Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ



**Курсовий проект**

На тему:

«Розробка програмного забезпечення для моніторингу системи в реальному часі.»

З дисципліни «Системне програмне забезпечення»

Виконав:

ст. гр. КІ-307

Срипник Т.О.

Прийняв: Олексів М.В.

Львів 2024

**Завдання**

Розробка програмного забезпечення для моніторингу системи в реальному часі

**Анотація**

У даному проекті розглядається процес створення диспетчера задач у вигляді програмної утиліти, яка забезпечує доступ до інформації про систему за допомогою системної бібліотеки «psutil». Для розробки такої утиліти використовувалося сучасне та потужне середовище PyCharm для мови програмування Python. В результаті роботи програми користувач отримує графічний та інтуїтивно зрозумілий інтерфейс диспетчера задач.

**Зміст**

Вступ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

1. Теоретичні відомості . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

2. Аналіз завдання . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

3. Розробка програми . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

3.1 Вибір мови та середовища програмування . . . . . . . . . . . . . . . . . . . . . 8

3.2 Бібліотеки, які використовуються при написанні програми . . . . . . . 8

4. Опис інтерфейсу та тестування програми . . .. . . . . . . . . . . . . . . . . . .9

4.1 Опис інтерфейсу програми . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

4.2 Тестування програми . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

Висновок . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

Список використаної літератури . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

Додаток А . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

Додаток Б . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

Додаток В . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

Додаток Г . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

**Вступ**

Операційна система - це комплекс програм, які призначені для ефективного управління різними ресурсами комп'ютера та обчислювальними процесами, а також для забезпечення коректної взаємодії користувача з комп'ютером та його обладнанням.

Одна з основних функцій операційної системи полягає в керуванні ресурсами комп'ютера та їх ефективному розподілі. Ресурсами можуть бути різноманітні компоненти комп'ютера, такі як оперативна пам'ять, простір на диску, периферійні пристрої та процесорний час. Операційна система не тільки надає користувачам та розробникам зручний інтерфейс доступу до апаратних засобів, але й відповідає за розподіл цих ресурсів з метою максимізації їх використання та оптимізації продуктивності комп'ютера.

Обчислювальні ресурси - це можливості, які надаються різними компонентами обчислювальної системи та використовуються (або "займаються") під час її роботи.

Ресурси розподіляються між процесами. Процес (задача) програма в стадії виконання.

**1.Теоретичні відомості**

Типи обчислювальних ресурсів:

- Процесорний час

- Пам'ять (оперативна і віртуальна)

- Місце на жорсткому диску (постійна пам'ять)

- Пропускна здатність мережі.

Ресурси розподіляються між процесами. Процес (задача) програма в стадії виконання.

Програма - це статичний об'єкт, що представляє собою файл з кодами і даними.

Процес - це динамічна одиниця роботи, яка виникає в операційній системі після запуску програми. Операційна система відповідає за ефективне розподілення ресурсів обчислювальної системи для забезпечення найбільш оптимального використання їх в процесі роботи.

Ресурси - це обсяг роботи або термін експлуатації, на який розраховується пристрій, будинок або інші об'єкти. Якщо ресурс вичерпано, безпечна робота пристрою не може бути гарантована, тому що потрібен капітальний ремонт або заміна.

Ресурси персонального комп'ютера

Ресурсом є будь-який компонент ЕОМ і надані їм можливості: центральний процесор, оперативна або зовнішня пам'ять, зовнішній пристрій, програма і т.д. Ресурси поділяються на чотири види.

види ресурсівперсонального комп'ютера:

Апаратні ресурси (Hardware), файлові ресурси, програмні ресурси (Software), мережеві ресурси

Апаратні ресурси - це системний блок, периферійні пристрої, будь-яке обладнання, підключене до комп'ютера.

Файлові ресурси - це файли і папки, а також вся файлова система.

Програмні ресурси - це все програми встановлені в комп'ютері. Часто називають програмним забезпеченням (ПЗ). Програмне забезпечення підрозділяється на два види: системне і прикладне ПЗ.

Мережеві ресурси- Ресурси доступні за коштами ЛВС. [1] Як правило, це ресурси інших комп'ютерів доступні за локальної або глобальної мережі.

