

ЗВІТ

із лабораторної роботи №2

з дисципліни "Технології програмування об'єктів лінгвістичної предметної галузі"

на тему: Реалізація графічного інтерфейсу для роботи з БД

Варіант №2-05

Мета

Метою лабораторної роботи №2 є створення графічного інтерфейсу для роботи з БД, використовуючи бібліотеки tkinter (власне для створення графічного інтерфейсу) та sqlite (взаємодія з БД в Python-програмах)

Середовище розробки

Мова програмування: Python

Назва та версія IDE: Visual Studio Code 1.78.2

Назва та версія ОС: Linux Mint 21.2 Cinnamon

Мова ОС: англійська

Дані з індивідуального варіанту:

Відмінок: pLl

Літера: G

Хід роботи

Створити новий графічний (віконний) проєкт мовою Python (через Tkinter). В заголовку форми вивести власне прізвище, ім'я та номер групи, номер ЛР.

```
In [18]: import tkinter as tk
from tkinter import ttk
import sqlite3

window = tk.Tk()
window.title("Кличлієв Кирило, група №2, ЛР №2")
```

Додати на форму:

- кнопку (Button);
- напис (Label);
- спадний список (ComboBox);
- таблицю для даних (TreeView або інші варіанти).

```
In [31]: button = ttk.Button(window, text="Отримати парадигми")
button.grid(row=0, column=2, padx=10, pady=10)

label = ttk.Label(window, text="Sample Label")
label.grid(row=0, column=0, padx=10, pady=10)

combo_var = tk.StringVar()
combo = ttk.Combobox(window, textvariable=combo_var)
combo.grid(row=1, column=0, padx=10, pady=10, sticky='n')

columns = ('ID', 'sgN', 'pLl')
tree = ttk.Treeview(window, columns=columns, show='headings')
tree.grid(row=1, column=1, columnspan=3, padx=10, pady=10)

tree.heading("ID", text="ID")
tree.heading("sgN", text="sgN")
tree.heading("pLl", text="pLl")
```

Скопіювати в папку з проєктом БД* pol_lab02.s3db зі словоформами іменників польської мови. Створити в проєкті під'єднання до БД, записавши у глобальну змінну шлях до файлу (абсолютний чи відносний).

```
In [38]: DATABASE_PATH = "pol_lab02.s3db"
conn = sqlite3.connect(DATABASE_PATH)
cursor = conn.cursor()
```

Перевірити з'єднання з БД, виконавши перший SQL-запит: знайти будь-яке одне слово в початковій формі (sgN) та вивести його в напис. Запит має виконуватись відразу при запуску програми, без додаткових дій користувача.

```
In [ ]: def get_sgn_word():
    query = "SELECT sgN FROM tnoun LIMIT 1"
    cursor.execute(query)
    result = cursor.fetchone()
    label.config(text=result[0])

get_sgn_word()
```

Завдання на максимальний бал: В п. 4 вивести слово, яке починається на літеру за інд. варіантом (літера G).

```
In [ ]: def get_word_starting_with_g():
    query = f"SELECT sgN FROM tnoun WHERE sgN LIKE 'g%' LIMIT 1"
    cursor.execute(query)
    result = cursor.fetchone()
    label.config(text=result[0])

get_word_starting_with_g()
```

Створити обробник натискання на кнопку, який виконує другий SQL-запит до БД і наповнює таблицю інформацією для будь-яких 12 слів із БД:

- індекс слова (id)
- називний відмінок однини (sgN)
- інший відмінок (за індивідуальним варіантом — див. нижче).

Завдання на максимальний бал: у першій колонці виводити не ID слова з відповідного поля в БД, а його порядковий номер у створеній таблиці на формі (1, 2, 3 і т.д.).

```
In [ ]: def fill_table():
    query = "SELECT id, sgN, pLl FROM tnoun LIMIT 12"
    cursor.execute(query)
    data = cursor.fetchall()
    for i, row in enumerate(data, start=1):
        tree.insert('', tk.END, values=(i, row[1], row[2]))

button.config(command=fill_table)
```

Завдання на максимальний бал: Вдосконалити наповнення таблиці з п. 6: якщо слово не має потрібної форми в БД, вивести в комірку прочерк (tire).

```
In [ ]: def fill_dashes():
    query = "SELECT id, sgN, pLl FROM tnoun LIMIT 12"
    cursor.execute(query)
    data = cursor.fetchall()
    for i, row in enumerate(data, start=1):
        other_case = row[2] if row[2] else "-"
        tree.insert("", i, values=(i, row[1], other_case))

button.config(command=fill_dashes)
```

