New York Times Bestseller List

December 18, 2020

University of St. Gallen

Skills: Programming with Advanced Computer Languages

Authors:	**Student ID:**
Navya Thottan	16-619-736
Belda Gökkaya	16-609-141
Diar Rachmann	19-625-359

1 Task and programming procedure

The New York Times newspaper has published “best seller” lists since 1942. Book sales are tracked nationwide, leading to a list of those books which have recently sold the most copies. The file bestsellers2.txt contains books which have reached #1 on either of two lists (fiction and nonfiction) since 1942. This file is the data set for the code we have created. We have chosen this project from the python projects that were offered to us. Our code only works in python. This code enables the user to search for title and author in the data set. Furthermore, the user can search for books in a specific month and year and let the code display all books in a year range. At the beginning a menu is displayed to the user where he can choose between these four functions. When the user is finished with searching through the data set, he also has the option to quit, which will end the running of the code. Our code is corrected for different potential input mistakes and returns appropriate error messages in those cases so the user can revise the input again. Due to the original formatting of the bestsellers2.txt file the titles, names of authors and publishers are displayed without spaces between two words or names.

2 Code

```
# At the beginning you have to make sure that you have downloaded the
bestsellers2.txt and add it to the same folder as this code, or the code
won`t function without the data set.
# First, the txt file has to be turned into a list, which is what we did in
this section
fp = open('bestsellers2.txt')
all_books = []
for line in fp:
# The file is split by using tabstops
    book_list = line.split('\t')
# Here the output is defined so that in every line there is only the
information for one book
    all_books.append(book_list)
# For book in all_books you only have to enter “print(book)” to get the
output from the dataset

# Secondly, we converted the date string into its date form so that our
search functions using dates could work
from datetime import datetime
for book in all_books:
    book_date = book[3]
    book_date = datetime.strptime(book_date, '%m/%d/%Y').date()
    book[3] = book_date

# Below is the search by title function. This code searches titles using
the index position of the titles in the txt file
def search_by_title():
    title = input("Enter a title: ")
    title = title.replace(" ", "")
    any_output = False
    for book in all_books:
# Here we made sure that the user can search the title by entering small
and big letters
        if title.lower() in book[0].lower():
            book[3] = book[3]
            any_output = True
            a = str(book)
```

```
# With the following step we made the output prettier by removing the
brackets and the apostrophes
            b = a.replace("'", "").replace("[", "").replace("]", "")
            print(b)
# If the user enters input that is not part of the data set or a set of
numbers, an appropriate message will be displayed
    if any_output is False:
        print("nothing found")

# Below is the search by author function
# By using the index position in the data set, the code executes the search
def search_by_author():
    author = input("Enter an author's name: ")
    author = author.replace(" ", "")
    any_output = False
    for book in all_books:
# Here we made sure that the user can search the author by entering small
as well as big letters
        if author.lower() in book[1].lower():
            book[3] = book[3]
            any_output = True
            a = str(book)
# We made the output prettier by removing the brackets and the apostrophes
            b = a.replace("'", "").replace("[", "").replace("]", "")
            print(b)
# If the user enters input that is not part of the data set or a set of
numbers, an appropriate message will be displayed
    if any_output is False:
        print('nothing found')

# Below is the function to display books in a year range
# The dates in the txt file are used to search for the year
def search_by_year_range():
    while True:
        try:
            y_1 = int(input("Enter beginning year: "))
            y_2 = int(input("Enter ending year: "))
# If the user enters invalid input like letters or wrong numbers, an
appropriate error message is displayed
        except ValueError:
            print("Oops!  That was not a valid input.  Try again...")
# If the user input is fine, the code will search for it
        else:
            break
    time_range = list(range(y_1, y_2 + 1))
    any_output = False
    print(f"All Titles between {y_1} and {y_2} are: ")
    for book in all_books:
        book_year = book[3].year
        if book_year in time_range:
            book[3] = book[3]
            any_output = True
            a = str(book)
# Here we made the output prettier by removing the brackets and the
apostrophes
            b = a.replace("'", "").replace("[", "").replace("]", "")
            print(b)
# If the user enters input that is not part of the data set, an appropriate
message will be displayed
    if any_output is False:
        print('nothing found')
```

```
# Below code has the function to search for books in a specific month of a
specific year
# The dates in the data set are used to search for month and year
def search_by_month_year():
    while True:
        try:
            month = int(input("enter a month in number: "))
            year = int(input("enter a year: "))
# If the user enters invalid input like letters or wrong numbers, an
appropriate error message is displayed
        except ValueError:
            print("Oops!  That was not a valid input.  Try again...")
# If the user input is fine, the code will search for it
        else:
            break
    any_output= False
    print(f"All Titles in month {month} of {year} :")
    for book in all_books:
        book_year = book[3].year
        book_month = book[3].month
        if month == book_month and year == book_year:
            book[3] = book[3]
            any_output= True
            a = str(book)
# Here we made the output prettier by removing the brackets and the
apostrophes
            b = a.replace("'", "").replace("[", "").replace("]", "")
            print(b)
# If the user enters input that is not part of the data set, an appropriate
message will be displayed
    if any_output is False:
            print("nothing found")


# Below is the menu that is shown at the beginning when the code starts
running and where the user can choose which search function he wants to use
import sys
while True:
 question = str(input("What would you like to do?:\n1: Search for title\n2:
Search for author\n"
                      "3: Look up year range\n4: Look up
month/year\nQ:Quit\n>"))
# If the user chooses the correct input numbers between 1-4, he can use the
function that is allocated to these numbers, which is what the following
code does
 if question == '1':
    search_by_title()
 elif question == '2':
    search_by_author()
 elif question == '3':
    search_by_year_range()
 elif question == '4':
    search_by_month_year()# Since our program will continue to execute
until the user decides to quit, the user has the possibility to quit the
program by entering q or Q
 elif question == "Q" or question == "q":
    print('Goodbye')
    sys.exit()

# If the user enters input that is neither defined by the numbers above nor
by the quit function, this appropriate error message is displayed
 else:
    print('Please select from the options in the menu')
```

