Lucrare de laborator

ELABORAREA APLICAȚIILOR CU INTERFAȚĂ GRAFICĂ DE UTILIZATOR (GUI) CU AJUTORUL MODULULUI TKINTER

1. Scopul lucrării: Introducere în programarea GUI. Familiarizarea cu procedura de utilizarea a limbajului Python în cadrul programării GUI. Familiarizarea cu etapele de elaborarea a unei aplicații GUI cu ajutorul Tkinter. Elaborarea unei aplicații GUI.

2. Noțiuni teoretice

Programarea GUI presupune elaborare unor aplicații cu interfață grafică de utilizator, adică aplicații în care utilizatorul interacționează cu calculatorul prin intermediul unor ferestre ce conțin butoane, bare de introducere a textului, texte informative si altele.

Pentru elaborarea aplicațiilor GUI în limbajul Python se utilizează module specializare în acest domeniu, cele mai des utilizate fiind: Tkinter, PyQt5, Kivy, wxPython, PySimpleGUI și altele. În cadrul acestei lucrări de laborator pentru elaborarea aplicațiilor GUI se va utiliza modulul Tkinter.

Pentru a putea elabora o aplicație GUI în Python cu ajutorul Tkinter este necesar instalarea unor aplicații de lucru în calculatorul personal și anume:

- Instalarea Python (vezi **Instalarea_Python.mp4** în anexa)
- Instalarea unui mediu de dezvoltare (de exemplu PyCharm) (vezi **Instalarea_PyCharm.mp4** în anexa)
- Instalarea și importul modulelor adiționale (vezi **Instalarea_biblioteci.mp4** în anexa)

Pentru lucru cu modulul Tkinter, inițial este necesară studierea documentația acesteia (https://docs.python.org/3/library/tk.html)

sau studierea unor tutoriale de pe internet (ex: https://realpython.com/python-gui-tkinter/) sau alte materiale video (ex: https://www.youtube.com/watch?v=YXPyB4XeYLA&ab_channel=freeCodeCamp.org)

În continuare se vor explica pe scurt etapele de elaborarea a unei aplicații GUI cu ajutorul Tkinter urmând ca acestea să fie apoi aplicate în cadrul unor exemple:

a) Importul modului Tkinter

Importul modului Tkinter se realizează în mod tradițional utilizând cuvântul cheie import:

```
import tkinter
```

sau

```
import tkinter as tk
```

sau utilizând și cuvântul cheie from:

```
from tkinter import *
```

Unele submodule din Tkinter (de exemplu messagebox sau filedialog) necesită importul explicit, nefiind importate când se specifica *:

```
from tkinter import messagebox, filedialog
```

b) Formarea ferestrei de baza

După importul tkinter se creează un obiect al clasei Tk care va reprezenta fereastra de baza

```
from tkinter import *
root = Tk()
```

Pentru ca fereastra să fie afișată în mod continuu cu ajutorul obiectului clasei Tk se va apela metoda mainloop()

```
root.mainloop()
```

Odată formata fereastra de baza, aceasta poate fi personalizată de exemplu introducându-se denumirea `(title())`, dimensiunile `(geometry())` etc.

c) Crearea componentelor

După crearea ferestrei se creează componentele ce vor fi plasare pe aceasta. Tkinter dispune de mai multe componente cele mai importante fiind:

- Label – se utilizează pentru introducerea unui text ce nu poate fi modificat

```
label = Label(root, text="Salutare studenti",
fg="white", bg="black")
```

- Button – se utilizează pentru apelarea unor funții

```
button = Button(root, text="Click me!",
width=25, height=5, bg="blue", fg="yellow",
command=None)
```

- Entry – se utilizează pentru introducerea unui text pe o singura linie

```
entry = Entry(root)
```

- Text – se utilizează pentru introducerea unui text pe mai multe linii

```
text_box = Text(root)
```

- Frame – se utilizează pentru gruparea mai multor componente

```
frame = Frame(root)
```

Toate componentele enumerate (dar și altele) au ca prim parametru fereastra pe care vor fi plasate (ex. fereastra root). Acestea însă pot conține și ați parametri de personalizare în funcție de tip dar pentru mai multe detalii apelați la documentația Tkinter.

d) Amplasarea componentelor în fereastră

În Tkinter, în funcție de necesitate, componentele pot fi amplasate în fereastră prin intermediul diferitor metode:

