Using os library to run multiple script’s at the same time

files in this project

Python scripts:

1.main.py
2.lot.py
3.fifo.py
4.filo.py
5.display_tk_V3.py

note that fifo, filo and lot can be swapped for other algorithm’s

temp files that allow scripts to communicate with each other:

1.input.txt
2.result.txt
3.status.txt

python os library

os library allows us to run terminal and shell command in python code
os.system(command) is for running one script and then waiting for the interpreter to finish
while os.popen(command) is for running multiple scripts at the sames time, by using os.popen(command).write() we can write args to scripts and by using os.popen(command).read() we can read the result and the log of the script

we start the program by running main.py

main.py

```
import os
import time
mypass = "0"
os.system("rm -rf result.txt && touch result.txt")
os.system("echo '0' > status.txt")

files_done = 0

command_1 = "python3 filo1.py"
command_2 = "python3 fifo1.py"
command_3 = "python3 lot1.py"

os.popen(command_1)
os.popen(command_2)
os.popen(command_3)

#os.popen("sudo -S %s"%(command_1),"w").write(mypass)
#os.popen("sudo -S %s"%(command_2),"w").write(mypass)
#os.popen("sudo -S %s"%(command_3),"w").write(mypass)

time.sleep(0.2)

while files_done != 3:
    st = os.popen("cat status.txt").readlines()

   #print("******" + str(files_done))
    if "1" in str(st):
        files_done += 1
        print(files_done)
print("All 3 Files are done!!")
time.sleep(2.3)

os.system("python3 display_tk_V3.py")
```

In this file we import os and time , then by using os.system we run rm -rf result.txt to delete previous result and touch result.txt to creat new result.txt file and echo 0 > status.txt to reset the status signal
the we use os.popen() to run all three algorithm commands

while loop is used to wait for all three algorithm to finish running then finaly by using os.system() , display_tk_V3.py get get started.

If you are using GNU Linux system like Fedora , Ubuntu , Arch etc
some might require sudo permission to fix this issue store your system password
in mypass var and run the command using popen().write(mypass):

```
os.popen("sudo -S %s"%(command_1),"w").write(mypass)
```

Input.txt

```
[["P1",10,0],["P2",20,2],["P3",100,1],["P4",200,4]]
```

this file is used to store the input values in a nested list format
there are four list and each of them have three items

PID [P1,P2,P3,P4]
Burst time or compute time [10,20,100,200]
Arrival time [0,2,1,4]

since we are using fifo filo and lot , Arrival time is not needed

Json lib is used to read input.txt file

```
Import  json

with open("input.txt",encoding="utf-8") as file_in:

    list_file_r = json.load(file_in)
```

lot1

```
import json
import os
import time
import random
with open("input.txt",encoding="utf-8") as file_in:
    #list_file_r = str(file_in.read().splitlines())
    list_file_r = json.load(file_in)
    #print(list_file_r)
    #list_file_r = "["+list_file_r[2:-2]+"]"
    #print(list_file_r)

result_list = []
def lot1(lot1_l):
    palceholder_list1 = []
    for i in range(len(lot1_l)):
        for j in range(lot1_l[i][1]):
            palceholder_list1.append(lot1_l[i][0])

    #print(palceholder_list1)
    #rand_list = shuffle(palceholder_list1)

    for i in range(len(palceholder_list1)):
        x = palceholder_list1.pop(random.randint(0,len(palceholder_list1)-1))
        global result_list
        result_list.append(x)
        #print("C"+x)
        time.sleep(0.01)
        command_result = str("echo 'C"+ x + "' >> result.txt")
        os.popen(command_result)

lot1(list_file_r)

os.popen("echo '1' > status.txt")

exit()
```

First json, os, time and random are imported
then input.txt is loaded and stored in list_file_r var using json lib

def lot1 generate a long list of pids and by using random lib , rearange them randomly then by using echo and os lib results are written in result.txt

```
command_result = str("echo 'C"+ x + "'" >> result.txt")
os.popen(command_result)
```

