**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**ĐỒ ÁN HỌC PHẦN**

TÊN HỌC PHẦN: **Lập trình Python (PYPRO)**

MÃ SỐ LỚP HP: **IPPA233277\_03**

Tên chủ đề: **LẬP TRÌNH PYTHON PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018**

**Họ tên sinh viên: NGUYỄN DIỆU HƯƠNG [29]**

**Mã số sinh viên: 21110489**

**Lớp: 211102D [CQ]**

**Ngày nộp: ..09../..05../2023**

**TP.HCM, ngày 09 Tháng 05 năm 2023**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI ĐỒ ÁN HỌC PHẦN**

TÊN HỌC PHẦN: **Lập trình Python (PYPRO)**

MÃ SỐ LỚP HP: **IPPA233277\_03**

Tên đề tài: **LẬP TRÌNH PYTHON PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018**

**Giảng viên giảng dạy: VÕ XUÂN THỂ**

**Họ tên sinh viên: NGUYỄN DIỆU HƯƠNG [29]**

**Mã số sinh viên: 21110489**

**Lớp: 211102D [CQ]**

**TÊN CÁC FILE SẢN PHẨM ĐỀ TÀI:**

**Tên sản phẩm đề tài: G329Huong\_DAHP.PyPro\_CO2Emissions.rar**

**Tập tin mã nguồn: G329Huong\_CO2Emissions.py**

**Tập dữ liệu thực nghiệm: G329Huong\_CO2Emissions.csv**

**Thư mục dữ liệu thực nghiệm: CSV29\_Folder**

**Giới thiệu thư mục bài làm: G329Huong\_DAHP.Intro.docx**

**Tập các thư viện sử dụng:** # Speech: speech\_recognition, gtts, playsound

# GUI: tkinter, os, datetime

# OS: os

# EDA: pandas, numpy, scipy, sklearn

# Graph: matplotlib

**NHẬN XÉT CỦA GIẢNG VIÊN**

**………………………………………………………..**

**………………………………………………………..**

**Ngày…. / …. / 202…**

**Giảng viên: Ký tên**

**TP.HCM, ngày 09 tháng 5 năm 2023**

**LỜI CẢM ƠN**

Em xin cam kết đồ án học phần này hoàn toàn do em thực hiện, không sao chép của các bạn sinh viên hay các nguồn khác.

Em đã nghiên cứu kỹ đồ án học phần và thực hiện kiểm tra nhiều lần, song không tránh khỏi sai sót. Mong nhận được sự góp ý của thầy để em có thể khắc phục và hoàn thiện hơn đồ án này.

Cuối cùng, em không thể hoàn thành đồ án này nếu không có sự giúp đỡ của thầy Võ Xuân Thể. Em chân thành cảm ơn sự giúp đỡ của thầy trong thời gian qua.

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| CSDL hoặc DB | Cơ sở dữ liệu: DataBase |
| NSD = Users = Account | Người Sử Dụng = là quyền làm việc trên WebApp được đảm bảo bởi tối thiểu là 2 yếu tố: tên đăng nhập (UserName) và mật khẩu (Password) |
|  |  |
| AI | Artificial intelligence: Trí tuệ nhân tạo |
| EDA | Exploratory Data Analysis: Phân tích Khám phá Dữ liệu (Còn gọi là: phân tích dữ liệu thăm dò) |
| GUI | Graphical User Interface: Giao diện đồ hoạ |
| PYPRO | Lập trình Python |

**DANH MỤC CÁC THUẬT NGỮ**

|  |  |
| --- | --- |
| Giao diện người dùng:  User-Interface | Là hệ thống các màn hình giao tiếp cho phép người sử dụng tương tác với các thành phần phần mềm trong HTTT, thường là 1 trong các dong: Win-form, Web-form, Mobile-Form. |
| Tài khoản (Account) | Là một quyền làm việc trên HTTT được cấp phát cho một cá nhân thông qua tên tài khoản (username) và mật khẩu (password). |
| Trợ lý ảo: Voice Assistant | Là các hệ thống có khả năng “nghe” và “nói” với con người, nhờ đó hỗ trợ con người trong một số chức năng như một “trợ lý”. |

**MỤC LỤC**

[**LỜI CẢM ƠN** 3](#_Toc135267560)

[DANH MỤC CÁC TỪ VIẾT TẮT 4](#_Toc135267561)

[**DANH MỤC CÁC THUẬT NGỮ** 4](#_Toc135267562)

[Chương 1: GIỚI THIỆU VỀ ĐỒ ÁN HỌC PHẦN 8](#_Toc135267563)

[1.1. Tổng quan về ĐỒ ÁN HỌC PHẦN 8](#_Toc135267564)

[1.2. Nội dung chuyên môn chính của ĐỒ ÁN HỌC PHẦN 8](#_Toc135267565)

[1.3. Công cụ và nền tảng kỹ thuật thực hiện ĐỒ ÁN HỌC PHẦN 8](#_Toc135267566)

[1.4. Bố cục của báo cáo 9](#_Toc135267567)

[Chương 2: CƠ SỞ LÝ THUYẾT CỦA HỌC PHẦN 10](#_Toc135267568)

[2.1. GIỚI THIỆU VỀ LẬP TRÌNH PYTHON 10](#_Toc135267569)

[2.1.1. Giới thiệu chung 10](#_Toc135267570)

[2.1.2. Cài đặt “bộ thảo chương” (soạn thảo chương trình IDE) và “bộ dịch” 10](#_Toc135267571)

[2.1.3. Giới thiệu các bộ tương tác lập trình 10](#_Toc135267572)

[2.1.4. Cấu hình mô trường tương tác lập trình 10](#_Toc135267573)

[2.2. TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PYTHON 10](#_Toc135267574)

[2.2.1. Danh hiệu (Identifier) 10](#_Toc135267575)

[2.2.2. Về viết mã lệnh lập trình (Code) 10](#_Toc135267576)

[2.2.3. Về chú giải (Comment) 10](#_Toc135267577)

[2.2.4. Các phép toán 10](#_Toc135267578)

[2.2.5. Chuỗi 10](#_Toc135267579)

[2.2.6. Lệnh 10](#_Toc135267580)

[2.2.7. Định nghĩa hàm 10](#_Toc135267581)

[2.2.8. Danh sách 10](#_Toc135267582)

[2.2.9. Bộ dữ liệu: Tuple 10](#_Toc135267583)

[2.2.10. Tập hợp: Set 10](#_Toc135267584)

[2.2.11. Từ điển: Dictionary 10](#_Toc135267585)

[2.2.12. Module 10](#_Toc135267586)

[2.2.13. Package 10](#_Toc135267587)

[2.2.14. Lớp 10](#_Toc135267588)

[2.2.15. Xử lý ngoại lệ 10](#_Toc135267589)

[2.3. GIỚI THIỆU CÁC THƯ VIỆN VÀ CÔNG CỤ CĂN BẢN 10](#_Toc135267590)

[2.3.1. Nạp thư viện 10](#_Toc135267591)

[2.3.2. Pandas 10](#_Toc135267592)

[2.3.3. Tensorflow 10](#_Toc135267593)

[2.3.4. NumPy 10](#_Toc135267594)

[2.3.5. SCIPY 10](#_Toc135267595)

[2.3.6. Theano 10](#_Toc135267596)

[2.3.7. Matplotlib 10](#_Toc135267597)

[2.3.8. scikit-learn = sklearn 11](#_Toc135267598)

[2.3.9. Keras 11](#_Toc135267599)

[2.3.10. NLTK 11](#_Toc135267600)

[2.3.11. PyTorch 11](#_Toc135267601)

[2.3.12. LightGBM 11](#_Toc135267602)

[2.3.13. Eli5 11](#_Toc135267603)

[2.4. LẬP TRÌNH PYTHON XỬ LÝ GIỌNG NÓI (Trợ lý ảo: Voice Assistant) 11](#_Toc135267604)

[2.4.1. Giới thiệu chủ đề 11](#_Toc135267605)

[2.4.2. Tài liệu và nguồn thực nghiệm 11](#_Toc135267606)

[2.4.3. Giới thiệu Các thư viện Python liên quan 11](#_Toc135267607)

[2.4.4. Giới thiệu bài toán 11](#_Toc135267608)

[2.4.5. Giới thiệu code tham khảo 11](#_Toc135267609)

[2.5. KỸ THUẬT LẬP TRÌNH ĐỆ QUY : THÁP HÀ NỘI (HANOI TOWER) 11](#_Toc135267610)

[2.6. LẬP TRÌNH PHÂN TÍCH KHÁM PHÁ (EDA) [Thăm dò] 11](#_Toc135267611)

[Chương 3: PHÂN TÍCH VÀ XÁC ĐỊNH CÁC CƠ SỞ KỸ THUẬT 12](#_Toc135267612)

[3.1. GIỚI THIỆU CHUNG VỀ CHỦ ĐỀ 12](#_Toc135267613)

[3.1.1. Mô tả yêu câu của chủ đề 12](#_Toc135267614)

[3.2. CÁC NỀN TẢNG KỸ THUẬT 12](#_Toc135267615)

[3.2.1. Hệ thống thư viện sử dụng 12](#_Toc135267616)

[3.2.2. Tập dữ liệu thực nghiệm 13](#_Toc135267617)

[3.2.3. Mô tả giải thuật 13](#_Toc135267618)

[3.3. LẬP TRÌNH XỬ GIỌNG NÓI (TRỢ LÝ ẢO: VOICE ASSISTANT) 14](#_Toc135267619)

[3.3.1. Giới thiệu chung về chủ đề 14](#_Toc135267620)

[3.3.1.1. Mô tả yêu cầu của chủ đề 14](#_Toc135267621)

[3.3.1.2. Phân tích yều cầu của chủ đề 14](#_Toc135267622)

[3.3.2. Các nền tảng kỹ thuật 15](#_Toc135267623)

[3.3.2.1. Nền tảng kỹ thuật cơ bản 15](#_Toc135267624)

[3.3.2.2. Hệ thống thư việc sử dụng 15](#_Toc135267625)

[3.3.3. Thiết kế giao diện 15](#_Toc135267626)

[3.3.4. Mã lệnh lập trình (python) 15](#_Toc135267627)

[Chương 4: LẬP TRÌNH PYTHON PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018 16](#_Toc135267628)

[4.1. Giới thiệu chung 16](#_Toc135267629)

[4.1.1. Các chức năng chính 16](#_Toc135267630)

[4.1.2. Các giao diện 16](#_Toc135267631)

[4.2. Mã lệnh lập trình (python) 21](#_Toc135267632)

[4.2.1. Các nền tảng kỹ thuật 21](#_Toc135267633)

[4.2.2. Full Code 29](#_Toc135267634)

[Chương 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỒ ÁN HỌC PHẦN 39](#_Toc135267635)

[5.1. Kết luận 39](#_Toc135267636)

[5.1.1. Những kết quả đạt được 39](#_Toc135267637)

[5.1.2. Hạn chế 39](#_Toc135267638)

[5.2. Hướng phát triển 39](#_Toc135267639)

[5.3. Hướng khắc phục hạn chế 39](#_Toc135267640)

[5.4. Hướng mở rộng ĐỒ ÁN HỌC PHẦN 39](#_Toc135267641)

[DANH MỤC TÀI LIỆU THAM KHẢO 40](#_Toc135267642)

# GIỚI THIỆU VỀ ĐỒ ÁN HỌC PHẦN

## Tổng quan về ĐỒ ÁN HỌC PHẦN

Đồ án Học phần **Lập trình Python Phân tích thăm dò mức độ phát sinh** **lượng khí thải CO2 của các nước trên thế giới từ năm 1990-2018**: đề tài vận dụng kiến thức và kỹ năng về Lập trình Python để xây dựng giao diện đồ hoạ (GUI) cho phép: tương tác với người dùng bằng giọng nói (Trợ lý ảo: Voice Assistant) và phân tích thăm dò (EDA: Epolore Data Analysing) đối với tập dữ liệu thực nghiệm về Dữ liệu của 266 nước trên thế giới qua các năm (1990-2018).