- .pack() – plasează componentele într-o anumită ordine una după alta

```
label.pack()
button.pack()
```

- .place() – plasează componentele specificându-se coordonatele de început a acestora

```
label.place(x=0,y=0)
button.place(x=25,y=25)
```

- .grid() – divizează fereastra într-un tabel și plasează componentele specificându-se coordonatele celulei tabelului

```
label.grid(row=0,column=0)
button.grid(row=1,column=1)
```

În practică de cele mai multe ori se utilizează metoda .pack() (pentru aplicații simple) și .grid() (pentru aplicații mai complexe)

3. Exemple de aplicații GUI elaborate cu Tkinter

3.1. Să se elaboreze o aplicați GUI ce va conține un buton prin intermediul căruia se va deschide pagina google.com în browser

```
# importul modului webbrowser ce permite interactiunea cu
# browser-ul
import webbrowser as wb

# importul modului tkinter
from tkinter import *

# crearea ferestrei de baza
root = Tk()

#setarea titlului ferestrei de baza
root.title("OpenGoogle")

#setarea dimensiunilor ferestrei de baza
root.geometry("300x200")

#crearea functiei ce ar permite deschiderea browserului
def google():
    # deschiderea paginii google in browser
    wb.open("www.google.com")

# crearea butonului ce apeleaza functia de deschidere a
```

```
# browser-ului
buton=Button(root, text="Open Google", command=google)

# plasarea butonului in ferestra
buton.pack(pady=50)

# crearea buclei de afisare continua a ferestrei
root.mainloop()
```

(în anexa fișierul **Google_Opener.py**)

3.2. Să se elaboreze o aplicați GUI ce va permite afișare informațiilor despre un clip de pe Youtube și descărcarea acestuia.

Aplicația va conține un câmp în care se va introduce link-ul clipului pe youtube, un prim buton la acționarea căruia se vor afișa informațiile despre clip și un alt buton ce va descărca acest clip.

Informațiile despre clip vor include:

- titlul clipului
- lungimea clipului în formatul: min:s
- autorul ce a încărcat clipul pe youtube
- data publicării
- numărul de vizualizări
- rating-ul clipului

La introducerea greșită a link-lui aplicația va afișa un mesaj de atenționare.

