БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ
Кафедра информатики

Индивидуальная практическая работа № 2
по дисциплине «Системное программирование»

Вариант 6

Факультет: ИНО
Специальность: ИиТП
Студент: Дубейковский А.А.
Группа № 893551
Зачётная книжка № 75350046

Условие

Оценка свободного пространства на диске и анализ его фрагментации: общее количество свободных кластеров, количество непрерывных последовательностей свободных кластеров, наибольшая непрерывная последовательность, и так далее.

Код с пояснениями

```
#include "framework.h"
#include "sp_lab2.h"
#include "windows.h"
#include "stdio.h"
#include "commdlg.h"
#include "fileapi.h"
#include "winioctl.h"
#include <iostream>
#include <atlstr.h>


#define MAX_LOADSTRING 64

HINSTANCE hInst;
ATOM MyRegisterClass(HINSTANCE hInstance);
BOOL InitInstance(HINSTANCE, int);
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
WCHAR szTitle[MAX_LOADSTRING] = L"sp_lab2";
WCHAR szWindowClass[MAX_LOADSTRING] = L"sp_lab2WindowClass";
```

Структура, хранящая информацию о диске:

```
struct DiskInfo {
    LPCWSTR lpDiskName;

    ULARGE_INTEGER lpFreeBytesAvailable;
    ULARGE_INTEGER lpTotalNumberOfBytes;
    ULARGE_INTEGER lpTotalNumberOfFreeBytes;

    DWORD lpSectorsPerCluster;
    DWORD lpBytesPerSector;
    DWORD lpNumberOfFreeClusters;
    DWORD lpTotalNumberOfClusters;

    DWORD longestClustersChainLength;
    DWORD clustersChainsCount;
};
```

Функция получения и подсчёта информации о диске:

```
bool GetDiskInfo(DiskInfo* diskInfo) {
    // функция для объема диска
    GetDiskFreeSpaceEx(
        L"C:",
        &diskInfo->lpFreeBytesAvailable,
        &diskInfo->lpTotalNumberOfBytes,
        &diskInfo->lpTotalNumberOfFreeBytes
    );

    // подсчёт свободных кластеров
    GetDiskFreeSpace(
        L"C:",
        &diskInfo->lpSectorsPerCluster,
        &diskInfo->lpBytesPerSector,
        &diskInfo->lpNumberOfFreeClusters,
        &diskInfo->lpTotalNumberOfClusters
    );

// открываем диск как HANDLE для DeviceIoControl, чтобы найти остальные атрибуты
diskInfo->lpDiskName = L"\\\\.\\C:";
HANDLE diskForCycle = CreateFile(diskInfo->lpDiskName, 0, FILE_SHARE_READ | FILE_SHARE_WRITE, 0, OPEN_EXISTING, 0, 0);

if (diskForCycle == INVALID_HANDLE_VALUE) {
    MessageBox(NULL, L"Failed to open disk.", L"Error", MB_OK | MB_ICONERROR);
    return false;
}
```

Переменные, которые будут нужны в ходе использования DeviceIoControl и подсчёта нужных нам параметров

```
DWORD tmp;
DWORD lastError;
DWORD clustersChainsCount = 0;
DWORD longestClustersChainLength = 0;
auto startingVcn = new STARTING_VCN_INPUT_BUFFER();
auto pointersBuffer = new RETRIEVAL_POINTERS_BUFFER();
```

Вызов функции в цикле, начинаем всегда со startingVCN, которое будет менять в теле цикла, цикл закончится если DeviceIoControl не найдёт больше данных на диске

```
do {
    DeviceIoControl(diskForCycle,
        FSCTL_GET_RETRIEVAL_POINTERS,
        startingVcn,
        sizeof(*startingVcn),
        pointersBuffer,
        sizeof(*pointersBuffer),
        &tmp,
        NULL);
```

