## Katrina Hauser

## 29 August 2021

## BIDD Fundamentals of Programming (Python)

## Assignment 08

# Assignment 08: Object Oriented Programing (OOP)

## Introduction

In Assignment 08, the focus was on object oriented programming and creating/working with objects in Python. To understand this functionality, several videos and references were reviewed.

## Topic 1: Python Features Review

The following Python modules were reviewed in addition to reading the second chapter of the course textbook: <https://saravji.github.io/saravjis_hut/FDN_Prog/Modules.html> - Module 8 only.

The following websites were reviewed and videos watched:

<https://www.youtube.com/watch?v=IHaTbJPdB-s>

1. What is the difference between a class and the objects made from a class?
2. What are the components that make up the standard pattern of a class?
3. What is the purpose of a class constructor?
4. When do you use the keyword "self"?
5. When do you use the keyword "@staticmethod"?
6. How are fields and attributes and property functions related?
7. What is the difference between a property and a method?
8. Why do you include a docstring in a class?

## Topic 2: Python Script Development

To demonstrate knowledge learned from the modules and references above, the previous version of the CD\_Inventory program that uses an inner data structure list of dictionaries was modified. In the modified version, the list of dicts was replaced with a list of objects. The focus of this assignment was to create the CD Inventory script similar to Assignment07 that uses object oriented programming and implement error handling, where applicable.

From the previous script, a menu option is presented to the user to control the CD inventory. The script loops through the options and executes the commands selected by the user. If the user enters an incorrect value, the script will catch this and present an error message. The user will be asked to enter a correct value. The basic looping and structure of the previous script was maintained.

The code was previously structured into classes – DataProcessor, FileIO and IO - with the following functions – add\_cd, load\_inventorys, save\_inventory, print\_menu, menu\_choice, show\_inventory and get\_cd\_info. The focus of the last assignment was the implement structured error handing. Classes had been implemented in Assignment 6 to separate the computer program into distinct sections such that each section addresses a separate concern. In the current assignment, a new class to setup the object was implemented. Functions involving reading and writing file data from .txt were modified to create objects to replace the dict structure that had been previously implemented.

Walking through the script, in the first section is code to setup the CD object. Properties and methods are created. Properties implemented are ID, title and album.

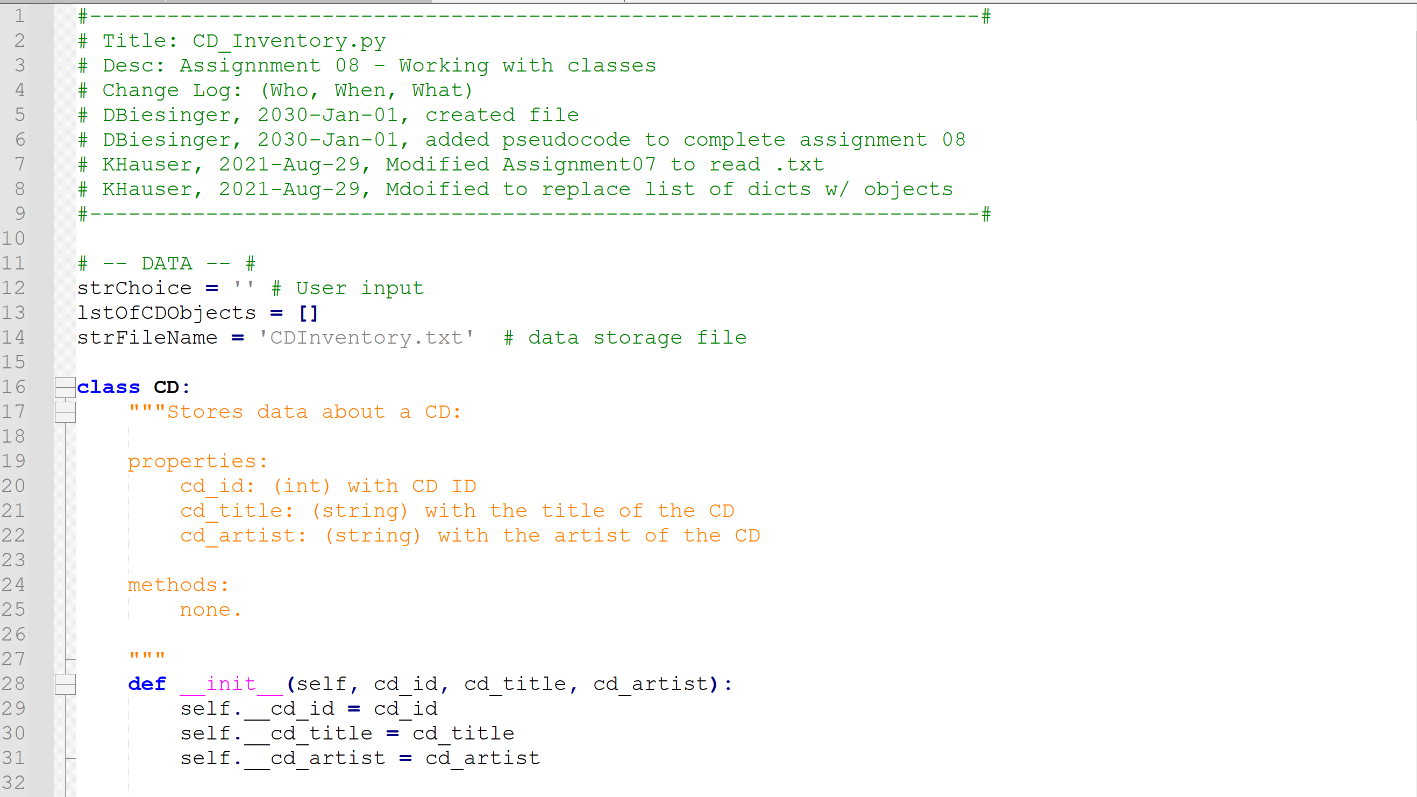


Figure 1: Class CD and init function to setup object with ID, title and album to replace dict structure.