Pentru salvarea clipului se va alege folderul dorit iar la final in aplicație se va afișa informația despre statutul operație de descărcare (s-a realizat cu succes sau a eșuat)

```
# importul modulelor
from pytube import YouTube
from tkinter import *
from tkinter import messagebox, filedialog

# crearea ferestrei root
root = Tk()

# setarea titlului ferestrei
root.title("Descărcător Youtube")
```

```
# setarea dimensiunilor ferestrei
root.geometry("400x290")

# blocare redimensionarii ferestrei
root.resizable(0,0)

# setarea culorii ferestrei
root.config(bg="#34ab67")

# definirea functiei de incarcare a materialului video (se va
# apela la apararea butornului de incarcare)
def load_func():
    """Functia ce incarca materialul video, extrage
informatiile despre acesta si le afiseza"""

    #introducerea blocului try-except pentru a evita aparitia
    # erorilor in cazul in care nu se introduce link
    # sau se introduce un link gresit
    try:
        # formarea unei variabile globale yt care ar pute fi
        # apelata si in afara functie
        global yt
        #crearea unui obiect al calsi Youtube
        yt = YouTube(entry_link.get())

        # inscriera informatiei despre datele clipului in
        # label-urile corespunzatoare
        label_titlu_v.config(text=yt.title)
        # sa transformat timpul din secunde in minute
        label_lung_v.config(text=f"{yt.length//60}:
        {yt.length%60}")
        label_autor_v.config(text=yt.author)
        label_data_v.config(text= yt.publish_date)
        label_vizual_v.config(text=yt.views)
        label_rating_v.config(text=yt.rating)

        # activarea label-urilor de descriere a informatie
        # afisate
        label_titlu_t.grid(row=4, column=0, sticky=E, padx=5)
        label_lung_t.grid(row=5, column=0, sticky=E, padx=5)
        label_autor_t.grid(row=6, column=0, sticky=E, padx=5)
        label_data_t.grid(row=7, column=0, sticky=E, padx=5)
        label_vizual_t.grid(row=8,    column=0,    sticky=E,
padx=5)
```

```
        label_rating_t.grid(row=9,    column=0,    sticky=E,
padx=5)

        # activarea label-urilor cu informatie despre clip
        label_titlu_v.grid(row=4, column=1, sticky=W, padx=5)
        label_lung_v.grid(row=5, column=1, sticky=W, padx=5)
        label_autor_v.grid(row=6, column=1, sticky=W, padx=5)
        label_data_v.grid(row=7, column=1, sticky=W, padx=5)
        label_vizual_v.grid(row=8,    column=1,    sticky=W,
padx=5)
        label_rating_v.grid(row=9,    column=1,    sticky=W,
padx=5)

        # activare butornului de descarcare
        button_save.grid(row=10,   column=0,   columnspan=2,
pady=5)

    except:
        # Afisare unei ferestre de atentionare cu mesajul
        # neintroducerii sau introducerii grsite a linkului
        messagebox.showinfo(title="Atentie", message="Nu ati
introdus link Youtube sau ati introdus un link gresit ")

# definirea functiei de descarcare a materialului video (se
# va apela la apararea butornului de descarcare)
def download_func():
    """Functia ce descarcare a materialul video in folderul
dorit"""

    # introducerea blocului try-except pentru a evita aparitia
    # erorilor datorate descarcarii
    try:
        #selectarea folderului unde se va salva clipul
        path    =    filedialog.askdirectory(initialdir="/",
title="Save file")
        #selectare fluxului din clip care se va salva
        yts     =     yt.streams.filter(file_extension="mp4",
progressive="True").order_by("resolution").desc().first()
        # salvarea fluxului selectat in folderul selectat
        yts.download(path)
        #Inscrierea informatiei de realizare cu succes a
        # operatie de salvare in label-ul corespunzator
        label_rez.config(text="Descărcare cu succes!!!")
        # activarea labelului ce va informa despre statutul
        # operatie de salvare
```

```
        label_rez.grid(row=11,    column=0,    columnspan=2,
pady=5)
    except:
        # Inscrierea informatiei de esuarea a operatie de
        # salvare in label-ul corespunzator
        label_rez.config(text="Descărcarea a esuat")
        # activarea labelului ce va informa despre statutul
        # operatie de salvare
        label_rez.grid(row=11,    column=0,    columnspan=2,
pady=5)

# crearea unui label in care ni se va cere sa introducem
# link-ul Youtube
label_intro = Label(root, text="Introduceți link-ul Youtube")
# activarea labelului in celula din randul 0, coloana 0 dar
# se va intinde pe 2 coloane
# continutul va si deplasat de la margini pe orizontal (padx)
# si pe vertical (pady) cu 5 si va fi plasat din stanga
# (W=vest)
label_intro.grid(row=0,   column=0,   columnspan=2,   pady=5,
sticky=W, padx=5)

# crearea unui entry in care se va introduce link-ul Youtube
entry_link = Entry(root, width=64)
# activarea entry-ului in celula din randul 1, coloana 0
entry_link.grid(row=1,   column=0,   columnspan=2,   pady=5,
padx=5)

# crearea unui buton care va incarca clipul apeland functia
# load_func
button_load    =    Button(root,    text="Încărcare    video",
command=load_func)
# activarea butonului in celula din randul 2, coloana 0
button_load.grid(row=3, column=0, columnspan=2, pady=5)

# crearea unor labelui prin intermediul carora se va specifica
# ce informatie despre clip se afiseaza
label_titlu_t = Label(root, text="Denumirea clipului:")
label_lung_t = Label(root, text="Lungimea clipului:")
label_autor_t = Label(root, text="Autorul clipului:")
label_data_t = Label(root, text="Data publicarii:")
label_vizual_t = Label(root, text="Numarul de vizualizari:")
label_rating_t = Label(root, text="Rating-ul piesei:")

# crearea unor labelui prin intermediul carora se va afisa
# informatiile despre clip
```

```
label_titlu_v = Label(root)
label_lung_v = Label(root)
label_autor_v = Label(root)
label_data_v = Label(root)
label_vizual_v = Label(root)
label_rating_v = Label(root)

# crearea unui buton care va descarca clipul apeland functia
# download_func
button_save   =   Button(root,   text="Descărcare   video",
command=download_func)
# crearea unui label in care se va afisa statutul operatie de
# salvare
label_rez = Label(root)

# crearea buclei de afisare permanentă a ferestrei
root.mainloop()
```

