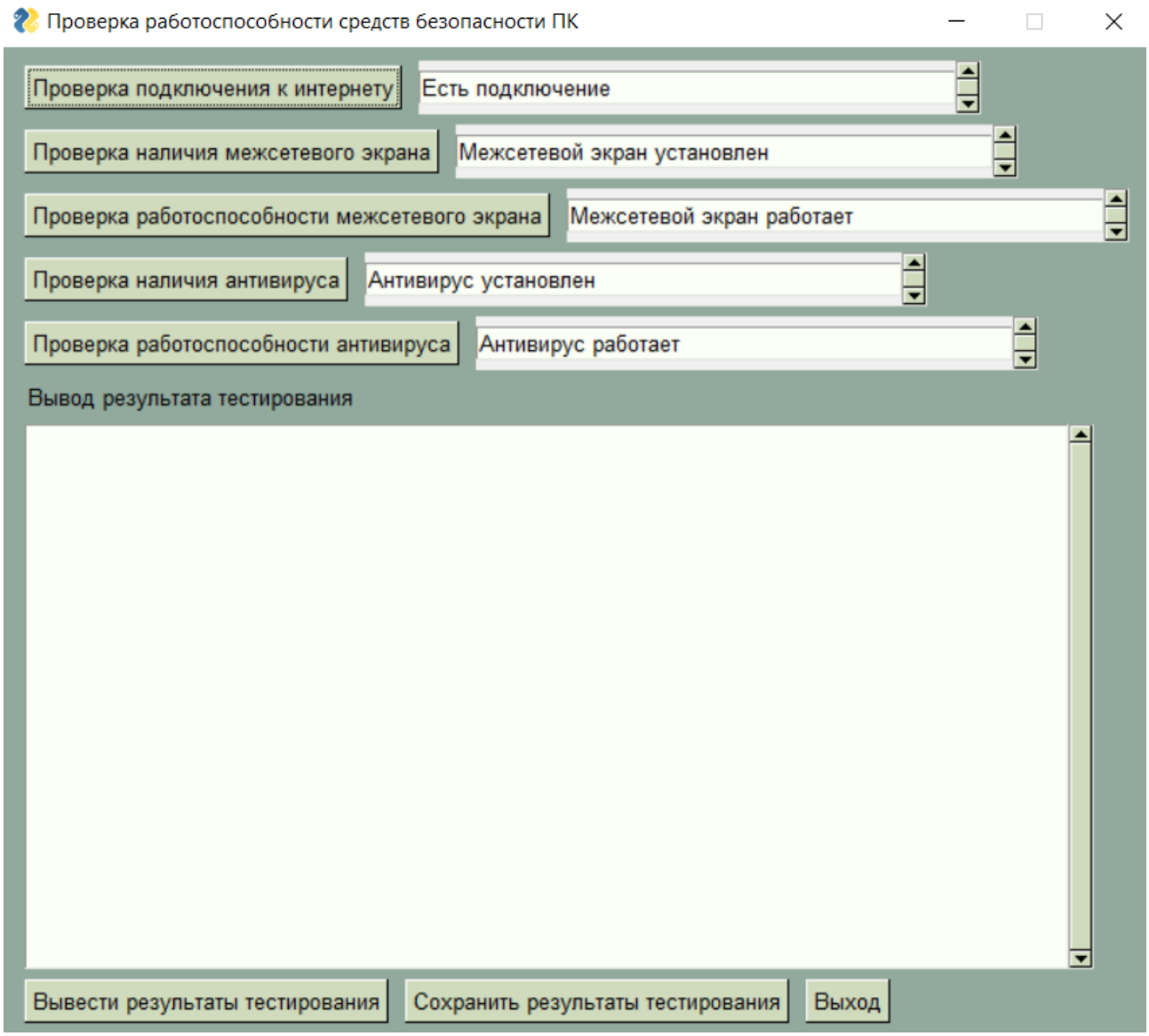

Рисунок 1.11. Проверка работоспособности антивирусного ПО

1.6.Модуль вывода результатов работы приложения.

Подытоживая результаты вышеописанных модулей, выводится общий результат работы программ, который содержит в себе информацию о состояние средств защиты тестируемого ПК. Блок-схема алгоритма представлена на рисунке 1.12.

Рисунок 1.12. Блок-схема алгоритма вывода результатов приложения

Код данного модуля представлен на листинге 1.7.

Листинг 1.7 Вывод результатов работы приложения.

```
if event == 'Вывести результаты тестирования':
    window[6].update(result
```

В предыдущих модулях осуществлялась соответствующие записи о работе средств защиты в переменную result, с сохранением предыдущего результата. При нажатие на соответствующую кнопку данные результаты выводится в Output элемент.

