# Main function to initiate the program

function main():

print("Welcome to the CS110 Book Recommender. Type the word in the")

print("left column to do the action on the right.")

print("recommend : recommend books for a particular user")

print("averages : output the average ratings of all books in the system")

print("quit : exit the program")

task()

# Function to read data from the input file

function read\_file():

open file "ratings-small.txt"

read lines from the file

extract unique books from every 3rd line

extract unique users from every 1st line

create empty dictionary ratings

# Iterate through each line in the file

for each line in lines:

extract user, book, and rating

if user is not in ratings:

initialize ratings[user] as a list of zeros with the length of books

find the index of the book in the books list

update the rating at the corresponding index in ratings[user]

return books, users, ratings

# Function to recommend books for a particular user

function recommend(user, books, users, ratings):

if user is not in ratings:

# If the user is not in the ratings, call the average function

call average function with books and ratings

return

initialize empty list similarities

# Iterate through each user in the system

for each other\_user in users:

if other\_user is not equal to user:

initialize similarity as 0

# Iterate through each book

for i in range(length of books):

update similarity by adding the product of ratings[user][i] and ratings[other\_user][i]

append (similarity, other\_user) to similarities

sort similarities in descending order

initialize recommended\_books as a list of zeros with the length of books

# Iterate through the minimum of 3 and the length of similarities

for i in range(minimum of 3 and length of similarities):

extract other\_user from similarities[i]

# Iterate through each book

for j in range(length of books):

update recommended\_books[j] by adding ratings[other\_user][j]

# Iterate through each book

for i in range(length of books):

if recommended\_books[i] is greater than 0:

print book[i] and recommended\_books[i] divided by the minimum of 3 and length of similarities

# Function to calculate and print the average ratings for all books

function average(books, ratings):

initialize averages as an empty list of tuples

# Iterate through each book

for i in range(length of books):

calculate average rating for book[i] by summing ratings[user][i] for all users

divide by the count of users with non-zero ratings

append (average, book[i]) to averages

sort averages in descending order

# Iterate through each average and book

for each avg, book in averages:

print book and avg

print a newline

# Function to handle user tasks

function task():

# Call read\_file to get books, users, and ratings

call read\_file to get books, users, and ratings

while true:

# Get the next task from user input

get next\_task from user input

if next\_task is "quit":

# If the task is to quit, break out of the loop

break

else if next\_task is "recommend":

# If the task is to recommend, get the user and call the recommend function

get user from user input

call recommend function with user, books, users, and ratings

print a newline

else if next\_task is "averages":

# If the task is to show averages, call the average function

call average function with books and ratings

# Check if the script is executed directly

if \_\_name\_\_ is "\_\_main\_\_":

# Call the main function

call main function