Комп'ютерна мережа - апаратне і програмне об'єднання двох і більше комп'ютерів з виділенням мережевих ресурсів. Для зв'язку комп'ютерів в комп'ютерну мережу можуть бути використані наступні апаратні засоби:

1. Модем

2. Мережева карта

3. Мережевий кабель

4. Мережеві комутатори

5. WI-FI адаптер

6. Бездротове обладнання

7. Маршрутизатор

8. Мережеві екрани

Мережевими ресурсами можуть бути:

- Обладнання (тобто апаратні ресурси іншого ПК або мережеві пристрої), наприклад мережевий принтер.

- Інформація (тобто файли і папки іншого комп'ютера), наприклад інформація в Інтернеті, або на сервері.

- Програмне забезпечення (встановлений на іншому комп'ютері)

**2.Аналіз завдання та способи його вирішення**

Метою цієї курсової роботи є розробка програми, яка буде мати ті ж функції, що й вбудована програма у Windows. Для того, щоб програма працювала правильно, користувач повинен легко розуміти інтерфейс програми та бути здатним керувати підключенням ПК до бездротової мережі Інтернет. Я використав системну бібліотеку «wlanapi.lib», яка містить готові методи для взаємодії з бездротовим модулем в ПК.

**3. Розробка програми**

**3.1 Вибір мови та середовища програмування**

Я вибрав мову програмування Python для написання утиліти, оскільки ця мова є об'єктно-орієнтованою, має динамічну систему типізації і надає широкий вибір інструментів для програміста. Я використовуватиму середовище програмування PyCharm для написання програми, оскільки воно має зручний інтерфейс, широкий набір інструментів для створення графічного інтерфейсу користувача. Крім того, PyCharm дозволяє легко тестувати та налагоджувати програми, що розробляються в ньому.

* **Бібліотеки, які використовуються при написанні програми**

Psutil - це бібліотека для мови програмування Python, яка дозволяє отримувати інформацію про процеси, які працюють на комп'ютері, а також про використання системних ресурсів (таких як ЦП, пам'ять, диски, мережа та датчики). Ця бібліотека дозволяє моніторити систему, профілювати та обмежувати ресурси процесів та управляти запущеними процесами. Вона надає багато функціональності, яка доступна в командному рядку UNIX, такі як ps, top, iotop, lsof, netstat, ifconfig, free та інші. Psutil підтримується на різних платформах.

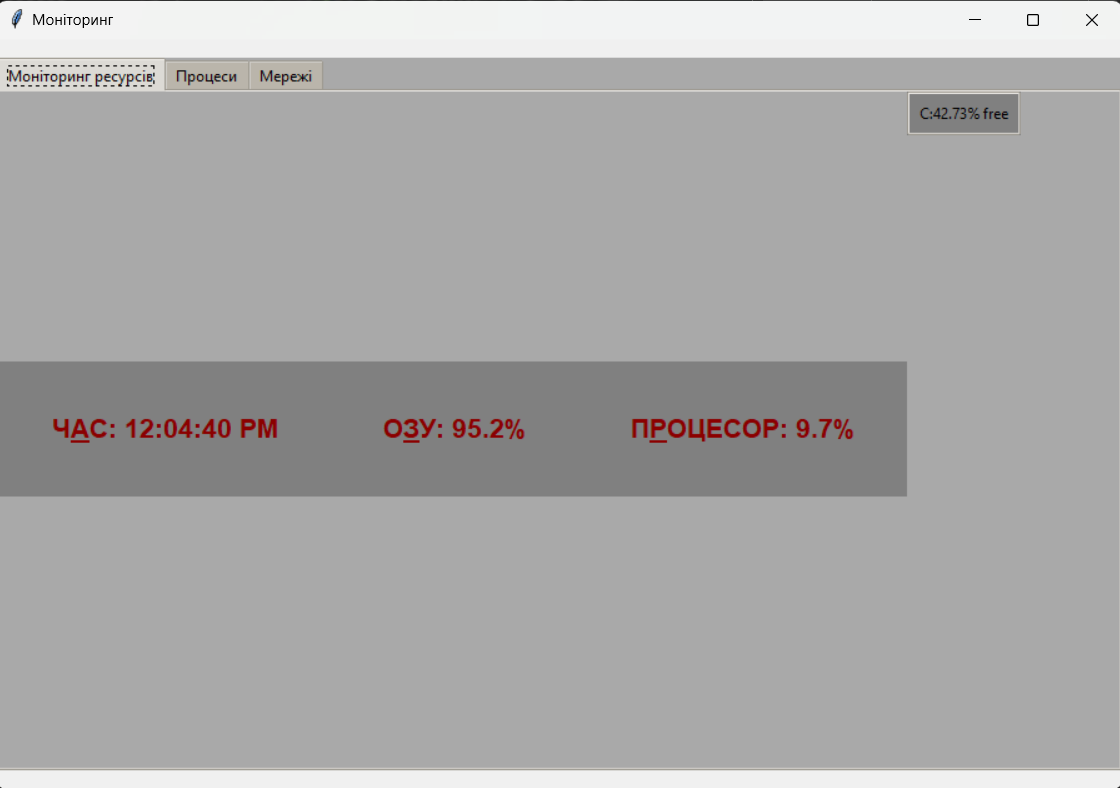
Tkinter - це графічна бібліотека для створення користувацького інтерфейсу програм на мові програмування Python. Вона використовує засоби бібліотеки Tk, яка є відкритою і поширюється разом з вихідним кодом. Tkinter є частиною стандартної бібліотеки Python, тому вона доступна на різних платформах.