Скриншот с итогами тестирования представлен на рисунке 1.13.

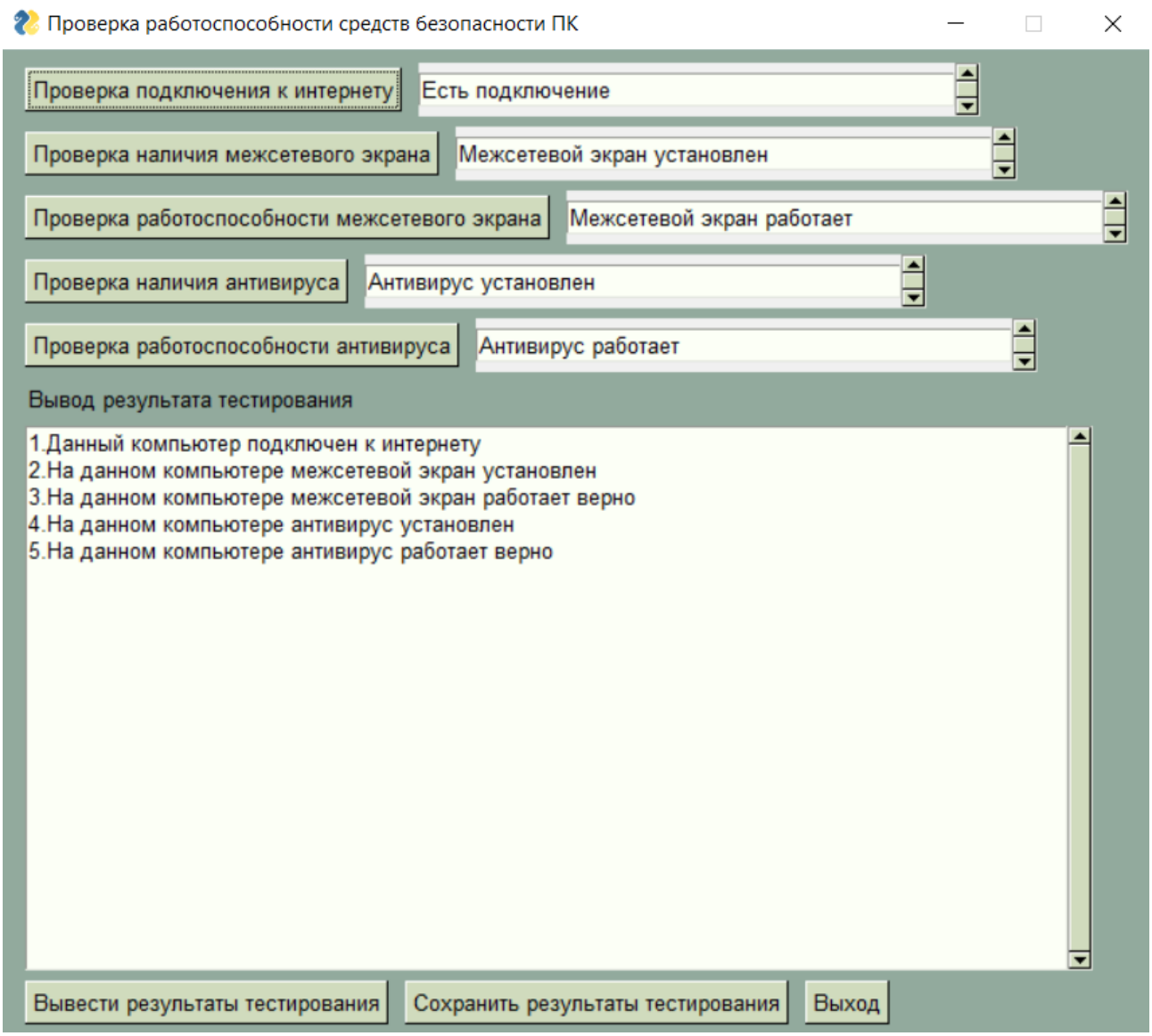

Рисунок 1.13. Вывод результатов работы приложения

1.7.Модуль сохранения результатов проверки.

Для сохранения результатов проверки создается текстовый файл, в который осуществляется запись тестирования. Блок-схема алгоритма представлена на рисунке 1.14.

Рисунок 1.14. Блок-схема алгоритма сохранения результатов проверки

Код программы представлен на листинге 1.8.

