МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по лабораторной работе № 8**

**по дисциплине «Операционные системы»**

**Тема: «Команды ОС и Win32 API (Python)»**

Выполнил: Сидоров Д. С., группа ИТС-123

Проверил: к.т.н., доц. Семёнов А. А.

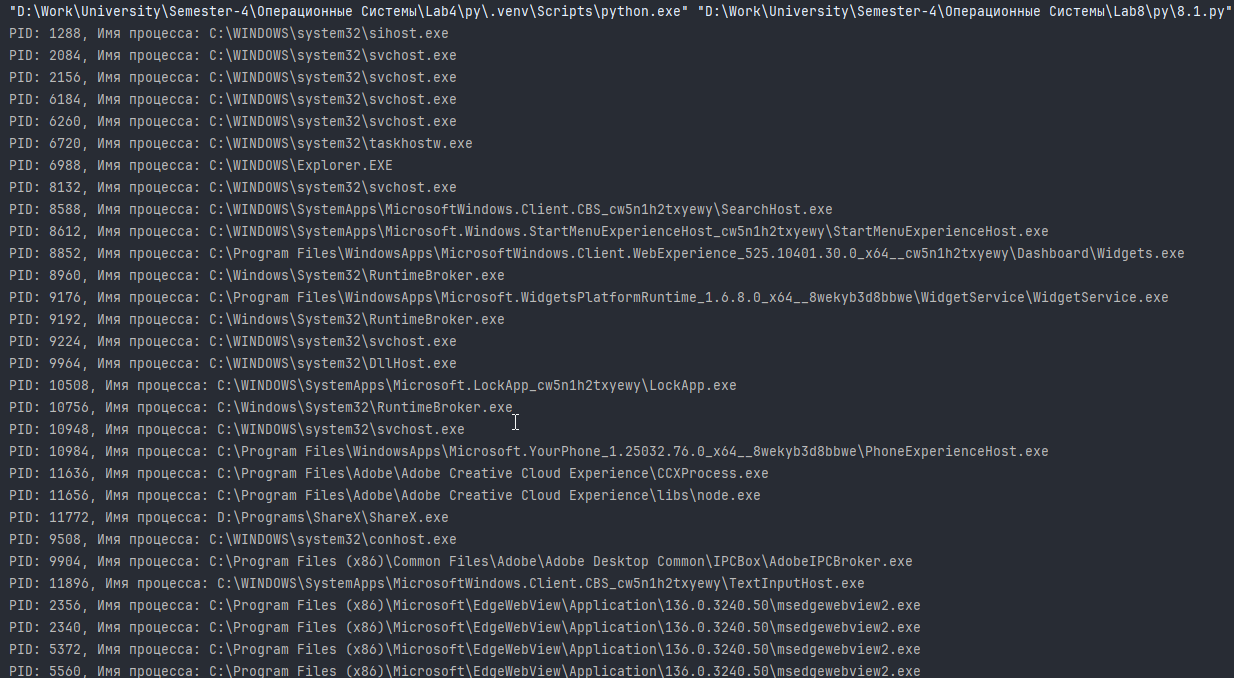
Москва, 2025г

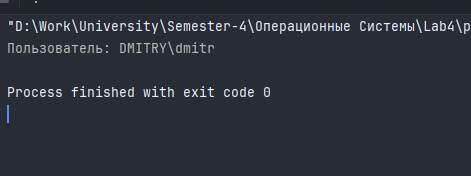
**Команды ОС и Win32 API (Python)**

Использование Win32API в Python позволяет создавать приложения, которые могут взаимодействовать с операционной системой Windows на низком уровне. Вот несколько примеров таких приложений:

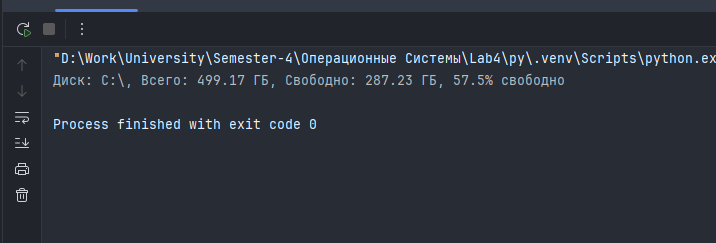
1. Получение информации о процессах.
2. Получение информации о пользователях.
3. Получение информации о сетевых подключениях.
4. Получение информации о дисковом пространстве.

