Shreya Padisetty

I made a library checkout system which the user can do by themselves and can expect the items they need to be ready to be ready for pickup by the time they arrive. This avoids any interaction with the staff at the library and allows the user to have an item ready for pickup to get later if they aren’t able to check it out right away. It also helps with putting an item on hold so the user knows they have it checked out to them instead of going to the library and finding out its all checked out. Once you run the program, it welcomes you to the library. I wanted to keep this part realistic so I made the program alternate between sleeping for half a second and a period. I made a nested dictionary to keep everything organized so I could have the categories of the items and have a key and value pair of a number and an item in that category. This helps me easily get the value of the item the user is trying to check out. I print out this list using a for loop. It then asks if you want to check out anything or not. If you say no, it ends the program but if you say yes, it asks what items you want to check out. By inputting a number from 1-15, the user is able to select an item from the list. I included a function to check if the user’s input is a valid option for the question prompted by the program. There are two options for this depending on if the input required is a yes or no question or if the user needs to input a range of numbers. The program runs in a while loop where the program prompts the user if they still want to check out and which item. Once done selecting items, the program prints out what items the user has checked out, how many items, and how long it took to check everything out. I also wanted to keep this part realistic and made the program sleep for a second to show that it’s loading all the information. I used the modules asyncio, time, and timeit. I used asyncio for the code at the end and during sleep, it alternates what code is being run between the two functions. This is where it prints the summary where it the program checks out these items and prints out what the user checked out. Asyncio sleeps the program and in that time, the program moves to the other function and prints out the total number of items. Then the function uses asyncio sleep, so this function is halted and then prints you have successfully checked out these items. Then it finally moves back to the other function where it prints out the transaction time. With time I used it for sleeping the function to make it feel realistic because there’s usually a loading screen. I used timeit to find the execution time of the code where I used asynchronous programming. I put the value of it as how long the transaction took. This code is when the used has finished checking out and we get a summary of all the items. There are a lot of limitations to this program. Checking the validity of the input is not completely accurate, I would make sure that when the input should be a number, letters won’t cause an error and the program would prompt the user for valid input. If I were to redo this project and implement code to help these limitations, I would try to create a system where you can make your own account and be able to check what you’ve checked out and you can return it. I think I would try to implement concurrency and also make the asynchronous part better. The program could have run without the asynchronous part but as I was unsure what to do for asynchronous programming, I had the program run code alternatively in two functions. To improve the project, I would try to find where I could use asynchronous programming and implement it there. I would probably use concurrency to keep a track of what items were checked out when as the user is checking out items. So, when the summary is printed, it’s also printed to a document where it’s like a checked out log.

IMPORT time

IMPORT timeit

IMPORT asyncio

FUNCTION validity with parameters varb, type

IF type is equal to “alpha”

WHILE varb does not equal ‘y’ and varb does not equal ‘n’

varb := please enter a valid option y or n (converted to lowercase)

ELIF type == “digit”

WHILE varb is less than 1 or varb is greater than 15

varb := please enter a valid option (converted to integer)

RETURN varb

prints welcome to the library!

prints here is a list of items you may checkout:

sleeping the program for 0.5 seconds

FOR x in range 3

prints .

sleeping the program for 0.5 seconds

items = a nested dictionary with the categories: textbooks, calculators, cameras, and games where they have items under each category

FOR x in items

prints a long line of –

prints x

prints a long line of –

FOR y in each category of the dictionary

prints the value of y converted to a string, :, and the value of the number given from the category

continue\_program := would you like to check something out y or n

continue\_program := the value after checking if the input is valid

checkedout\_dict := an empty dictionary

IF continue\_program is equal to ‘n’

prints thank you for visiting the library!

quits the program

ELSE

WHILE continue\_program is equal to ‘y’

prints a long line of –

item\_checkout := what would you like to check out in a range of 1-15 (converted to integer)

item\_checkout := the value after checking if the input is valid

IF item\_checkout is less than or equal to 5

in the checkedout\_dict a key is made from item\_checkout and is set to the value it is from in the category textbooks

ELIF item\_checkout is greater than 5 and less than or equal to 8

in the checkedout\_dict a key is made from item\_checkout and is set to the value it is from in the category calculators

ELIF item\_checkout is greater than 8 and less than or equal to 10

in the checkedout\_dict a key is made from item\_checkout and is set to the value it is from in the category cameras

ELIF item\_checkout is greater than 10 and less than or equal to 15

in the checkedout\_dict a key is made from item\_checkout and is set to the value it is from in the category games

continue\_program := would you like to check something else out y or n (converted to lowercase)

continue\_program := the value after checking if the input is valid

prints a long line of –

ASYNC FUNCTION async\_checkingout

prints you have a total of (number of items that were checked out) items

sleeps the function for 5 seconds

prints your transaction took (execution time) seconds

prints your items will be ready in an hour

ASYNC FUNCTION main

creating the task async\_checkingout

prints checking out items this may take a second

sleeping the program for 1 second

prints here is a list of items you have checked out:

FOR x, y in checkedout\_dict items

prints y

prints you have successfully checked out these items

loop := getting the event loop

loop will run until main is completed

prints have a nice day!