Лабораторна робота №4

Системний реєстр

Мета: Вивчити призначення та методи роботи з системним реєстром Windows.

Хід роботи

1. Вивчити призначення та структуру системного реєстру.

2. Вивчити призначення та методи роботи з утилітою RegEdit.

3. Знайти відповідні розділи реєстру в яких є інформація про програми та служби які завантажуються автоматично.

4. Програмно, вивести список усіх програм та служб які завантажуються автоматично для усіх користувачів та поточного користувача.

5. Додати програмно до автозавантаження програм для поточного користувача завантаження програми WinWord або іншої.

6. Вивести список повторно, та показали що зареєстрована програма є у списку.

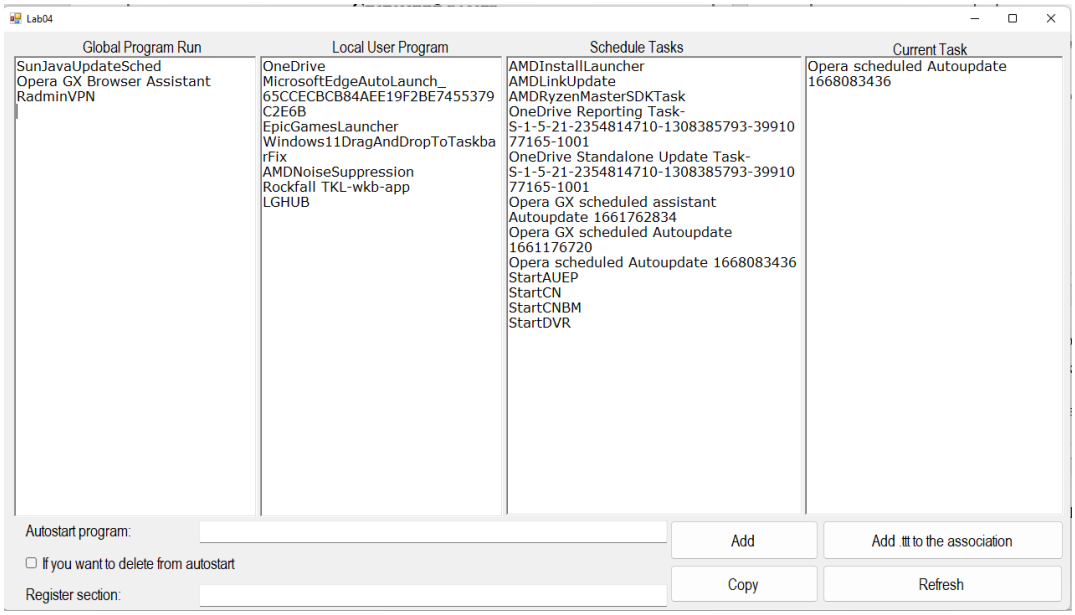
7. Вивести список усіх завдань, які зареєстровані у планувальнику задач системи. Інформації отримати з відповідного розділу реєстру, як для усіх користувачів так і для поточного користувача.

8. Зробити програмно копію будь якого розділу реєстру у файл відповідного формату .reg

9. За допомогою текстового редактора створити REG файл, за допомогою якого в реєстр у відповідний розділ буде внесено інформацію про асоціацію відкриття файлів .ttt програмою notepad.

10. Описати усі виконані дії у звіті.

