**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN LẬP TRÌNH TÍNH TOÁN**

**TÊN ĐỀ TÀI:**

**DỰ ĐOÁN ĐIỂM ĐÁNH GIÁ NGƯỜI TIÊU DÙNG**

**Người hướng dẫn : TS.NGUYỄN VĂN HIỆU**

**Sinh viên thực hiện:**

**Tên sinh viên 1 : LÊ QUỐC RÔN LỚP 20TCLC\_DT4**

**Tên sinh viên 2 : TRẦN DUY QUANG LỚP 20TCLC\_DT4**

**Đà Nẵng, 04/2020**

MỤC LỤC

[MỤC LỤC](#_Toc72525266)

[DANH MỤC HÌNH VẼ](#_Toc72525267)

[MỞ ĐẦU](#_Toc72525268)…………………………………………………………………………

[1. TỔNG QUAN ĐỀ TÀI](#_Toc72525269) 1

[2. CƠ SỞ LÝ THUYẾT](#_Toc72525278) 3

[2.1. Ý tưởng](#_Toc72525279) 3

[2.2. Cơ sở lý thuyết](#_Toc72525280) 4

[3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN](#_Toc72525281) 6

[3.1. Phát biểu bài toán](#_Toc72525282) 6

[3.2. Cấu trúc dữ liệu](#_Toc72525285) 6

[3.3. Thuật toán](#_Toc72525286) 8

[4. CHƯƠNG TRÌNH VÀ KẾT QUẢ](#_Toc72525287) 16

[4.1. Tổ chức chương trình.](#_Toc72525288) 16

[4.2. Ngôn ngữ cài đặt](#_Toc72525289) 16

[4.3. Kết quả](#_Toc72525290) 16

[4.3.1. Giao diện chính của chương trình](#_Toc72525291) 16

[4.3.2. Kết quả thực thi của chương trình](#_Toc72525292) 19

[4.3.3. Nhận xét đánh giá](#_Toc72525298) 23

[5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN](#_Toc72525299) 23

[5.1. Kết luận](#_Toc72525300) 23

[5.2. Hướng phát triển](#_Toc72525301) 24

[TÀI LIỆU THAM KHẢO](#_Toc72525302) 24

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

*Ngày …. tháng …. năm …*

Giảng viên hướng dẫn

TS.Nguyễn Văn Hiệu

DANH MỤC HÌNH VẼ

Hình 1. Ma trận User-Items

Hình 2. Collaborative recommendation systems

Hình 3. Content-based recommendation systems

Hình 4. Knowledge-based recommendation systems

Hình 5. Mô hình phương pháp Collaborative filtering

Hình 6. Ma trận đầu vào

Hình 7. Ma trận đầu ra

Hình 8. Menu chính của chương trình

Hình 9. Input menu

Hình 10. Output menu

Hình 11. Báo lỗi khi người dùng chưa nhập vào input

MỞ ĐẦU

Ngày nay, ngành công nghệ thông tin đang phát triển vô cùng mạnh mẽ. Những thành tựu của ngành CNTT được ứng dụng ở mọi lĩnh vực và dần trở thành một phần quan trọng trong cuộc sống hiện đại. Đặc biệt là trong ngành thương mại điện tử, với việc ứng dụng các hệ thống gợi ý (reccomendation system) để gợi ý sản phẩm cho người dùng đã mang lại một cách tiếp cận gần gũi hơn của các sản phẩm tới người dùng, giúp cải thiện trải nghiệm, tiết kiệm được thời gian cho người dùng và nâng cao doanh thu rõ rệt cho doanh nghiệp.

Khi người dùng sử dụng một trang web hay một ứng dụng thì việc để giữ chân người dùng ở lại và ủng hộ lâu dài chính là trải nghiệm của nó mang lại. Hiện nay, một số trang web hay các ứng dụng có luồng thông tin, tài nguyên lớn, tuy nhiên không kiểm soát được nó, mang lại trải nghiệm rất kém cho người dùng. Các tin rác hay các tin tức mà người dùng không quan tâm thường xuyên hiển thị, người dùng phải tự đi tìm các thông tin phù hợp với bản thân trên trang web hay ứng dụng gây mất thời gian. Giả sử rằng, nếu một người đi mua sắm trực tuyến mất một giây cho mỗi lần vuốt trên ứng dụng di động, để duyệt qua 2 tỷ sản phẩm có sẵn trên một trang web thương mại điện tử thì cô ta sẽ mất 65 năm, gần như toàn bộ cuộc đời để đi qua toàn bộ các danh mục(sản phẩm) của ứng dụng. Đây là một trong những lí do chúng ta cần phải khắc phục ngay lập tức nếu không hàng tỷ người dùng Internet trên thế giới không thể tiếp cận được với các sản phẩm, dịch vụ,...

Để giải quyết vấn đề trên, chúng ta cần phải xác định được độ quan tâm của người dùng đối với các sản phẩm trên trang các web hay ứng dụng. Từ đó, ta có được dữ liệu để sàn lọc ra các sản phẩm mà người dùng quan tâm đến nó nhất, sau đó hiển thị đến với người dùng. Tóm lại, mấu chốt để giải quyết vấn đề trên là ta cần phải dự đoán được độ quan tâm hay cụ thể là điểm rating của người dùng đối với các sản phẩm trên trang web.

Xuất phát từ nhận thức trên, với mong muốn áp dụng công nghệ thông tin vào việc dự đoán điểm rating người dùng đối với các sản phẩm, dựa trên những hoạt động thực tiễn nhóm chúng em xin tìm hiểu và thực hiện đề tài “*Xây dựng chương trình Dự đoán điểm đánh giá người dùng*”, với các chức năng tính toán, dự đoán rating người dùng.

Em xin chân thành cảm ơn sự hướng dẫn tận tình của thầy Nguyễn Văn Hiệu đã trao đổi, góp ý giúp em hoàn thành đề tài này. Với sự hạn chế về mặt kiến thức nên báo cáo của em vẫn còn nhiều thiếu sót và hạn chế, vậy em rất mong nhận được sự góp ý thêm từ quý thầy cô và các bạn.

Đà Nẵng, ngày 22 tháng 05 năm 2021

# TỔNG QUAN ĐỀ TÀI

## 1.1 Mục đích của đề tài.

Xây dựng một chương trình có thể dự đoán được điểm rating cùa người dùng đối với sản phẩm chính xác nhất có thể, từ đó có cơ sở để áp dụng và phát triển hệ thống gợi ý (Recommendation system) đẩy mạnh các hình thức tiếp cận của sản phẩm đến với người dùng giúp nâng cao trải nghiệm của người dùng khi sử dụng trang web hay ứng dụng.

## 1.2 Ý nghĩa của đề tài.

## Giúp tiết kiệm chi phí và thời gian cho người dùng tối ưu nhất.

Là cơ sở để xây dựng một hệ thống gợi ý hoàn chỉnh, giúp nâng cao trải nghiệm của người dùng và đẩy mạnh doanh thu cho các doanh nghiệp.