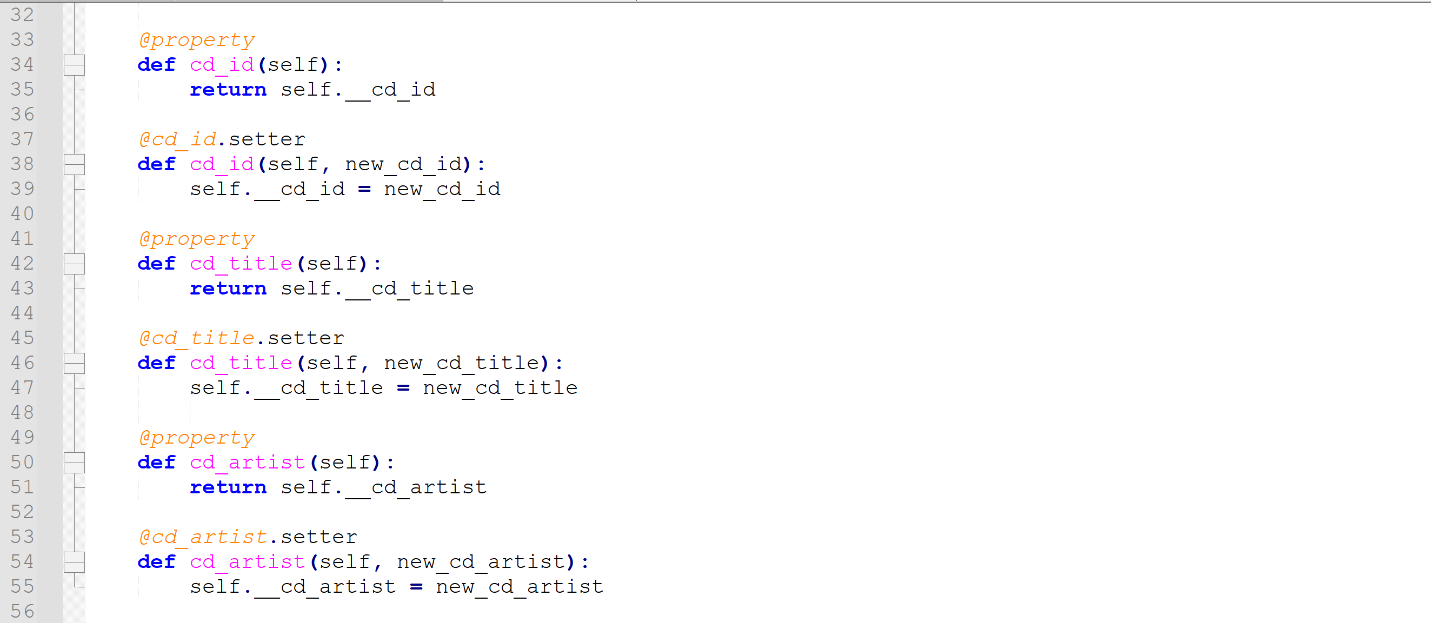


Figure 2: CD class sets up object with ID, title and album properties.

The DataProcessor class with the add\_inventory function focus is on processing the data in the internal data structure that is a list of objects. The user is provided functionality to add cds from this list (list of objects). Structured error handling added to catch type error if user enters value other than integer. Functionality is implemented to add to empty list and save that list to create new CDInventory.txt file. This allows a user to create a new list and save it, if one does not exist.