Алгоритм подсчёта, он даже частично описан в документации к DeviceIoControl

```
        lastError = GetLastError();

        if (lastError != ERROR_HANDLE_EOF) {
            ++clustersChainsCount;
            auto currentClustersChainLength = pointersBuffer->Extents->NextVcn.QuadPart -
                pointersBuffer->StartingVcn.QuadPart;
            if (currentClustersChainLength > longestClustersChainLength) {
                longestClustersChainLength = currentClustersChainLength;
            }
            startingVcn->StartingVcn.QuadPart = pointersBuffer->Extents->NextVcn.QuadPart;
        }
    } while (lastError == ERROR_MORE_DATA);

    diskInfo->clustersChainsCount = clustersChainsCount;
    diskInfo->longestClustersChainLength = longestClustersChainLength;

    CloseHandle(diskForCycle);
    return true;
}
```

```cpp
int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
    _In_opt_ HINSTANCE hPrevInstance,
    _In_ LPWSTR lpCmdLine,
    _In_ int nCmdShow)
{
    UNREFERENCED_PARAMETER(hPrevInstance);
    UNREFERENCED_PARAMETER(lpCmdLine);
    MyRegisterClass(hInstance);

    if (!InitInstance(hInstance, nCmdShow))
    {
        return FALSE;
    }
    MSG msg;

    while (GetMessage(&msg, nullptr, 0, 0))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return (int)msg.wParam;
}
ATOM MyRegisterClass(HINSTANCE hInstance)
{
    WNDCLASSEXW wcex;
    wcex.cbSize = sizeof(WNDCLASSEX);
    wcex.style = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc = WndProc;
    wcex.cbClsExtra = 0;
    wcex.cbWndExtra = 0;
    wcex.hInstance = hInstance;
    wcex.hIcon = LoadIcon(hInstance, IDI_APPLICATION);
    wcex.hCursor = LoadCursor(nullptr, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
    wcex.lpszMenuName = 0;
    wcex.lpszClassName = szWindowClass;
    wcex.hIconSm = LoadIcon(wcex.hInstance, IDI_APPLICATION);
    return RegisterClassExW(&wcex);
}
```

```
BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
{
    hInst = hInstance;
    HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS_OVERLAPPEDWINDOW ^
        WS_THICKFRAME,
        CW_USEDEFAULT, 0, 600, 600, nullptr, nullptr, hInstance, nullptr);
    if (!hWnd)
    {
        return FALSE;
    }
    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);
    return TRUE;
}
```

Работа с окном и кнопками:

переменные

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    static HWND hDiskDescriptionText;
    static HWND hFreeBytesCountText;
    static HWND hNumberOfFreeClustersText;
    static HWND hClustersChainsCountText;
    static HWND hLongestClustersChainLengthText;
    static HWND hStartButton;
    switch (message)
    {
```

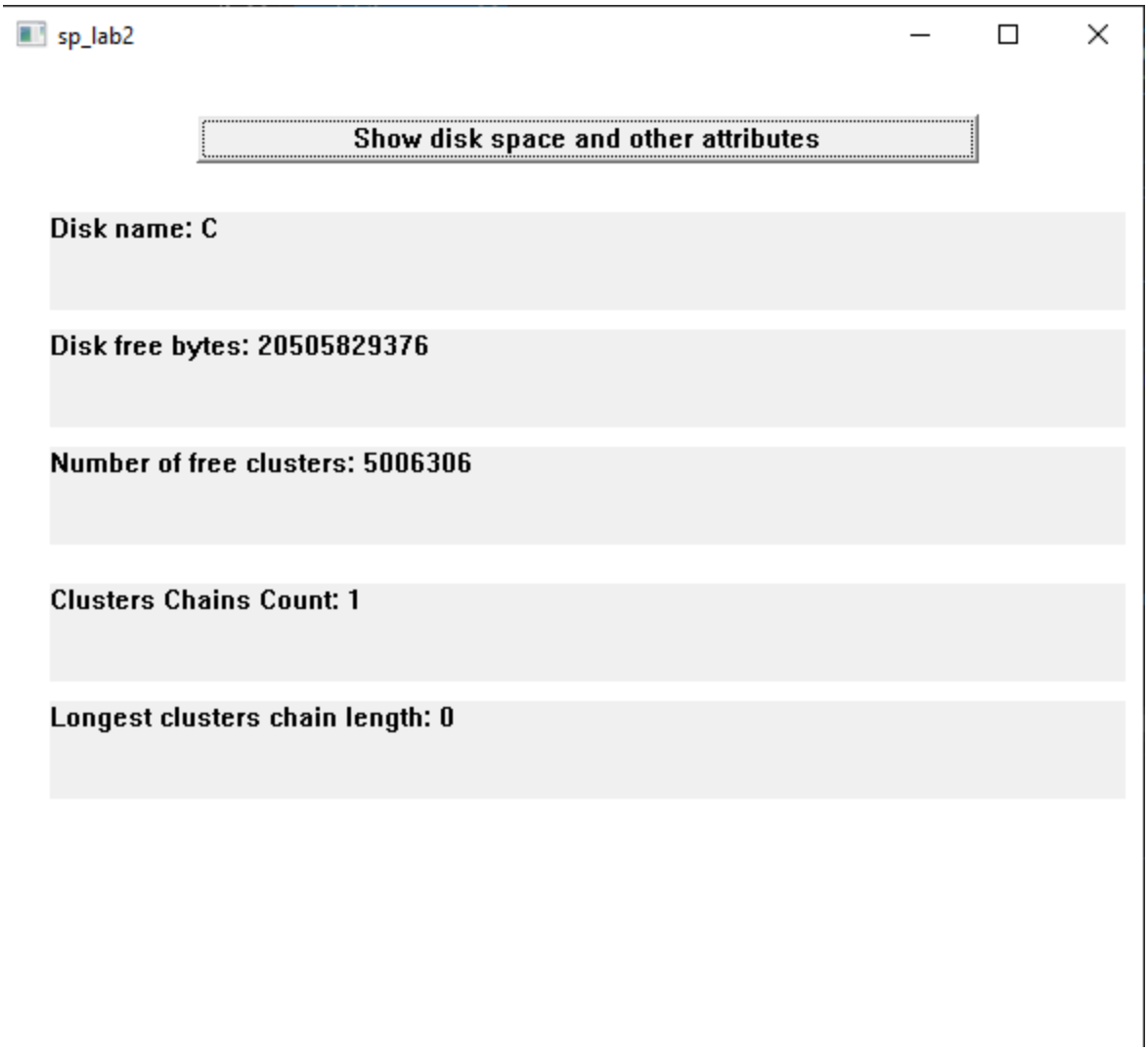
Создание частей окна