Bên cạnh đó, thông qua đề tài này giúp chúng em phát triển các kỹ năng: phân tích, thiết kế, lựa chọn cấu trúc dữ liệu và xây dựng thuật toán, lập trình ,...

## 1.3 Phương pháp thực hiện.

## 1.3.1 Giới thiệu về hệ thống gợi ý (Recommender system).

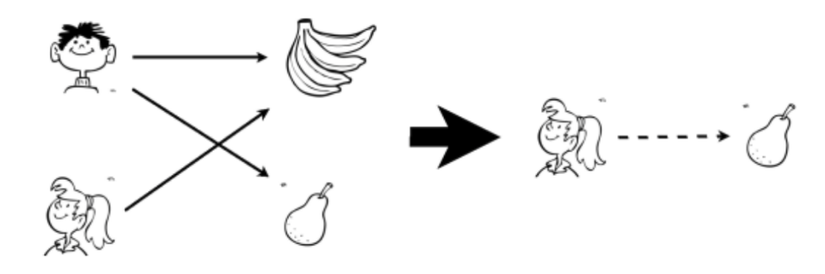
Hệ thống gợi ý Recommender System là một hệ thống lọc thông tin để dự đoán về sở thích của người dùng đối với mặt hàng mặt họ quan tâm, sau đó đưa ra các đề xuất hợp lí có hiệu quả. Hình thức này được sử dụng để cung cấp các kiến nghị, gợi ý trên nhiều lĩnh vực như: YouTube giới thiệu video, Shopee & Tiki giới thiệu sản phẩm liên quan cần mua, Netflix đề xuất phim theo sở thích và Facebook để giới thiệu bạn bè quen biết cũng như các sản phẩm quảng cáo khác,...

Những yếu tố góp phần hình thành nên hệ thống Recommender System bao gồm: người dùng (Users), mặt hàng (Items), phản hồi (Feedback). Sau khi đã thu thập được các thông tin số liệu, việc biểu diễn nó trên ma trận User-Items là phương pháp hiểu quả để xem xét sở thích của người dùng lên từng items cụ thể.

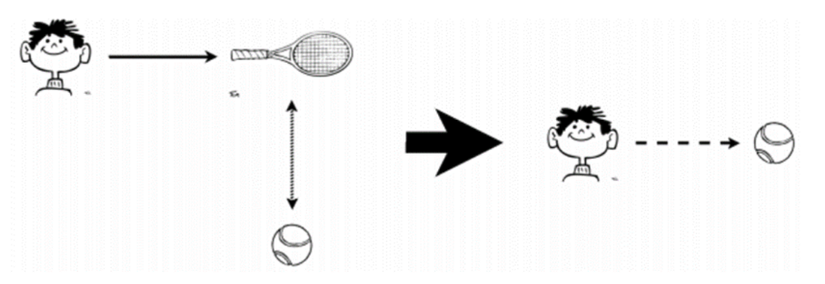
## 

## Hình 1. Ma trận User-Items

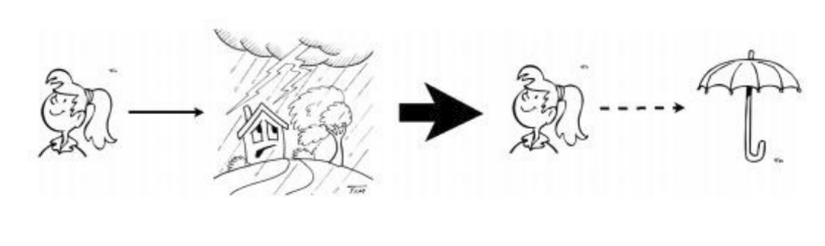
Các phương pháp gợi ý:

* Hệ thống gợi ý dựa theo lọc cộng tác (Collaborative recommendation systems): là phương pháp gợi ý được triển khai rộng rãi nhất và thành công nhất trong thực tế.

**Hình 2.** **Collaborative recommendation systems**

* Hệ thống gợi ý dựa theo nội dung (Content-based recommendation systems): là sự kế thừa và mở rộng của lĩnh vực nghiên cứu lọc thông tin.

**Hình 3.** **Content-based recommendation systems**

* Hệ thống gợi ý dựa trên cơ sở tri thức (Knowledge-based recommendation systems): gợi ý các đối tượng dựa trên các suy luận về nhu cầu và sở thích của người dùng.

**Hình 4. Knowledge-based recommendation systems**

## 1.3.2 Phương pháp triển khai.

Có thể thấy là việc dự đoán điểm rating người dùng có thể được triển khai với nhiều phương pháp khác nhau như đã nêu ở trên. Qua nghiên cứu và tìm hiểu, nhóm chúng em quyết định triển khai phương pháp lọc cộng tác(Collaborative Filtering) vì tính đơn giản và hiệu quả của nó mang lại.

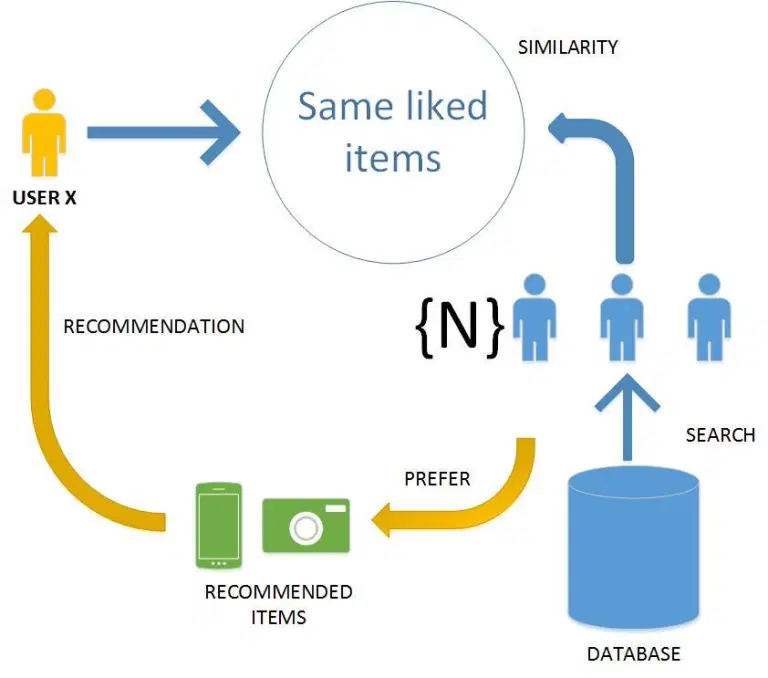
# CƠ SỞ LÝ THUYẾT

## Ý tưởng

Ý tưởng cơ bản của Collaborative Filtering là xác định *mức độ quan tâm* của một *user* tới một *item* dựa trên các *users* khác *gần giống* với *user* này. Việc *gần giống nhau* giữa các *users* có thể được xác định thông qua *mức độ quan tâm* của các *users* này tới các *items* khác mà hệ thống đã biết.