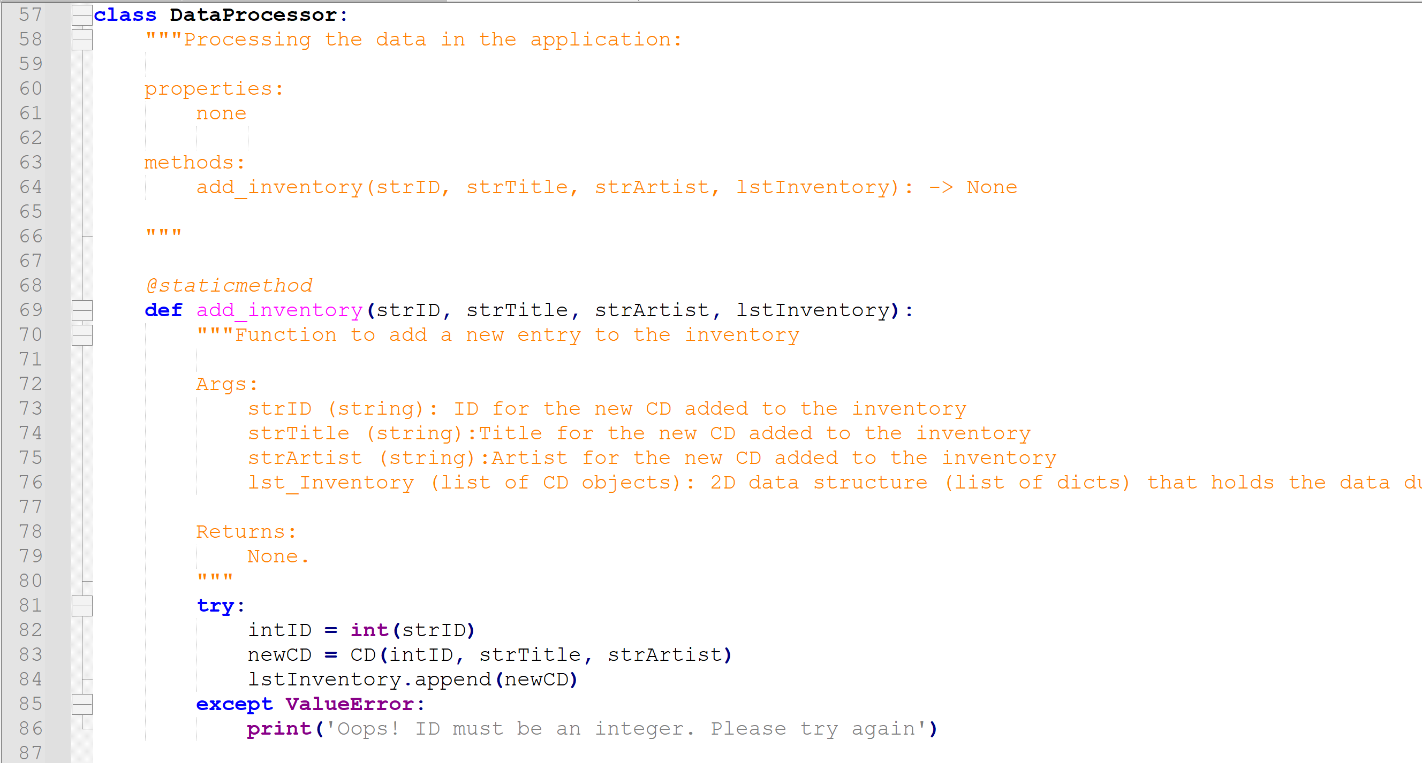


Figure 3: FileProcessor class add\_inventory function creates object with ID, title and album. Data is saved as a list of objects.

The FIleIO class with the load\_inventory and save\_inventory functions focus is on reading and writing to the CDInventory.txt.