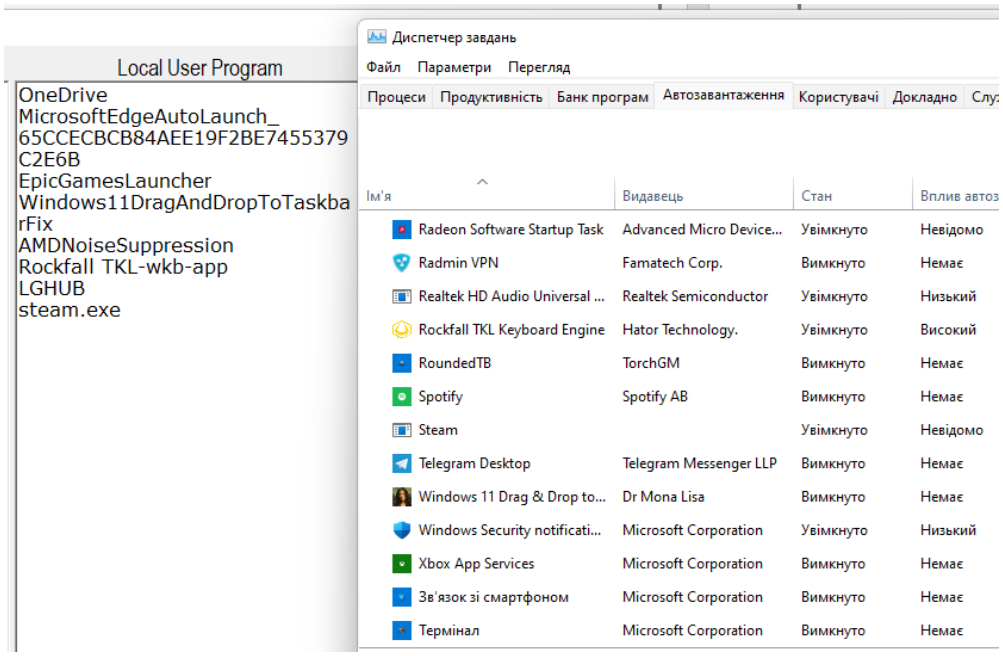
Головне вікно програми:



На головному екрані можна побачити 4 основні блоки з інформацією, можна побачити програми з автозапуску, поточні програми планувальника системи.

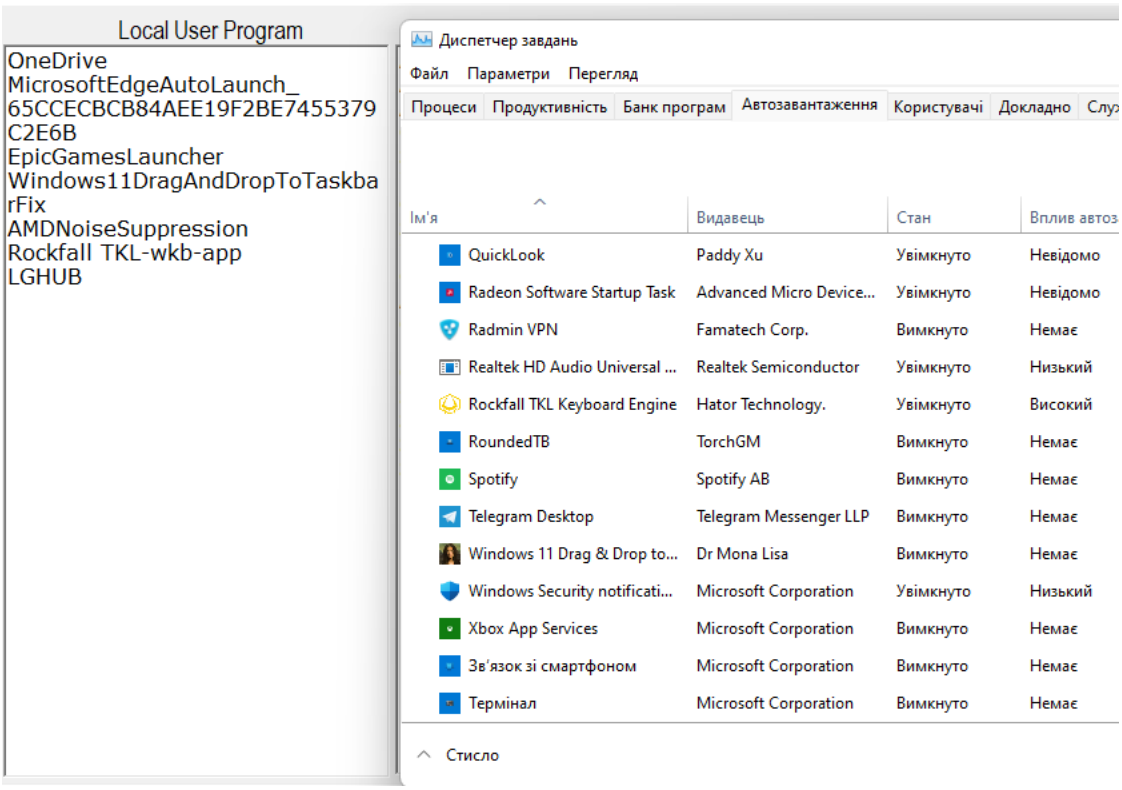
Спробую додати програму до автозапуску:

