School of Computer Science and informatics
Coursework Assessment Proforma

Module & Lecturer:	CM1208: Maths for Computer Science, Y. Lai
Date Set:	23rd February 2017
Submission Date:	31st March 2017

Submission Arrangements:
Your coursework program – a Python application program in the file named `Recommend.py` – should be submitted by midnight (23:59) of Friday 31st March, 2017 (Week 10, Semester 2) via Learning Central.

Make sure to include (as comments) your student number (but *not* your name) at the top of your program file (`Recommend.py`). Also, follow this by any notes (as comments) regarding your submission. For instance, specify here if your program does not generate the proper output or does not do it in the correct manner.

Title: Implementation of e-commerce recommendation using item-to-item collaborative filtering

This coursework is worth 15% of the total marks available for this module. The penalty for late or non-submission is an award of zero marks. You are reminded of the need to comply with Cardiff University's Student Guide to Academic Integrity. Your work should be submitted using the official Coursework Submission Cover sheet.

INSTRUCTIONS

Write a Python application program (main file called `Recommend.py`) that implements e-commerce recommendation based on item-to-item collaborative filtering, as discussed in the lectures.

Input and Output. The input to your program involves two text files **history.txt** which contains the purchase history of all the customers and **queries.txt** which includes all the queries (shopping cart items based on which you need to provide recommendation).

The file **history.txt** is a text file describing the complete purchase history of all the customers. The first line includes three numbers:

Number_of_Customers Number_of_Items Number_of_Transactions

This is followed by *Number_of_Transactions* lines of text, each line containing two numbers:

Customer_ID Item_ID

This means the customer *Customer_ID* bought the item *Item_ID*. Both *Customer_ID* and *Item_ID* are integers starting from 1. Note that the same customer may have bought the same item multiple times.

The file **queries.txt** is a text file with each line containing a query, describing a list of items in the current shopping cart. Each query is composed of one or more numbers, separated by whitespace, corresponding to the item IDs in the current shopping cart.

For output, print the required messages (see details below and sample outputs overleaf) following exactly the same format (except for the actual number of whitespace) as specified and as in the examples. When printing real numbers, you should print them with at least two fractional digits.

Reading the transaction history. Your program should read in all the transactions from `history.txt`, and build the customer-item purchase history table (as explained in the lecture) where an entry of 1 means the customer has bought the item and 0 otherwise. Print the total number of non-zero entries (i.e. with a value of 1) in the customer-item purchase history table ("Positive entries: *number*").

Precomputing item-to-item angles. Your program should work out the angles (in degrees) between every pair of items (excluding an item with itself). Print the average of all the pairwise angles ("Average angle: *average_angle*").

Recommendation. For each query (each line in **queries.txt**), your program should perform the following:

- Print the query in a line "Shopping cart: *query*".
- For each item *Item_ID* in the query (in the order as it appears), find an item *Match_ID not in the current shopping cart* which has the minimum angle *Min_Angle* with the item *Item_ID*.
  - If *Min_Angle* is less than 90°, print "Item: *Item_ID*; match: *Match_ID*; angle: *Min_Angle*".
  - Otherwise, no match is accepted, and you should simply print "Item: *Item_ID* no match".

  Note that when multiple items have the same minimum angle, your algorithm can print any one of these. Each printed item *Match_ID* is considered as a candidate for recommendation.
- Produce the recommendation list by combining all the candidates and order them in increasing order of angles. For items which are considered relevant via different shopping cart items, the minimum angle it makes with any item in the shopping cart should be used in ranking, and it should only appear in the recommendation list once. If multiple candidates have the same angle, they may appear in arbitrary order. Print "Recommend: *list_of_recommended_items*".

Error checking is *not* required (invalid input, etc.) *Remember that I will perform extensive tests on your program, using more than just the test data I have provided.*

Submission Instructions

Description		Type	Name
Cover Sheet	**Compulsory**	One PDF (.pdf) file	[student number].pdf
Solution	**Compulsory**	One Python source file (.py)	Recommend.py
	Optional	Additional Python source files (.py)	No restriction

Note: both the cover sheet and the source code should be submitted to Learning Central as detailed in Submission Arrangements.

Criteria for Assessment

Assessment and marking of your program will be done by automatically checking the output of your program against my program on a large set of test conditions. This will be done using a text comparison tool, *so make sure your outputs match mine as formatting of text will be critical!*
The breakdown of marks will be:

- 10% – reading in purchase history
- 25% – precomputing item-to-item angles
- 5% – reading and printing queries
- 30% – working out candidate for each item in the shopping cart
- 15% – printing recommendation in the correct order
- 15% – efficiency (i.e. excessive run-times will be penalised)

Further Details

Feedback on your coursework will address the above criteria and will be returned in approximately three working weeks. This will be supplemented with oral feedback in lectures. If you have any questions relating to your individual solutions talk to the lecturer.

Sample Outputs

Given the following `history.txt`:

```
5 5 12
1 1
1 2
2 1
2 3
3 1
3 2
3 4
4 2
4 4
1 2
2 3
5 5
```

and the following `queries.txt`:

```
2
1 2
5
5 1
5 1 4
1 3 5
```

the exact output of your program should be:

```
Positive entries: 10
Average angle: 74.41
Shopping cart: 2
Item: 2; match: 4; angle: 35.26
Recommend: 4
Shopping cart: 1 2
Item: 1; match: 3; angle: 54.74
Item: 2; match: 4; angle: 35.26
Recommend: 4 3
Shopping cart: 5
Item: 5 no match
Recommend:
Shopping cart: 5 1
Item: 5 no match
Item: 1; match: 2; angle: 48.19
Recommend: 2
Shopping cart: 5 1 4
Item: 5 no match
Item: 1; match: 2; angle: 48.19
Item: 4; match: 2; angle: 35.26
Recommend: 2
Shopping cart: 1 3 5
Item: 1; match: 2; angle: 48.19
Item: 3 no match
Item: 5 no match
Recommend: 2
```