## Nội dung chuyên môn chính của ĐỒ ÁN HỌC PHẦN

Bao gồm 5 nội dung lập trình chính dùng ngôn ngữ lập trình Python:

+ Trợ lý ảo

+ Phân tích dữ liệu khám phá || thăm dò

+ Giao diện đồ họa dạng WinForm

+ Game

+ Plot: data visualization

## Công cụ và nền tảng kỹ thuật thực hiện ĐỒ ÁN HỌC PHẦN

+ Ngôn ngữ lập trình **Python**

+ Công cụ lập trình và biên dịch: **Spyder (Anaconda3)**

+ Các thư viện chính:

# Voice

import speech\_recognition as sr

from gtts import gTTS

import playsound

# thư viện cần thiết EDA

import numpy as np # thư viện về đại số tuyến tính

import pandas as pd # thư viện xử lý

from sklearn import preprocessing # thư viện tiền xử lý

from scipy import stats # thư viện cung cấp công cụ thống kê

from sklearn.preprocessing import Binarizer

from sklearn.feature\_selection import SelectKBest, chi2

from sklearn.linear\_model import LinearRegression #sử dụng hồi quy tuyến tính

# tkinter

import tkinter as tk

from tkinter import ttk

from tkinter import messagebox

from tkinter import filedialog

# Thư viện thời gian thực

from datetime import datetime

# Thư viện vẽ đồ thị

import matplotlib.pyplot as plt

# Thư viện chạy các file bên ngoài

import subprocess

# thư viện os

import os

import time

#thư viện game

import subprocess

## Bố cục của báo cáo

Báo cáo gồm những nội dung như sau:

Chương 1: Giới thiệu đồ án của Học phần

Chương 2: Các cơ sở lý thuyết của Học phần phục việc thực hiện đề tài

Chương 3: Phân tích và xác đinh các cơ sở kỹ thuật thực hiện đề tài

Chương 4: Lập trình xây dựng sản phầm đề tài: LẬP TRÌNH PYTHON PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018

Chương 5 Tổng kết các kết quản đạt được và còn hạn chế của đồ án, đồng thời đề xuất hướng khắc phục hạn chế và phát triển Đồ án.

# CƠ SỞ LÝ THUYẾT CỦA HỌC PHẦN

## GIỚI THIỆU VỀ LẬP TRÌNH PYTHON

### Giới thiệu chung

### Cài đặt “bộ thảo chương” (soạn thảo chương trình IDE[[1]](#footnote-2)) và “bộ dịch”

### Giới thiệu các bộ tương tác lập trình

### Cấu hình mô trường tương tác lập trình

## TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PYTHON

### Danh hiệu (Identifier)

### Về viết mã lệnh lập trình (Code)

### Về chú giải (Comment)

### Các phép toán

### Chuỗi

### Lệnh

### Định nghĩa hàm

### Danh sách

### Bộ dữ liệu: Tuple

### Tập hợp: Set

### Từ điển: Dictionary

### Module

### Package

### Lớp

### Xử lý ngoại lệ

## GIỚI THIỆU CÁC THƯ VIỆN VÀ CÔNG CỤ CĂN BẢN

### Nạp thư viện

### Pandas

### Tensorflow

### NumPy

### SCIPY

### Theano

### Matplotlib

### scikit-learn = sklearn

### Keras

### NLTK

### PyTorch

### LightGBM

### Eli5

## LẬP TRÌNH PYTHON XỬ LÝ GIỌNG NÓI (Trợ lý ảo: Voice Assistant)

### Giới thiệu chủ đề

### Tài liệu và nguồn thực nghiệm

### Giới thiệu Các thư viện Python liên quan

### Giới thiệu bài toán

### Giới thiệu code tham khảo

**[1] Nghe tiếng Việt => Text**

**[2] Text => đọc tiếng Việt**

## KỸ THUẬT LẬP TRÌNH ĐỆ QUY : THÁP HÀ NỘI (HANOI TOWER)

## LẬP TRÌNH PHÂN TÍCH KHÁM PHÁ (EDA) [Thăm dò]

1. **PHÂN TÍCH VÀ XÁC ĐỊNH CÁC CƠ SỞ KỸ THUẬT**

## GIỚI THIỆU CHUNG VỀ CHỦ ĐỀ

### Mô tả yêu câu của chủ đề

Vận dụng kỹ thuật lập trình EDA kết hợp với GUI và Voice Assistant cho phép dùng tập dữ liệu thực nghiệm về lượng khí thải của 266 nước trên thế giới qua các năm và dự đoán năm 2018.

Dữ liệu thực nghiệm gồm các giá trị liên quan tới các thông số lượng khí thải của 266 nước trên thế giới qua các năm :

G329Huong\_CO2Emissions.csv dạng file Excel.csv

. Số records: 266

. Số thuộc tính input: 30 để đoán thuộc tính thứ 31: 2019

Phân tích yều cầu của chủ đề

+ Chủ đề lập trình python này là cơ sở của các ứng dụng về DataWarehouse, Bigdata, Data Analysis, Data Sciences, . . .

+ Đây là xử lý nền tảng cho các nội dung chuyên môn (nêu trên) chủ yếu là giai đoạn tiền xử lý dữ liệu đầu vào (Input Preporcessing), như: xử lý NULL, lọc dữ liệu, tích hợp dữ liệu (Integration), …

**Một số trường hợp tiền xử lý dữ liệu đầu vào (input) của các hệ thống EDA**

[1] Xử lý NULL là xử lý các giá trị bị trống, làm ảnh hưởng tới quá trình tính toán số liệu.

[2] Xử lý dữ liệu quá khác biệt = “ngoại lệ” (isolated): là loại bỏ các giá trị có z-score quá chênh lệch so với các bản ghi khác trong dataset.

## CÁC NỀN TẢNG KỸ THUẬT

### Hệ thống thư viện sử dụng

**Bước 1: Nhập các thư viện cần thiết**

# Voice

import speech\_recognition as sr

from gtts import gTTS

import playsound

# thư viện cần thiết EDA

import numpy as np # thư viện về đại số tuyến tính

import pandas as pd # thư viện xử lý

from sklearn import preprocessing # thư viện tiền xử lý

from scipy import stats # thư viện cung cấp công cụ thống kê

from sklearn.preprocessing import Binarizer

from sklearn.feature\_selection import SelectKBest, chi2

# tkinter

import tkinter as tk

from tkinter import ttk

from tkinter import messagebox

from tkinter import filedialog

# Thư viện thời gian thực

from datetime import datetime

# Thư viện vẽ đồ thị

import matplotlib.pyplot as plt

# Thư viện chạy các file bên ngoài

import subprocess

# thư viện os

import os

#thư viện game

import subprocess

### Tập dữ liệu thực nghiệm

**Bước 2: Tải tập dữ liệu**

#Load the data set (Nạp tập dữ liệu)

df = pd.read\_csv('./G329Huong\_CO2Emissions.csv')

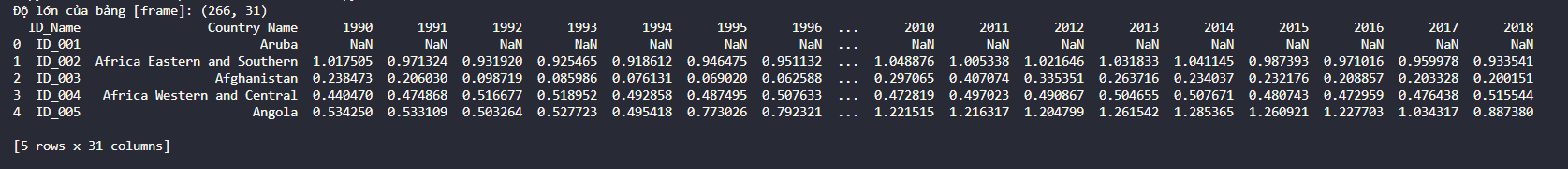
#Display the shape of the data set (Vẽ biểu đồ tập dữ liệu)

print('Độ lớn của bảng [frame]:', df.shape)

#Display data (Hiển thị dữ liệu dạng mảng 5 dòng đầu)

print(df[0:5])

Chạy python G329Huong\_CO2Emissions.py hiển thị kết quả:



### Mô tả giải thuật

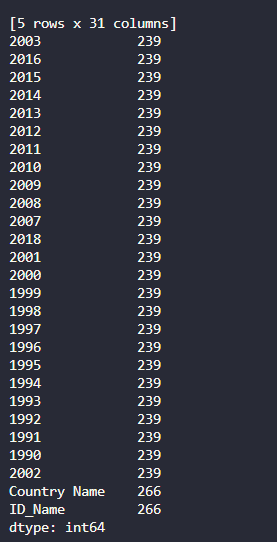
**Bước 3: Xử lý dữ liệu**

# Checking for null values (Kiểm tra giá trị null = đếm số dòng có dữ liệu ứng từng thuộc tính)

print(df.count().sort\_values())

**Kết quả:**

* Có cột dữ liệu Null



## LẬP TRÌNH XỬ GIỌNG NÓI (TRỢ LÝ ẢO: VOICE ASSISTANT)

## Giới thiệu chung về chủ đề

### Mô tả yêu cầu của chủ đề

+ Lập trình App xử lý "lời nới" (Speech): Ứng dụng “Trợ lý ảo” (Voice Assistant)

+ Gồm 2 phần (Part) cơ bản sau

1. Nghe tiếng Việt

2. Trả lời bằng tiếng Việt: Text => Nói tiếng Việt

### Phân tích yều cầu của chủ đề

+ Chủ đề lập trình python này là cơ sở của các ứng dụng về AI, ML, Deep ML.

