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 Косяков М.М.

Проверил: ассистент кафедры информатики Гриценко Н.Ю.

Минск 2024

**СОДЕРЖАНИЕ**

[1 Формулировка задачи 3](#_Toc178093096)

[2 Краткие теоритические сведения 4](#_Toc178093097)

[3 Описание функций программы 5](#_Toc178093098)

[3.1 Запрос желания на получение информации о системе 5](#_Toc178093099)

[3.2 Вывод информации о системе 5](#_Toc178093100)

[Заключение 6](#_Toc178093101)

[Список использованных источников 7](#_Toc178093102)

[Приложение А (обязательное) Исходный код программы 8](#_Toc178093103)

1 ФОРМУЛИРОВКА ЗАДАЧИ

В рамках этой лабораторной работы ставлю перед собой цель углубить и закрепить навыки работы с отдельными задачами, связанных с конфигурированием, мониторингом, управлением системой в операционной системе Windows. В ходе выполнения работы необходимо изучить подходы к хранению и использованию конфигураций.

Особое внимание будет уделено реестру Windows, его назначению, структуре, типу и характеру хранимых данных, интерфейсу, группам функций API, порядку использования реестра.

Также предстоит изучить работу с журналированием, а именно назначением, решаемыми задачами, основными используемыми подходами. Предстоит ознакомиться с подходами к хранению и использованию конфигураций.

Для выполнения лабораторной работы по созданию утилиты для просмотра информации о системе необходимо реализовать приложение, которое будет осуществлять сбор информации о системе и отображение ее в удобном виде. Состав информации — на усмотрение разработчика, в качестве ориентира — стандартные «системные» сведения, можно расширить или специализировать на конкретном разделе. Источники информации: в основном реестр и специализированные системные функции, содержимое файловой системы, WMI, собственные измерения характеристик. Программа будет написана на языке C с использованием WinAPI.

В качестве задачи необходимо реализовать утилиту для просмотра информации о системе, а для этого:

– запросить желание на получение информации о системе;

– реализовать вывод информации о системе.

В результате выполнения этой лабораторной работы будут не только получены теоретические знания работы с отдельными задачами, связанных с конфигурированием, мониторингом, управлением системой в операционной системе Windows, но и практический опыт в разработке приложений, которые могут эффективно использовать ресурсы компьютера, а также корректно и безопасно взаимодействовать с системой.

Понимание концепций реестра Windows, порядка использования реестра, журналирования станет необходимым для создания высокопроизводительных программных решений.

2 КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ

Оверлеи — программные библиотеки, загружаемые и исполняемые в составе других программ. Они не находятся постоянно в памяти, а сменяют друг друга при необходимости [1].

Динамически подключаемые библиотеки в Windows являются исполняемыми модулями, которые не могут выполняться самостоятельно, а присоединяются к другим исполняемым файлам. Они включаются в адресное пространство процесса, обладают собственными экземплярами. Экспортируют символы (функции, методы) через таблицу экспорта. DLL служат для повторного использования кода, уменьшения размера исполняемых файлов и упрощения обновлений программного обеспечения. Могут содержать ресурсы помимо функций и данных.

Реестр Windows — это специальная база данных, где приложения и операционная система хранят разнообразную информацию, включая данные об аппаратной конфигурации, установленных программах, драйверах, административных настройках и пользовательских данных. Реестр организован иерархически как дерево из ключей и значений, с ограничением в 512 уровней.

Ключи представляют собой каталоги, содержащие подключи и значения, а также могут быть ссылками на другие ключи. Значения в реестре характеризуются типами данных, такими как строки, двоичные данные, двойные слова и другие. Рекомендуется хранить данные конфигурации приложений в определенных разделах реестра, в зависимости от их общности или специфичности для пользователей.

Regedit — утилита для работы с реестром Windows. Она позволяет работать с реестром через GUI и выполнить операции по изменению данных с помощью файлов .reg в командном режиме. Файлы .reg содержат данные реестра для импорта и экспорта [2].

Протоколирование (журналирование) — фиксация информации о событиях в системе для обнаружения ошибок, отладки, мониторинга процессов, сбора данных и контроля безопасности. Важно для отладки программ.

Windows Management Instrumentation (WMI) — управление Windows-системой через объекты с методами и свойствами, хранящимися в файлах .mof. Доступ осуществляется через PowerShell.