Листинг 1.8. Сохранение результатов тестирования.

```
if event == 'Сохранить результаты тестирования':
    my_file = open("result.txt", "w+")
    my_file.write(result)
```

При нажатие на соответствующую кнопку создается текстовый документ result.txt с помощью функции open(). В данный файл записывается переменная result, в которой находится выводы по тестированию средств защиты.

Текстовый документ с результатом проверки продемонстрирован на рисунке 1.15.

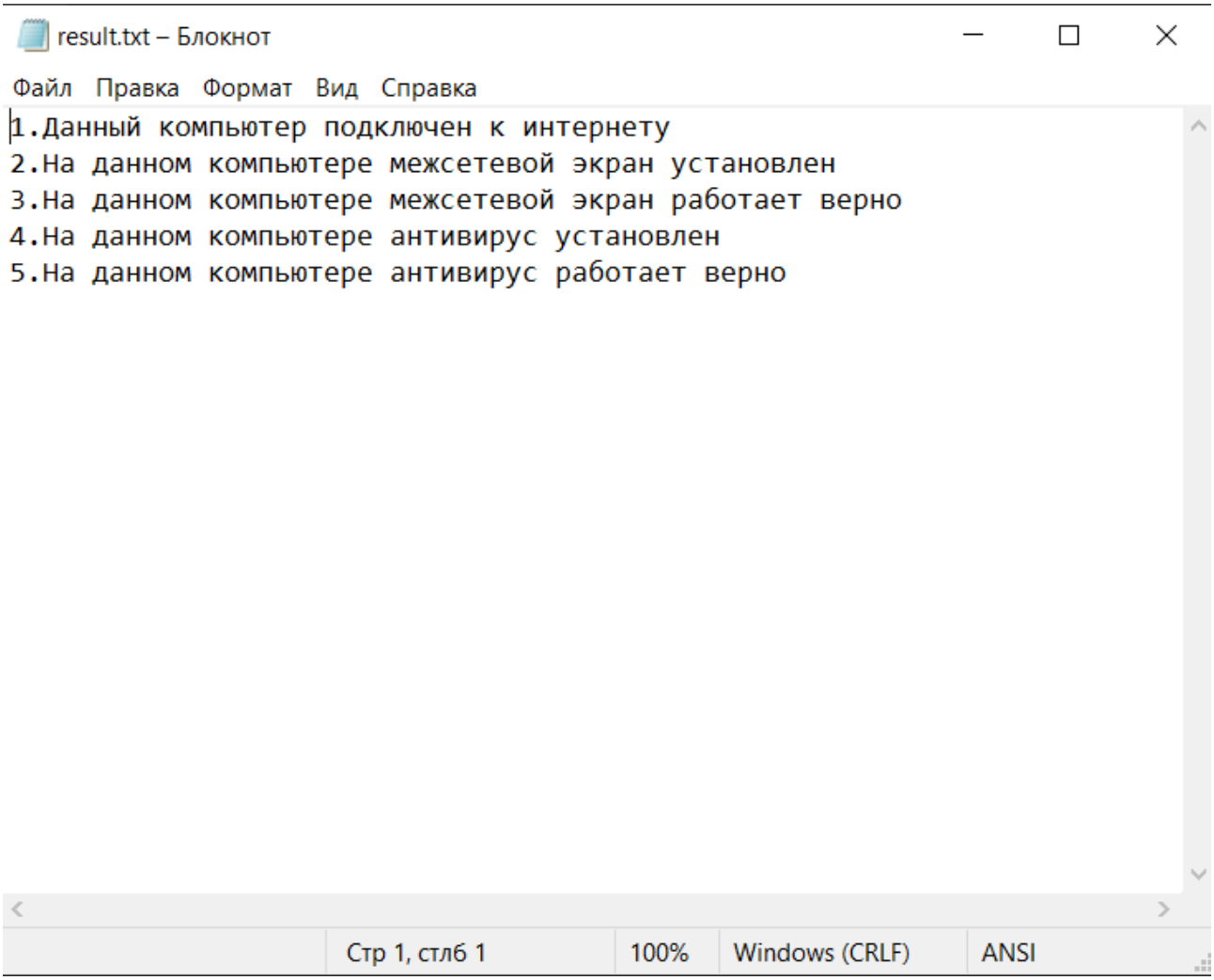

Рисунок 1.15. Сохранение результатов проверки

Заключение

В данной работе были изучены теоритические основы построения систем защиты компьютерных систем от вредоносных программ и систем применяемых для защиты межсетевых взаимодействий. Освоены и реализованы программные способы проверки подключения к сети Интернет, работоспособности антивируса и межсетевого экрана. Разработаны алгоритмы работы модулей ПО, а также интерфейс ПО. По проделанной работе можно сделать следующие выводы.