(în anexa fișierul **Youtube_Downloader.py**)

3.3. Să se elaboreze o aplicați GUI ce va funcționa ca un music player. Aplicația va permite următoarele funcționalități:

- Încărcarea mai multor fișier mp3 într-un playlist
- Pornirea cântecului înserat în playlist
- Oprirea cântecului care este pornit
- Punerea pe pauză și scoaterea din pauza a cântecului pornit
- Ștergerea playlist-ului

```
# importul modulelor
from tkinter import *
from tkinter import filedialog, messagebox
from pygame import mixer

# crearea ferestrei root
root=Tk()
# setarea titlului ferestrei
root.title('Music player')
# setarea dimensiunilor ferestrei
root.geometry('370x300')
# blocare redimensionarii ferestrei
root.resizable(0,0)
# setarea culorii ferestrei
```

```
root.config(bg="#228B22")
# initierea pachetului mixel pentru lucru cu muzica
mixer.init()
# crearea functiei de incarcare a cantecelor
def add_songs():
    """
    Functia de incarcarea mai multor cantece in playlist
    """
    # adaugarea cantecelor prin deschiderea directoriului
    songs=filedialog.askopenfilenames(initialdir="/",
title='Adaugare   cantece',   filetypes=(('mp3   Files'   ,
'*.mp3'),))
    # se selecteaza primul cantec din lista de cantece
    song_0 = songs[0]
    # se creaza o lista in urma divizarii directorului dupa
    # simbolul /
    lista = song_0.split("/")
    # se creaza o variabila globala care va reprezenta calea
    # catre folderul cu cantece
    global path
    # se creaza calea din uniunea cu ajutorul simbolului / a
    # tuturor elemtelor liste exceptandul pe ultimul
    path="/".join(lista[0:len(lista)-1])+"/"
    # se formeaza o bucla cu care se trece prin lista de
    #  cantece
    for song in songs:
        # selectarea denumirii cantecului din directoriu
        # cartre acesta
        songs1 = song.replace(path, '').replace(".mp3", '')
        # fixarea cantecelor in lista cu cantece
        cantec_list.insert(END,songs1)
        # se activeaza primul cantec din lista
        cantec_list.selection_set(0)

# crearea functiei de pornirea cantecului selectat
def play_song():
    """
    Functia de pornire a cantecului selectat
    """
    # verificare daca playlist-ul nu este gol
    if cantec_list.size() == 0:
        # se afiseaza mesajul de atentionare
        messagebox.showinfo(title='Informație' , message="Nu
ați incarcat nici un cântec")
    # Variabila ce specifica ca cantecul este pe pauza se
    # seteaza in false
```

```
    global paused
    paused = False

    # alegerea cantecului selectat
    song=cantec_list.get(ACTIVE)
    # adaugarea intregului directoriu catre cantec
    song = f'{path}{song}.mp3'
    # incarcarea cantelului
    mixer.music.load(song)
    #pornirea cantecului
    mixer.music.play(loops=0)

# crearea functiei de oprire a cantecului
def stop_song():
    """
    Functia de oprire a cantecului
    """
    # verificare daca playlist-ul nu este gol
    if cantec_list.size() == 0:
        # se afiseaza mesajul de atentionare
        messagebox.showinfo(title='Informație' , message="Nu
ați incarcat nici un cântec")
    # oprirea cantecului
    mixer.music.stop()

# se creaza o variabila globala ce va fixa faptul ca cantecul
# este pe pauza
paused=False

# crearea functiei de punere in pauza a cantecului
def pause_song():
    """
    Functia de pauzare a cantecului
    """
    # verificare daca playlist-ul nu este gol
    if cantec_list.size() == 0:
        # se afiseaza mesajul de atentionare
        messagebox.showinfo(title='Informație' , message="Nu
ați incarcat nici un cântec")
    #verificarea daca cantecul este pe pauza
    global paused
    if paused:
        # cantecul se scoate de pe pauza
        mixer.music.unpause()
        # se specifica ca cantecul nu mai e pe pauza
        paused = False
```

```
    # se verifica daca cantecul nu este pe pauza
    else:
        # cantecul se scoate de pe pauza
        mixer.music.pause()
        # se specifica ca cantecul e pe pauza
        paused = True

# crearea unui functii de stergere a tuturor cantecelor din
# playlist
def del_songs():
    """
    Functia de stergere a tuturor cantecelor din playlist
    """
    # oprirea cantecului care canta la moment
    stop_song()
    # stergerea tuturor cantecelor
    cantec_list.delete(0,END)

# crearea unui buton de incarcare a cantecelor in playlist
buton_incarc = Button(root, text="Încărcare cântece",
bg="blue", fg="white", command=add_songs)
buton_incarc.grid(row=0, column=0, columnspan=3, pady=5)

# crearea unui buton de sterege a cantecelor din playlist
buton_sterg = Button(root, text="Stergere cântece",
bg="blue", fg="white", command=del_songs)
buton_sterg.grid(row=1, column=0, columnspan=3, pady=5)

# crearea unui listbox pentru playlist
cantec_list=Listbox(root, width=60, bg="#9ACD32",
fg="#0000FF", selectbackground='#B22222', activestyle=None)
cantec_list.grid(row=2, column=0, columnspan=3, pady=5,
padx=3)

# crearea butonului de start
buton_start = Button(root, text="Start",
bg="green",fg="white", command=play_song)
buton_start.grid(row=3, column=0, pady=5)

# crearea butonului de start
buton_pauza = Button(root, text="Pauza",bg="yellow",
command=pause_song)
buton_pauza.grid(row=3, column=1, pady=5)

# crearea butonului de start
```

```
buton_stop = Button(root, text="Stop", bg="red",fg="white",
command=stop_song)
buton_stop.grid(row=3, column=2, pady=5)

# crearea buclei de afisare permanenta
root.mainloop()
```