**4. Опис інтерфейсу та тестування програми**

4.1 Опис інтерфейсу програми

Головне меню програми, яке наведено на малюнку 4.1, складається з таких елементів:

* Resource monitoring (Мал. 4.1)
* Processes (Мал. 4.2)
* Networks (Мал. 4.3)



Малюнок 4.1

Програма відображає актуальні дані про стан системи, такі як поточний час, відсоток використання оперативної пам'яті та центрального процесора. Це дозволяє користувачу моніторити поточний стан системи та вчасно реагувати на потенційні проблеми з її продуктивністю.

A screenshot of a computer

Description automatically generated

Малюнок 4.2

Дані, що були надані вище, вказують на наявність утиліти, яка надає список усіх запущених на комп'ютері процесів. Це може бути корисно для моніторингу та управління роботою програмного забезпечення на комп'ютері.

A screenshot of a computer

Description automatically generated

4.2 Тестування програми

* Порівнюю дані моніторингу із справжнім диспетчером задач

A screenshot of a computer

Description automatically generated

* Процеси

A screenshot of a computer

Description automatically generated

**Висновок**

При створенні утиліти моніторингу використання ресурсів системи було важливо розуміти, як саме працюють компоненти системи, які ресурси можна моніторити та які є важливими метрики для аналізу. В процесі розробки програми, були вивчені технології та інструменти, які можуть допомогти у зборі та аналізі даних про систему, що є важливими навичками для будь-якого програміста.

**Список використаної літератури**

* <https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D1%96_%D1%80%D0%B5%D1%81%D1%83%D1%80%D1%81%D0%B8> - інформація про значимість сисвтемних ресурсів
* <https://psutil.readthedocs.io/en/latest/> - документація для бібліотеки psutil
* <https://docs.python.org/3/library/tkinter.html> - документація для бібліотеки tkinter, призначеної для розробки графічних інтерфейсі

**Додаток А**

app.py

import tkinter as tk  
from time import strftime  
from tkinter import ttk  
  
import psutil  
  
from app\_table\_wiget import tableFrame  
from componentUsaged import componentUsaged  
  
from main import runningProcesses, showNetwork, FreeSpace  
  
# root window  
root = tk.Tk()  
root.geometry('900x600')  
root.title('Моніторинг')  
  
# create a notebook  
notebook = ttk.Notebook(root)  
notebook.pack(pady=15, expand=True)  
  
# create frames  
frame1 = ttk.Frame(notebook, width=600, height=600)  
frame2 = ttk.Frame(notebook, width=600, height=600)  
frame3 = ttk.Frame(notebook, width=600, height=600)  
frame1.pack(fill='both', expand=True)  
frame2.pack(fill='both', expand=True)  
frame3.pack(fill='both', expand=True)  
  
# add frames to notebook  
notebook.add(frame1, text='Resource monitoring ')  
notebook.add(frame2, text='Processes')  
notebook.add(frame3, text='Networks')  
  
  
  
tableFrame(root=root,frame=frame3,  
 dataGener=showNetwork,  
 columns = ["Опис", "MAC-адреса", "IP-адреса"])  
  
tableFrame(root=root,frame=frame2,  
 dataGener=runningProcesses,  
 columns = ["ід", "ім'я", "Віртуальний розмір (VirtualSize)"])  
  
  
  