Для повышения безопасности средства защиты должны работать в комплексе, дополняя друг друга. Межсетевой экран отвечает за входящий/исходящий трафик, но не осуществляет защиту от вирусов с внешних носителей – за это отвечает антивирус. Если использовать представленные средства защиты, то можно обезопасить себя от различных угроз: удаление, копирование, утечка персональных данных, заражение ПК вирусами и т.д.

Список используемых источников

1. Марк Лутц Python. Карманный справочник . - 5-е изд. - Вильямс, 2015
2. Марк Лутц Изучаем Python . - 5-е изд. - Вильямс, 2013. - 900 с.
3. Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. - 1-е изд. - ДМК Пресс, 2012. - 700 с.
4. Климентьев К.Е. Компьютерные вирусы и антивирусы. Взгляд программиста, ДМК Пресс, 656 стр., 2015г.

Приложение А

Листинг приложения А

```
import socket
import PySimpleGUI as sg
import os
import requests
from winreg import *
from urllib.request import Request, urlopen
from urllib.error import URLError, HTTPError

result = ''
sg.theme('LightGreen5')
layout = [  [sg.Button('Проверка подключения к интернету'), sg.Output(key =
1,)],
            [sg.Button('Проверка наличия межсетевого экрана'), sg.Output(key =
2)],
            [sg.Button('Проверка работоспособности межсетевого экрана'),
sg.Output(key = 3)],
            [sg.Button('Проверка наличия антивируса'), sg.Output(key = 4)],
            [sg.Button('Проверка работоспособности антивируса'), sg.Output(key
= 5)],
            [sg.Text('Вывод результата тестирования ')],
            [sg.Output(key = 6,size=(88, 20))],
            [sg.Button('Вывести результаты тестирования'),sg.Button('Сохранить
результаты тестирования'),sg.Button('Выход')]
            ]
window = sg.Window('Проверка информационной безопасности', layout)




while True:
    event, values = window.read()
    if event in (None, 'Выход'):
        break
    if event == 'Проверка подключения к интернету':
            try:
                socket.gethostbyaddr('www.google.com')
            except socket.gaierror:
                window[1].update('Нет подключения')
                result = '1.Данный компьютер не подключен к интернету\n'
            else:
                window[1].update('Есть подключение')
                result = '1.Данный компьютер подключен к интернету\n'
    if event == 'Проверка наличия межсетевого экрана':
        test_path = r"C:\Users\Darina\zafwSetupWeb.exe"

        if os.path.exists(test_path):
            window[2].update('Межсетевой экран установлен')
            result = result + '2.На данном компьютере межсетевой экран
установлен\n'
        else:
            window[2].update('Межсетевой экран не установлен')
            result = result + '2.На данном компьютере межсетевой экран не
установлен\n'
    if event == 'Проверка работоспособности межсетевого экрана':
     req = Request("https://learndb.ru//")
     response = urlopen(req).status
     if response == 200:
          window[3].update('Межсетевой экран работает')
          result = result + '3.На данном компьютере межсетевой экран
работает верно\n'
```

```
    else:
        window[3].update('Межсетевой экран не работает')
        result = result + '3.На данном компьютере межсетевой экран не
работает\n'

    if event == 'Проверка наличия антивируса':
        try:
            aReg = ConnectRegistry(None, HKEY_LOCAL_MACHINE)
            aKey = OpenKey(aReg, r"SOFTWARE\KasperskyLab")
        except IOError as e:
            window[4].update("Антивирус не установлен")
            result = result + '4.На данном компьютере антивирус не
установлен\n'
        else:
            window[4].update("Антивирус установлен")
            result = result + '4.На данном компьютере антивирус установлен\n'
    if event == 'Проверка работоспособности антивируса':
        try:
            my_file = open("antiv.txt", "w+")
            my_file.write("X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-
ANTIVIRUS-TEST-FILE!$H+H*")
            my_file.readline('antiv.txt')
        except IOError as e:
            window[5].update("Антивирус работает")
            result = result + '5.На данном компьютере антивирус работает
верно\n'
        else:
            window[5].update("Антивирус не работает")
            result = result + '5.На данном компьютере антивирус работает
неисправно\n'
    if event == 'Вывести результаты тестирования':
        window[6].update(result)
    if event == 'Сохранить результаты тестирования':
        my_file = open("result.txt", "w+")
        my_file.write(result)
```