Оновлюю інформацію і перевіряю додавання до автозапуску:



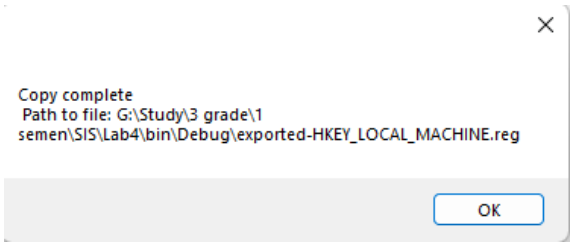
Тепер поставимо галочку і спробую видалити програму з автозапуску:

Оновлю інформацію і перевірю видалення:

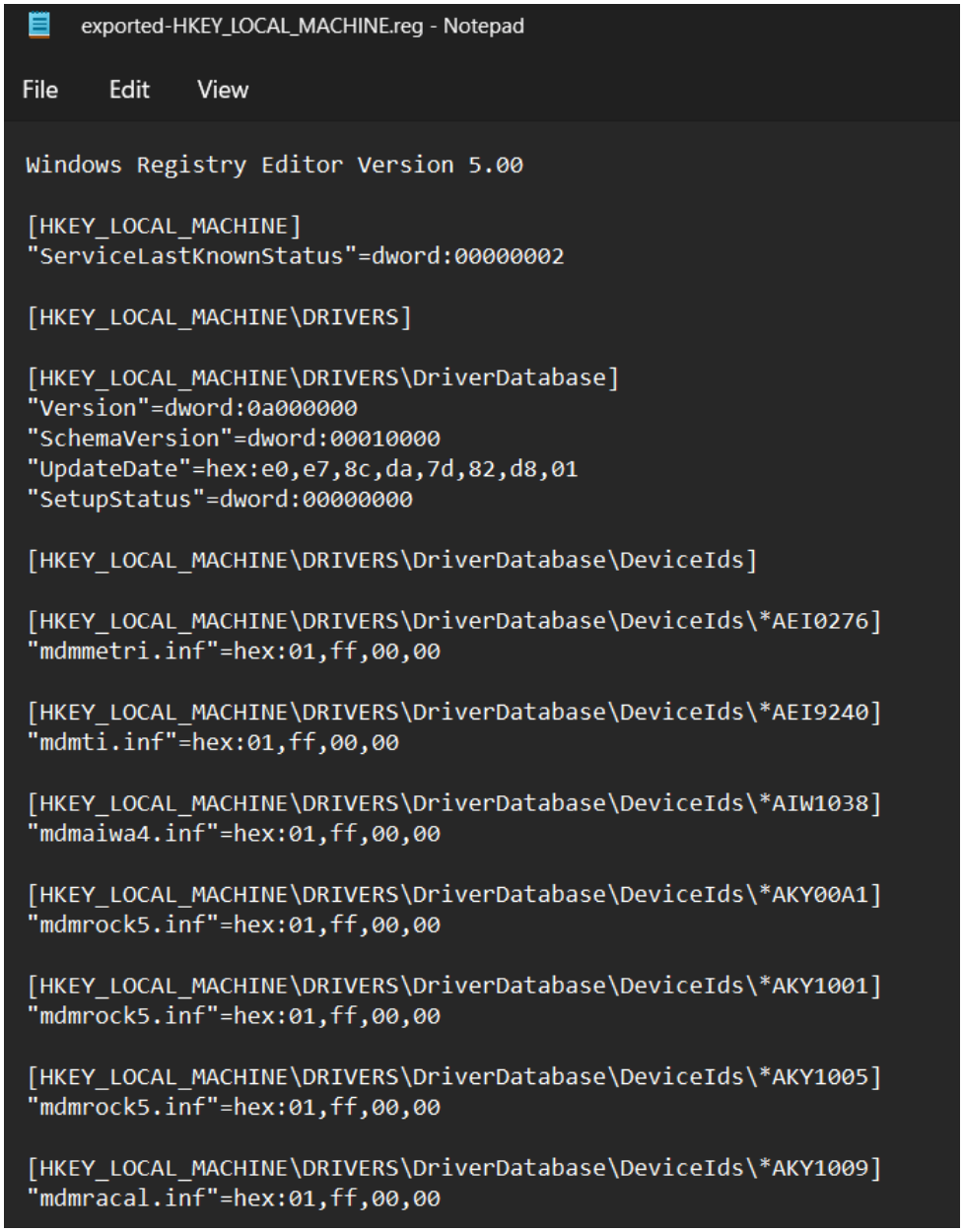


Спроба скопіювати розділ реєстру за його назвою:

Успішне копіювання розділу:



Скопійований розділ у файл:



