DSA Hackathon
Explanation of Functions in the Program

This program implements a basic e-commerce system with functionality to handle users, products, and browsing history while generating product recommendations. Here's an explanation of how each function works:

1) Adding Users

The addUser function allows the system to add new users by assigning them a unique ID and name. We ensured that the user's information is stored efficiently by using a hash table. Each user is linked to their respective index in the hash table using a linked list. This helps manage any collisions that might occur when two user IDs hash to the same index. The hash table allows us to access user data quickly.

2) Adding Products

The addProduct function registers products in the system by assigning them a unique ID and name. Just like the users, the products are stored in a hash table for fast access. We ensure that all product data is linked properly using linked lists at each index, making it easy to retrieve and process information even if multiple products hash to the same index.

3) Tracking Browsing History

The addBrowsingHistory function tracks which products each user has browsed. We store the user ID and product ID together in a linked list. This makes it simple to traverse the browsing history and identify the products a user has shown interest in. This history forms the basis for generating personalized recommendations.

4) Generating Recommendations

The generateRecommendations function is the highlight of the program. It analyzes a user's browsing history and identifies the products they have interacted with. By matching the product IDs in the browsing history with the products stored in the hash table, it generates a list of relevant recommendations. If no products are found in the browsing history, the function gracefully informs the user that no recommendations are available.

Main Function

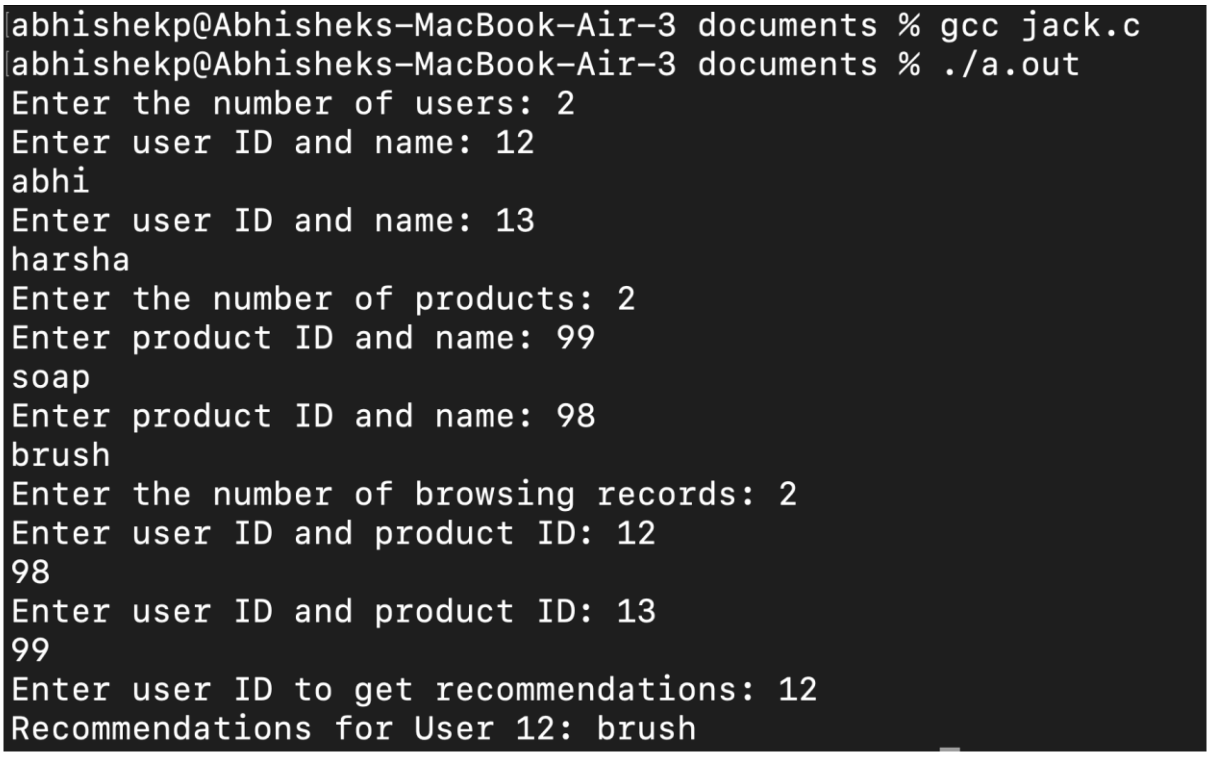
The main function ties everything together. It prompts the user to add their details, register products, and record browsing activities. Once all the data is collected, it allows the user to retrieve personalized recommendations. The program ensures a smooth flow, handling all interactions step by step.

Data Structures Used

We chose to use hash tables for managing users and products because of their efficiency in accessing data. Each user and product is represented as a node in their respective hash table, and linked lists help manage collisions effectively. The browsing history is stored in a separate linked list, making it easy to traverse and analyze.

By designing the system in this way, we achieved a modular and efficient solution for managing user data, product catalogs, and personalized recommendations.

If you’d like this explanation saved in a document format, let me know!



```
abhishekp@Abhisheks-MacBook-Air-3 documents % gcc jack.c
abhishekp@Abhisheks-MacBook-Air-3 documents % ./a.out
Enter the number of users: 2
Enter user ID and name: 12
abhi
Enter user ID and name: 13
harsha
Enter the number of products: 2
Enter product ID and name: 99
soap
Enter product ID and name: 98
brush
Enter the number of browsing records: 2
Enter user ID and product ID: 12
98
Enter user ID and product ID: 13
99
Enter user ID to get recommendations: 12
Recommendations for User 12: brush
```