Доповнити той же обробник натискання кнопки третім SQL-запитом: у спадний список мають завантажитись усі слова, які починаються на задану літеру (g) в початковій формі (sgN).

```
In [ ]: def load_tnoun_to_combo():
    query = f"SELECT sgN FROM tnoun WHERE sgN LIKE 'g%'"
    cursor.execute(query)
    tnoun = cursor.fetchall()
    combo['values'] = [word[0] for word in tnoun]

load_tnoun_to_combo()
```

```
In [ ]: window.mainloop()

conn.close()
```

Повний пайтонівський скрипт:

```
In [ ]: import tkinter as tk
from tkinter import ttk
import sqlite3

# 1
window = tk.Tk()
window.title("Кличлієв Кирило, група №2, ЛР №2")

# 2
button = ttk.Button(window, text="Отримати парадигми")
button.grid(row=0, column=2, padx=10, pady=10)

label = ttk.Label(window, text="Sample Label")
label.grid(row=0, column=0, padx=10, pady=10)

combo_var = tk.StringVar()
combo = ttk.Combobox(window, textvariable=combo_var)
combo.grid(row=1, column=0, padx=10, pady=10, sticky='n')

columns = ('ID', 'sgN', 'pLl')
tree = ttk.Treeview(window, columns=columns, show='headings')
tree.grid(row=1, column=1, columnspan=3, padx=10, pady=10)

tree.heading("ID", text="ID")
tree.heading("sgN", text="sgN")
tree.heading("pLl", text="pLl")

# 3
DATABASE_PATH = "pol_lab02.s3db"
conn = sqlite3.connect(DATABASE_PATH)
cursor = conn.cursor()

# 4
def execute_first_query():
    query = "SELECT sgN FROM tnoun LIMIT 1"
    cursor.execute(query)
    result = cursor.fetchone()
    label.config(text=result[0])

execute_first_query()

# 5
def execute_second_query():
    query = f"SELECT sgN FROM tnoun WHERE sgN LIKE 'g%' LIMIT 1"
    cursor.execute(query)
    result = cursor.fetchone()
    label.config(text=result[0])

execute_second_query()

# 6 + 7
def execute_third_query():
    query = "SELECT id, sgN, pLl FROM tnoun LIMIT 12"
    cursor.execute(query)
    data = cursor.fetchall()
    for i, row in enumerate(data, start=1):
        tree.insert('', tk.END, values=(i, row[1], row[2]))

button.config(command=execute_third_query)

# 8
def execute_fifth_query():
    query = "SELECT id, sgN, pLl FROM tnoun LIMIT 12"
    cursor.execute(query)
    data = cursor.fetchall()
    for i, row in enumerate(data, start=1):
        other_case = row[2] if row[2] else "-"
        tree.insert("", i, values=(i, row[1], other_case))

button.config(command=execute_fifth_query)

# 9
def load_tnoun_to_combo():
    query = f"SELECT sgN FROM tnoun WHERE sgN LIKE 'g%'"
    cursor.execute(query)
    tnoun = cursor.fetchall()
    combo['values'] = [word[0] for word in tnoun]

load_tnoun_to_combo()

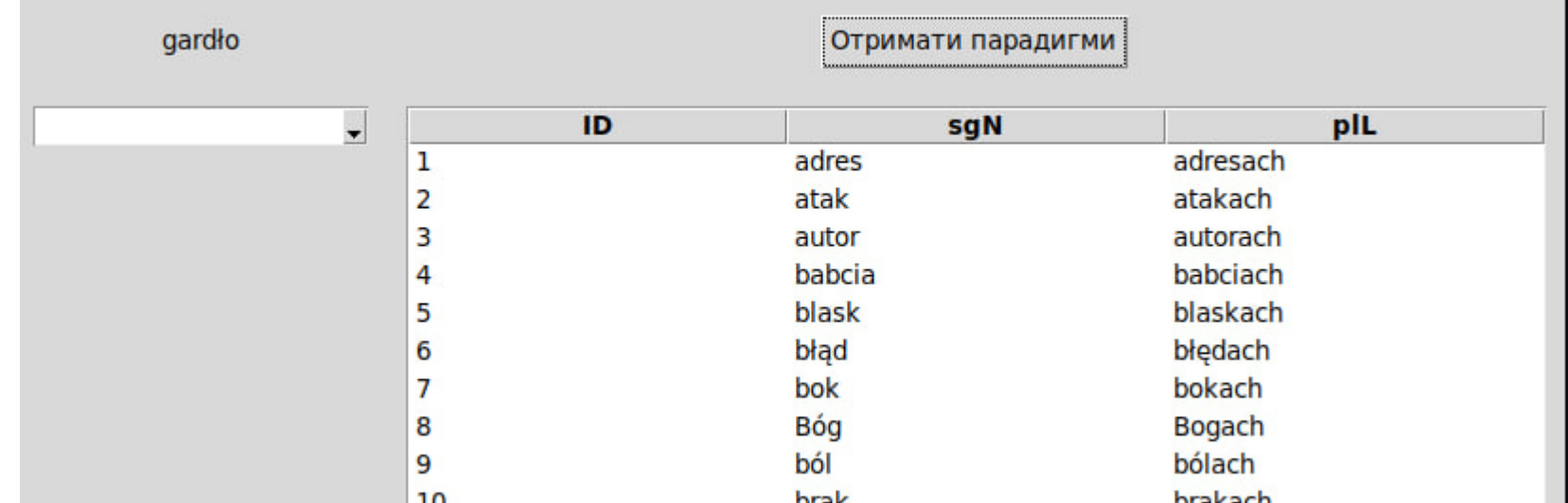
window.mainloop()

conn.close()
```