Công việc mà chúng ta cần làm để dự đoán được rating của một người dùng *u* lên sản phẩm *i* gồm hai bước sau:

1. Đầu tiên, chúng ta phải tìm được nhóm người có chung sở thích với người dùng *u* này thông qua bảng ratings mà chúng ta đã có.
2. Sau đó, dựa trên nhóm người mà chúng ta có chung sở thích với u mà chúng ta tìm được và các đánh giá của nhóm người này lên sản phẩm *i*, chúng ta sẽ dự đoán đánh giá của người dùng *u* lên sản phẩm i.

**Hình 5. Mô hình phương pháp Collaborative filtering**

## Cơ sở lý thuyết

Đầu tiên, để tính được sự tương đồng giữa các người dùng, chúng ta sẽ sử dụng điểm đánh giá của người dùng đó lên tất cả các sản phẩm làm vector đại diện cho người dùng, rồi so sánh các vector với nhau để tìm sự giống nhau. Có nhiều hàm toán học để tính toán độ giống nhau giữa các vector như Pearson,Cosine,...Qua thảo luận, nhóm chúng em quyết định sẽ triển khai hai hàm phổ biến nhất là *Pearson* và *Cosine*.

**Pearson Similarity function**

**Cosine Similarity function**

là tập hợp các sản phẩm đã được cả hai user u và v cùng đánh giá.

lần lượt là điểm của user *u* và *v* đánh giá item *k*.

Ta có thể thấy được là các hàm Similarity trên sẽ có giá trị từ [-1; 1] hay là độ Similarity giữa các user sẽ có giá trị từ [-1; 1]. Giá trị bằng 1 thể hiện hai vector hoàn toàn giống nhau. Ngược lại, giá trị bằng -1 thể hiện hai vector này hoàn toàn trái ngược nhau, tức không có điểm chung nào.

Sau khi có được điểm tương đồng giữa các user, ta có thể tính được điểm rating bằng công thức sau:

*Sim(u, v)* là giá trị tương đồng giữa hai user *u* và *v*.

là giá trị rating trung bình của user *u*.

là tập hợp các item mà user *u* đã đưa ra đánh giá.

là tập hợp các hàng xóm hay tập hợp các user có điểm tương đồng với user *u* cao nhất.

là điểm rating còn thiếu của user *u* đối với item *j*.

# TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN

## Phát biểu bài toán

* Ta có:

+  **numberOfUsers** là số lượng người dùng.

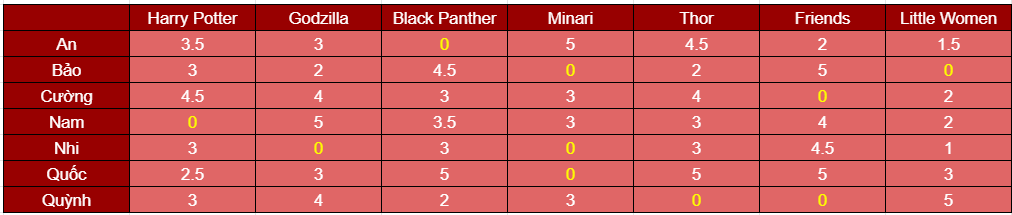
* **numberOfItems** là số lượng các sản phẩm.

## 3.1.1. Input

Là một ma trận cấp (**numberOfUsers** x **numberOfItems**), với mỗi hàng là điểm rating của user đánh giá item tương ứng và mỗi cột là điểm rating mà item đó nhận được từ user.

Điểm rating do người dùng đánh giá sẽ giao động từ [1; 5] tăng dần theo mức độ yêu thích của người dùng.

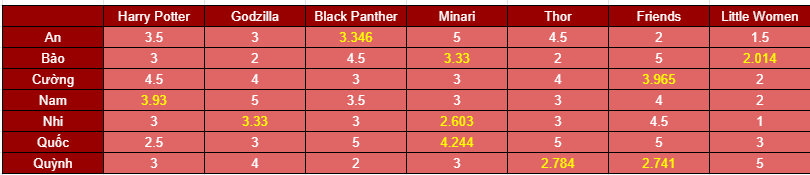
Trong ma trận đầu vào sẽ có các ô có giá trị là ‘0’, tương đương với việc người dùng chưa đưa ra đánh giá cho item đó.



**Hình 6. Ma trận đầu vào**

## 3.1.2. Output

Là một ma trận cấp (**numberOfUsers** x **numberOfItems**) mà các giá trị ‘0’ hay các giá trị chưa được người dùng rating ở ma trận input đã được dự đoán.



**Hình 7. Ma trận đầu ra**

## Cấu trúc dữ liệu

Chính vì đây là chương trình dự đoán điểm rating người dùng, trong đó rating của người dùng đối với các item được biểu diễn dưới dạng ma trận cấp (numberOfUsers x numberOfItems), với mỗi hàng là điểm rating của user đánh giá item tương ứng và mỗi cột là điểm rating mà item đó nhận được từ user nên chúng em sẽ làm việc chủ yếu với ma trận. Chính vì thế, chúng em sử dụng cấu trúc dữ liệu chủ yếu là mảng hai chiều để lưu trữ và tính toán, ngoài ra còn sử dụng mảng một chiều để lưu trữ hàng xóm và xử lí các bài toán nhỏ hơn.

Tổ chức lưu trữ dữ liệu:

* ***int numberOfUsers, numberOfItems:*** Lưu giá trị số lượng người dùng và số lượng item.
* ***float matrix[numberOfUsers + 1][numberOfItems + 1]:*** Lưu ma trận đầu vào.
* ***float avgMatrix[numberOfUsers + 1][numberOfItems + 1]:*** Lưu ma trận đầu vào khi đã trừ cho giá trị trung bình của user.

()

()

* ***float simMatrixPearson[numberOfUsers + 1][numberOfUsers + 1]:*** Lưu ma trận chứa giá trị tương đồng giữa các user được tính theo hàm *Pearson*.
* ***float simMatrixCosine[numberOfUsers + 1][numberOfUsers + 1]:*** Lưu ma trận chứa giá trị tương đồng giữa các user được tính theo hàm Cosine.
* ***int neighbor[k + 1]:*** Lưu trữ *k* hàng xóm của user *u.*
* ***float resultMatrixPearson[numberOfUsers + 1][numberOfItems + 1]:*** Lưu trữ ma trận đầu ra được tính theo hàm *Pearson.*
* ***float resultMatrixCosine[numberOfUsers + 1][numberOfItems + 1]:***Lưu trữ ma trận đầu ra được tính theo hàm *Cosine.*

## Thuật toán

* + 1. **Tính rating trung bình của user (Độ phức tạp: O(n))**

Bắt đầu

Count = 0

Sum = 0

i = 1

i <= numberOfItems

matrix[user][i] != 0

Count++

Sum += matrix[user][i]

i++

return Sum/Count

Kết thúc

Đúng

Đúng

Sai

Sai

**Lưu đồ thuật toán tính giá trị rating trung bình của user**

Step 1: Gán sum = 0, count = 0;

Step 2: Lặp các item, nếu item đã được rating thì sẽ tăng biến sum và count.