+ Các hệ thống này thường có 3 phần :

Part1: Speech….: chính là part 1 của chủ đề này

Part2: Các kỹ thuật AI

Part2: Voice….: chính là part 2 của chủ đề này

## Các nền tảng kỹ thuật

### Nền tảng kỹ thuật cơ bản

### Hệ thống thư việc sử dụng

B1: Cài đặt thư viện

conda install …. Các thư viện sau:

------------------gốc hướng dẫn / dùng python trong visual (pip3)

pip3 install SpeechRecognition

*pip3 install SpeechRecognition pydub # Long audio source*

pip3 install pyaudio

pip3 install gtts

pip3 install playsound==1.2.2

B2 : Nạp thư viện vào App

# Voice

import speech\_recognition as sr

from gtts import gTTS

import playsound

## Thiết kế giao diện

+ Xử lý trong cửa sổ Console Output của Spyder

## Mã lệnh lập trình (python)

**FULL CODE**

# B1: NẠP THƯ VIỆN

# Voice

import speech\_recognition as sr

from gtts import gTTS

import playsound

# B2: CHỌN PHƯƠNG ÁN NHẬP ÂM THANH TỪ MICROPHONE

def VoiceAssistant():

    vlenh=""

    label=tk.Label(foot,text=vlenh)

    label.place(x=800, y=20)

    label.destroy()

    r = sr.Recognizer()

    with sr.Microphone() as Source:

        messagebox.showinfo("Cảnh báo", "Hiệu chỉnh nhiễu trước khi nói!")

        r.adjust\_for\_ambient\_noise(Source, duration=1)

        messagebox.showinfo("Sẵn sàng", "Bấm OK để bắt đầu nói tiếng Việt trong 3 giây")

        audio\_data = r.record(Source, duration=3)

        try:

            vlenh = r.recognize\_google(audio\_data, language="vi")

            messagebox.showinfo("Bạn đã nói: ", vlenh)

        except:

            vlenh = "Bạn nói gì tôi nghe không hiểu!"

        label=tk.Label(foot,text=vlenh)

        label.place(x=800, y=20)

        ten = gTTS(text=vlenh, lang='vi')

        # Xuất ra lời nói theo văn bản đã nhập

        filename =os.path.abspath('')+'\huong29.mp3'

        ten.save(filename)

1. **LẬP TRÌNH PYTHON PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018**

## Giới thiệu chung

* + 1. Các chức năng chính

- Mở file CSV: chọn file csv bất kỳ trong máy để xử lý

- Tiền xử lý: xử lý dòng cột, các giá trị Null hoặc quá nhiều số 0, xử lý cá biệt, tính toán z-score

- Đồ thị z-score

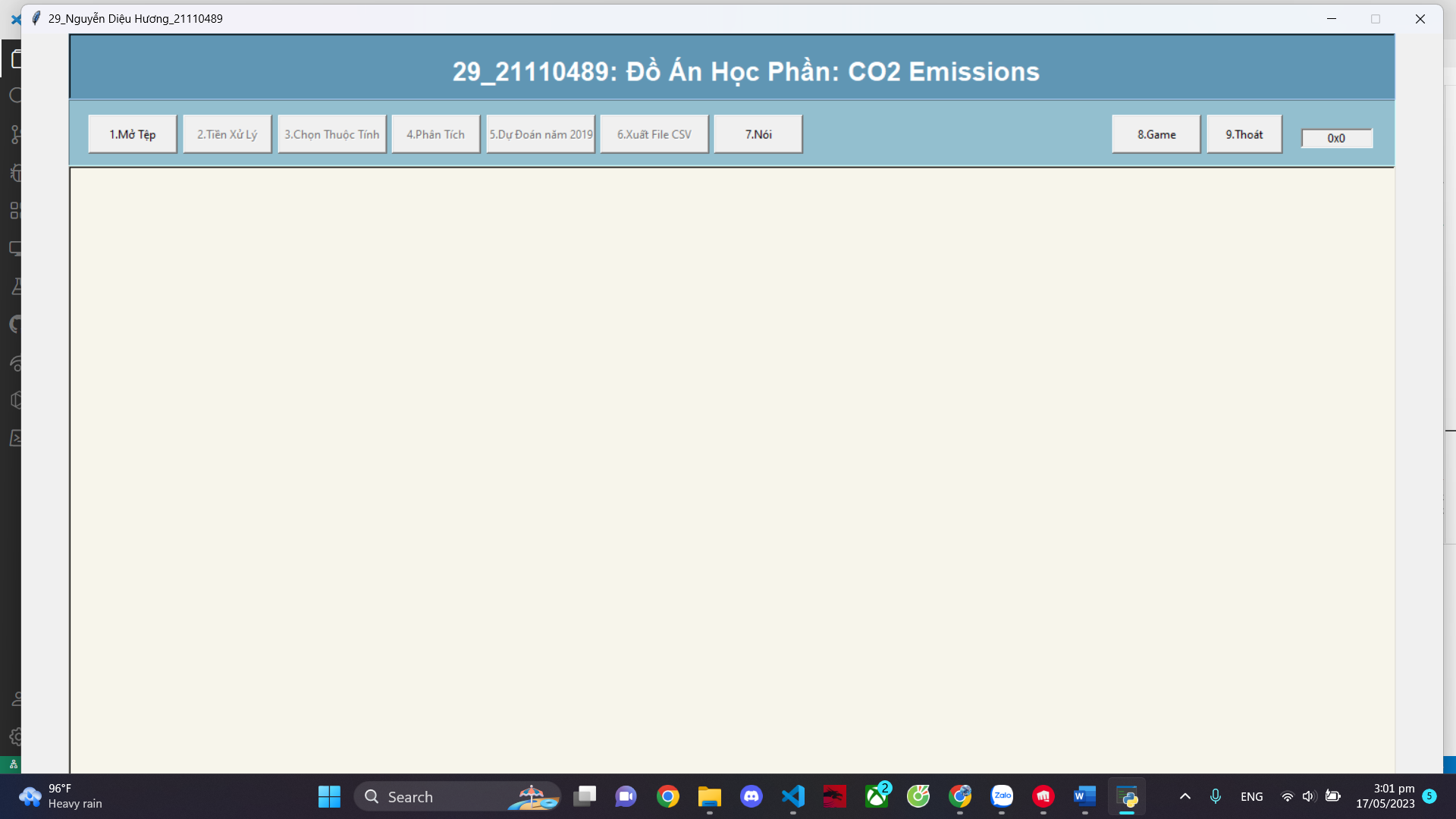
- Phân tích số liệu: so sánh số liệu năm gần nhất ‘2018’ với trung bình các cột được chọn phân tích

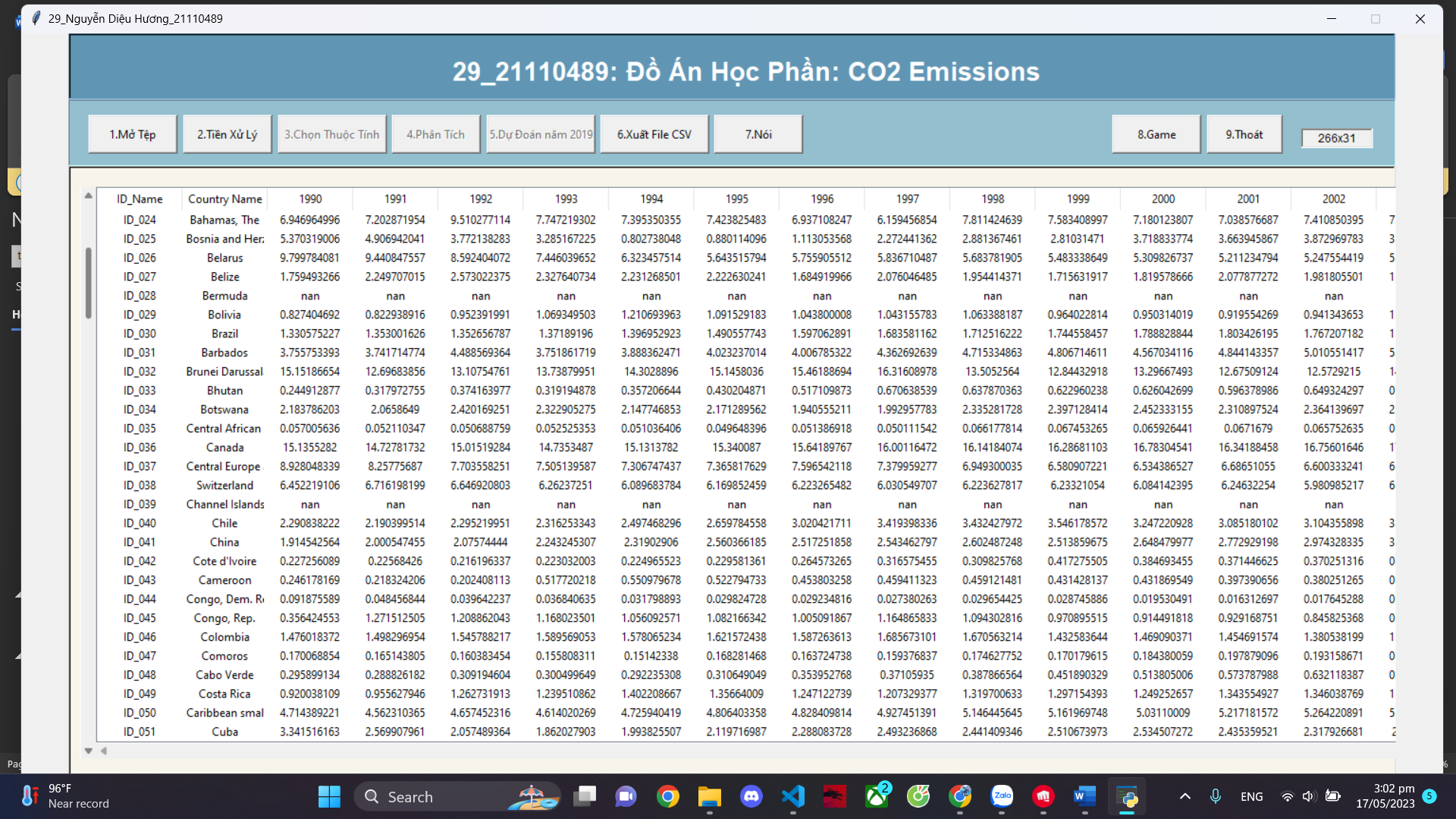
- Dự đoán số liệu: tạo thuộc tính mới 2019 để dự đoán số liệu năm này, sử dụng hàm hồi quy tuyến tính của thư viện sklearn: from sklearn.linear\_model import LinearRegression