Лістинг програми:

```
namespace Lab04
{
    public partial class Form1 : Form
    {
        public Form1()
        {

            InitializeComponent();
            UpdateGlobalProgramRun();
            UpdateLocalUserProgram();
            UpdateSheduleTask();
            UpdateCurrentTask();

        }

        private void UpdateGlobalProgramRun()
        {
            GlobalProgramRun.Clear();
            var services = Registry.LocalMachine
                .OpenSubKey(@"Software\Microsoft\Windows\CurrentVersion\Run")
                .GetValueNames()
                .ToList();
            services.ForEach(s => GlobalProgramRun.AppendText(s + Environment.New-
Line));
        }

        private void UpdateLocalUserProgram()
        {
            LocalUserProgram.Clear();
            var services = Registry.CurrentUser
                .OpenSubKey(@"Software\Microsoft\Windows\CurrentVersion\Run")
                .GetValueNames()
                .Where(s => !string.IsNullOrEmpty(s))
                .ToList();
            services.ForEach(s => LocalUserProgram.AppendText(s + Environment.New-
Line));
        }

        private void UpdateSheduleTask()
        {
            SheduleTask.Clear();
            ParseScheduleTasks().ToList().ForEach(s => SheduleTask.AppendText(s + Envi-
ronment.NewLine));
        }

        private IList<string> ProcessTaskFolder(ITaskFolder tFolder, string author =
"")
        {
            var scheduledTasks = new List<string>();
            var tCol = tFolder.GetTasks((int)_TASK_ENUM_FLAGS.TASK_ENUM_HIDDEN);

            for (int idx = 1; idx <= tCol.Count; idx++)
            {
                if (string.IsNullOrEmpty(author))
                    scheduledTasks.Add(tCol[idx].Name);

                var principal = GetAuthorFromXmlString(tCol[idx].Xml);
                if (!string.IsNullOrEmpty(principal) && principal == author)
                    scheduledTasks.Add(tCol[idx].Name);
            }
```

```
            var tFolderCol = tFolder.GetFolders(0);
            for (int idx = 1; idx <= tFolderCol.Count; idx++)
                ProcessTaskFolder(tFolderCol[idx]);

            return scheduledTasks;
        }

        private string GetAuthorFromXmlString(string xmlString)
        {
            if (string.IsNullOrEmpty(xmlString))
                return string.Empty;

            var xml = new XmlDocument();
            xml.LoadXml(xmlString);
            var registrationInfo = xml.GetElementsByTagName("RegistrationInfo");
            var childNodes = registrationInfo[0].ChildNodes;
            if (registrationInfo.Count == 0 || childNodes.Count == 0)
                return string.Empty;

            var authorNode = string.Empty;
            foreach (XmlNode node in childNodes)
                if (node.Name == "Author")
                    authorNode = node.InnerText;

            return authorNode;
        }

        private IList<string> ParseScheduleTasks(bool forCurrentUser = false)
        {
            var TaskServ = new TaskScheduler.TaskScheduler();
            TaskServ.Connect();

            if (!forCurrentUser)
                return ProcessTaskFolder(TaskServ.GetFolder("\\"));

            var author = string.Join("\\", GetAuthorName(TaskServ));
            return ProcessTaskFolder(TaskServ.GetFolder("\\"), author);
        }

        private IEnumerable<string> GetAuthorName(TaskScheduler.TaskScheduler TaskServ)
        {
            if (TaskServ != null)
