**18/10/2024**

pyinstaller -n “Scoring Game” 🡪 if wanted to give the name with space, use the name inside “”, else, simply type the name without “”.

pyinstaller -n or --name “Scoring Game” -F or --onefile scoring\_game.py 🡪 To get the .exe file of the python code.

pyinstaller –distpath (full path like - E:\Python\chapters\APPS) 🡪 This will get the .exe file executable under Folder (APPS)

pyinstaller -i (path like - E:\Python\chapters\APPS\pic.jpg) 🡪 This will give the .exe file icon of the provided image but for this **pip install Pillow** needs to be installed.(Pillow needs to be installed to convert the .jpg file to .ico file)

**17/11/2024**

Made the Flow Chart in Google sheets Drawing and then Slept.

**18/11/2024 (updated SECTION and pulled using git pull )**

Learned to update .docx in vs code codespace(GitHub)

1. download the .docx file from GitHub.
2. Make changes to that file.
3. Re-upload it to GitHub.
4. git pull it to the codespace (git pull origin main)
5. View that file using Office viewer(markdown) extension already installed in your created Code space.

**\*\*GitHub Learning\*\***

-

18/11/2024--> status|commit|push from terminal itself

git status (to check if any changes is there or not)

git add file\_name (if the file is not added to the repository)

git add . (for the stages of changes made)

git commit -m "you message to show up"

git push

for adding any new file also, do the same.

git restore file\_name (to restore the program before git push command)

**if getting error while git push,**

use--> git fetch origin

git status | git merge origin/main

git add <file name and so on..>

git push origin main | git push origin main --force (if you are absolutely certain that you won't lose important changes that might be present on the remote branch. )

**To pull any changes that are made in the directory into the codespace**

git pull origin main

git add <file name>

-

**VDD**

**Category wise**

* **SUMIF**(category1, *what to search*,prices\numbers)
* **IF**(condition1(met),print this(could be a text | blank space | a value of a cell),Else print this same way)
* For Other category use **SUMPRODUCT**🡪 sumproduct(range(B2:B)<>”category name” |value of a cell)\*(*range*(B2:B)<>”category2 name” | value of a cell)\*…,Price\number range(*form responses*) + sumproduct(range(B2:B)=”Other”, Price range(*form responses*))
* <> means Not Equal to

**Daily wise**

* Already defined a text date in A2, A3,… and add one day to the below cell like that.
* Price column 🡪 **sumif(**range of all the entries of the timestampt(Raw daily sheet’!A:A)**,**A2(daily dates, not the each entries)**,**range of all the price entries (Raw daily sheet’!B:B)**)**
* To sum each Price Entries for the Each Time Entries, and then A2 is each day DATES(daily)

**Raw daily sheet**

* Arrayformula(Range of all the Price Entries | Range of all the Time Entries)
* Use ARRAYFORMULA to automatically prints what comes in the range of cell provided (i.e. form responses range)
* For the dates Arrayformula(DATE(Year(range),Month(range),Day(Range))

**Bill**

* Using arrayformula(B6:B100-B$5), to get the B5 cell locked, so increase B6-B5, B7-B5 and get the total units from cell 5 till the ***Entries***
* **Units for daily** 🡪 if((D7-D6)>0,D7-D6,””), get the latest two subtracted by each other and if greater than 0 then print, else not.

**FDD**

* Used **iferror**(if condition is met then 🡪 condition, elif some error then in the place of that error print whatever you want)
* Multiple times used, **if**(range🡪fdd[study] =”Yes”,15,-20) with **iferror**
* used mod(value,divisor)=remainder , if this condition of row() is true then print the vale of sum, else not 🡪 if(**mod**(row(2)-row(o$2),7)=0,sum(02:0$2),””) This means to print the value of sum after every 7th cell
* Multiple conditional formatting for different colors to show for 🡪 text starts with | >= | range | etc..
* **=IF(ROW()=9, SUM(O2:O9), IF(MOD(ROW(), 7) = 2, SUM(OFFSET(O$2, (ROW()-8)/7\*7, 0, 7)), ""))** this is what not understandable, but it is for sum of every new week score.
* Then totally the score, giving percentage with Concatenate. Giving a target, then calculating the percentage for the total score as of now.

\*\*NOT DONE\*\*

In \*\***pandas\_dataframe\_timeslots.py**\*\*

-

**\*\*Disclaimer 1-->\*\*** Ask for time slots and ask 7 or till user

did not want to stop using while or for loop, whichever suits you.

\*\*Disclaimer 2-->\*\* Make this data printed as a schedule.csv or xlsx as schedule 1, Schedule 2 or Date 1, Date 2 (Learn Read and write to the file using python)

\*\*Disclaimer 3-->\*\* Use the append method to store N number of time slots and also to update and delete use the list.

\*\*Disclaimer 3-->\*\* As soon as you have typed all your Slots, Display it also. So, 1<sup>st</sup> Table for Time Slot display & 2<sup>nd</sup> for Full Schedule Display.

\*\*Disclaimer 4-->\*\* Make def functions for \*\*ias={tslots,full\_schedule,update:(tslots,dt(defined tasks)),delete-a slot (dt\_data.drop(tslotslist[user\_input\_S.No-1]),rename}\*\* and then the name of the def function in the dictionary and ask the user, if the user wanted to add the timeslots, defined tasks, reminders|Help. Then call the action taken by the user as ias[update], ias[full\_schedule]

\*\*1-->\*\* df = df.rename(columns={'A': 'X', 'B': 'Y'}) or| df.rename(columns={'A': 'X'}, index={'X': 'W'}, inplace=True) or| df.index.name = 'Index\_Name'

-

1--> Use User input to ask for \*\*Time Slots\*\* as t1,t2,t3... t7 or user input until user want's to stop

2--> Same as above, ask user to enter the \*\*Defined Tasks\*\* as d1,d2,d3... same number as \*\*Time Slots\*\*

3--> Same goes with \*\*Reminders\*\* as r1,r2... same number as \*\*Time Slots\*\*

4--> Use data\_dict={"Time Slots":[t1,t2,t3,...],"Defined Tasks":[d1,d2,d3....],"Reminders":[r1,r2,r3....]}

\*\*Second Normal approach\*\* --> Typing in the schedule simply to the dictionary

------------------------------------------------------------------------------------------------------------------------------------------------------------------

\*\*NOT DONE\*\*

In \*\*A New File of Python\*\*

use the CSV or xlsx file to retrieve the data from it and use it in python using the pandas \*\*DataFrame\*\* or any other library. maybe this is in the \*\*Python TuteDude course\*\*

------------------------------------------------------------------------------------------------------------------------------------------------------------------

\*\*NOT DONE\*\*

Get the wifi names and the passwords for all the wifi connected to the laptop using python

------------------------------------------------------------------------------------------------------------------------------------------------------------------

\*\*NOT DONE\*\*

get your \*\*ideas\_while\_work.md\*\* file copy paste in your \*\*JFoldknewme.docx\*\* file saved in Drive and see what happens.

------------------------------------------------------------------------------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------------------------------------------------------------------------------