(în anexa fișierul **Simple_Music_Player.py**)

4. Sarcină de laborator

Să se elaboreze o aplicație cu interfață grafica de utilizator care va avea rolul unui calculator ce efectuează operațiile:

- adunarea a 2 numere tastând butonul "+"
- scăderea unui număr din altul tastând butonul "-"
- înmulțirea a 2 numere tastând butonul "*"
- împărțirea unui număr la altul tastând butonul "/"
- determinarea restului divizării tastând butonul "%"
- ridicare unui număr la o putere tastând butonul "^"

Pe lângă butoanele operațiilor enumerate mai sus, aplicația va mai conține și butoane pentru toate cifrele (de la 0 la 9), butonul "C" de ștergere și butonul "=" pentru afișarea rezultatului.

În câmpul în care se va introduce expresia de calcul, datele vor pute fi introduse atât de la tastatura calculatorului cât și prin intermediul butoanelor aplicației.

În cazul operațiilor de divizare și de determinare a restului, dacă al doilea termen este 0 (zero) atunci se va afișa mesajul "`Eroare: împărțirea la 0`".

În cazul introducerii unei expresii greșite (nu va conține nici un semn al operației sau va conține litere în locul cifrelor) atunci se va afișa mesajul "`Eroare: expresie greșită`"

Exemple de format a expresiilor de calcul ce vor fi introduse și de format a rezultatelor după tastarea butonului "=":

Exemplu de adunare: `23+34` -> rezultat afișat: `23+34=57`

Exemplu de scădere: `50-18` -> rezultat afișat: `50-18=32`

Exemplu de înmulțire: `24*2` -> rezultat afișat: `24*2=48`

Exemplu de împărțire: `15/2` -> rezultat afișat: `15/2=7.5`

Exemplu de determinare rest: `23%3` -> rezultat afișat: `23%3=2`

Exemplu de ridicare la putere: `3^3` -> rezultat afișat: `3^3=27`

Exemplu de împărțire la 0: `35/0` -> rezultat afișat: `Eroare: împărțirea la 0`

Exemplu de expresie greșită: `3t+3y` -> rezultat afișat: `Eroare: expresie greșită`

Aplicația se va personaliza (fixarea culorilor, dimensiunilor, aspectului butoanelor, etc) individual de către fiecare student

Opțional se poate adăuga un buton "." care va permite realizarea tuturor operațiilor cu numere flotante (cu virgulă) în locul numerelor întregi.

Rezultatele se transmit într-un fișier cu format .py pe poșta electronică a profesorului.

Aprecierea rezultatelor se va realiza în funcție de respectarea tuturor cerințelor sarcinii.