- Giọng nói

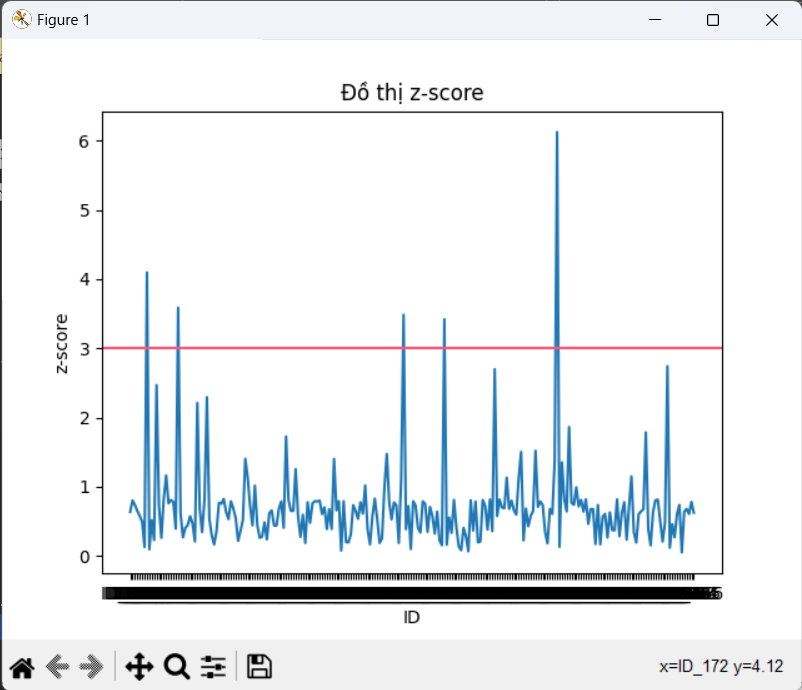
- Game

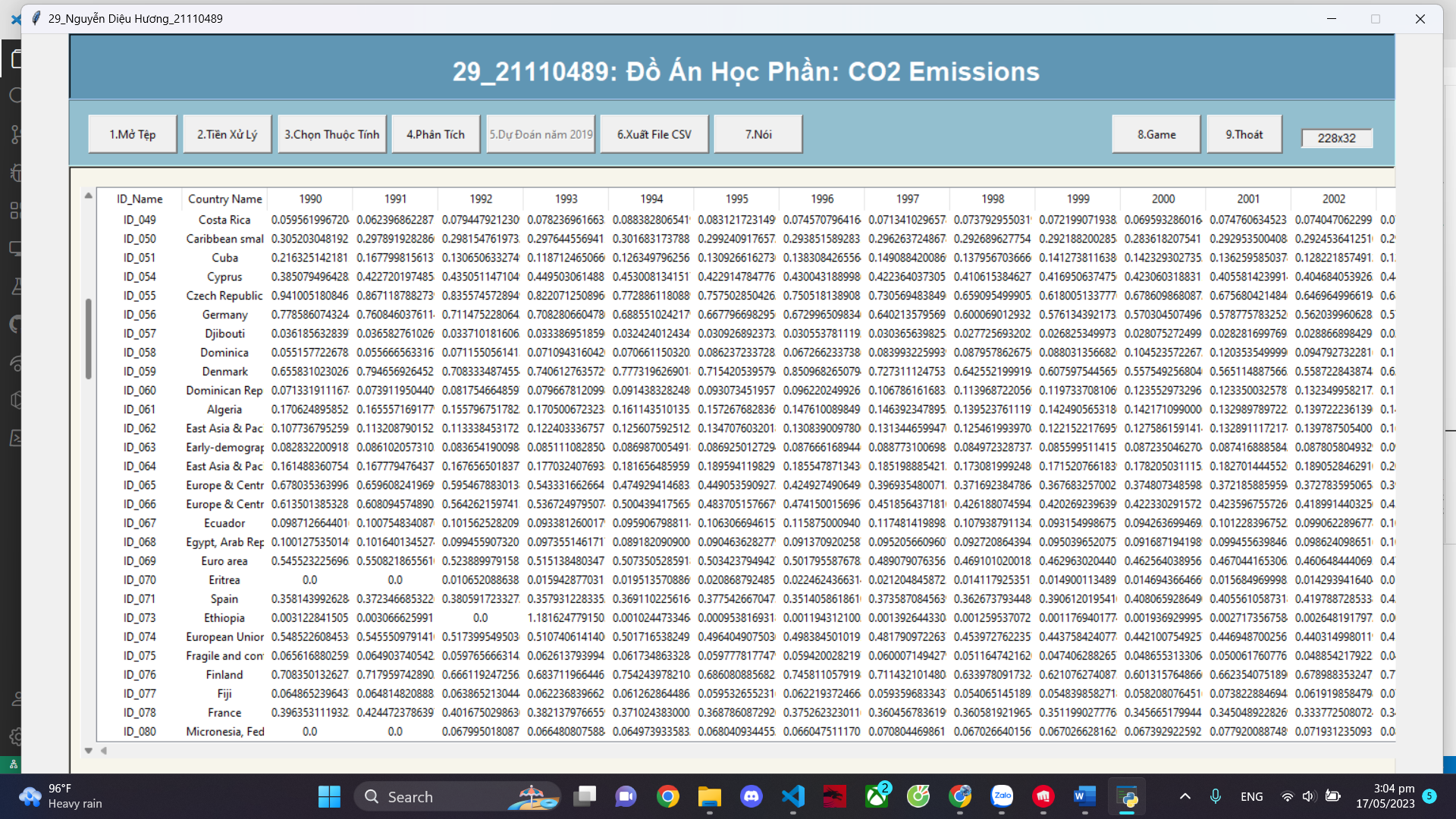
* + 1. Các giao diện
* Giao diện chính



* Giao diện của chức năng “Mở tệp”
* Giao diện của chức năng tiền xử lý và đồ thị z-score

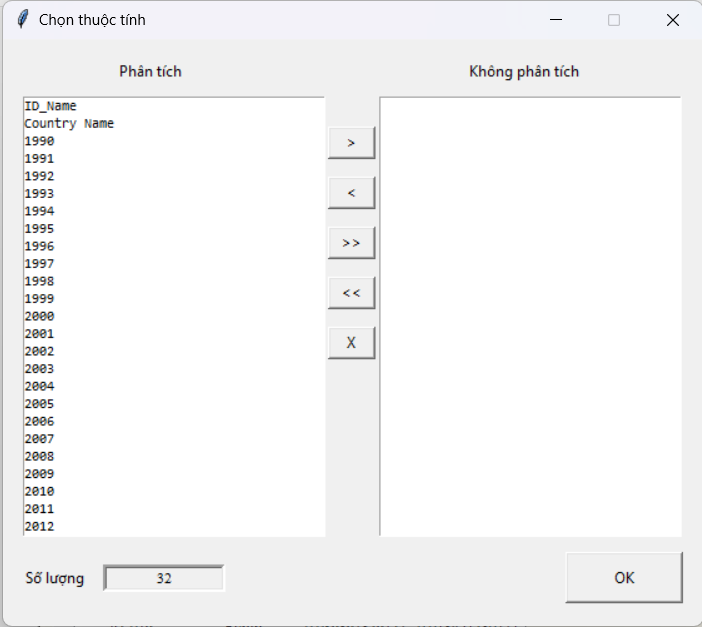
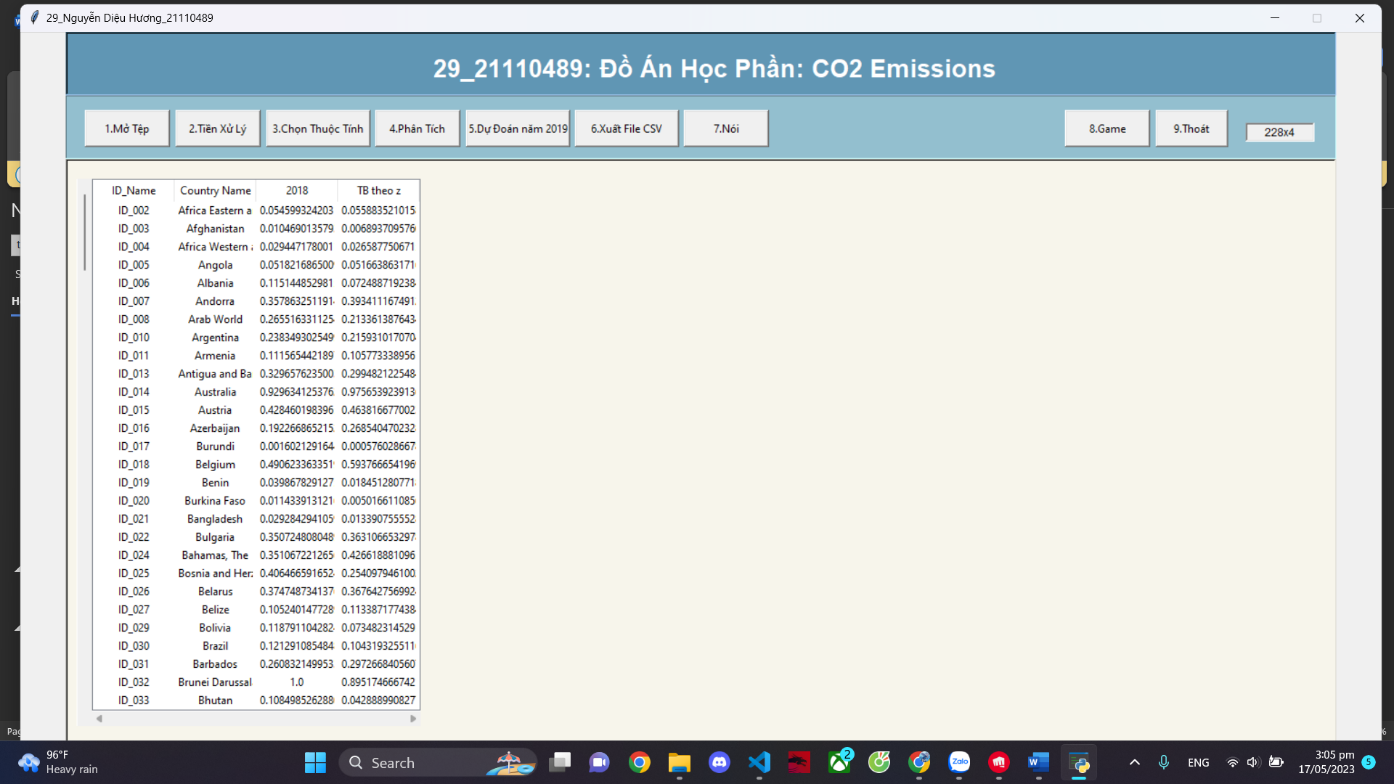
+ Đồ thị z-score