For i: 1 -> numberOfItems

If matrix[user][i] != 0 then {count++; sum+= matrix[user][i]}

Step3: return sum/count

* + 1. **Pearson(Độ phức tạp: O(n))**

Bắt đầu

tuSo = 0

sumOfUser1 = sumOfUser2 = 0

i = 1

i <= numberOfItems

matrix[user1][i] \* matrix[user2][i] != 0

tuSo += avgMatrix[user1][i]\*avgMatrix[user2][i]

sumOfUser1 += pow(avgMatrix[user1][i],2)

sumOfUser2 += pow(avgMatrix[user2][i],2)

i++

return tuSo/sqrt(sumOfUser1\*

sumOfUser2)

Kết thúc

Đúng

Đúng

Sai

Sai

**Lưu đồ thuật toán tính tương đồng giữa *user1* và *user2* theo hàm *Pearson***

Step 1: Gán tuSo = sumOfUser1 = sumOfUser2 = 0

Step 2: For i: 1 -> numberOfItems

If matrix[user1][i]\*matrix[user2][i] (Nếu cả hai *user* đều đã đánh giá cho item *i*) then

tuSo += avgMatrix[user1][i]\*avgMatrix[user2][i]

sumOfUser1 += pow(avgMatrix[user1][i],2)

sumOfUser2 += pow(avgMatrix[user2][i],2)

Step 3: return

* + 1. **Cosine(Độ phức tạp: O(n))**

Bắt đầu

tuSo = 0

sumOfUser1 = sumOfUser2 = 0

i = 1

i <= numberOfItems

matrix[user1][i] \* matrix[user2][i] != 0

tuSo += matrix[user1][i]\*matrix[user2][i]

sumOfUser1 += pow(matrix[user1][i],2)

sumOfUser2 += pow(matrix[user2][i],2)

i++

return tuSo/sqrt(sumOfUser1\*

sumOfUser2)

Kết thúc

Đúng

Đúng

Sai

Sai

**Lưu đồ thuật toán tính tương đồng giữa *user1* và *user2* theo hàm *Cosine***

Các bước tiến hành tương tự như hàm *Pearson*, chỉ khác ở step 2 là ở hàm *Cosine* sử dụng các rating của người dùng trong khi hàm *Pearson* đã trừ đi rating trung bình của người dùng đó(tức là sử dụng giá trị của ma trận *avgMatrix*)

* + 1. **Tìm k hàng xóm (k người dùng tương đồng với user nhất)**

Bắt đầu

n = numberOfUsers, j = 1

j <= numberOfUsers

simMatrix\_Copy = simMatrix[user][j]

arr[j] = j

j++

Đúng

swapFloat(&simMatrix\_Copy[user], &simMatrix\_Copy[n])

swapInt(&arr[user], &arr[n]

Sai

i = 1

i < numberOfUser

minRating = simMatrix\_Copy[1]

index = 1

j = 1

Kết thúc

Sai

Đúng

j < n

simMatrix\_Copy[j] < minRating

minRating = simMatrix\_Copy[j]

index = j

j++

swapFloat(&simMatrix\_Copy[index],

&simMatrix\_Copy[n])

swapInt(&arr[index], &arr[n])

i++

Đúng

Sai

Đúng

Sai

**Lưu đồ thuật toán tìm *k* hàng xóm của *user***

**Ý tưởng:** Tạo ra 2 mảng một chiều, một cái dùng để lưu các hàng xóm được đánh dấu từ 1 -> numberOfUsers, một cái là các giá trị tương đồng của user với các người dùng khác. Sau đó sử dụng Selection Sort để sắp xếp trên mảng chứa các giá trị tương đồng, mỗi lần swap 2 vị trí ở mảng chứa giá trị tương đồng thì sẽ swap luôn 2 vị trí ở mảng chứa các hàng xóm. Khi kết thúc hàm, sẽ thu được mảng arr (mảng chứa hàng xóm) là các hàng xóm sao cho sim của các arr[i] là giảm dần.

Step 1: Gán n = numberOfUser, khởi tạo hai mảng simMatrix\_Copy[n + 1], arr[n + 1]

Step 2: For j: 1 -> numberOfUser

simMatrix\_Copy[j] = simMatrix[user][j]

arr[j] = j

Step 3: Hoán vị sim của người dùng user và người dùng user về cuối cùng (vì simMatrix\_Copy[user] sẽ luôn luôn bằng 1 và ta sẽ không sử dụng nó để tính toán)

swapFloat(&simMatrix\_Copy[user], &simMatrix\_Copy[n])

swapInt(&arr[user], &arr[n])

Step 4: For i: 1 -> numberOfUser – 1 (không xét *user*)

minRating = simMatrix\_Copy[1]

index = 1

For j: 1 -> n – 1

If simMatrix\_Copy[j] < minRating then

minRating = simMatrix\_Copy[j]

index = j

swapFloat(&simMatrix\_Copy[index], &simMatrix\_Copy[n – 1])

swapInt(&arr[index], &arr[n - 1])

* Độ phức tạp thuật toán:
* Số lần thực hiện phép toán là: 1 + 2 + 3 + 4 + …. + n – 1 =
* Vậy độ phức tạp thuật toán là O( O() = O()
  + 1. **Dự đoán Rating(Độ phức tạp: O(n))**

Bắt đầu

tuSo = 0

mauSo = 0

i = 1

i <= k

tuSo += simMatrix[user][arr[i]]\*avgMatrix[arr[i]][item]

mauSo += abs(simMatrix[arr[i]][user]

i++

return tuSo/mauSo

Kết thúc

Đúng

Sai

**Lưu đồ thuật toán dự đoán điểm đánh giá người dùng**

Step 1: Gán tuSo = 0 và mauSo = 0

Step 2: For i: 1 -> k (k: số người dùng tương đồng nhất)

tuSo += simMatrix[user][arr[i]]\*avgMatrix[arr[i]][item]

mauSo += abs(simMatrix[arr[i]][user]

Trong đó: arr[i] là người dùng tương đùng với *user*

Step 3: return tuSo/mauSo

# CHƯƠNG TRÌNH VÀ KẾT QUẢ

## Tổ chức chương trình.

Sau khi nghiên cứu bài toán, nhóm chúng em chia chương trình thành các hàm nhỏ để dễ tổ chức, quản lí và bảo trì chương trình.