  
  
def time(label, description):  
 string = description + strftime('%H:%M:%S %p')  
 label.config(text=string)  
 label.after(1000, time, label, description)  
  
def ramUsed():  
 return psutil.virtual\_memory().percent  
  
  
def ram(label, description):  
 string = description + str(ramUsed()) + '%'  
 label.config(text=string)  
 label.after(1000, ram, label, description)  
  
def cpuUsed():  
 yield psutil.cpu\_percent(interval=None, percpu=False)  
  
  
def cpu(label, description):  
 string = description + str(next(cpuUsed())) + '%'  
 label.config(text=string)  
 label.after(1000, cpu, label, description)  
  
  
  
  
componentUsaged(root=root, frame=frame1,  
 dataGener=time,  
 position='w',  
 description = 'ЧАС: ')  
  
componentUsaged(root=root, frame=frame1,  
 dataGener=ram,  
 position='w',  
 description='ОЗУ: ')  
  
componentUsaged(root=root, frame=frame1,  
 dataGener=cpu,  
 position='w',  
 description='ПРОЦЕСОР: ')  
  
  
for i in FreeSpace():  
 print(i)  
 lbl = ttk.Label(frame1, text=i)  
 lbl.pack(anchor="w")  
  
root.mainloop()

**Додаток Б**

main.py

import wmi  
from pefile import long  
import time  
import psutil  
  
c = wmi.WMI()  
  
# 1 список запущених процесів  
def runningProcesses():  
 for process in c.Win32\_Process():  
 yield process.ProcessId, process.Name, int(process.VirtualSize) / (1024\*1024)  
  
# 11 список всіх процесів із даним іменем  
def runningProcess(name = "notepad.exe"):  
 for process in c.Win32\_Process(name=name):  
 print(process.ProcessId, process.Name, process.VirtualSize / ())  
 print(process)  
  
# 2 відсоток вільного простору на дисках  
def FreeSpace():  
 for disk in c.Win32\_LogicalDisk (DriveType=3):  
 yield (disk.Caption + "%0.2f%% free" % (100.0 \* long(disk.FreeSpace) / long(disk.Size)))  
  
  
# 3 відкриває текстовий файл, після виводить його вміст  
def RunTXT(path = r"C:\temp\temp.txt"):  
 filename = path  
 process = c.Win32\_Process  
 process\_id, result = process.Create (CommandLine="notepad.exe " + filename)  
 watcher = c.watch\_for (  
 notification\_type="Deletion",  
 wmi\_class="Win32\_Process",  
 delay\_secs=1,  
 ProcessId=process\_id  
 )  
  
 watcher ()  
 print("This is what you wrote:")  
 print(open (filename).read ())  
  
  
# 4 Показати IP та MAC-адреси для мережевих інтерфейсів із підтримкою IP  
def showNetwork():  
 for interface in c.Win32\_NetworkAdapterConfiguration (IPEnabled=1):  
 yield interface.Description, interface.MACAddress, interface.IPAddress[0]

**Додаток В**

app\_table\_wiget.py

from ttkwidgets import Table  
import tkinter as tk  
from tkinter import ttk  
  
def tableFrame(root, frame, dataGener, columns):  
 root.columnconfigure(0, weight=1)  
 root.rowconfigure(0, weight=1)  
  
 style = ttk.Style(root)  
 style.theme\_use('clam')  
 style.configure(root,background='DarkGray')  
 sortable = tk.BooleanVar(root, False)  
  
  
 table = Table(frame, columns=columns, sortable=sortable.get(), height=30)  
 for col in columns:  
 table.heading(col, text=col)  
 table.column(col, width=300, stretch=False)  
  
 # sort column A content as int instead of strings  
 table.column(columns[0], type=int)  
  
 for i in dataGener():  
 table.insert('', 'end', iid=i,  
 values=i)  
  
 # toggle table properties  
 def toggle\_sort():  
 table.config(sortable=sortable.get())  
  
 tk.Checkbutton(frame, text='sortable', variable=sortable, command=toggle\_sort).pack(side='bottom')  
 table.pack(anchor=tk.W)

**Додаток Г**

componentUsaged.py

import tkinter as tk  
from tkinter import ttk  
  
def componentUsaged(root, frame, dataGener, position, description):  
 lbl = ttk.Label(frame, font=('Times New Romans', 22, 'bold'))  
 dataGener(lbl, description)  
 lbl.pack(anchor=position)