3 Example: Input and Output

The user input is green. In this example, the output is shown when the user input is correct.

```
What would you like to do?:

1: Search for title

2: Search for author

3: Look up year range

4: Look up month/year

Q: Quit

>1

Enter a title: secret

HarryPotterandtheChamberofSecrets, J.K.Rowling, Levine/Scholastic,
datetime.date(1999, 6, 20), Fiction\n

TheSecretofSantaVittoria, RobertCrichton, Simon&Schuster,
datetime.date(1966, 11, 20), Fiction\n

TheSecretPilgrim, JohnleCarre, Knopf, datetime.date(1991, 1, 20), Fiction\n

What would you like to do?:

1: Search for title

2: Search for author

3: Look up year range

4: Look up month/year

Q: Quit

>2

Enter an author's name: tolkein

Silmarillion, J.R.R.Tolkein, HoughtonMifflin, datetime.date(1977, 10, 2),
Fiction\n

TheChildrenoftheHurin, J.R.R.Tolkein, HoughtonMifflin, datetime.date(2007,
5, 6), Fiction\n

What would you like to do?:

1: Search for title

2: Search for author

3: Look up year range

4: Look up month/year

Q: Quit

>3

Enter beginning year: 1960

Enter ending year: 1962

All Titles between 1960 and 1962 are:
```

AShadeofDifference, AllenDrury, Doubleday, datetime.date(1962, 10, 28), Fiction\n

FrannyandZooey, J.D.Sallinger, Little,Brown, datetime.date(1961, 10, 29), Fiction\n

Hawaii, JamesMichener, RandomHouse, datetime.date(1960, 1, 17), Fiction\n

SevenDaysinMay, FletcherKnebel, Harper, datetime.date(1962, 11, 18), Fiction\n

ShipofFools, KatherineAnnePorter, Little,Brown, datetime.date(1962, 4, 29), Fiction\n