And lastly after finishing main.py get a done signal

```
os.popen("echo '1' > status.txt")
```

main.py reads the signal from status.txt and when all three algorithms are done GUI get started

```
while files_done != 3:
    st = os.popen("cat status.txt").readlines()
    if "1" in str(st):
        files_done += 1
        print(files_done)

print("All 3 Files are done!!")

time.sleep(2.3)

os.system("python3 display_tk_V3.py")
```

result.txt

```
BP4
CP3
AP1
BP4
CP4
AP1
BP4
CP3
AP1
BP4
CP4
AP1
CP4
BP4
AP1
BP4
CP4
AP1
BP4
CP3
AP1
BP4
CP4
AP1
BP4
CP2
AP1
BP4
CP4
AP1
…...etc
```

all three algorithms work simultaneously so Chars are used to distinguish them

Each algorithm gets a Char:

A for fifo
B for filo
C for lottery

for example CP3 is a lottery process with PID 3
BP1 is a filo process with PID 1

display_tk_V3.py

```
from tkinter import *
from tkinter import ttk
import time
import json

time.sleep(0.5)

with open("input.txt",encoding="utf-8") as file_in:
    #list_file_r = str(file_in.read().splitlines())
    list_file_r = json.load(file_in)

total_1 = list_file_r[0][1]
total_2 = list_file_r[1][1]
total_3 = list_file_r[2][1]
total_4 = list_file_r[3][1]

ap1 = 0
ap2 = 0
```

```
ap3 = 0
ap4 = 0

bp1 = 0
bp2 = 0
bp3 = 0
bp4 = 0

cp1 = 0
cp2 = 0
cp3 = 0
cp4 = 0

label_fg = "#000000"
label_bg = "#3d828a"

font_b = ("Fira Code",15)
font_s = ("Fira Code",12)

root = Tk()
root.title("promax1st")
root.geometry("1000x1000")
root.config(bg="#1a1b26")


fifo_frame = LabelFrame(root,text="SJF",font=font_b,bd=1,bg=label_bg,fg=label_fg)
fifo_frame.pack(side=TOP,padx=5,pady=20)
filo_frame = LabelFrame(root,text="LJF",font=font_b,bd=1,bg=label_bg,fg=label_fg)
filo_frame.pack(padx=5,pady=20)
lot_frame = LabelFrame(root,text="LOT",font=font_b,bd=1,bg=label_bg,fg=label_fg)
lot_frame.pack(padx=5,pady=20)

TROUGH_COLOR = "black"
BAR_COLOR = "#00fff7"


s = ttk.Style()
s.theme_use('clam')
s.configure("green.Horizontal.TProgressbar", troughcolor=TROUGH_COLOR,
                bordercolor=TROUGH_COLOR, background=BAR_COLOR,
lightcolor=BAR_COLOR,
                darkcolor=BAR_COLOR)
```

```
a1d = StringVar()
a1d.set("P1: 0%")
a2d = StringVar()
a2d.set("P2: 0%")
a3d = StringVar()
a3d.set("P3: 0%")
a4d = StringVar()
a4d.set("P4: 0%")

b1d = StringVar()
b1d.set("P1: 0%")
b2d = StringVar()
b2d.set("P2: 0%")
b3d = StringVar()
b3d.set("P3: 0%")
b4d = StringVar()
b4d.set("P4: 0%")

c1d = StringVar()
c1d.set("P1: 0%")
c2d = StringVar()
c2d.set("P2: 0%")
c3d = StringVar()
c3d.set("P3: 0%")
c4d = StringVar()
c4d.set("P4: 0%")

per_1 = StringVar()
per_2 = StringVar()
per_3 = StringVar()
per_4 = StringVar()


l_ap1 = Label(fifo_frame,textvariable=a1d,font=font_s,bg=label_bg)
l_ap1.pack()

p_ap1 = ttk.Progressbar(fifo_frame,style="green.Horizontal.TProgressbar",
orient="horizontal", length=300, mode="determinate")
p_ap1.pack(pady=10,padx=20)
```

```
l_ap2 = Label(fifo_frame,textvariable=a2d,font=font_s,bg=label_bg)
l_ap2.pack()

p_ap2 = ttk.Progressbar(fifo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_ap2.pack(pady=10)


l_ap3 = Label(fifo_frame,textvariable=a3d,font=font_s,bg=label_bg)
l_ap3.pack()

p_ap3 = ttk.Progressbar(fifo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_ap3.pack(pady=10)


l_ap4 = Label(fifo_frame,textvariable=a4d,font=font_s,bg=label_bg)
l_ap4.pack()

p_ap4 = ttk.Progressbar(fifo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_ap4.pack(pady=10)


l_bp1 = Label(filo_frame,textvariable=b1d,font=font_s,bg=label_bg)
l_bp1.pack()

p_bp1 = ttk.Progressbar(filo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_bp1.pack(pady=10,padx=20)

l_bp2 = Label(filo_frame,textvariable=b2d,font=font_s,bg=label_bg)
l_bp2.pack()

p_bp2 = ttk.Progressbar(filo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_bp2.pack(pady=10)

l_bp3 = Label(filo_frame,textvariable=b3d,font=font_s,bg=label_bg)
l_bp3.pack()

p_bp3 = ttk.Progressbar(filo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_bp3.pack(pady=10)

l_bp4 = Label(filo_frame,textvariable=b4d,font=font_s,bg=label_bg)
l_bp4.pack()

p_bp4 = ttk.Progressbar(filo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_bp4.pack(pady=10)