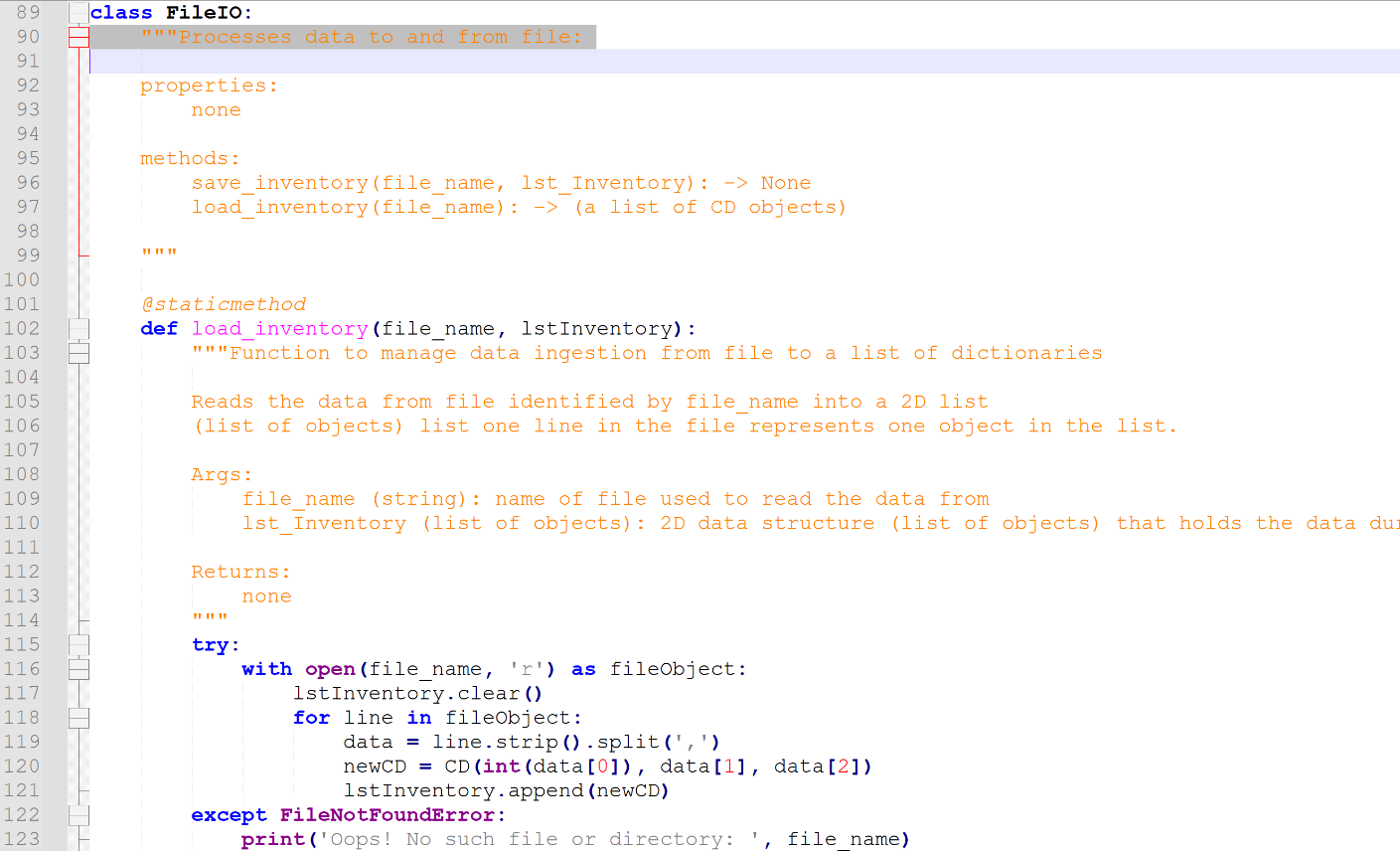


Figure 4: FileProcessor class load\_inventory imports data from .txt file into a list of objects. Error handling to caught file not found issues in addition to other load issue that can arise if the file fails to initially load.

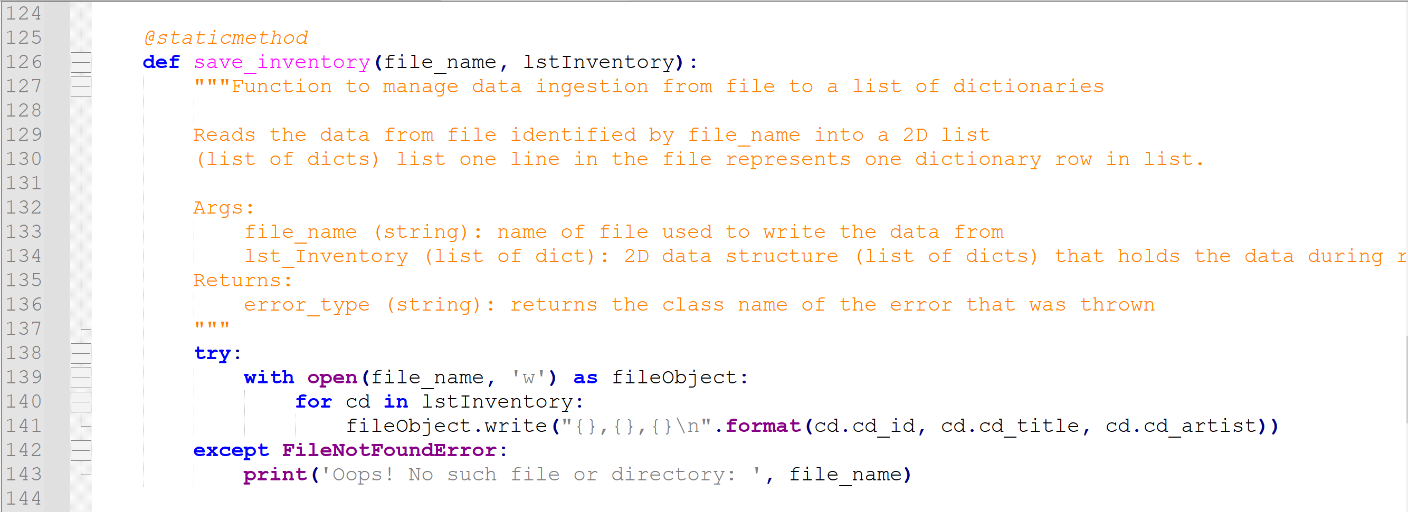


Figure 5: IO class save\_inventory function saves CDs to file.

The IO class with print\_menu, menu\_choice, show\_inventory and get\_cd\_info functions focus is on input and output to the user. Error handling added to catch when list is empty and the user chooses to display list.

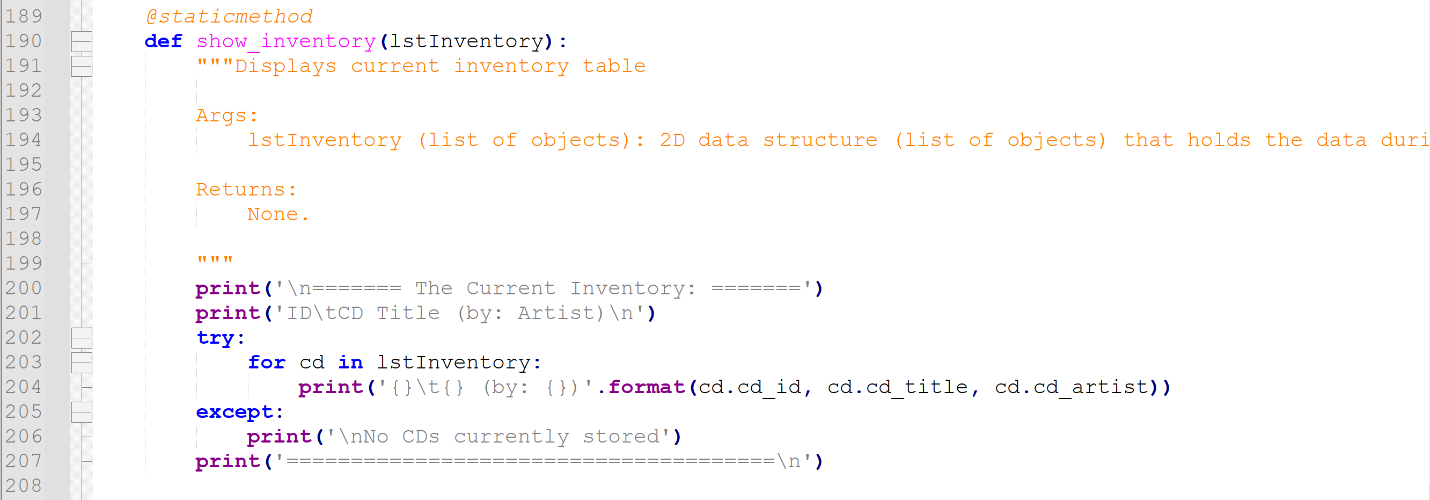


Figure 6: Function show\_inventory to display list of objects.