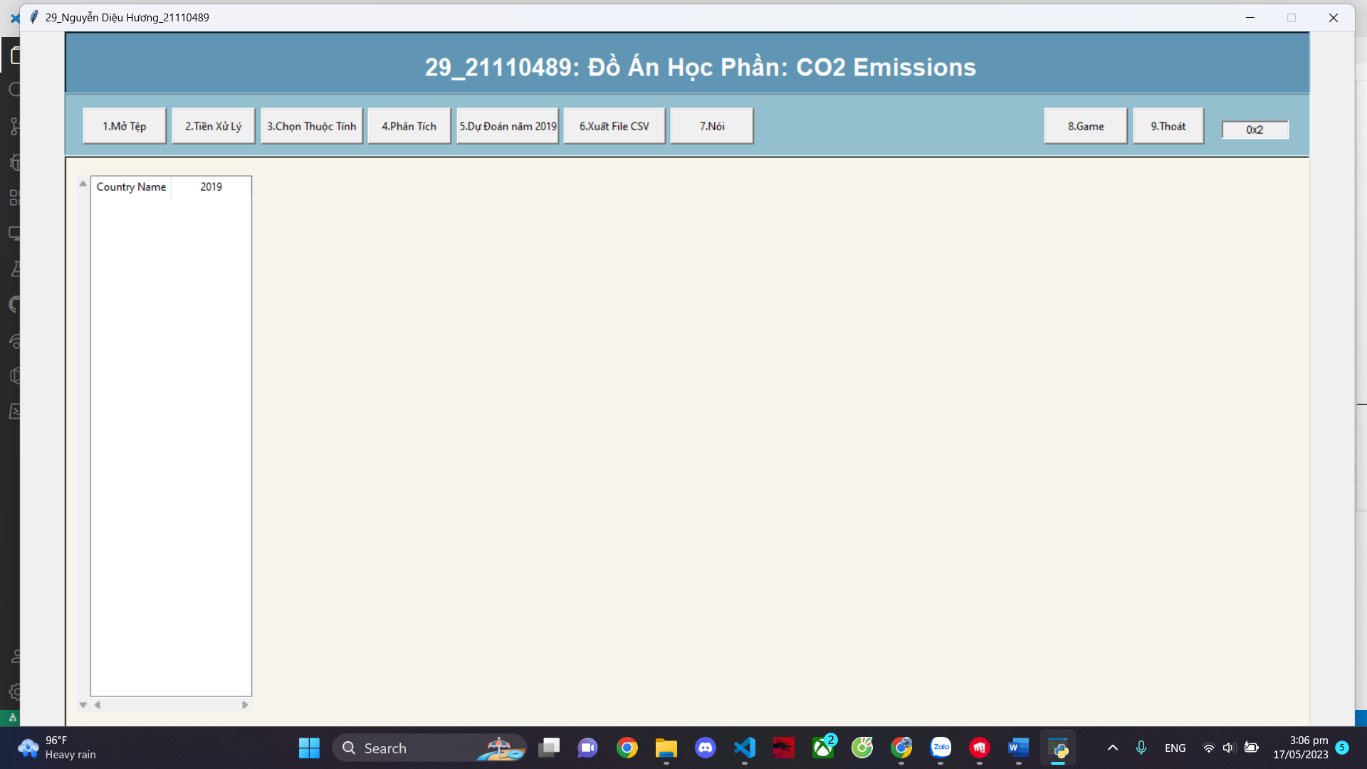
+ Bảng số liệu tiền xử lí

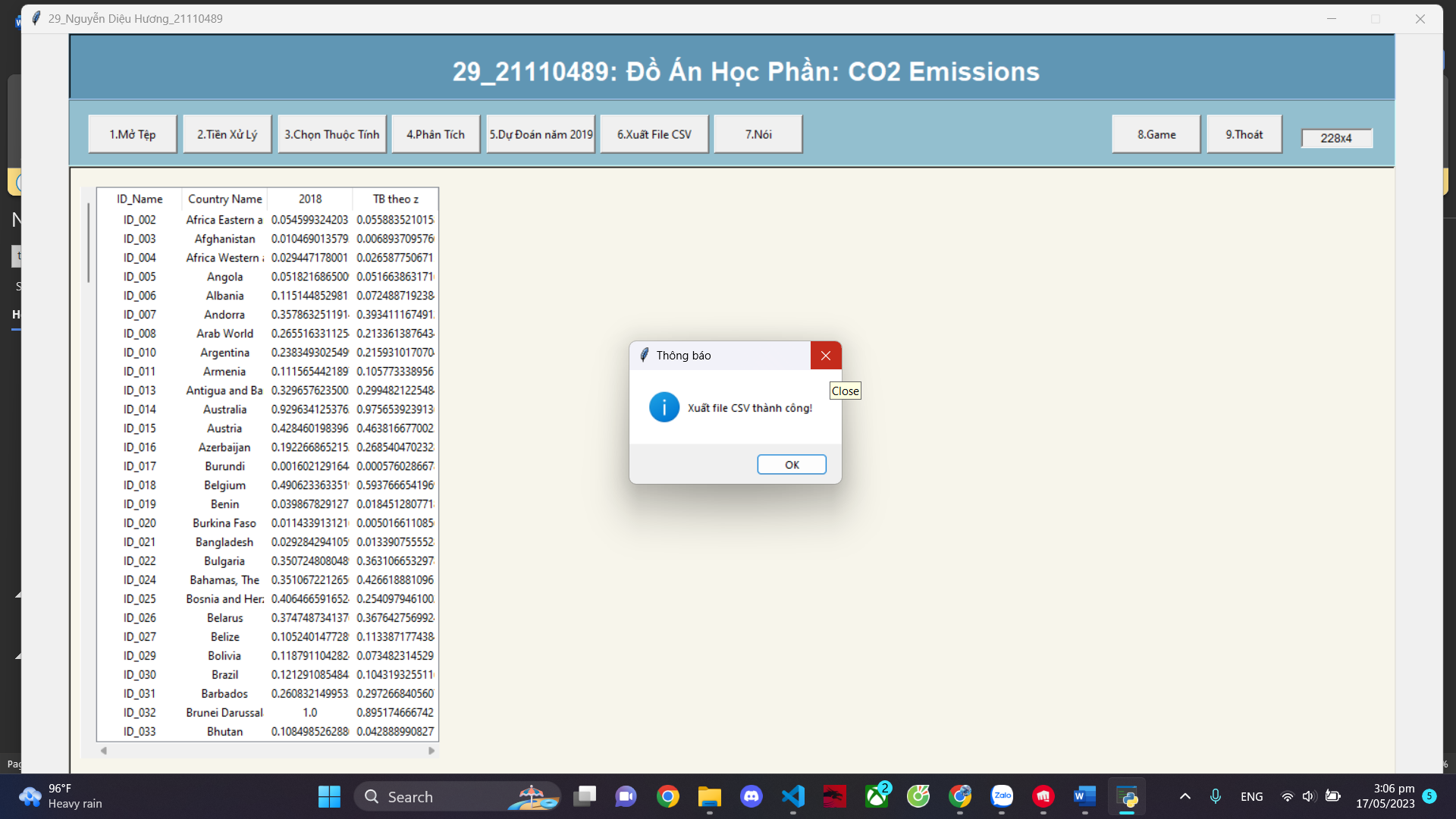
* Giao diện phân tích số liệu

+ Chọn thuộc tính phân tích: chọn các năm để tính toán so sánh: Không chọn các cột ID\_Name, Country Name, 2018, 2019

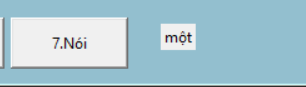


+ Phân tích sau khi chọn

* Giao diện của chức năng “Dự đoán năm 2019”
* Chức năng “Xuất file CSV”



* Chức năng “Nói”



* Chức năng “Game”



## Mã lệnh lập trình (python)

* + 1. Các nền tảng kỹ thuật
* ***a. Hàm Voice***

def VoiceAssistant():

    vlenh=""

    label=tk.Label(foot,text=vlenh)

    label.place(x=800, y=20)

    label.destroy()

    r = sr.Recognizer()

    with sr.Microphone() as Source:

        messagebox.showinfo("Cảnh báo", "Hiệu chỉnh nhiễu trước khi nói!")

        r.adjust\_for\_ambient\_noise(Source, duration=1)

        messagebox.showinfo("Sẵn sàng", "Bấm OK để bắt đầu nói tiếng Việt trong 3 giây")

        audio\_data = r.record(Source, duration=3)

        try:

            vlenh = r.recognize\_google(audio\_data, language="vi")

            messagebox.showinfo("Bạn đã nói: ", vlenh)

        except:

            vlenh = "Bạn nói gì tôi nghe không hiểu!"

        label=tk.Label(foot,text=vlenh)

        label.place(x=800, y=20)

        ten = gTTS(text=vlenh, lang='vi')

        # Xuất ra lời nói theo văn bản đã nhập

        filename =os.path.abspath('')+'\huong29.mp3'

        ten.save(filename)

* ***b. Hàm mở file CSV***

def Open():

    global data\_origin, data\_preprocessed, current\_dataset

    btnProcessEDA['state'] = tk.ACTIVE

    btnAnalyze['state'] = tk.DISABLED

    btnChooseAttribute['state'] = tk.DISABLED

    btnExportCSV['state'] = tk.ACTIVE

    btnPredict['state'] = tk.DISABLED

    filename = filedialog.askopenfilename(

        title="Chọn file csv", initialdir='./', filetypes=(("CSV Files", "\*.csv"),))

    data\_origin = pd.read\_csv(filename)

    current\_dataset = data\_origin

    # Hiển thị tập dữ liệu gốc

    ShowData(data\_origin)

    print(data\_origin.count().sort\_values())

* ***c. Hàm thoát chương trình***

def Close():

    result = messagebox.askokcancel('Thoát', 'Bạn có muốn thoát không?', )

    if result:

        master.destroy()

* ***d. Hàm xử lý, tiền xử lý***

def ProcessEDA():

    # Hiển thị các nút chức năng sau khi tiền xử lý

    btnChooseAttribute['state'] = tk.ACTIVE

    btnAnalyze['state'] = tk.ACTIVE

    btnExportCSV['state'] = tk.ACTIVE

    global data\_preprocessed, current\_dataset

    PreProcessing()

    ClearFrame()

    # Hiển thị tập dữ liệu đã được tiền xử lý

    current\_dataset = data\_preprocessed

    ShowData(data\_preprocessed)

def PreProcessing():

    global data\_origin, data\_preprocessed

    data\_preprocessed = data\_origin

    # Bước 3: Xử lý cột null

    print(data\_preprocessed.count().sort\_values())

    # Tất cả các cột đều có 205 giá trị ==> Không có cột nào có dữ liệu null

    # Bước 4: Xử lý dòng null

    # Xóa các cột không cần thiết cho việc xử lý dữ liệu

    data\_preprocessed = data\_preprocessed.dropna(how='any')

    thirdcol = data\_preprocessed.shape[1]/3

    for idx, row in data\_preprocessed.iterrows():

        num\_zeroes = (row == 0).sum()

        if num\_zeroes > thirdcol:

            data\_preprocessed = data\_preprocessed.drop(idx, axis=0)

    # Bước 5: Loại bỏ các giá trị ngoại lệ isolated

    cols = data\_preprocessed.columns[(data\_preprocessed.columns != '2019') & (

        data\_preprocessed.columns != 'Country Name') & (data\_preprocessed.columns != 'ID\_Name')]

    z = np.abs(stats.zscore(data\_preprocessed[cols].\_get\_numeric\_data()))

    zscoreGraph(data\_preprocessed, z)

    data\_preprocessed[cols] = (data\_preprocessed[cols])[(z <3).all(axis=1)]

    data\_preprocessed = data\_preprocessed.dropna(how='any')

