EXP: 5th and 6th

5: Implement concurrent echo client-server application

6: To write Program multi-threaded client/server processes

Server:

import \_thread as thread

from ChatFns import \*

#---------------------------------------------------#

#---------INITIALIZE CONNECTION VARIABLES-----------#

#---------------------------------------------------#

#Initiate socket and bind port to host PC

WindowTitle = 'Rohan Tech - Host'

s = socket(AF\_INET, SOCK\_STREAM)

HOST = gethostname()

PORT = 8011

conn = ''

s.bind((HOST, PORT))

def ClickAction():

    #Write message to chat window

    EntryText = FilteredMessage(EntryBox.get("0.0",END))

    LoadMyEntry(ChatLog, EntryText)

    #Scroll to the bottom of chat windows

    ChatLog.yview(END)

    #Erace previous message in Entry Box

    EntryBox.delete("0.0",END)

    #Send my mesage to all others

    conn.sendall(EntryText.encode('utf-8'))

#---------------------------------------------------#

#----------------- KEYBOARD EVENTS -----------------#

#---------------------------------------------------#

def PressAction(event):

    EntryBox.config(state=NORMAL)

    ClickAction()

def DisableEntry(event):

    EntryBox.config(state=DISABLED)

#---------------------------------------------------#

#-----------------GRAPHICS MANAGEMENT---------------#

#---------------------------------------------------#

#Create a window

base = Tk()

base.title(WindowTitle)

base.geometry("400x500")

base.resizable(width=FALSE, height=FALSE)

#Create a Chat window

ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatLog.insert(END, "Waiting for your partner to connect..\n")

ChatLog.config(state=DISABLED)

#Bind a scrollbar to the Chat window

scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")

ChatLog['yscrollcommand'] = scrollbar.set

#Create the Button to send message

SendButton = Button(base, font=30, text="Send", width="12", height=5,

                    bd=0, bg="#eb3434", activebackground="#34ebc0",

                    command=ClickAction)

#Create the box to enter message

EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")

EntryBox.bind("<Return>", DisableEntry)

EntryBox.bind("<KeyRelease-Return>", PressAction)

#Place all components on the screen

scrollbar.place(x=376,y=6, height=386)

ChatLog.place(x=6,y=6, height=386, width=370)

EntryBox.place(x=128, y=401, height=90, width=265)

SendButton.place(x=6, y=401, height=90)

def GetConnected():

    s.listen(1)

    global conn

    conn, addr = s.accept()

    LoadConnectionInfo(ChatLog, 'Connected with: ' + str(addr) + '\n---------------------------------------------------------------')

    while 1:

        try:

            data = conn.recv(1024)

            LoadOtherEntry(ChatLog, data.decode('utf-8'))

            if base.focus\_get() == None:

                FlashMyWindow(WindowTitle)

                playsound('SOUND1.wav')

        except:

            LoadConnectionInfo(ChatLog, '\n [ Your partner has disconnected ]\n [ Waiting for him to connect..] \n  ')

            GetConnected()

    conn.close()

thread.start\_new\_thread(GetConnected,())

base.mainloop()

**Client:**

import \_thread as thread

from ChatFns import \*

#---------------------------------------------------#

#---------INITIALIZE CONNECTION VARIABLES-----------#

#---------------------------------------------------#

WindowTitle = 'JChat v0.1 - Client'

HOST = gethostname()

PORT = 8011

s = socket(AF\_INET, SOCK\_STREAM)

#---------------------------------------------------#

#------------------ MOUSE EVENTS -------------------#

#---------------------------------------------------#

def ClickAction():

    #Write message to chat window

    EntryText = FilteredMessage(EntryBox.get("0.0",END))

    LoadMyEntry(ChatLog, EntryText)

    #Scroll to the bottom of chat windows

    ChatLog.yview(END)

    #Erace previous message in Entry Box

    EntryBox.delete("0.0",END)

    #Send my mesage to all others

    s.sendall(EntryText.encode('utf-8'))

#---------------------------------------------------#

#----------------- KEYBOARD EVENTS -----------------#

#---------------------------------------------------#

def PressAction(event):

    EntryBox.config(state=NORMAL)

    ClickAction()

def DisableEntry(event):

    EntryBox.config(state=DISABLED)

#---------------------------------------------------#

#-----------------GRAPHICS MANAGEMENT---------------#

#---------------------------------------------------#

#Create a window

base = Tk()

base.title(WindowTitle)

base.geometry("400x500")

base.resizable(width=FALSE, height=FALSE)

#Create a Chat window

ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatLog.insert(END, "Connecting to your Loki..\n")

ChatLog.config(state=DISABLED)

#Bind a scrollbar to the Chat window

scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")

ChatLog['yscrollcommand'] = scrollbar.set

#Create the Button to send message

SendButton = Button(base, font=30, text="Send", width="12", height=5,

                    bd=0, bg="#34ebc0", activebackground="#eb3434",

                    command=ClickAction)

#Create the box to enter message

EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")

EntryBox.bind("<Return>", DisableEntry)

EntryBox.bind("<KeyRelease-Return>", PressAction)

