# Importando as bibliotecas necessárias.

import socket

import tkinter as tk

# Criando a classe App que contém as definições da GUI e dos métodos de jogo.

class App:

# Definindo o construtor da classe.

def \_\_init\_\_(self, master):

# Configurando a janela principal.

self.master = master

master.title("Adivinhe o número")

# Adicionando um rótulo para instruções ao usuário.

self.label = tk.Label(

master, text="Tente adivinhar um número de 1 a 20. Você tem 10 tentativas.")

self.label.pack()

# Adicionando uma caixa de entrada para adivinhação do usuário.

self.number\_entry = tk.Entry(master)

self.number\_entry.pack()

# Adicionando um botão para envio da adivinhação.

self.guess\_button = tk.Button(

master, text="Chutar", command=self.make\_guess)

self.guess\_button.pack()

# Adicionando uma etiqueta para exibir o resultado do jogo.

self.result\_label = tk.Label(master, text="")

self.result\_label.pack()

# Adicionando um botão para jogar novamente.

self.play\_again\_button = tk.Button(

master, text="Jogar novamente", command=self.play\_again)

self.play\_again\_button.pack()

# Adicionando um botão para sair do jogo.

self.quit\_button = tk.Button(master, text="Sair", command=master.quit)

self.quit\_button.pack()

# Criando um objeto socket para se comunicar com o servidor.

self.client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

self.client\_socket.connect(('localhost', 8000))

# Inicializando o número de tentativas do usuário.

self.num\_guesses = 0

# Definindo o método make\_guess que é chamado quando o usuário faz uma adivinhação.

def make\_guess(self):

# Obtendo a adivinhação do usuário a partir da caixa de entrada.

palpite = self.number\_entry.get()

# Enviando a adivinhação para o servidor por meio do socket.

self.client\_socket.send(palpite.encode('utf-8'))

# Recebendo a resposta do servidor e decodificando-a em utf-8.

resultado = self.client\_socket.recv(1024)

resultado = resultado.decode('utf-8')

# Incrementando o número de tentativas do usuário.

self.num\_guesses += 1

# Tratando os diferentes casos de resposta do servidor.

if resultado == 'Maior':

self.result\_label.config(text=" ► Tente um número maior.")

elif resultado == 'Menor':

self.result\_label.config(text=" ► Tente um número menor.")

elif resultado == 'Ganhou':

self.result\_label.config(

text="Parabéns! Você ganhou em {} tentativas.".format(self.num\_guesses))

self.client\_socket.close()

self.guess\_button.config(state="disabled")

elif resultado == 'Perdeu':

self.result\_label.config(

text="Você perdeu o jogo.")

self.client\_socket.close()

self.guess\_button.config(state="disabled")

elif resultado == 'Empate':

self.result\_label.config(text="Empate! \n Nenhum jogador venceu.")

self.client\_socket.close()

self.guess\_button.config(state="disabled")

# Verificando se o usuário atingiu o limite de tentativas.

if self.num\_guesses == 10:

self.result\_label.config(

text="Você atingiu o limite de tentativas.")

# Definindo o método play\_again que é chamado quando o usuário escolhe jogar novamente.

def play\_again(self):

# Limpando a caixa de entrada e a etiqueta de resultado.

self.number\_entry.delete(0, tk.END)

self.result\_label.config(text="")

# Reinicializando o número de tentativas do usuário.

self.num\_guesses = 0

# Habilitando o botão de adivinhação.

self.guess\_button.config(state="normal")

# Criando um novo objeto socket para se comunicar com o servidor.

self.client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

self.client\_socket.connect(('localhost', 8000))

# Criando a janela principal da GUI e instanciando a classe App.

root = tk.Tk()

app = App(root)

root.mainloop()