    # Bước 6: Thay thế các giá trị 0 và 1 bởi YES và NO

    #thêm một cột dự đoán 2019

    data\_preprocessed['2019'] = 0

    # Bước 7: Rời rạc hóa dùng MinMaxScaler

    scaler = preprocessing.MinMaxScaler()

    scaler.fit(data\_preprocessed[cols])

    data\_preprocessed[cols] = pd.DataFrame(scaler.transform(

        data\_preprocessed[cols]), index=data\_preprocessed.index, columns=cols)

    # Bước 8: Xác định mô hình trích lọc các thuộc tính đặc trưng EDA

    X = data\_preprocessed.loc[:, (data\_preprocessed.columns != '2018') & (data\_preprocessed.columns != '2019') & (

        data\_preprocessed.columns != 'Country Name') & (data\_preprocessed.columns != 'ID\_Name')]

    y = data\_preprocessed[['2019']]

    binarizer = Binarizer(threshold=0.5)

    y = binarizer.fit\_transform(y)

    selector = SelectKBest(chi2, k=3)

    selector.fit(X, y)

    X\_new = selector.transform(X)

print(X.columns[selector.get\_support(indices=True)])

* ***e. Hàm chọn thuộc tính***

def ChooseAttribute():

    global data\_preprocessed

    # Hiển thị cửa sổ mới

    chooseAtr\_form = tk.Toplevel()

    chooseAtr\_form.geometry('560x470')

    chooseAtr\_form.resizable(tk.FALSE, tk.FALSE)

    chooseAtr\_form.title("Chọn thuộc tính")

    chooseAtr\_form.focus()

    chooseAtr\_form = chooseAtr\_form

    tk.Label(chooseAtr\_form, text='Phân tích').place(x=90, y=15)

    listbox1 = tk.Listbox(chooseAtr\_form, height=25, width=40,

                          font="Consolas 8", selectmode=tk.EXTENDED)

    listbox1.place(x=15, y=45)

    tk.Label(chooseAtr\_form, text='Không phân tích').place(x=370, y=15)

    listbox2 = tk.Listbox(chooseAtr\_form, height=25, width=40,

                          font="Consolas 8", selectmode=tk.EXTENDED)

    listbox2.place(x=300, y=45)

    tk.Label(chooseAtr\_form, text="Số lượng").place(x=15, y=420)

    lblSoLuong = tk.Label(chooseAtr\_form, relief=tk.SUNKEN, font="Times 8",

                          borderwidth=3, width=15, height=1)

    lblSoLuong.place(x=80, y=420)

    tk.Button(chooseAtr\_form, text='OK', width=12, height=2,

              command=lambda: DropColumns(list(listbox2.get(0, tk.END)), data\_preprocessed, chooseAtr\_form)).place(x=450, y=410)

    columns = data\_preprocessed.columns

    for col in columns:

        listbox1.insert(tk.END, col)

    lblSoLuong.configure(text=listbox1.size())

    # Các nút chức năng

    btnMoveRight = tk.Button(chooseAtr\_form, text=">",

                             width=4, command=lambda: MoveToRight(listbox1, listbox2, lblSoLuong))

    btnMoveRight.place(x=260, y=70)

    btnMoveLeft = tk.Button(chooseAtr\_form, text="<",

                            width=4, command=lambda: MoveToLeft(listbox1, listbox2, lblSoLuong))

    btnMoveLeft.place(x=260, y=110)

    btnMoveAllRight = tk.Button(

        chooseAtr\_form, text=">>", width=4, command=lambda: MoveAllToRight(listbox1, listbox2, lblSoLuong))

    btnMoveAllRight.place(x=260, y=150)

    btnMoveAllLeft = tk.Button(

        chooseAtr\_form, text="<<", width=4, command=lambda: MoveAllToLeft(listbox1, listbox2, lblSoLuong))

    btnMoveAllLeft.place(x=260, y=190)

    btnDeleteAll = tk.Button(chooseAtr\_form, text='X', width=4,

                             command=lambda: DeleteAll(listbox1, listbox2, lblSoLuong))

    btnDeleteAll.place(x=260, y=230)

* ***f. Hàm Phân tích:***

def Analyze():

    # Bước 9: Xác định mô hình trích lọc các thuộc tính đặc trưng

    global data\_preprocessed, data\_analyzed, current\_dataset

    data\_newa = pd.DataFrame()

    data\_newa['TB theo z'] = data\_preprocessed.select\_dtypes(include='number').drop(columns='2019').mean(axis=1)

    data\_newb = data\_preprocessed[['ID\_Name', 'Country Name', '2018']]

    data\_analyzed=pd.concat([data\_newb, data\_newa], axis=1)

    current\_dataset = data\_analyzed

    ShowData(data\_analyzed)

    data\_preprocessed['2019']=data\_newa['TB theo z']

    btnPredict['state'] = tk.ACTIVE

* ***f. Hàm dự đoán:***

def Predict():

    # Chọn các cột quốc gia và số liệu từ 1990 đến 2018

    data\_newa = data\_preprocessed[['Country Name']]

    data\_newb = data\_preprocessed.select\_dtypes(include='number')

    data\_train = pd.concat([data\_newa, data\_newb], axis=1)

    # Tạo DataFrame chứa dữ liệu dự đoán cho năm 2019

    data\_predict = pd.DataFrame(columns=['Country Name', '2019'])

    # Duyệt qua từng quốc gia trong DataFrame

    for index, row in data\_train.iterrows():

        country = row['Country Name']

        x\_train = row.drop('Country Name').values.reshape(-1, 1)

        y\_train = row['2018']

        # Kiểm tra xem có đủ dữ liệu để huấn luyện mô hình hay không

        if not isinstance(y\_train, float):

            # Xây dựng mô hình hồi quy tuyến tính

            model = LinearRegression()

            model.fit(x\_train, y\_train)

            # Dự đoán số liệu cho năm 2019

            x\_predict = [[2019]]

            y\_predict = model.predict(x\_predict)

            # Thêm dữ liệu dự đoán vào DataFrame

            data\_predict.loc[index] = [country, y\_predict[0]]

    ShowData(data\_predict)

* ***g. Hàm xóa***

def ClearFrame():

    for widget in body.winfo\_children():

        widget.destroy()

* ***h. Hàm thể hiện dữ liệu***

def ShowData(df):

    global trv

    ClearFrame()

    r\_set = df.to\_numpy().tolist()

    l1 = list(df)

    # Tạo khung cảnh

    frame = ttk.Frame(body)

    frame.grid(row=4, column=1, columnspan=3, padx=10, pady=20)

    # Tạo thanh cuộn dọc

    scrollbar\_y = ttk.Scrollbar(frame)

    scrollbar\_y.pack(side=tk.LEFT, fill=tk.Y)

    # Tạo thanh cuộn ngang

    scrollbar\_x = ttk.Scrollbar(frame, orient=tk.HORIZONTAL)

    scrollbar\_x.pack(side=tk.BOTTOM, fill=tk.X)

    # Tạo Treeview với thanh cuộn

    trv = ttk.Treeview(frame, selectmode='browse',

                       height=28, show='headings', columns=l1,

                       yscrollcommand=scrollbar\_y.set, xscrollcommand=scrollbar\_x.set)

    trv.pack(side=tk.LEFT, fill=tk.BOTH)

    scrollbar\_y.config(command=trv.yview)

    scrollbar\_x.config(command=trv.xview)

    for i in l1:

        trv.column(i, width=90, anchor='c')

        trv.heading(i, text=str(i))

    for dt in r\_set:

        v = [r for r in dt]

        item\_id = v[0]

    # Kiểm tra xem mục đã tồn tại trong cây danh sách hay chưa

        if not trv.exists(item\_id):

            trv.insert("", 'end', iid=item\_id, values=v)

    lblShape.configure(text=str(df.shape[0]) + 'x' + str(df.shape[1]))

* ***i. Hàm các nút chức năng***

def MoveToRight(listbox1, listbox2, lblSoLuong):

    i = 0

    selection = listbox1.curselection()

    for i in reversed(selection):

        listbox2.insert(tk.END, listbox1.get(i))

        listbox1.delete(i)

    dem = listbox1.size()

    lblSoLuong.configure(text=dem)

def MoveToLeft(listbox1, listbox2, lblSoLuong):

    i = 0

    selection = listbox2.curselection()

    for i in reversed(selection):

        listbox1.insert(tk.END, listbox2.get(i))

        listbox2.delete(i)

    dem = listbox1.size()

    lblSoLuong.configure(text=dem)

def MoveAllToRight(listbox1, listbox2, lblSoLuong):

    while listbox1.size() != 0:

        listbox2.insert(tk.END, listbox1.get(listbox1.size()-1))

        listbox1.delete(listbox1.size()-1)

    lblSoLuong.configure(text=0)

def MoveAllToLeft(listbox1, listbox2, lblSoLuong):

    while listbox2.size() != 0:

        listbox1.insert(tk.END, listbox2.get(listbox2.size()-1))

        listbox2.delete(listbox2.size()-1)

    lblSoLuong.configure(text=listbox1.size())

def DeleteAll(listbox1, listbox2, lblSoLuong):

    listbox1.delete(0, tk.END)

    listbox2.delete(0, tk.END)

    global data\_preprocessed

    columns = data\_preprocessed.columns

    for col in columns:

        listbox1.insert(tk.END, col)

    lblSoLuong.configure(text=listbox1.size())

def DropColumns(columns, data, chooseAtr\_form):

    if len(columns) == len(data.columns):

        messagebox.showwarning("Thông báo", "Phải chọn ít nhất 1 thuộc tính")

        chooseAtr\_form.focus()

        return

    if 'Country Name' in columns or '2018' in columns or '2019' in columns or 'ID\_Name' in columns:

        messagebox.showwarning(

            "Thông báo", "Không chọn ID\_Name, Country, 2018, 2019 ")

        chooseAtr\_form.focus()

        return

    data.drop(columns=columns, inplace=True, axis=1)

    chooseAtr\_form.destroy()

    ShowData(data)

    messagebox.showinfo("Thông báo", "Đã drop thành công " +str(len(columns)) + " thuộc tính!")