To start, variables, lists and dictionaries are initialized. Error handing is managed in the functions. The is serialized into binary files and read in using pickle.

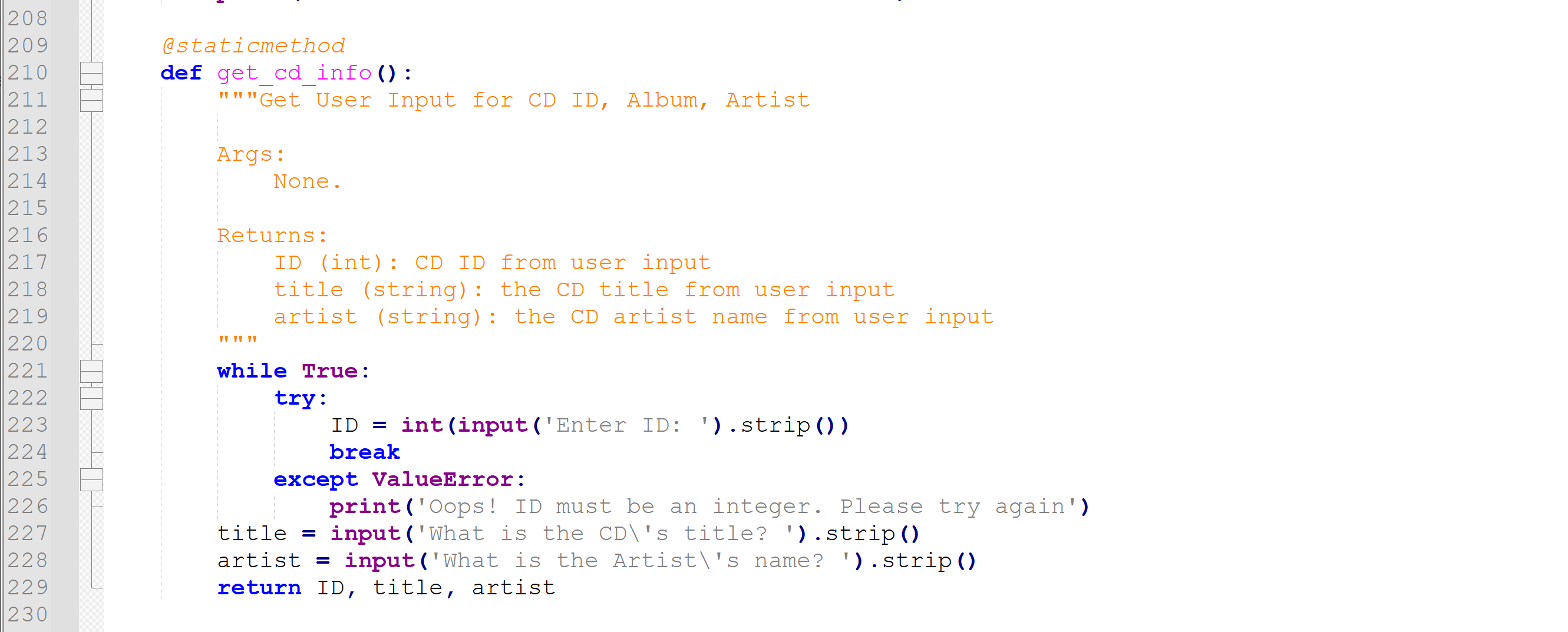


Figure 7: get\_cd\_info method that asks user for CD, title and artist. Error checking implemented to handle case of str entered as non-int.

## Summary

In assignment 08, object oriented programming was reviewed. To demonstrate knowledge of the basics from this assignment, the CDInventory Python pseudo-code was modified to structure script into a list of objects with structured error handling to manage errors. The script functionality allows users to load data from a file, add data, view data and save the data back to a file.

## Appendix

Assignment08 running in Spyder (Python 3.8):