TheAgonyandtheEcstasy, IrvingStone, Doubleday, datetime.date(1961, 4, 23), Fiction\n

TheLastoftheJust, AndreSchwarz-Bart, Atheneum, datetime.date(1961, 3, 26), Fiction\n

BornFree, JoyAdamson, Pantheon, datetime.date(1960, 8, 7), Nonfiction\n

"CaloriesDontCount", HermanTaller, Simon&Schuster, datetime.date(1962, 3, 25), Nonfiction\n

MayThisHouseBeSafefromTigers, AlexanderKing, Simon&Schuster, datetime.date(1960, 3, 13), Nonfiction\n

SilentSpring, RachelCarson, HoughtonMifflin, datetime.date(1962, 10, 28), Nonfiction\n

TheMakingofthePresident-1960, TheodoreH.White, Atheneum, datetime.date(1961, 9, 10), Nonfiction\n

TheNewEnglishBible, OxfordUniversityPress(Editor), OxfordUniversityPress, datetime.date(1961, 5, 28), Nonfiction\n

TheRiseandFalloftheThirdReich, WilliamShirer, Simon&Schuster, datetime.date(1960, 12, 4), Nonfiction\n

TheRothchilds, FredericMorton, Atheneum, datetime.date(1962, 6, 24), Nonfiction\n

TheWasteMakers, VancePackard, McKay, datetime.date(1960, 11, 6), Nonfiction\n

TravelswithCharley, JohnSteinbeck, Viking, datetime.date(1962, 10, 21), Nonfiction\n

What would you like to do?:

1: Search for title

2: Search for author

3: Look up year range

4: Look up month/year

Q: Quit

>4

enter a month in number: 9

enter a year: 1990

All Titles in month 9 of 1990 :

```
FourPastMidnight, StephenKing, Viking, datetime.date(1990, 9, 16),
Fiction\n

MemoriesofMidnight, SidneySheldon, Morrow, datetime.date(1990, 9, 2),
Fiction\n

DarknessVisible, WilliamStyron, RandomHouse, datetime.date(1990, 9, 16),
Nonfiction\n

"MilliesBook", BarbaraBush, Morrow, datetime.date(1990, 9, 30),
Nonfiction\n

Trump:SurvivingattheTop, DonaldTrump, RandomHouse, datetime.date(1990, 9,
9), Nonfiction\n

What would you like to do?:

1: Search for title

2: Search for author

3: Look up year range

4: Look up month/year

Q: Quit

>q

Goodbye


Process finished with exit code 0
```

4 Problems in the Code

In the example above you can see that for each date of a book, there is written `datetime.date` in front of the date. This information is displayed because we converted the dates into their date form to enable the search by dates. We tried to convert the date back into string using `strftime()` but in this case the code would stop running from time to time and display an error message (see Figure 1). We were not able to fix this, so we decided to leave the date as it is since the code works perfectly otherwise.

```
>2
Enter an author's name: diana
Traceback (most recent call last):
  File "C:\Users\George Thottan\PycharmProjects\pythonProject1\main.py", line 101, in <module>
    search_by_author()
  File "C:\Users\George Thottan\PycharmProjects\pythonProject1\main.py", line 38, in search_by_author
    book[3] = book[3].strftime('%m/%d/%Y')
AttributeError: 'str' object has no attribute 'strftime'

Process finished with exit code 1
```

Figure 1: The error message when using `strftime()`

This is how the code for searching the author looked like with the `strftime()` function.

```
def search_by_author():
    author = input("Enter an author's name: ")
    author = author.replace(" ", "")
    any_output = False
    for book in all_books:
        if author.lower() in book[1].lower():
            book[3] = book[3].strftime('%m/%d/%Y')
            any_output = True
            a = str(book)
            b = a.replace("'", "").replace("[", "").replace("]", "")
            print(b)
    if any_output is False:
        print('nothing found')
```

5 Sources

For splitting the txt file we used the source code from this YouTube video:
http://youtu.be/O4hNpq3Aiig