* ***void inputMatrixFromKeyBoard(float matrix[][150], int &numberOfUsers, int &numberOfItems):*** Hàm dùng để nhập input từ bàn phím.
* ***int inputMatrixFromFile(char fileName[], float matrix[][150], int &numberOfUsers, int &numberOfItems):*** Hàm đọc giá trị ma trận từ file, đọc thành công trả về 1, ngược lại trả về 0.
* ***void outputMatrix(char fileName[], float matrix[][150], int rows, int columns):*** Hàm dùng để ghi kết quả ra file.
* ***float getAvgRatingOfUser(float matrix[][150], int user, int numberOfItems):*** Hàm tính giá trị rating trung bình của người dùng.
* ***float getSimilarityPearson(float matrix[][150], float avgMatrix[][150], int user1, int user2, int numberOfItems):*** Hàm tính giá trị tương đồng giữa*user1* và*user2* theo hàm *Pearson*.
* ***float getSimilarityCosine(float matrix[][150], int user1, int user2, int numberOfItems):*** Hàm tính giá trị tương đồng giữa*user1* và*user2* theo hàm Cosine.
* ***void getNeighbor(float simMatrix[][150], int numberOfUsers, int user, int k, int arr[]):*** Hàm để lấy các hàng xóm của *user*, các hàng xóm trả về lưu trữ ở mảng một chiều *arr*.
* ***float getRating(float matrix[][150], float simMatrix[][150],float calMatrix[][150], int numberOfUsers, int numberOfItems, int k, int arr[], int user, int item):*** Hàm trả về điểm rating dự đoán của *user* cho *item*.

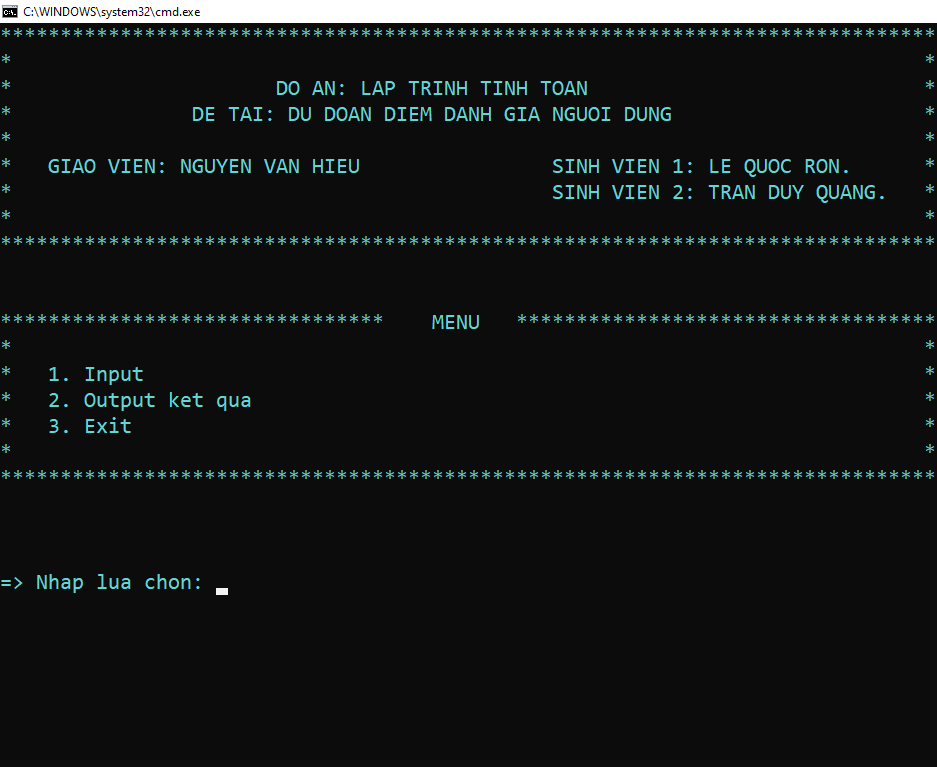
## Ngôn ngữ cài đặt

Là sinh viên năm nhất, nhóm chúng em chỉ mới học qua học phần *“Kỹ thuật lập trình”* và chỉ tiếp cận với ngôn ngữ C. Nên nhóm chúng em sẽ sử dụng ngôn ngữ C để cài đặt chương trình.