Необходимо провести эксперименты с этими примерами, подправить их при необходимости.

*  
Рисунок 1. Работающее приложение из п. 1.*



*Рисунок 2. Работающее приложение из п. 2.*



*Рисунок 3. Работающее приложение из п. 4.*

**Листинг кода**

**1.py**import win32api

import win32process

import win32con

def get\_process\_list():

processes = []

snapshot = win32process.EnumProcesses()

for pid in snapshot:

try:

hProcess = win32api.OpenProcess(win32con.PROCESS\_QUERY\_INFORMATION | win32con.PROCESS\_VM\_READ, False, pid)

modules = win32process.EnumProcessModules(hProcess)

exe\_name = win32process.GetModuleFileNameEx(hProcess, modules[0]) if modules else "Unknown"

processes.append((pid, exe\_name))

win32api.CloseHandle(hProcess)

except (win32api.error, IndexError):

continue

return processes

def get\_process\_details(pid):

try:

hProcess = win32api.OpenProcess(win32con.PROCESS\_QUERY\_INFORMATION | win32con.PROCESS\_VM\_READ, False, pid)

modules = win32process.EnumProcessModules(hProcess)

process\_name = win32process.GetModuleFileNameEx(hProcess, modules[0]) if modules else "Unknown"

win32api.CloseHandle(hProcess)

return f"PID: {pid}, Имя процесса: {process\_name}"

except win32api.error:

return f"PID: {pid}, Имя процесса: Недоступно"

if \_\_name\_\_ == "\_\_main\_\_":

process\_list = get\_process\_list()

for pid, process\_name in process\_list:

print(get\_process\_details(pid))

**2.py**import win32api

import win32con

import win32security

def get\_logged\_users():

users = []

try:

hToken = win32security.OpenProcessToken(win32api.GetCurrentProcess(), win32con.TOKEN\_QUERY)

user\_sid, \_ = win32security.GetTokenInformation(hToken, win32security.TokenUser)

user\_name, domain, \_ = win32security.LookupAccountSid(None, user\_sid)

users.append(f"{domain}\\{user\_name}")

win32api.CloseHandle(hToken)

except win32api.error:

pass

return users

if \_\_name\_\_ == "\_\_main\_\_":

users = get\_logged\_users()

for user in users:

print(f"Пользователь: {user}")

**3.py**import win32file

import win32net

def get\_network\_connections():

connections = []

resume\_handle = 0

level = 1

try:

while True:

network\_connections, total\_resume, resume\_handle = win32net.NetUseEnum(None, level, resume\_handle)

print(network\_connections, total\_resume, resume\_handle)

for connection in network\_connections:

local\_name = connection.get('local', '')

remote\_name = connection.get('remote', '')

status = get\_connection\_status(remote\_name)

connections.append((local\_name, remote\_name, status))

if not resume\_handle:

break

except Exception as e:

print(f"Ошибка при получении сетевых подключений: {e}")

return connections

def get\_connection\_status(remote\_name):

try:

win32file.GetFileAttributes(remote\_name)

return "Подключено"

except Exception:

return "Отключено"

connections = get\_network\_connections()

for local\_name, remote\_name, status in connections:

print(f"Локальное имя: {local\_name}, Удаленное имя: {remote\_name}, Статус: {status}")

**4.py**import win32file

def get\_disk\_space(drive):

free\_bytes, total\_bytes, \_ = win32file.GetDiskFreeSpaceEx(drive)

percent\_free = (free\_bytes / total\_bytes) \* 100

free\_bytes /= 1000\*\*3

total\_bytes /= 1000\*\*3

return f"Диск: {drive}, Всего: {round(total\_bytes, 2)} ГБ, Свободно: {round(free\_bytes, 2)} ГБ, {percent\_free:.1f}% свободно"

# Пример использования

print(get\_disk\_space("C:\\"))

**Вывод:** В ходе лабораторной работы были изучены возможности взаимодействия с операционной системой Windows на низком уровне с использованием Win32 API через Python. Проведены эксперименты с получением информации о запущенных процессах, пользователях системы, сетевых подключениях и дисковом пространстве. Выявлены некоторые несоответствия в исходном коде и внесены необходимые корректировки для корректной работы примеров. Практическое применение данных функций позволяет разрабатывать мониторинговые и диагностические инструменты для Windows. Данный опыт демонстрирует гибкость Python при работе с системными API и его потенциал для разработки системных утилит.