3 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Согласно формулировке задачи, были спроектированы следующие функции программы [3]:

– запрос желания на получение информации о системе;

– вывод информации о системе;

– завершение работы программы.

**3.1 Запрос желания на получение информации о системе**

Для начала обработки программа запрашивает у пользователя желание на получение информациио системе (рисунок 3.1).

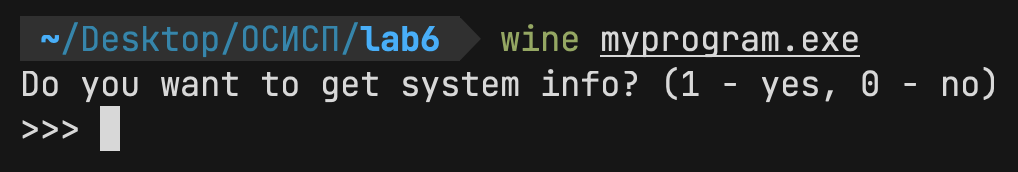


Рисунок 3.1 – Запрос желания на получение информации о системе

**3.2 Вывод информации о системе**

Программа выводит информацию о системе, включая данные о процессоре (архитектура и количество логических процессоров), модели процессора из реестра Windows, версии операционной системы, имени компьютера и имени текущего пользователя (рисунок 3.2).

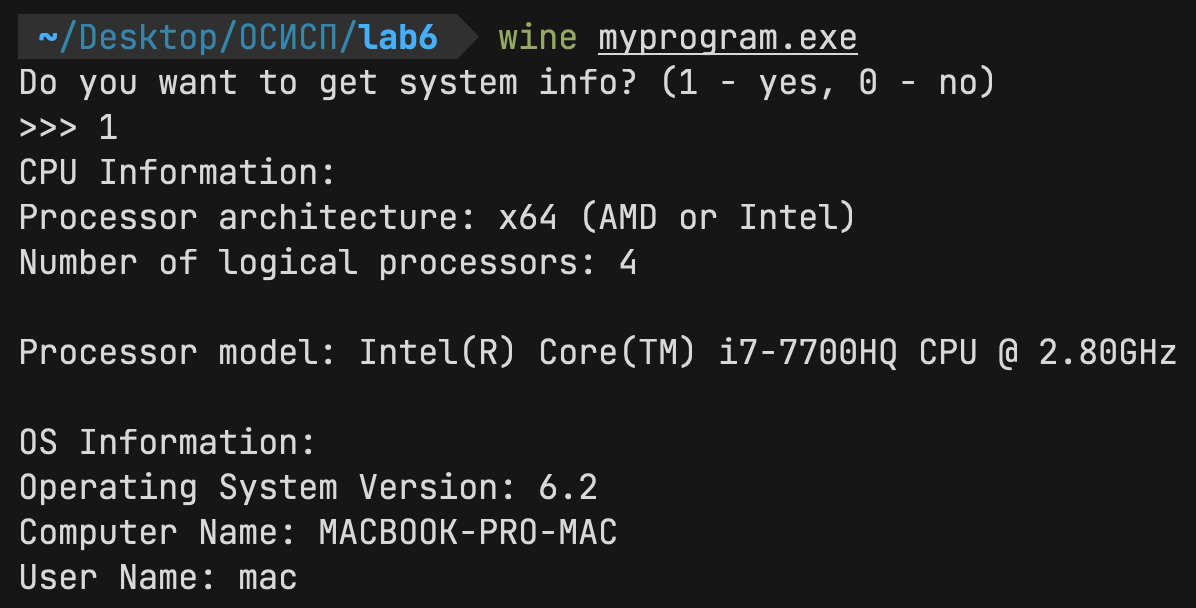


Рисунок 3.2 – Сообщение о выводе информации о системе

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной лабораторной работы были изучены и закреплены навыки программирования приложений для операционной системы Windows с использованием отдельных задач, связанных с конфигурированием, мониторингом, управлением системой. Мы погрузились в основы подходов к хранению и использованию конфигураций и научились их реализовывать в среде Windows с помощью WinAPI.