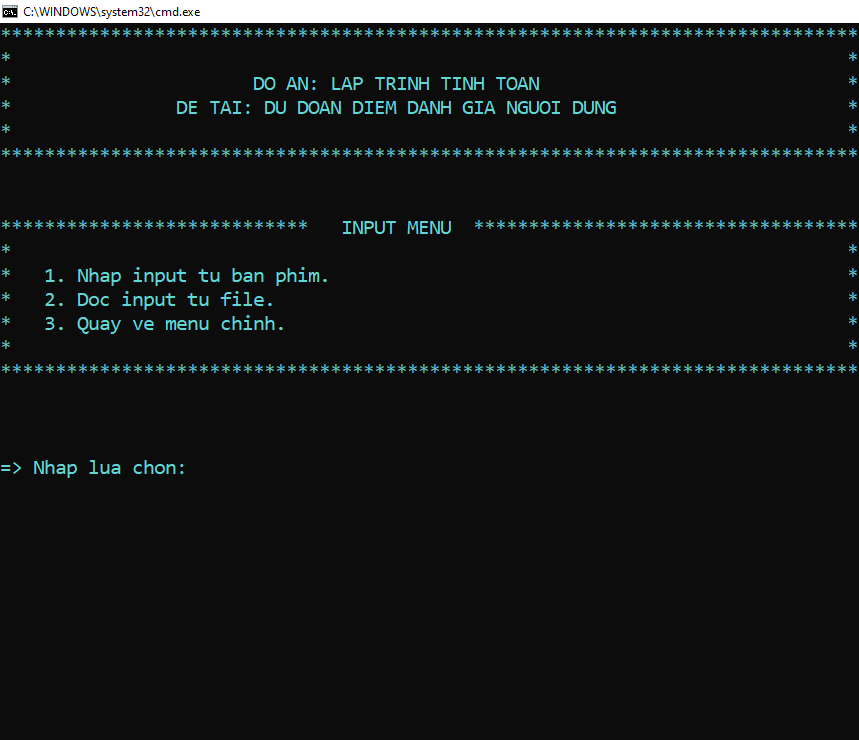
## Kết quả

### Giao diện chính của chương trình

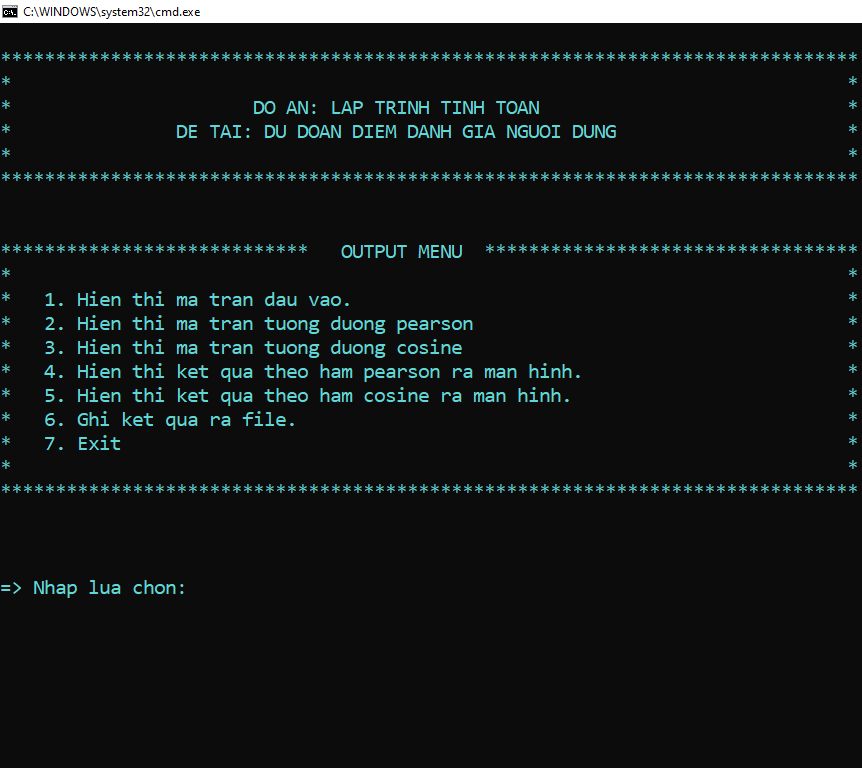
Nhóm chúng em chia chương trình thành 3 menu chính như dưới đây:

* Menu chính dùng để truy cập vào các menu input và menu output.

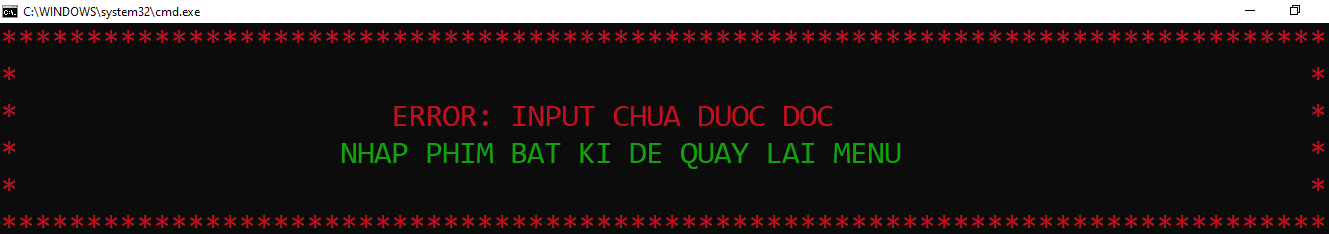
**Hình 8.** **Menu chính của chương trình**

* Input menu dùng để nhập input với các lựa chọn như hình dưới.

**Hình 9. Input menu**

* Output menu dùng để hiển thị output ra cửa sổ console và dùng để ghi kết quả ra file. Để có thể vào được output menu thì người dùng phải nhập input trước, nếu không chương trình sẽ báo lỗi. 

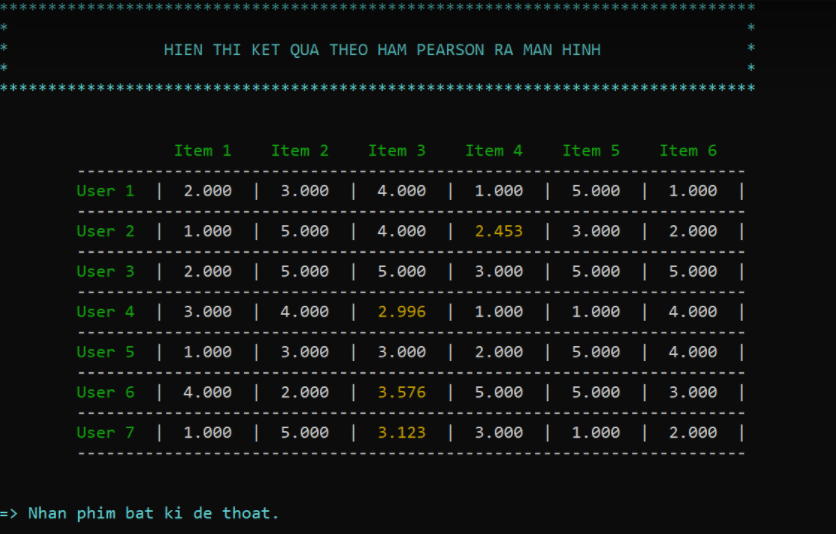
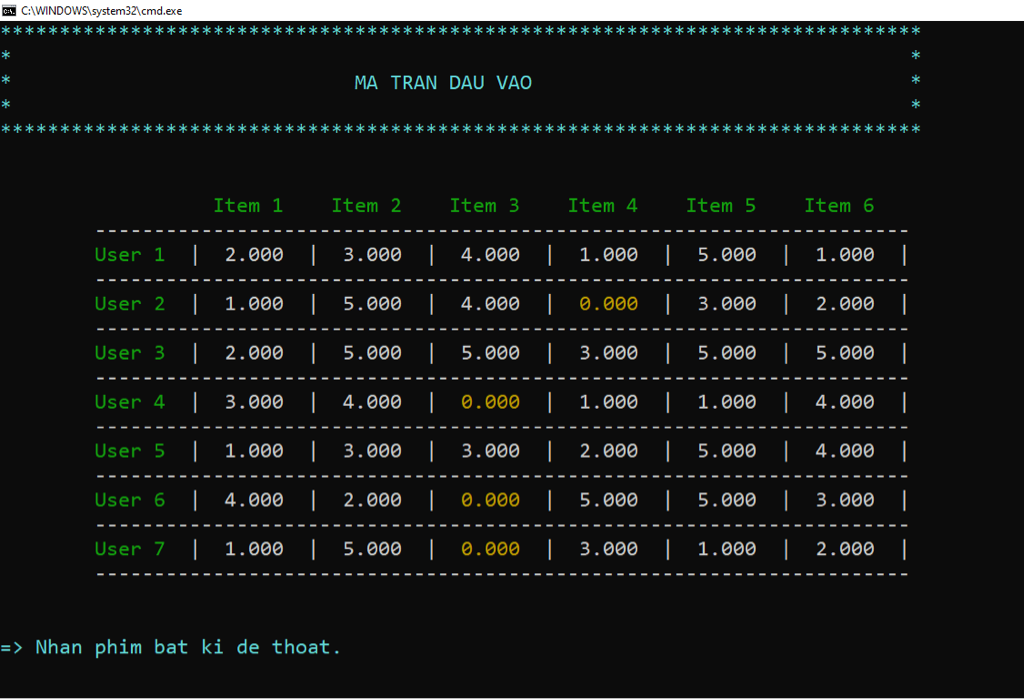
**Hình 10. Output menu**