            {
                if (!string.IsNullOrEmpty(TaskServ.ConnectedDomain))
                    yield return TaskServ.ConnectedDomain;

                if (!string.IsNullOrEmpty(TaskServ.ConnectedUser))
                    yield return TaskServ.ConnectedUser;
            }
        }

        private void UpdateCurrentTask()
        {
            CurrentTask.Clear();
            ParseScheduleTasks(true).ToList().ForEach(s => CurrentTask.AppendText(s +
Environment.NewLine));
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void buttonRefresh_Click(object sender, EventArgs e)
        {
```

```
            UpdateGlobalProgramRun();
            UpdateLocalUserProgram();
            UpdateSheduleTask();
            UpdateCurrentTask();
        }

        private void buttonAddToAutoStart_Click(object sender, EventArgs e)
        {
            string ExePath = AutostartProgram.Text.ToString();
            RegistryKey reg;
            reg = Registry.CurrentUser.OpenSubKey(@"SOFTWARE\Microsoft\Windows\Cur-
rentVersion\Run", true);
            try
            {
                if(checkBox1.Checked)
                {
                    reg.DeleteValue(ExePath);
                }
                else
                {
                    reg.SetValue(ExePath, "\"" + ExePath + "\"");
                }
                reg.Flush();
                reg.Close();
            }
            catch (Exception)
            {

                MessageBox.Show("Could not add program to autostart");
            }
        }

        private void label5_Click(object sender, EventArgs e)
        {

        }

        private void ExportKey(string key, string path)
        {
            string arguments = $"reg export \"{key}\" \"{path}\" /y";
            string strCmdText = "/C " + arguments;
            const string FILE_NAME = "cmd.exe";

            var process = new Process()
            {
                StartInfo = new ProcessStartInfo()
                {
                    FileName = FILE_NAME,
                    Arguments = strCmdText,
                    UseShellExecute = false,
                    CreateNoWindow = false,
                }
            };

            try
            {
                process.Start();
                if (process != null)
                    process.WaitForExit();
            }
            catch(Exception)
            {
                MessageBox.Show("Failed to copy partition");
            }
```

```
        }

        private void buttonCopy_Click_1(object sender, EventArgs e)
        {
            string key = RegisterSection.Text;
            var path = $@"{Environment.CurrentDirectory}\exported-{key}.reg";

            ExportKey(key, path);
            string PathToFile = path.ToString();
            MessageBox.Show($"Copy complete \n Path to file: {PathToFile}");

            RegisterSection.Text = string.Empty;
        }

        private void buttonAddToAssociation_Click(object sender, EventArgs e)
        {
            var path = $@"{Environment.CurrentDirectory}\tttAssociation.reg";
            string arguments = $"/s {path}";
            var regProcess = Process.Start($"regedit.exe", arguments);
            regProcess.WaitForExit();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void richTextBoxCurrentUserPrograms_TextChanged(object sender, Even-
tArgs e)
        {

        }
    }
}
```

Висновки: вивчив призначення та методи роботи з системним реєстром Windows.