l_cp1 = Label(lot_frame,textvariable=c1d,font=font_s,bg=label_bg)
l_cp1.pack()

p_cp1 = ttk.Progressbar(lot_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_cp1.pack(pady=10,padx=20)

l_cp2 = Label(lot_frame,textvariable=c2d,font=font_s,bg=label_bg)
l_cp2.pack()

p_cp2 = ttk.Progressbar(lot_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_cp2.pack(pady=10)

l_cp3 = Label(lot_frame,textvariable=c3d,font=font_s,bg=label_bg)
l_cp3.pack()

p_cp3 = ttk.Progressbar(lot_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_cp3.pack(pady=10)

l_cp4 = Label(lot_frame,textvariable=c4d,font=font_s,bg=label_bg)
l_cp4.pack()

p_cp4 = ttk.Progressbar(lot_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_cp4.pack(pady=10)


l_name = Label(root,text="Alireza Mansoori",font=font_b,bg="#1a1b26",fg="#88ddff")
#l_name.anchor('se')
l_name.pack(side=BOTTOM)




p_ap1.start()
p_ap2.start()
p_ap3.start()
p_ap4.start()
p_bp1.start()
p_bp2.start()
p_bp3.start()
p_bp4.start()
p_cp1.start()
p_cp2.start()
p_cp3.start()
p_cp4.start()


p_ap1['value'] = 0
p_ap2['value'] = 0
p_ap3['value'] = 0
p_ap4['value'] = 0
p_bp1['value'] = 0
p_bp2['value'] = 0
p_bp3['value'] = 0
p_bp4['value'] = 0
p_cp1['value'] = 0
```

```
p_cp2['value'] = 0
p_cp3['value'] = 0
p_cp4['value'] = 0



with open("result.txt") as in_file:
    for line in in_file:
        line = line.strip()
        print(line)
        time.sleep((total_1+total_2+total_3+total_4)/10000)
        if line == "AP1":
            ap1 += 1
            a1d.set("P1: " + str(int((ap1/total_1)*100))+"%")
            p_ap1['value'] = (ap1/total_1)*100
            #print(int((ap1/total_1)*100))
        elif line == "AP2":
            ap2 += 1

            a2d.set("P2: " + str(int((ap2/total_2)*100))+"%")
            p_ap2['value'] = (ap2/total_2)*100

        elif line == "AP3":
            ap3 += 1
            a3d.set("P3: " + str(int((ap3/total_3)*100))+"%")
            p_ap3['value'] = (ap3/total_3)*100

        elif line == "AP4":
            ap4 += 1
            a4d.set("P4: " + str(int((ap4/total_4)*100))+"%")
            p_ap4['value'] = (ap4/total_4)*100

        elif line == "BP1":
            bp1 += 1
            b1d.set("P1: " + str(int((bp1/total_1)*100))+"%")
            p_bp1['value'] = (bp1/total_1)*100

        elif line == "BP2":
            bp2 += 1
            b2d.set("P2: " + str(int((bp2/total_2)*100))+"%")
            p_bp2['value'] = (bp2/total_2)*100
        elif line == "BP3":
            bp3 += 1
            b3d.set("P3: " + str(int((bp3/total_3)*100))+"%")
            p_bp3['value'] = (bp3/total_3)*100

        elif line == "BP4":
            bp4 += 1
            b4d.set("P4: " + str(int((bp4/total_4)*100))+"%")
            p_bp4['value'] = (bp4/total_4)*100


        elif line == "CP1":
            cp1 += 1
            c1d.set("P1: " + str(int((cp1/total_1)*100))+"%")
            p_cp1['value'] = (cp1/total_1)*100

        elif line == "CP2":
            cp2 += 1
            c2d.set("P2: " + str(int((cp2/total_2)*100))+"%")
            p_cp2['value'] = (cp2/total_2)*100

        elif line == "CP3":
            cp3 += 1
            c3d.set("P3: " + str(int((cp3/total_3)*100))+"%")
            p_cp3['value'] = (cp3/total_3)*100

        elif line == "CP4":
            cp4 += 1
            c4d.set("P4: " + str(int((cp4/total_4)*100))+"%")
            p_cp4['value'] = (cp4/total_4)*100


        root.update_idletasks()
        #fifo_frame.update_idletasks()
        #filo_frame.update_idletasks()
        #lot_frame.update_idletasks()

#time.sleep(20)
p_ap1.stop()
p_ap2.stop()
p_ap3.stop()
p_ap4.stop()
```

```
p_bp1.stop()
p_bp2.stop()
p_bp3.stop()
p_bp4.stop()
p_cp1.stop()
p_cp2.stop()
p_cp3.stop()
p_cp4.stop()

p_ap1['value'] = 100
p_ap2['value'] = 100
p_ap3['value'] = 100
p_ap4['value'] = 100
p_bp1['value'] = 100
p_bp2['value'] = 100
p_bp3['value'] = 100
p_bp4['value'] = 100
p_cp1['value'] = 100
p_cp2['value'] = 100
p_cp3['value'] = 100
p_cp4['value'] = 100


print("all " + str(total_1+total_2+total_3+total_4) + " process  are complited!")

root.mainloop()
```