Результати роботи програми:

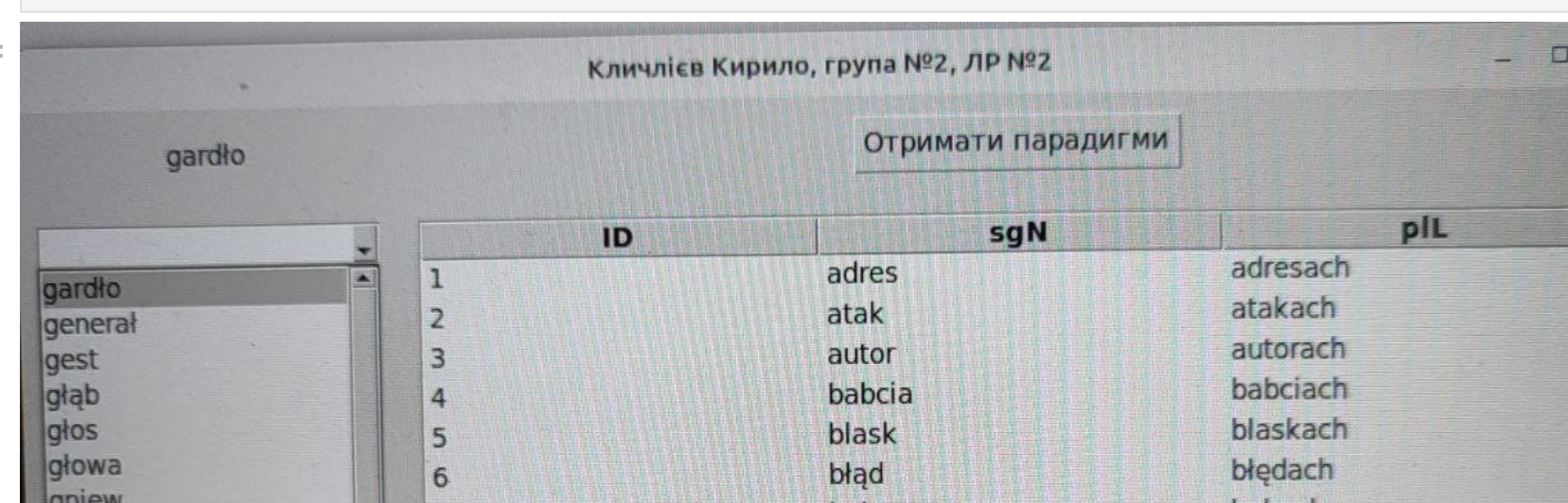
```
In [1]: from IPython import display
display.Image("images/photo-2023-11-11-00-18-47.jpg")
```

```
Out[1]:
```



```
In [2]: from IPython import display
display.Image("images/photo-2023-11-11-00-19-04.jpg")
```

```
Out[2]:
```



PS. Вставив фото, оскільки не можна зробити скрін на ноуті з відкритим спадним списком.

Висновки

У ході лабораторної роботи №2 було створено графічний інтерфейс для взаємодії з БД лінгвістичної інформації. Були реалізовані наступні віджети: напис, кнопка, випадний список, таблиця. Таблиця відображається при натисканні на кнопку.