#Place all components on the screen

scrollbar.place(x=376,y=6, height=386)

ChatLog.place(x=6,y=6, height=386, width=370)

EntryBox.place(x=128, y=401, height=90, width=265)

SendButton.place(x=6, y=401, height=90)

#---------------------------------------------------#

#----------------CONNECTION MANAGEMENT--------------#

#---------------------------------------------------#

def ReceiveData():

    try:

        s.connect((HOST, PORT))

        LoadConnectionInfo(ChatLog, '[ Succesfully connected ]\n---------------------------------------------------------------')

    except:

        LoadConnectionInfo(ChatLog, '[ Unable to connect ]')

        return

    while 1:

        try:

            data = s.recv(1024)

        except:

            LoadConnectionInfo(ChatLog, '\n [ Your partner has disconnected ] \n')

            break

        if data != '':

            LoadOtherEntry(ChatLog, data.decode('utf-8'))

            if base.focus\_get() == None:

                FlashMyWindow(WindowTitle)

                playsound('SOUND.wav')

        else:

            LoadConnectionInfo(ChatLog, '\n [ Your partner has disconnected ] \n')

            break

    #s.close()

thread.start\_new\_thread(ReceiveData,())

base.mainloop()

**ChatGUI:**

from tkinter import \*

from socket import \*

import urllib

import re

import pygame

import win32gui

def getmixerargs():

    pygame.mixer.init()

    freq, size, chan = pygame.mixer.get\_init()

    return freq, size, chan

def initMixer():

    BUFFER = 3072  # audio buffer size, number of samples since pygame 1.8.

    FREQ, SIZE, CHAN = getmixerargs()

    pygame.mixer.init(FREQ, SIZE, CHAN, BUFFER)

def playsound(soundfile):

    """Play sound through default mixer channel in blocking manner.

       This will load the whole sound into memory before playback

    """

    pygame.init()

    pygame.mixer.init()

    sound = pygame.mixer.Sound(soundfile)

    clock = pygame.time.Clock()

    sound.play()

    while pygame.mixer.get\_busy():

        clock.tick(1000)

def playmusic(soundfile):

    """Stream music with mixer.music module in blocking manner.

       This will stream the sound from disk while playing.

    """

    pygame.init()

    pygame.mixer.init()

    clock = pygame.time.Clock()

    pygame.mixer.music.load(soundfile)

    pygame.mixer.music.play()

    while pygame.mixer.music.get\_busy():

        clock.tick(1000)

def stopmusic():

    """stop currently playing music"""

    pygame.mixer.music.stop()

#HOW TO PLAY SONG:

initMixer()

#playmusic(filename)

def FlashMyWindow(title):

    ID = win32gui.FindWindow(None, title)

    win32gui.FlashWindow(ID,True)

def FlashMyWindow2(title2):

    ID2 = win32gui.FindWindow(None, title2)

    win32gui.FlashWindow(ID2,True)

def GetExternalIP():

    url = "http://checkip.dyndns.org"

    request = urllib.urlopen(url).read()

    return str(re.findall(r"\d{1,3}\.\d{1,3}\.\d{1,3}.\d{1,3}", request))

def GetInternalIP():

    return str(gethostbyname(getfqdn()))

def FilteredMessage(EntryText):

    """

    Filter out all useless white lines at the end of a string,

    returns a new, beautifully filtered string.

    """

    EndFiltered = ''

    for i in range(len(EntryText)-1,-1,-1):

        if EntryText[i]!='\n':

            EndFiltered = EntryText[0:i+1]

            break

    for i in range(0,len(EndFiltered), 1):

            if EndFiltered[i] != "\n":

                    return EndFiltered[i:]+'\n'

    return ''

def LoadConnectionInfo(ChatLog, EntryText):

    if EntryText != '':

        ChatLog.config(state=NORMAL)

        if ChatLog.index('end') != None:

            ChatLog.insert(END, EntryText+'\n')

            ChatLog.config(state=DISABLED)

            ChatLog.yview(END)

def LoadMyEntry(ChatLog, EntryText):

    if EntryText != '':

        ChatLog.config(state=NORMAL)

        if ChatLog.index('end') != None:

            LineNumber = float(ChatLog.index('end'))-1.0

            ChatLog.insert(END, "Rohan : " + EntryText)

            ChatLog.tag\_add("Rohan ", LineNumber, LineNumber+0.4)

            ChatLog.tag\_config("Rohan " , foreground="#eb3434", font=("Arial", 18, "bold"))

            ChatLog.config(state=DISABLED)

            ChatLog.yview(END)

def LoadOtherEntry(ChatLog, EntryText):

    if EntryText != '':

        ChatLog.config(state=NORMAL)

        if ChatLog.index('end') != None:

            try:

                LineNumber = float(ChatLog.index('end'))-1.0

            except:

                pass

            ChatLog.insert(END, "Loki: " + EntryText)

            ChatLog.tag\_add("Loki ", LineNumber, LineNumber+0.6)

            ChatLog.tag\_config("Loki ", foreground="#34ebc0", font=("Arial", 18, "bold"))

            ChatLog.config(state=DISABLED)

            ChatLog.yview(END)

**Output:**