**Hình 11. Báo lỗi khi người dùng chưa nhập vào input**

### Kết quả thực thi của chương trình

## 4.3.2.1. Trường hợp 1

**Ma trận đầu vào**

**Kết quả theo hàm *Pearson***

### Kết quả theo hàm Cosine

## 4.3.2.2. Trường hợp 2

**Ma trận đầu vào**

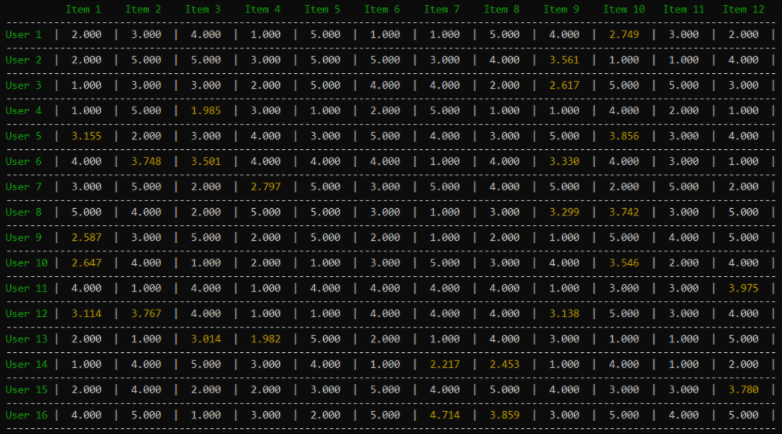
### 

### Kết quả theo hàm Pearson

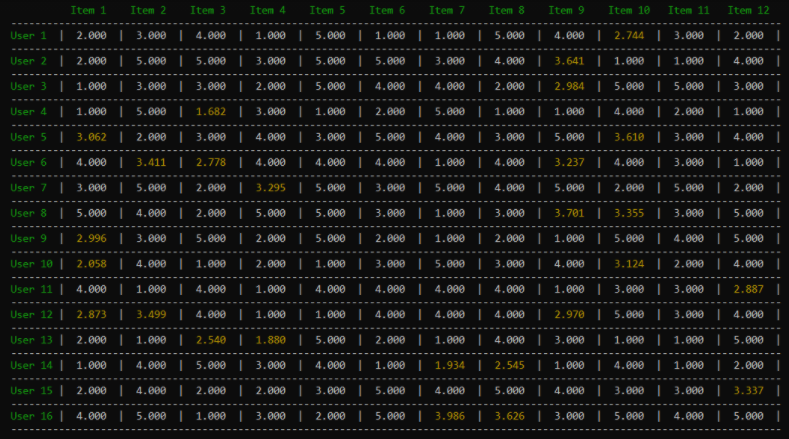
**Kết quả theo hàm** ***Cosine***

## 4.3.2.3. Trường hợp 3.

**Ma trận đầu vào**



**Kết quả theo hàm *Pearson***

**Kết quả theo hàm *Cosine***

### Nhận xét đánh giá

Vì tính chất của bài toán là “dự đoán” nên sẽ không có một kết quả chính xác cụ thể, chính vì thế các kết quả đưa trên đều mang tính chất tương đối.

Với phương pháp lọc cộng tác, dữ liệu đầu vào đóng vai trò rất quan trọng trong việc tính độ tương đồng giữa các người dùng, dự đoán rating. Ta có thể thấy là ở trường hợp 1, ma trận đầu vào là ma trận cấp (7x6), tức là dữ liệu để tính toán vẫn chưa nhiều nên dẫn đến kết quả có thể sai số tương đối lớn, ngược lại ở các trường hợp 2 và 3, ta thấy sai số giữa các kết quả theo hàm Cosine và hàm Pearson có giảm đi đáng kể.

Tóm lại, dữ liệu đầu vào đóng vai trò rất quan trọng, dữ liệu càng nhiều sai số sẽ càng giảm, kết quả càng chính xác.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Qua chương trình ta có thể dự đoán được rating của người dùng với các sản phẩm.

Việc dự đoán sẽ không có một kết quả chính xác nhất định, ta chỉ có thể thu được một kết quả tương đối, dữ liệu đầu vào càng lớn thì kết quả càng chính xác.

Với các người dùng mới hoặc các sản phẩm mới thì chương trình sẽ gặp khó khăn vì thiếu dữ liệu tính toán.

## Hướng phát triển

Có thể thấy, đây là cơ sở, nền tảng để xây dựng nên một hệ thống gợi ý hoàn chỉnh.

Ở đây dữ liệu đầu vào thì nó sẽ tự động thu thập ở trên chính hệ thống thông qua tương tác của người dùng như là số lần click vào sản phẩm, dữ liệu người dùng nhập ở thanh tìm kiếm, các câu hỏi khi người dùng truy cập hoặc là đánh giá trực tiếp sản phẩm như Youtube, Netflix,...

Sau khi có dữ liệu, ta sẽ dự đoán được rating và sẽ gợi ý đến với người dùng.

Ta có thể áp dụng nó vào các trang web thương mại, tin tức,... giúp tăng trải nghiệm cho người dùng và tăng doanh thu cho các doanh nghiệp.

TÀI LIỆU THAM KHẢO

**[1]**. Introduction to Algorithmic Marketing, Lya Katsov, 2018.

**[2]**. Wikipedia, Pearson correlation coefficient,<https://en.wikipedia.org/wiki/Pearson_>

correlation\_coefficient

**[3]**. Wikipedia, Recommender system, <https://en.wikipedia.org/wiki/Recommender>

\_system

**[4]**. Machine Learning cơ bản, Neighborhood-Based Collaborative Filtering, <https://machinelearningcoban.com/2017/05/24/collaborativefiltering/>

**[5]**. Nguyễn Thành Hậu, Xây dựng một hệ thống gợi ý Collaborative Filtering dễ dàng như thế nào?, https://viblo.asia/p/xay-dung-mot-he-thong-goi-y-collaborative-filtering

-de-dang-nhu-the-nao-GrLZDXv3Zk0

PHỤ LỤC

## Tính rating trung bình của người dùng*.*

*float getAvgRatingOfUser(float matrix[][150],int user,int numberOfItems){*

*int count = 0;*

*float sum = 0;*

*for (int j = 1;j <= numberOfItems;j++){*

*if (matrix[user][j] != 0) {*

*count++;*

*sum += matrix[user][j];*

*}*

*}*

*return (sum/count);*

*}*

## Tính độ tương đồng giữa user1 & user2

*float getSimilarityPearson(float matrix[][150],float avgMatrix[][150],int user1,int user2,int numberOfItems){*

*float tuSo = 0;*

*float sumOfUser1 = 0;*

*float sumOfUser2 = 0;*

*for (int i = 1;i <= numberOfItems;i++){*

*if (matrix[user1][i] \* matrix[user2][i] != 0){*

*tuSo += avgMatrix[user1][i]\*avgMatrix[user2][i];*

*sumOfUser1 += pow(avgMatrix[user1][i], 2);*

*sumOfUser2 += pow(avgMatrix[user2][i], 2);*

*}*

*}*

*return (tuSo/(sqrt(sumOfUser1)\*sqrt(sumOfUser2)));*

*}*

## Tính độ tương đồng giữa user1 & user2

*float getSimilarityCosine(float matrix[][150],int user1,int user2,int numberOfItems){*

