g***Министерство образования Республики Беларусь***

***Учреждение образования***

***«Брестский государственный технический университет»***

***Кафедра ИИТ***

**Лабораторная работа №6**

**По дисциплине КМЗИ за IV семестр**

**Тема: «Стандарты безопасного удаления данных»**

**Выполнил:**

Студент группы ИИ-15 (1)

2-го курса

Волк И. А.

**Проверил:**

Хацкевич М. В.

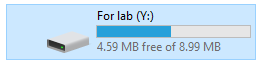
Брест 2019

Цель работы: изучить стандарты безопасного удаления данных.

Ход работы

В данной лабораторной работе произведу проверку работы американского стандарта DoD 5220.22-М.

1. Создадим раздел на жестком диске. Назовем его Y



1. Заполним его мусором. Файл так и назовем: trash

Скрипт:

import os

import ctypes

import numpy as np

def get\_free\_space():

dir\_name = os.path.dirname(\_\_file\_\_)

free\_bytes = ctypes.c\_ulonglong(0)

ctypes.windll.kernel32.GetDiskFreeSpaceExW(ctypes.c\_wchar\_p(dir\_name), None, None, ctypes.pointer(free\_bytes))

return int(free\_bytes.value)

free\_space = get\_free\_space()

print(free\_space)

tresh\_file = open('trash', 'w+')

for i in range(free\_space - 500 \* 1025):

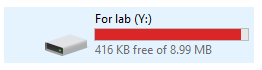
if i % 50 == 0 and i != 0:

print('\n', end='', file=tresh\_file)

else:

print('1' if np.random.randint(0, 2) % 2 == 0 else '0', end='', file=tresh\_file)

После запуска:



Содержимое файла:



Напишем программу для реализации алгоритма безопасного удаления:

#include <fstream>

#include <windows.h>

#include <random>

#include <iostream>

void fillRandomly(std::fstream& stream);

size\_t findSize(std::fstream& stream);

void inverse(std::fstream& stream);

int main()

{

std::fstream fsTrash {"Y:/trash", std::ios::out | std::ios::in| std::ios::binary };

if (!fsTrash)

{

std::cout << "Error while opening!" << std::endl;

}

std::cout << "Size of file: " << findSize(fsTrash);

fillRandomly(fsTrash);

inverse(fsTrash);

fillRandomly(fsTrash);

system("pause");

}

size\_t findSize(std::fstream& stream)

{

auto init\_pos = stream.tellg();

stream.seekg(0, std::ios::end);

size\_t end = stream.tellg();

stream.seekg(0, std::ios::beg);

size\_t begin = stream.tellg();

stream.seekg(init\_pos);

return end - begin;

}

void fillRandomly(std::fstream& stream)

{

size\_t size = findSize(stream);

stream.seekp(stream.end);

char random\_byte;

std::default\_random\_engine gen;

for(size\_t i = 0; i < size; i++)

{

random\_byte = gen();

stream.write(&(random\_byte), sizeof(char));

}

}

void inverse(std::fstream& stream)

{

size\_t size = findSize(stream);

char symbol;

stream.seekg(stream.beg);

stream.seekp(stream.beg);

for(size\_t i = 0; i < size; i++)

{

stream.read(&symbol, sizeof(char));

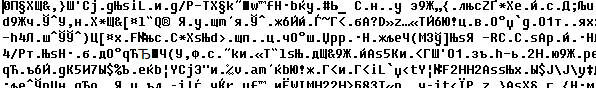
symbol = ~symbol;

stream.write(&symbol, sizeof(char));

}

}

Содержимое файла после выполнения:



Вывод: по ходу лабораторной работы научился производить безопасное удаление данных.