* ***k. Hàm vẽ đồ thị***

def zscoreGraph(data, z):

    x = data['ID\_Name'].values.tolist()

    numberCol = z.select\_dtypes(include='number')

    y = numberCol.mean(axis=1).values.tolist()

    plt.plot(x, y)

    plt.title('Đồ thị z-score')

    plt.axhline(y=3, color='#FF5677', linestyle='-')

    plt.xlabel('ID')

    plt.ylabel('z-score')

 plt.show()

* ***l. Hàm Xuất file CSV***

def ExportCSV():

    global current\_dataset

    # Tạo thư mục nếu không tồn tại

    os.makedirs('./CSV29\_Folder', exist\_ok=True)

    try:

        current\_time = datetime.now().strftime("%d%m%Y")

        path = f'./CSV29\_Folder/Processed\_CO2Emission\_{current\_time}.csv'

        current\_dataset.to\_csv(path)

        messagebox.showinfo('Thông báo', 'Xuất file CSV thành công!')

    except FileNotFoundError:

        messagebox.showerror('Thông báo', 'Không tìm thấy đường dẫn hoặc tên tệp!')

    except PermissionError:

        messagebox.showerror('Thông báo', 'Không có quyền ghi file!')

    except Exception as e:

        messagebox.showerror('Thông báo', f'Lỗi xuất file CSV: {str(e)}')

* + 1. Full Code

# Voice

import speech\_recognition as sr

from gtts import gTTS

import playsound

# thư viện cần thiết EDA

import numpy as np  # thư viện về đại số tuyến tính

import pandas as pd  # thư viện xử lý

from sklearn import preprocessing  # thư viện tiền xử lý

from scipy import stats  # thư viện cung cấp công cụ thống kê

from sklearn.preprocessing import Binarizer

from sklearn.feature\_selection import SelectKBest, chi2

from sklearn.linear\_model import LinearRegression #sử dụng hồi quy tuyến tính

# tkinter

import tkinter as tk

from tkinter import ttk

from tkinter import messagebox

from tkinter import filedialog

# Thư viện thời gian thực

from datetime import datetime

# Thư viện vẽ đồ thị

import matplotlib.pyplot as plt

# Thư viện chạy các file bên ngoài

import subprocess

# thư viện os

import os

import time

#thư viện game

import subprocess

#================================================================

#tạo của sổ

master = tk.Tk()

master.geometry('1500x800')

master.title("29\_Nguyễn Diệu Hương\_21110489")

master.pack\_propagate()

master.resizable(tk.FALSE, tk.FALSE)

def VoiceAssistant():

    vlenh=""

    label=tk.Label(foot,text=vlenh)

    label.place(x=800, y=20)

    label.destroy()

    r = sr.Recognizer()

    with sr.Microphone() as Source:

        messagebox.showinfo("Cảnh báo", "Hiệu chỉnh nhiễu trước khi nói!")

        r.adjust\_for\_ambient\_noise(Source, duration=1)

        messagebox.showinfo("Sẵn sàng", "Bấm OK để bắt đầu nói tiếng Việt trong 3 giây")

        audio\_data = r.record(Source, duration=3)

        try:

            vlenh = r.recognize\_google(audio\_data, language="vi")

            messagebox.showinfo("Bạn đã nói: ", vlenh)

        except:

            vlenh = "Bạn nói gì tôi nghe không hiểu!"

        label=tk.Label(foot,text=vlenh)

        label.place(x=800, y=20)

        ten = gTTS(text=vlenh, lang='vi')

        # Xuất ra lời nói theo văn bản đã nhập

        filename =os.path.abspath('')+'\huong29.mp3'

        ten.save(filename)

def Open():

    global data\_origin, data\_preprocessed, current\_dataset

    btnProcessEDA['state'] = tk.ACTIVE

    btnAnalyze['state'] = tk.DISABLED

    btnChooseAttribute['state'] = tk.DISABLED

    btnExportCSV['state'] = tk.ACTIVE

    btnPredict['state'] = tk.DISABLED

    filename = filedialog.askopenfilename(

        title="Chọn file csv", initialdir='./', filetypes=(("CSV Files", "\*.csv"),))

    data\_origin = pd.read\_csv(filename)

    current\_dataset = data\_origin

    # Hiển thị tập dữ liệu gốc

    ShowData(data\_origin)

    print(data\_origin.count().sort\_values())

def Close():

    result = messagebox.askokcancel('Thoát', 'Bạn có muốn thoát không?', )

    if result:

        master.destroy()

def ProcessEDA():

    # Hiển thị các nút chức năng sau khi tiền xử lý

    btnChooseAttribute['state'] = tk.ACTIVE

    btnAnalyze['state'] = tk.ACTIVE

    btnExportCSV['state'] = tk.ACTIVE

    global data\_preprocessed, current\_dataset

    PreProcessing()

    ClearFrame()

    # Hiển thị tập dữ liệu đã được tiền xử lý

    current\_dataset = data\_preprocessed

    ShowData(data\_preprocessed)

def PreProcessing():

    global data\_origin, data\_preprocessed

    data\_preprocessed = data\_origin

    # Bước 3: Xử lý cột null

    print(data\_preprocessed.count().sort\_values())

    # Tất cả các cột đều có 205 giá trị ==> Không có cột nào có dữ liệu null

    # Bước 4: Xử lý dòng null

    # Xóa các cột không cần thiết cho việc xử lý dữ liệu

    data\_preprocessed = data\_preprocessed.dropna(how='any')

    thirdcol = data\_preprocessed.shape[1]/3

    for idx, row in data\_preprocessed.iterrows():

        num\_zeroes = (row == 0).sum()

        if num\_zeroes > thirdcol:

            data\_preprocessed = data\_preprocessed.drop(idx, axis=0)

    # Bước 5: Loại bỏ các giá trị ngoại lệ isolated

    cols = data\_preprocessed.columns[(data\_preprocessed.columns != '2019') & (

        data\_preprocessed.columns != 'Country Name') & (data\_preprocessed.columns != 'ID\_Name')]

    z = np.abs(stats.zscore(data\_preprocessed[cols].\_get\_numeric\_data()))

    zscoreGraph(data\_preprocessed, z)

    data\_preprocessed[cols] = (data\_preprocessed[cols])[(z <3).all(axis=1)]

    data\_preprocessed = data\_preprocessed.dropna(how='any')

    # Bước 6: Thay thế các giá trị 0 và 1 bởi YES và NO

    #thêm một cột dự đoán 2019

    data\_preprocessed['2019'] = 0

    # Bước 7: Rời rạc hóa dùng MinMaxScaler

    scaler = preprocessing.MinMaxScaler()

    scaler.fit(data\_preprocessed[cols])

    data\_preprocessed[cols] = pd.DataFrame(scaler.transform(

        data\_preprocessed[cols]), index=data\_preprocessed.index, columns=cols)

    # Bước 8: Xác định mô hình trích lọc các thuộc tính đặc trưng EDA

    X = data\_preprocessed.loc[:, (data\_preprocessed.columns != '2018') & (data\_preprocessed.columns != '2019') & (

        data\_preprocessed.columns != 'Country Name') & (data\_preprocessed.columns != 'ID\_Name')]

    y = data\_preprocessed[['2019']]

    binarizer = Binarizer(threshold=0.5)

    y = binarizer.fit\_transform(y)

    selector = SelectKBest(chi2, k=3)

    selector.fit(X, y)

    X\_new = selector.transform(X)

    print(X.columns[selector.get\_support(indices=True)])

def ChooseAttribute():

    global data\_preprocessed

    # Hiển thị cửa sổ mới

    chooseAtr\_form = tk.Toplevel()

    chooseAtr\_form.geometry('560x470')

    chooseAtr\_form.resizable(tk.FALSE, tk.FALSE)

    chooseAtr\_form.title("Chọn thuộc tính")

    chooseAtr\_form.focus()

    chooseAtr\_form = chooseAtr\_form

    tk.Label(chooseAtr\_form, text='Phân tích').place(x=90, y=15)

    listbox1 = tk.Listbox(chooseAtr\_form, height=25, width=40,

                          font="Consolas 8", selectmode=tk.EXTENDED)

    listbox1.place(x=15, y=45)

    tk.Label(chooseAtr\_form, text='Không phân tích').place(x=370, y=15)

    listbox2 = tk.Listbox(chooseAtr\_form, height=25, width=40,

                          font="Consolas 8", selectmode=tk.EXTENDED)

    listbox2.place(x=300, y=45)

    tk.Label(chooseAtr\_form, text="Số lượng").place(x=15, y=420)

    lblSoLuong = tk.Label(chooseAtr\_form, relief=tk.SUNKEN, font="Times 8",

                          borderwidth=3, width=15, height=1)

    lblSoLuong.place(x=80, y=420)

    tk.Button(chooseAtr\_form, text='OK', width=12, height=2,

              command=lambda: DropColumns(list(listbox2.get(0, tk.END)), data\_preprocessed, chooseAtr\_form)).place(x=450, y=410)

    columns = data\_preprocessed.columns

    for col in columns:

        listbox1.insert(tk.END, col)

    lblSoLuong.configure(text=listbox1.size())

    # Các nút chức năng

    btnMoveRight = tk.Button(chooseAtr\_form, text=">",

                             width=4, command=lambda: MoveToRight(listbox1, listbox2, lblSoLuong))

    btnMoveRight.place(x=260, y=70)

    btnMoveLeft = tk.Button(chooseAtr\_form, text="<",

                            width=4, command=lambda: MoveToLeft(listbox1, listbox2, lblSoLuong))

    btnMoveLeft.place(x=260, y=110)

    btnMoveAllRight = tk.Button(

        chooseAtr\_form, text=">>", width=4, command=lambda: MoveAllToRight(listbox1, listbox2, lblSoLuong))

    btnMoveAllRight.place(x=260, y=150)

    btnMoveAllLeft = tk.Button(

        chooseAtr\_form, text="<<", width=4, command=lambda: MoveAllToLeft(listbox1, listbox2, lblSoLuong))

    btnMoveAllLeft.place(x=260, y=190)

    btnDeleteAll = tk.Button(chooseAtr\_form, text='X', width=4,

                             command=lambda: DeleteAll(listbox1, listbox2, lblSoLuong))

    btnDeleteAll.place(x=260, y=230)

def Analyze():

    # Bước 9: Xác định mô hình trích lọc các thuộc tính đặc trưng

    global data\_preprocessed, data\_analyzed, current\_dataset

    data\_newa = pd.DataFrame()

    data\_newa['TB theo z'] = data\_preprocessed.select\_dtypes(include='number').drop(columns='2019').mean(axis=1)

    data\_newb = data\_preprocessed[['ID\_Name', 'Country Name', '2018']]

    data\_analyzed=pd.concat([data\_newb, data\_newa], axis=1)

    current\_dataset = data\_analyzed

    ShowData(data\_analyzed)

    data\_preprocessed['2019']=data\_newa['TB theo z']

    btnPredict['state'] = tk.ACTIVE

def Predict():

    # Chọn các cột quốc gia và số liệu từ 1990 đến 2018

    data\_newa = data\_preprocessed[['Country Name']]

    data\_newb = data\_preprocessed.select\_dtypes(include='number')

    data\_train = pd.concat([data\_newa, data\_newb], axis=1)

    # Tạo DataFrame chứa dữ liệu dự đoán cho năm 2019

    data\_predict = pd.DataFrame(columns=['Country Name', '2019'])

    # Duyệt qua từng quốc gia trong DataFrame

    for index, row in data\_train.iterrows():

        country = row['Country Name']

        x\_train = row.drop('Country Name').values.reshape(-1, 1)

        y\_train = row['2018']

        # Kiểm tra xem có đủ dữ liệu để huấn luyện mô hình hay không

        if not isinstance(y\_train, float):