*float tuSo = 0;*

*float sumOfUser1 = 0;*

*float sumOfUser2 = 0;*

*for (int i = 1;i <= numberOfItems;i++){*

*if (matrix[user1][i]\*matrix[user2][i] != 0){*

*tuSo += matrix[user1][i]\*matrix[user2][i];*

*sumOfUser1 += pow(matrix[user1][i], 2);*

*sumOfUser2 += pow(matrix[user2][i], 2);*

*}*

*}*

*return tuSo/(sqrt(sumOfUser1)\*sqrt(sumOfUser2));*

*}*

## 4. Hoán vị a và b(số thực)

*void swapFloat(float \*a, float \*b){*

*float temp = \*a;*

*\*a = \*b;*

*\*b = temp;*

*}*

**5.          Hoán vị a và b(số nguyên)**

*void swapInt(int \*a, int \*b){*

*int temp = \*a;*

*\*a = \*b;*

*\*b = temp;*

*}*

**6.             Tìm hàng xóm(các user tương đồng nhất).**

*void getNeighbor(float simMatrix[][150],int numberOfUsers,int user, int k, int arr[]) {*

*int n = numberOfUsers;*

*float simMatrix\_Copy[numberOfUsers + 1];*

*for (int j = 1;j <= numberOfUsers;j++){*

*simMatrix\_Copy[j] = simMatrix[user][j];*

*arr[j] = j;*

*}*

*swapFloat(&simMatrix\_Copy[user],&simMatrix\_Copy[n]);*

*swapInt(&arr[user],&arr[n]);*

*for (int i = 1;i < numberOfUsers;i++){*

*float minRating = simMatrix\_Copy[1];*

*int index = 1;*

*for (int j = 1;j < n;j++){*

*if (simMatrix\_Copy[j] < minRating){*

*minRating = simMatrix\_Copy[j];*

*index = j;*

*}*

*}*

*swapFloat(&simMatrix\_Copy[index], &simMatrix\_Copy[n - 1]);*

*swapInt(&arr[index], &arr[n - 1]);*

*n--;*

*}*

*}*

**7.         Dự đoán rating**

*float getRating(float matrix[][150], float simMatrix[][150],float avgMatrix[][150], int k,*

*int arr[], int user, int item){*

*float tuSo = 0;*

*float mauSo = 0;*

*for (int i = 1;i <= k;i++){*

*tuSo += simMatrix[arr[i]][user]\*avgMatrix[arr[i]][item];*

*mauSo += abs(simMatrix[arr[i]][user]);*

*}*

*return (tuSo/mauSo);*

*}*

**8.         Hàm xử lí(tính toán và xử lí toàn bộ bài toán)**

*void run(float matrix[][150],float avgMatrix[][150], float simMatrixPearson[][150], float simMatrixCosine[][150],float resultMatrixCosine[][150],*

*float resultMatrixPearson[][150],int indexOfNoneRating[],int arrPearson[],int arrCosine[], int k, int numberOfUsers, int numberOfItems){*

*int q = 0;*

*for (int i = 1;i <= numberOfUsers;i++){*

*float avgRatingOfUser = getAvgRatingOfUser(matrix, i, numberOfItems);*

*for (int j = 1;j <= numberOfItems;j++){*

*if (matrix[i][j] != 0) avgMatrix[i][j] = matrix[i][j] - avgRatingOfUser;*

*else avgMatrix[i][j] = 0;*

*}*

*}*

*for (int i = 1;i <= numberOfUsers;i++){*

*for (int j = 1;j <= numberOfUsers;j++){*

*simMatrixPearson[i][j] = getSimilarityPearson(matrix, avgMatrix, i, j, numberOfItems);*

*simMatrixCosine[i][j] = getSimilarityCosine(matrix, i, j, numberOfItems);*

*}*

*}*

*for (int i = 1;i <= numberOfUsers;i++){*

*float avgRating = getAvgRatingOfUser(matrix,i,numberOfItems);*

*getNeighbor(simMatrixPearson, numberOfUsers, i, k, arrPearson);*

*for (int j = 1;j <= numberOfItems;j++){*

*if (matrix[i][j] == 0) {*

*indexOfNoneRating[q++] = ((i-1)\*numberOfItems) + j;*

*resultMatrixPearson[i][j] = avgRating + getRating(matrix, simMatrixPearson, avgMatrix,  k, arrPearson, i, j);*

*}else{*

*resultMatrixPearson[i][j] = matrix[i][j];*

*}*

*}*

*}*

*for (int i = 1;i <= numberOfUsers;i++){*

*float avgRating = getAvgRatingOfUser(matrix,i,numberOfItems);*

*getNeighbor(simMatrixCosine, numberOfUsers, i, k, arrCosine);*

*for (int j = 1;j <= numberOfItems;j++){*

*if (matrix[i][j] == 0) {*

*resultMatrixCosine[i][j] = avgRating + getRating(matrix, simMatrixCosine, avgMatrix,  k, arrCosine, i, j);*

*}else{*

*resultMatrixCosine[i][j] = matrix[i][j];*

*}*

*}*

*}*

*}*

**9.        Nhập input từ bàn phím.**

*void inputMatrixFromKeyBoard(float matrix[][150], int &numberOfUsers, int &numberOfItems){*

*printf("Nhap so luong nguoi dung: ");scanf("%d",&numberOfUsers);*

*printf("Nhap so luong san pham: ");scanf("%d",&numberOfItems);*

*printf("Nhap vao ma tran.\n");*

*for (int i = 1;i <= numberOfUsers;i++){*

*for (int j = 1;j <= numberOfItems;j++){*

*scanf("%f",&matrix[i][j]);*

*}*

*}*

*}*

**10.       Nhập input từ file.**

*int inputMatrixFromFile(char fileName[], float matrix[][150], int &numberOfUsers, int &numberOfItems){*

*FILE \*fptr;*

*fptr = fopen(fileName,"r");*

*int rows = 1, columns = 0;*

*char c;*

*if (fptr == NULL){*

*return 0;*

*}else{*

*while((c = fgetc(fptr)) != EOF){*

*if (c == '\n'){*

*rows += 1;*

*columns = 1;*

*}else{*

*fseek(fptr, -1, SEEK\_CUR);*

*columns += 1;*

*}*

*fscanf(fptr,"%f",&matrix[rows][columns]);*

*}*

*}*

*numberOfUsers = rows;*

*numberOfItems = columns;*

*fclose(fptr);*

*return 1;*

*}*

**11.      Ghi kết quả vào file**

*void outputMatrix(char fileName[], float matrix[][150], int rows, int columns){*

*FILE \*fptr = fopen(fileName,"w");*

*if (fptr == NULL){*

*printf("Can't open file.");*

*exit(1);*

*}else{*

*for (int i = 1;i <= rows;i++){*

*for (int j = 1;j <= columns;j++)  fprintf(fptr, "%2.2f\t", matrix[i][j]);*

*fprintf(fptr, "\n" );*

*}*

*}*

*fclose(fptr);*

*}*