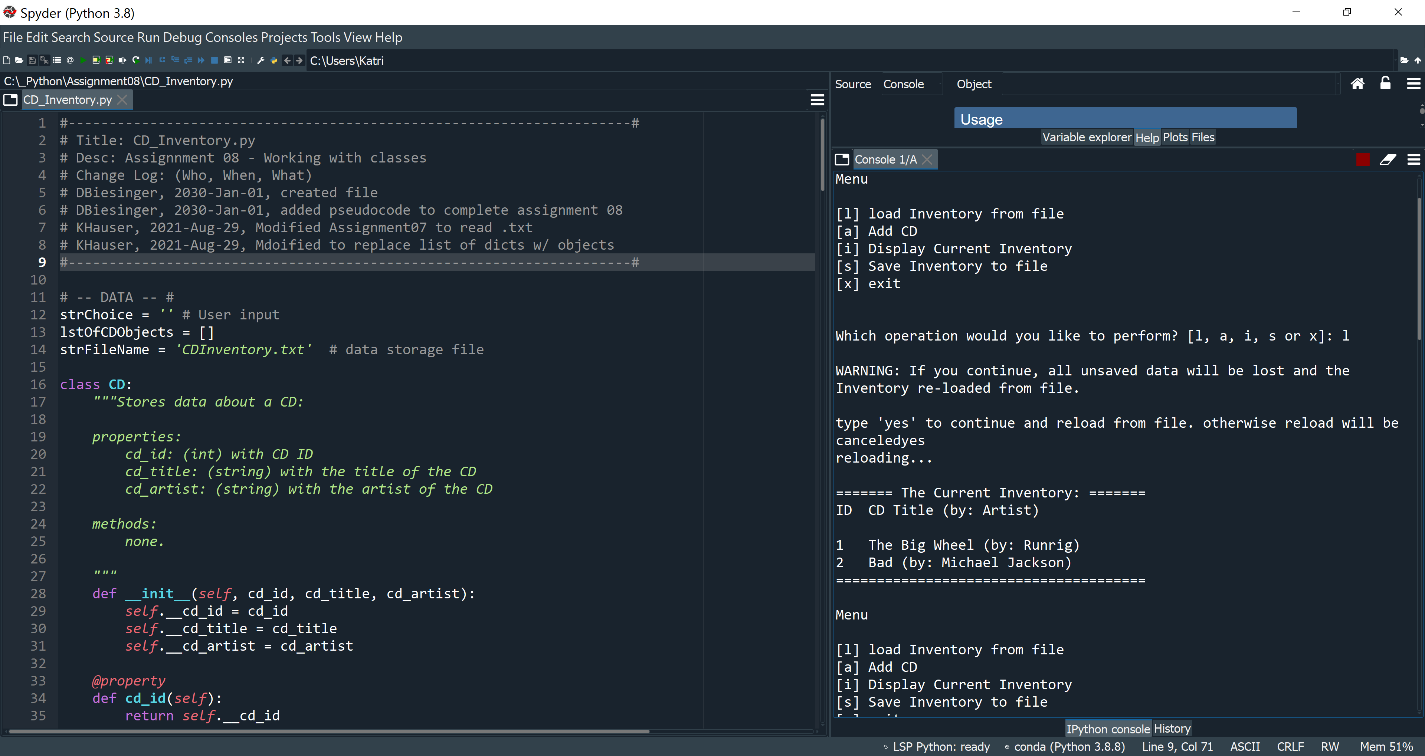


Figure 8: Script CD\_Inventory.py running in Spyder processing load (‘l’) command from user. Program loads from CDInventory.txt file.

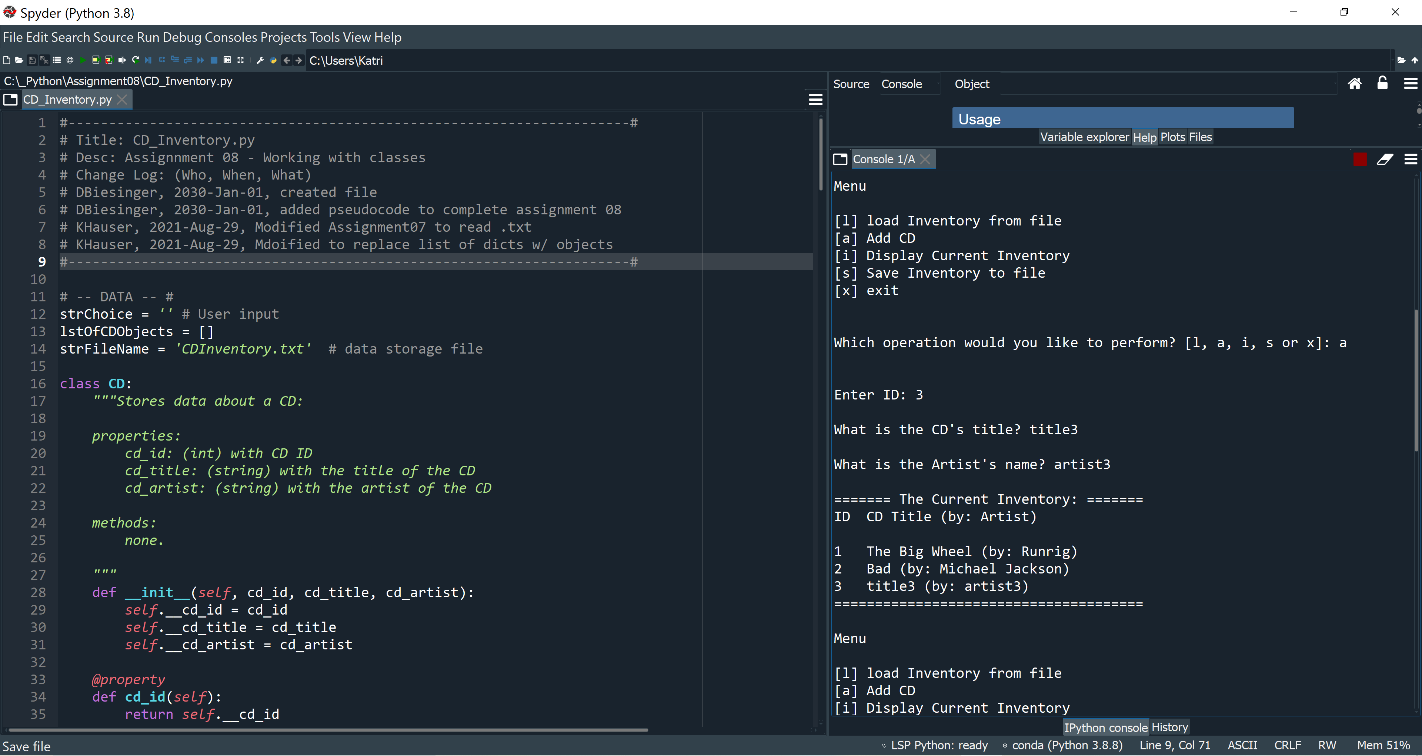


Figure 9: Script CD\_Inventory.py running in Spyder processing display (‘a’) command from user. Program adds new object with ID, Title and Artist entered from user..