Основными аспектами, с которыми мы ознакомились, стали реестр Windows, его назначение, структура, тип и характер хранимых данных, интерфейс, группы функций API, порядок использования реестра. Мы также освоили типичные проблемы, возникающие при работе с журналированием, а именно назначением, решаемыми задачами, основными используемыми подходами, что сыграет важную роль в разработке производительных приложений.

Изучили принципы работы с подходами к хранению и использованию конфигураций.

В ходе работы над созданием утилиты для просмотра информации о системе мы реализовали приложение, которое осуществляет сбор информации о системе и отображает ее в удобном виде.

Эта лабораторная работа не только позволила углубить наши теоретические знания в области служебных и технологичных задач, но и предоставила ценный практический опыт в разработке эффективных приложений, способных эффективно использовать ресурсы компьютера.

Понимание концепций реестра Windows, порядка использования реестра, журналирования, а также навыки управления ими стали ключевыми в создании высокопроизводительных программных решений.

Выполнение этой лабораторной работы позволило нам не только расширить нашу базу знаний, но и приобрести практические навыки, которые будут полезны в дальнейшей разработке программного обеспечения для Windows.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Build desktop Windows apps using the Win32 API [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/

[2] Основные сообщения ОС Windows (Win32 API). Программирование в ОС Windows. Лекция 1. – Режим доступа: https://www.youtube.com/watch?v=  
wTArIolxch0

[3] Разработка приложений с помощью WinAPI. – Режим доступа: https://shorturl.at/BDJW8

ПРИЛОЖЕНИЕ А

(обязательное)

**Исходный код программы**

#include <windows.h>

#include <stdio.h>

#define nl printf("\n");

void printCPUInfo()

{

// Declare a SYSTEM\_INFO structure to hold system details

SYSTEM\_INFO sysInfo;

GetSystemInfo(&sysInfo);

printf("CPU Information:\n");

if (sysInfo.wProcessorArchitecture == 9)

{

printf("Processor architecture: x64 (AMD or Intel)\n"); // 9 - x64 (AMD or Intel)

}

printf("Number of logical processors: %d\n", sysInfo.dwNumberOfProcessors); // Number of logical processors

}

void printProccessModel()

{

HKEY hKey;

const char \*subKey = "HARDWARE\\DESCRIPTION\\System\\CentralProcessor\\0";

const char \*valueName = "ProcessorNameString";

char processorName[256];

DWORD bufferSize = sizeof(processorName);

DWORD type = 0;

if (RegOpenKeyExA(HKEY\_LOCAL\_MACHINE, subKey, 0, KEY\_READ, &hKey) != ERROR\_SUCCESS)

{

printf("Failed to open the registry to get the processor model.\n");

return;

}

if (RegQueryValueExA(hKey, valueName, NULL, &type, (LPBYTE)processorName, &bufferSize) != ERROR\_SUCCESS || type != REG\_SZ)

{

RegCloseKey(hKey);

printf("Failed to get the processor model from the registry.\n");

return;

}

RegCloseKey(hKey);

printf("Processor model: %s\n", processorName);

}

void printOSInfo()

{

// Retrieve and display the computer name

char computerName[MAX\_COMPUTERNAME\_LENGTH + 1];

DWORD size = sizeof(computerName);

GetComputerNameA(computerName, &size);

// Retrieve and display the current user name

char userName[256];

DWORD userNameSize = sizeof(userName);

GetUserNameA(userName, &userNameSize);

// Declare and initialize an OSVERSIONINFOEX structure to hold OS details

OSVERSIONINFOEX osInfo;

ZeroMemory(&osInfo, sizeof(OSVERSIONINFOEX));

osInfo.dwOSVersionInfoSize = sizeof(OSVERSIONINFOEX);

GetVersionEx((OSVERSIONINFO \*)&osInfo);

printf("OS Information:\n");

printf("Operating System Version: %d.%d\n", osInfo.dwMajorVersion, osInfo.dwMinorVersion);

printf("Computer Name: %s\n", computerName);

printf("User Name: %s\n", userName);

}

// Function to retrieve and display system information

void displaySystemInfo()

{

printCPUInfo();

nl;

printProccessModel();

nl;

printOSInfo();

nl;

}

int main()

{

int input;

printf("Do you want to get system info? (1 - yes, 0 - no) \n");

printf(">>> ");

scanf("%d", &input);

switch (input)

{

case 1:

displaySystemInfo();

break;

default:

break;

}

return 0;

}