            # Xây dựng mô hình hồi quy tuyến tính

            model = LinearRegression()

            model.fit(x\_train, y\_train)

            # Dự đoán số liệu cho năm 2019

            x\_predict = [[2019]]

            y\_predict = model.predict(x\_predict)

            # Thêm dữ liệu dự đoán vào DataFrame

            data\_predict.loc[index] = [country, y\_predict[0]]

    ShowData(data\_predict)

def ClearFrame():

    for widget in body.winfo\_children():

        widget.destroy()

trv = None

def ShowData(df):

    global trv

    ClearFrame()

    r\_set = df.to\_numpy().tolist()

    l1 = list(df)

    # Tạo khung cảnh

    frame = ttk.Frame(body)

    frame.grid(row=4, column=1, columnspan=3, padx=10, pady=20)

    # Tạo thanh cuộn dọc

    scrollbar\_y = ttk.Scrollbar(frame)

    scrollbar\_y.pack(side=tk.LEFT, fill=tk.Y)

    # Tạo thanh cuộn ngang

    scrollbar\_x = ttk.Scrollbar(frame, orient=tk.HORIZONTAL)

    scrollbar\_x.pack(side=tk.BOTTOM, fill=tk.X)

    # Tạo Treeview với thanh cuộn

    trv = ttk.Treeview(frame, selectmode='browse',

                       height=28, show='headings', columns=l1,

                       yscrollcommand=scrollbar\_y.set, xscrollcommand=scrollbar\_x.set)

    trv.pack(side=tk.LEFT, fill=tk.BOTH)

    scrollbar\_y.config(command=trv.yview)

    scrollbar\_x.config(command=trv.xview)

    for i in l1:

        trv.column(i, width=90, anchor='c')

        trv.heading(i, text=str(i))

    for dt in r\_set:

        v = [r for r in dt]

        item\_id = v[0]

    # Kiểm tra xem mục đã tồn tại trong cây danh sách hay chưa

        if not trv.exists(item\_id):

            trv.insert("", 'end', iid=item\_id, values=v)

    lblShape.configure(text=str(df.shape[0]) + 'x' + str(df.shape[1]))

def MoveToRight(listbox1, listbox2, lblSoLuong):

    i = 0

    selection = listbox1.curselection()

    for i in reversed(selection):

        listbox2.insert(tk.END, listbox1.get(i))

        listbox1.delete(i)

    dem = listbox1.size()

    lblSoLuong.configure(text=dem)

def MoveToLeft(listbox1, listbox2, lblSoLuong):

    i = 0

    selection = listbox2.curselection()

    for i in reversed(selection):

        listbox1.insert(tk.END, listbox2.get(i))

        listbox2.delete(i)

    dem = listbox1.size()

    lblSoLuong.configure(text=dem)

def MoveAllToRight(listbox1, listbox2, lblSoLuong):

    while listbox1.size() != 0:

        listbox2.insert(tk.END, listbox1.get(listbox1.size()-1))

        listbox1.delete(listbox1.size()-1)

    lblSoLuong.configure(text=0)

def MoveAllToLeft(listbox1, listbox2, lblSoLuong):

    while listbox2.size() != 0:

        listbox1.insert(tk.END, listbox2.get(listbox2.size()-1))

        listbox2.delete(listbox2.size()-1)

    lblSoLuong.configure(text=listbox1.size())

def DeleteAll(listbox1, listbox2, lblSoLuong):

    listbox1.delete(0, tk.END)

    listbox2.delete(0, tk.END)

    global data\_preprocessed

    columns = data\_preprocessed.columns

    for col in columns:

        listbox1.insert(tk.END, col)

    lblSoLuong.configure(text=listbox1.size())

def DropColumns(columns, data, chooseAtr\_form):

    if len(columns) == len(data.columns):

        messagebox.showwarning("Thông báo", "Phải chọn ít nhất 1 thuộc tính")

        chooseAtr\_form.focus()

        return

    if 'Country Name' in columns or '2018' in columns or '2019' in columns or 'ID\_Name' in columns:

        messagebox.showwarning(

            "Thông báo", "Không chọn ID\_Name, Country, 2018, 2019 ")

        chooseAtr\_form.focus()

        return

    data.drop(columns=columns, inplace=True, axis=1)

    chooseAtr\_form.destroy()

    ShowData(data)

    messagebox.showinfo("Thông báo", "Đã drop thành công " +str(len(columns)) + " thuộc tính!")

def zscoreGraph(data, z):

    x = data['ID\_Name'].values.tolist()

    numberCol = z.select\_dtypes(include='number')

    y = numberCol.mean(axis=1).values.tolist()

    plt.plot(x, y)

    plt.title('Đồ thị z-score')

    plt.axhline(y=3, color='#FF5677', linestyle='-')

    plt.xlabel('ID')

    plt.ylabel('z-score')

    plt.show()

def ExportCSV():

    global current\_dataset

    # Tạo thư mục nếu không tồn tại

    os.makedirs('./CSV29\_Folder', exist\_ok=True)

    try:

        current\_time = datetime.now().strftime("%d%m%Y")

        path = f'./CSV29\_Folder/Processed\_CO2Emission\_{current\_time}.csv'

        current\_dataset.to\_csv(path)

        messagebox.showinfo('Thông báo', 'Xuất file CSV thành công!')

    except FileNotFoundError:

        messagebox.showerror('Thông báo', 'Không tìm thấy đường dẫn hoặc tên tệp!')

    except PermissionError:

        messagebox.showerror('Thông báo', 'Không có quyền ghi file!')

    except Exception as e:

        messagebox.showerror('Thông báo', f'Lỗi xuất file CSV: {str(e)}')

def Game():

    subprocess.run(['python', 'G329Huong\_Game.py'])

#tạo frame

#head

head = tk.Frame(master,bg='#6096B4',  width=1400, height=70, relief=tk.SUNKEN, borderwidth=2)

head.pack()

head\_label = tk.Label(head, text="29\_21110489: Đồ Án Học Phần: CO2 Emissions", bg='#6096B4', fg="white",font=("Arial", 20 ,"bold"))

head\_label.place(x=(1400-600)//2, y=(70-30)//2)

#foot

foot = tk.Frame(master, bg='#93BFCF', width=1400, height=70,  relief=tk.SUNKEN, borderwidth=1)

foot.pack()

btnOpen = tk.Button(foot, text='1.Mở Tệp', width=12, height=2, command=Open)

btnOpen.place(x=20, y=15)

btnProcessEDA = tk.Button(foot, text='2.Tiền Xử Lý', width=12, height=2, state=tk.DISABLED, command=ProcessEDA)

btnProcessEDA.place(x=120, y=15)

btnChooseAttribute = tk.Button(foot, text='3.Chọn Thuộc Tính', width=15, height=2, state=tk.DISABLED, command=ChooseAttribute)

btnChooseAttribute.place(x=220, y=15)

btnAnalyze = tk.Button(foot, text='4.Phân Tích', width=12, height=2, state=tk.DISABLED, command=Analyze)

btnAnalyze.place(x=340, y=15)

btnPredict = tk.Button(foot, text='5.Dự Đoán năm 2019', width=15, height=2, state=tk.DISABLED, command=Predict)

btnPredict.place(x=440, y=15)

btnExportCSV = tk.Button(foot, text='6.Xuất File CSV', width=15, height=2, state=tk.DISABLED, command=ExportCSV)

btnExportCSV.place(x=560, y=15)

lblShape = tk.Label(master, text='0x0', width=10, font=('arial 9'), height=1, relief=tk.SUNKEN)

lblShape.place(x=1350, y=100)

btnVoiceAssistant = tk.Button(foot, text='7.Nói', width=12, height=2, command=VoiceAssistant)

btnVoiceAssistant.place(x=680, y=15)

btnGame = tk.Button(foot, text='8.Game', width=12, height=2, command=Game)

btnGame.place(x=1100, y=15)

btnClose = tk.Button(foot, text='9.Thoát', width=10, height=2, command=Close)

btnClose.place(x=1200, y=15)

#body

body = tk.Frame(master, bg='#F7F5EB', width=1400, height=650, relief=tk.SUNKEN, borderwidth=2)

body.pack()

body.grid\_propagate(False)

body = body

#==========================================================================

master.mainloop()

1. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỒ ÁN HỌC PHẦN

## Kết luận

* + 1. Những kết quả đạt được

ĐỒ ÁN HỌC PHẦN đã thực hiện việc PHÂN TÍCH THĂM DÒ MỨC ĐỘ PHÁT SINH LƯỢNG KHÍ THẢI CO2 CỦA CÁC NƯỚC TRÊN THẾ GIỚI TỪ NĂM 1990-2018; bao gồm:

+ Giao diện đồ họa: GUI

+ Xử lý giọng nói trợ lý ảo: Voice Assistant

+ Phân tích khám phá trên tập dữ liệu thực nghiệm: G329Huong\_CO2Emissions.csv

* + 1. Hạn chế
* Chưa liên kết voice assistant để điều khiển phân tích bằng lời nói.
* Chưa dự đoán được kết quả
* Chưa có nhiều chức năng phân tích theo nhiều dạng
* Chưa có nhiều đồ thị biểu diễn

## Hướng phát triển

Trong tương lai, đồ án có thể mở rộng thêm các chức năng như cho phép người dùng chọn tập dữ liệu tùy ý, có thể xử lý các thuộc tính phức tạp, sử dụng các hàm, các thuật toán dự đoán để đưa ra khách quan theo nhiều khía cạnh.

## Hướng khắc phục hạn chế

## Đồ án sẽ tiếp tục thực hiện liên kết voice assistant để điều khiển phân tích bằng lời nói

## Thêm nhiều dạng phân tích, so sánh trực quan

## Phân tích kĩ và sâu hơn để dự đoán hiệu quả

## Thêm nhiều đồ thị để thấy số liệu một cách trực quan

## Hướng mở rộng ĐỒ ÁN HỌC PHẦN

Sẽ tiếp tục mở rộng sản phẩm đề tài với các ứng dụng thực tiễn. Kết hợp với các thuật toán trí tuệ nhân tạo để cho ra khả năng dự đoán thuộc tính từ tập dữ liệu thực nghiệm.

# DANH MỤC TÀI LIỆU THAM KHẢO

1. Võ Xuân Thể (2023), Tài liệu học tập Lập trình Python, Trường Đại học Sư phạm kỹ thuật Tp.HCM, Tài liệu lưu hành nội bộ được cung cấp theo từng buổi học
2. Võ Xuân Thể (2023), Bài tập thực hành Lập trình Python, Trường Đại học Sư phạm kỹ thuật Tp.HCM, Tài liệu lưu hành nội bộ được cung cấp theo từng buổi học
3. Dữ liệu nguồn: CO2 Emissions \_ 1960 – 2018 Link: <https://www.kaggle.com/datasets/kkhandekar/co2-emissions-1960-2018>, truy cập tháng 03/2023

1. IDE (Integrated Development Environment) là môi trường tích hợp dùng để viết code để phát triển ứng dụng [↑](#footnote-ref-2)