Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Операционные среды и системное программирование

ОТЧЁТ

к лабораторной работе №6

на тему

**НЕКОТОРЫЕ СЛУЖЕБНЫЕ И ТЕХНОЛОГИЧЕСКИЕ ЗАДАЧИ**

Выполнил студент гр. 253505 Ющук И.А.

Проверил ассистент кафедры информатики Гриценко Н.Ю.

Минск 2024

**СОДЕРЖАНИЕ**

[1 Формулировка задачи 3](#_Toc182241314)

[2 Описание функций программы 4](#_Toc182241315)

[2.1 Вывод информации о заданном значении реестра 4](#_Toc182241316)

[2.2 Рекурсивный поиск ключей реестра 4](#_Toc182241317)

[2.3 Поиск и вывод значений внутри указанного ключа 4](#_Toc182241318)

[Заключение 6](#_Toc182241319)

[Список использованных источников 7](#_Toc182241320)

[Приложение А (обязательное) Исходный код программы 8](#_Toc182241321)

1 ФОРМУЛИРОВКА ЗАДАЧИ

Поиск в реестре ключей (ветвей реестра) и/или значений по заданному образцу:

– key – по имени; результат – иерархия подключей и значений данного ключа (фактически поддерево);

– value – по имени и/или собственно значению; результат – единственное значение или список значений.

Найденные данные выводятся на консоль и/или в файл, желательно в формате reg-файлов (перспектива возможного импорта обратно в реестр).

Интерфейс может быть минимальным (простейший GUI или командная строка).

2 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Согласно формулировке задачи, были спроектированы следующие функции программы:

– функция для вывода информации о заданном значении реестра;

– функция для рекурсивного поиска ключей реестра;

– функция для поиска и вывода значений внутри указанного ключа.

## **2.1 Вывод информации о заданном значении реестра**

Данная функция предназначена для вывода всей доступной информации о заданном значении реестра Windows в удобном и легко читаемом формате. Она принимает несколько ключевых параметров, включая имя значения, его тип и данные, что делает её универсальным инструментом для работы с реестром.

Функция автоматически определяет тип данных, к которому относится значение, например, строковый (REG\_SZ), числовой (REG\_DWORD) или бинарный (REG\_BINARY). В зависимости от типа данные отображаются соответствующим образом: строковые значения выводятся в привычном текстовом формате, числовые (REG\_DWORD) — в виде шестнадцатеричного числа, а бинарные данные (REG\_BINARY) представляются как последовательность байтов в формате hex-кода(рисунок 2.1).

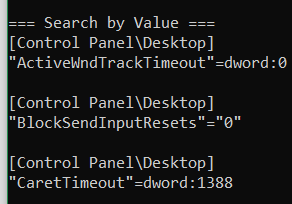


Рисунок 2.1 – Вывод информации о заданном значении реестра

## **2.2 Рекурсивный поиск ключей реестра**

Эта функция выполняет ключевую роль в рекурсивном поиске по реестру Windows. Она открывает заданный ключ реестра с использованием функции RegOpenKeyExA() [1] и осуществляет последовательный перебор всех его подкаталогов (подключей) через RegEnumKeyExA() [2] (рисунок 2.2).



Рисунок 2.2 – Вывод результатов поиска

В процессе работы функция проверяет соответствие текущего ключа или его подкаталогов заданному шаблону поиска. Если совпадение обнаружено, выводится имя соответствующего ключа, что позволяет пользователю легко идентифицировать нужный элемент.

При наличии вложенных подкаталогов функция рекурсивно вызывает сама себя, чтобы углубить поиск и проанализировать все возможные уровни структуры реестра. Такой подход обеспечивает полный охват заданной области поиска, делая процесс максимально эффективным и удобным для пользователя.

## **2.3 Поиск и вывод значений внутри указанного ключа**

Эта функция выполняет важную задачу работы с реестром Windows, обеспечивая открытие указанного ключа с помощью вызова функции RegOpenKeyExA(). После успешного открытия ключа она последовательно перебирает все содержащиеся в нём значения, используя RegEnumValueA() [3].

В процессе перебора для каждого найденного значения выполняется проверка на соответствие указанным критериям — совпадение имени или данных, представленных в строковом либо числовом формате. Если обнаруживается совпадение, функция вызывает PrintRegValue() для отображения детализированной информации о найденном значении, что позволяет пользователю мгновенно получить необходимые данные в удобной форме.

Благодаря такому подходу, данная функция является универсальным инструментом для выполнения поиска в реестре, упрощая анализ ключей и значений, а также предоставляя удобные средства для работы с конфигурацией системы (рисунок 2.3).



Рисунок 2.3 – Результат поиска при поиске по значению

ЗАКЛЮЧЕНИЕ

В рамках данной работы была разработана многофункциональная программа на языке C++, которая предоставляет возможность выполнения поиска по реестру Windows, используя широкий спектр функций WinAPI. Программа обладает расширенными возможностями для рекурсивного поиска ключей и значений реестра, что делает её крайне полезным инструментом для детального анализа и управления системными настройками.

Одной из ключевых особенностей приложения является поддержка поиска по имени и данным значений различных типов, включая строки и целые числа, что обеспечивает гибкость и удобство в использовании. Благодаря применению таких функций WinAPI, как RegOpenKeyExA(), RegEnumKeyExA() и RegEnumValueA(), удалось достичь высокой производительности и точности в обработке данных реестра, что значительно упрощает поиск и управление параметрами.