Tkinter lib is used for the GUI

each algorithm display its values in a frame
and within these frames four progressbars are used for each PID
so 12 progressbar in total

to calculate progress for PIDs:
you divide current process by its total burst time then multiply it by 100
(you can get starting burts time by reading input.txt file using json)

```
with open("input.txt",encoding="utf-8") as file_in:
    #list_file_r = str(file_in.read().splitlines())
    list_file_r = json.load(file_in)

total_1 = list_file_r[0][1]
total_2 = list_file_r[1][1]
total_3 = list_file_r[2][1]
total_4 = list_file_r[3][1]
```

```
a1d.set("P1: " + str(int((ap1/total_1)*100))+"%")
            p_ap1['value'] = (ap1/total_1)*100
```

Fonts , colors and the window geometry are configerd using vars

```
label_fg = "#000000"
label_bg = "#3d828a"

font_b = ("Fira Code",15)
font_s = ("Fira Code",12)

root = Tk()
root.title("promax1st")
root.geometry("1000x1000")
root.config(bg="#1a1b26")
```

Progress bar color configerd using tkk.Style()

```
TROUGH_COLOR = "black"
BAR_COLOR = "#00fff7"



s = ttk.Style()
s.theme_use('clam')
```

```
s.configure("green.Horizontal.TProgressbar", troughcolor=TROUGH_COLOR,
              bordercolor=TROUGH_COLOR, background=BAR_COLOR,
lightcolor=BAR_COLOR,
              darkcolor=BAR_COLOR)
```

Each PID get pecentage lable and a progressbars

for example PID 2 , fifo:

```
l_ap2 = Label(fifo_frame,textvariable=a2d,font=font_s,bg=label_bg)
l_ap2.pack()

p_ap2 = ttk.Progressbar(fifo_frame,
style="green.Horizontal.TProgressbar",orient="horizontal", length=300,
mode="determinate")
p_ap2.pack(pady=10)
```

Alireza Mansoori

to see an example of the code running click this text