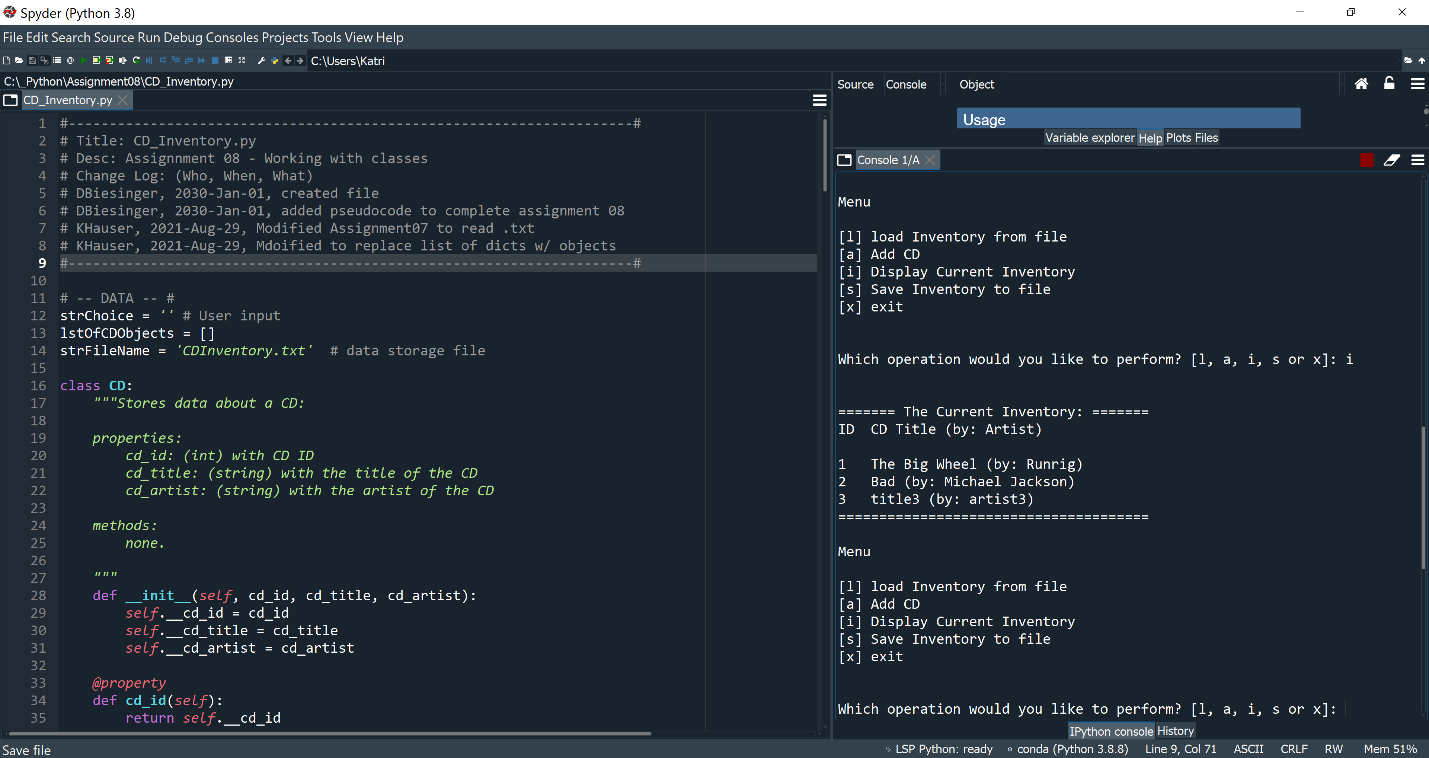


Figure 10: Script CD\_Inventory.py running in Spyder processing display inventory (‘i’) command. Script displays CDs saved in the list of objects.