```
    {
    case WM_CREATE: {
        hStartButton = CreateWindow(L"button", L"Show disk space and other attributes", WS_CHILD | WS_VISIBLE, 100, 25,
            400, 25, hWnd, 0, hInst, NULL);

        hDiskDescriptionText = CreateWindow(L"static", NULL, WS_CHILD | WS_VISIBLE, 25, 75, 550, 50, hWnd, 0, hInst, NULL);
        hFreeBytesCountText = CreateWindow(L"static", NULL, WS_CHILD | WS_VISIBLE, 25, 135, 550, 50, hWnd, 0, hInst, NULL);
        hNumberOfFreeClustersText = CreateWindow(L"static", NULL, WS_CHILD | WS_VISIBLE, 25, 195, 550, 50, hWnd, 0, hInst, NULL
        hClustersChainsCountText = CreateWindow(L"static", NULL, WS_CHILD | WS_VISIBLE, 25, 265, 550, 50, hWnd, 0, hInst, NULL)
        hLongestClustersChainLengthText = CreateWindow(L"static", NULL, WS_CHILD |  WS_VISIBLE, 25, 325, 550, 50, hWnd,
            0, hInst, NULL);
break;
```

Вывод полученной информации при нажатии на кнопку

```
case WM_COMMAND: {
    if ((HWND)lParam == hStartButton) {
        DiskInfo disk;
        GetDiskInfo(&disk);

        TCHAR attribute[64];

        swprintf_s(attribute, L"Disk name: C");
        SetWindowText(hDiskDescriptionText, attribute);

        swprintf_s(attribute, L"Disk free bytes: %lld", disk.lpFreeBytesAvailable);
        SetWindowText(hFreeBytesCountText, attribute);

        swprintf_s(attribute, L"Number of free clusters: %d", disk.lpNumberOfFreeClusters);
        SetWindowText(hNumberOfFreeClustersText, attribute);

        swprintf_s(attribute, L"Clusters Chains Count: %d", disk.clustersChainsCount);
        SetWindowText(hClustersChainsCountText, attribute);

        swprintf_s(attribute, L"Longest clusters chain length: %d", disk.longestClustersChainLength);
        SetWindowText(hLongestClustersChainLengthText, attribute);
    }
}break;
case WM_DESTROY:
    PostQuitMessage(0);
    break;
default:
    default:
        return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}
```

Результат:



Собственно целый день разбирался что не так с моим циклом с DeviceIoControl, почему не идёт полный подсчёт, поначалу пробовал различные способы создания Handle через CreateFile, пробовал открывать и диск как файловую систему через \\\\.\\C:\\, чтобы работать последовательно через каждый каталог, пробовал подключаться и как к физическим дискам, через PhysicalDrive0 и ещё пару способов, но всё равно DeviceIoControl думает что внутри ничего нет, там нет ошибки при исполнении никакой, он спокойно заканчивает работу и выводит ноль. Честно, в течении дня гуглил это и искал способы, не знаю, с этой функцией возникли трудности, хотя GetDiskFreeSpace и подобвные работали прекрасно. Я подумал, может ли это быть связано с тем, что у меня один диск для двух операционных систем, при том виндоус занимает небольшую часть только. Буду рад если подскажете, как это можно было бы сделать, но часть атрибутов успешно нашёл и вывел.

Вывод

Познакомился с многими функцими WinApi по работе с дисками и файлами, узнал больше о том, как вся информация распаложена внутри, получил знания о вычислениях различных аттрибутов диска собственноручно.