Разработанное приложение предоставляет пользователям, включая системных администраторов и опытных специалистов, мощный инструмент для навигации по сложной структуре реестра Windows. Это особенно актуально для задач по диагностике, настройке системы, а также для глубокого анализа её состояния. Подобный функционал может быть полезен в разнообразных сценариях: от исправления ошибок конфигурации до выполнения сложной оптимизации и исследования параметров системы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] RegOpenKeyExA function [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regopenkeyexa.

[2] RegEnumKeyExA function [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regenumkeyexa.

[3] RegEnumValueA function [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regenumvaluea.

ПРИЛОЖЕНИЕ А  
(обязательное)  
Исходный код программы

#include <windows.h>

#include <iostream>

#include <string>

#include <vector>

void PrintRegValue(HKEY hKey, const std::string& subKey, const std::string& valueName, DWORD type, const BYTE\* data, DWORD dataSize) {

std::cout << "[" << subKey << "]\n";

std::cout << "\"" << valueName << "\"=";

if (type == REG\_SZ) {

std::cout << "\"" << reinterpret\_cast<const char\*>(data) << "\"";

}

else if (type == REG\_DWORD) {

DWORD value = \*reinterpret\_cast<const DWORD\*>(data);

std::cout << "dword:" << std::hex << value;

}

else if (type == REG\_BINARY) {

std::cout << "hex:";

for (DWORD i = 0; i < dataSize; ++i) {

printf("%02x", data[i]);

if (i < dataSize - 1) {

std::cout << ",";

}

}

}

else {

std::cout << "Unsupported value type";

}

std::cout << "\n\n";

}

void SearchRegistryKey(HKEY hRootKey, const std::string& subKey, const std::string& searchKeyName) {

HKEY hKey;

if (RegOpenKeyExA(hRootKey, subKey.c\_str(), 0, KEY\_READ, &hKey) != ERROR\_SUCCESS) {

return;

}

char keyName[256];

DWORD keyNameSize;

DWORD index = 0;

while (true) {

keyNameSize = sizeof(keyName);

LONG result = RegEnumKeyExA(hKey, index, keyName, &keyNameSize, nullptr, nullptr, nullptr, nullptr);

if (result == ERROR\_NO\_MORE\_ITEMS) {

break;

}

else if (result == ERROR\_SUCCESS) {

std::string currentKey = subKey + "\\" + keyName;

if (searchKeyName.empty() || currentKey.find(searchKeyName) != std::string::npos) {

std::cout << "[" << currentKey << "]\n\n";

}

SearchRegistryKey(hRootKey, currentKey, searchKeyName);

}

++index;

}

RegCloseKey(hKey);

}

void SearchRegistryValue(HKEY hRootKey, const std::string& subKey, const std::string& searchValueName, const std::string& searchValueData = "", DWORD searchValueDWORD = 0) {

HKEY hKey;

if (RegOpenKeyExA(hRootKey, subKey.c\_str(), 0, KEY\_READ, &hKey) != ERROR\_SUCCESS) {

return;

}

DWORD index = 0;

char valueName[256];

DWORD valueNameSize;

BYTE data[512];

DWORD dataSize;

DWORD type;

while (true) {

valueNameSize = sizeof(valueName);

dataSize = sizeof(data);

LONG result = RegEnumValueA(hKey, index, valueName, &valueNameSize, nullptr, &type, data, &dataSize);

if (result == ERROR\_NO\_MORE\_ITEMS) {

break;

}

else if (result == ERROR\_SUCCESS) {

std::string valueStr(valueName);

bool matchName = searchValueName.empty() || valueStr.find(searchValueName) != std::string::npos;

bool matchData = true;

if (type == REG\_SZ && !searchValueData.empty()) {

std::string dataStr(reinterpret\_cast<const char\*>(data), dataSize - 1);

matchData = dataStr.find(searchValueData) != std::string::npos;

}

else if (type == REG\_DWORD && searchValueDWORD != 0) {

DWORD dwordValue = \*reinterpret\_cast<const DWORD\*>(data);

matchData = (dwordValue == searchValueDWORD);

}

else if (type == REG\_BINARY && !searchValueData.empty()) {

std::string hexData;

for (DWORD i = 0; i < dataSize; ++i) {

char hexByte[3];

sprintf\_s(hexByte, "%02x", data[i]);

hexData += hexByte;

}

matchData = hexData.find(searchValueData) != std::string::npos;

}

if (matchName && matchData) {

PrintRegValue(hKey, subKey, valueStr, type, data, dataSize);

}

}

++index;

}

RegCloseKey(hKey);

}

/\*

Control Panel\\Desktop папка с настройками рабочего стола

Software\\Microsoft\\Windows\\DWM папка с цветом системы

\*/

int main() {

std::string searchKey = "Control Panel\\Desktop";

std::string searchValueName = "";

std::string searchValueData = "";

DWORD searchValueDWORD = 0;

std::cout << "=== Search by Key ===\n";

SearchRegistryKey(HKEY\_CURRENT\_USER, searchKey, searchKey);

std::cout << "\n=== Search by Value ===\n";

SearchRegistryValue(HKEY\_CURRENT\_USER, searchKey, searchValueName, searchValueData, searchValueDWORD);

return 0;

}