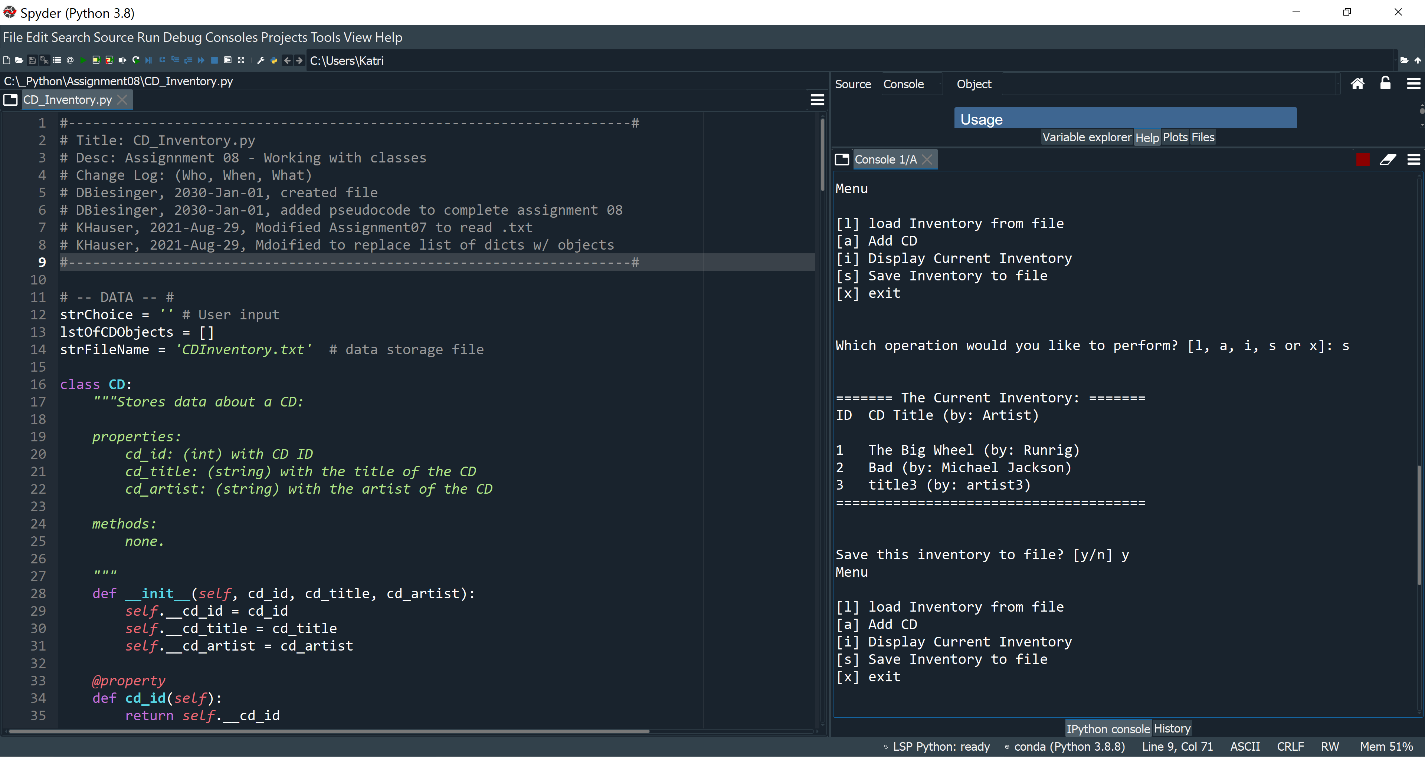


Figure 11: Script CD\_Inventory.py running in Spyder processing save (‘s’) command from user. Program saves data from inner data structure list of objects to CDInventory.txt file.

<https://github.com/hauserk/Assignment_08>

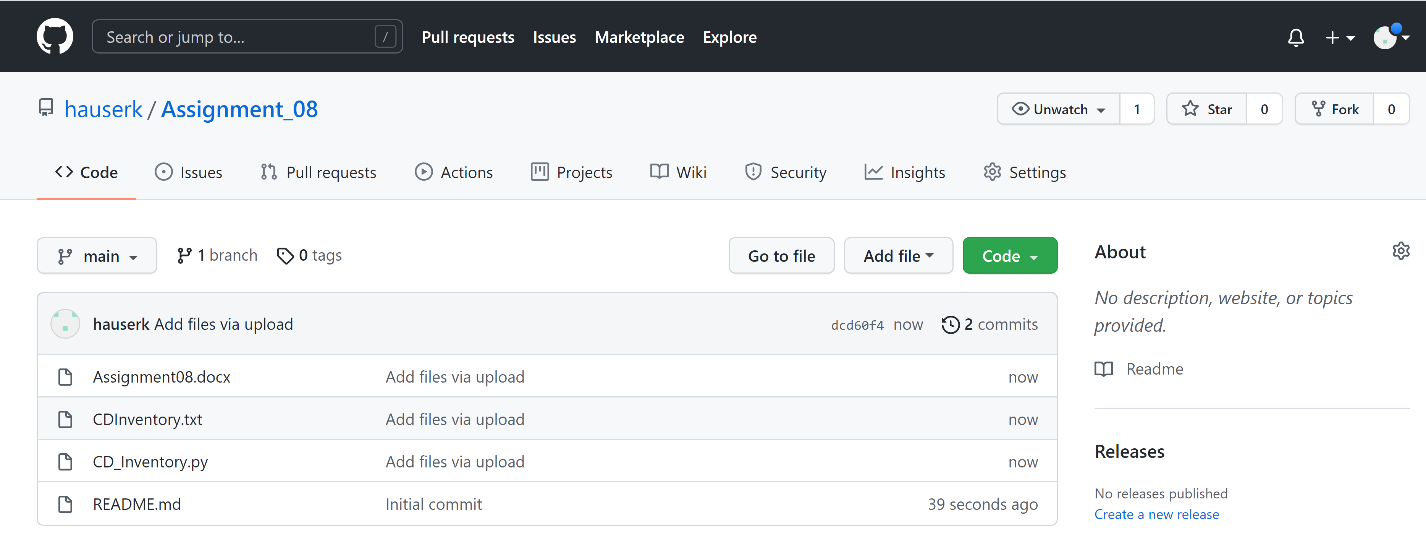


Figure 12: GitHub repository of Assignment08