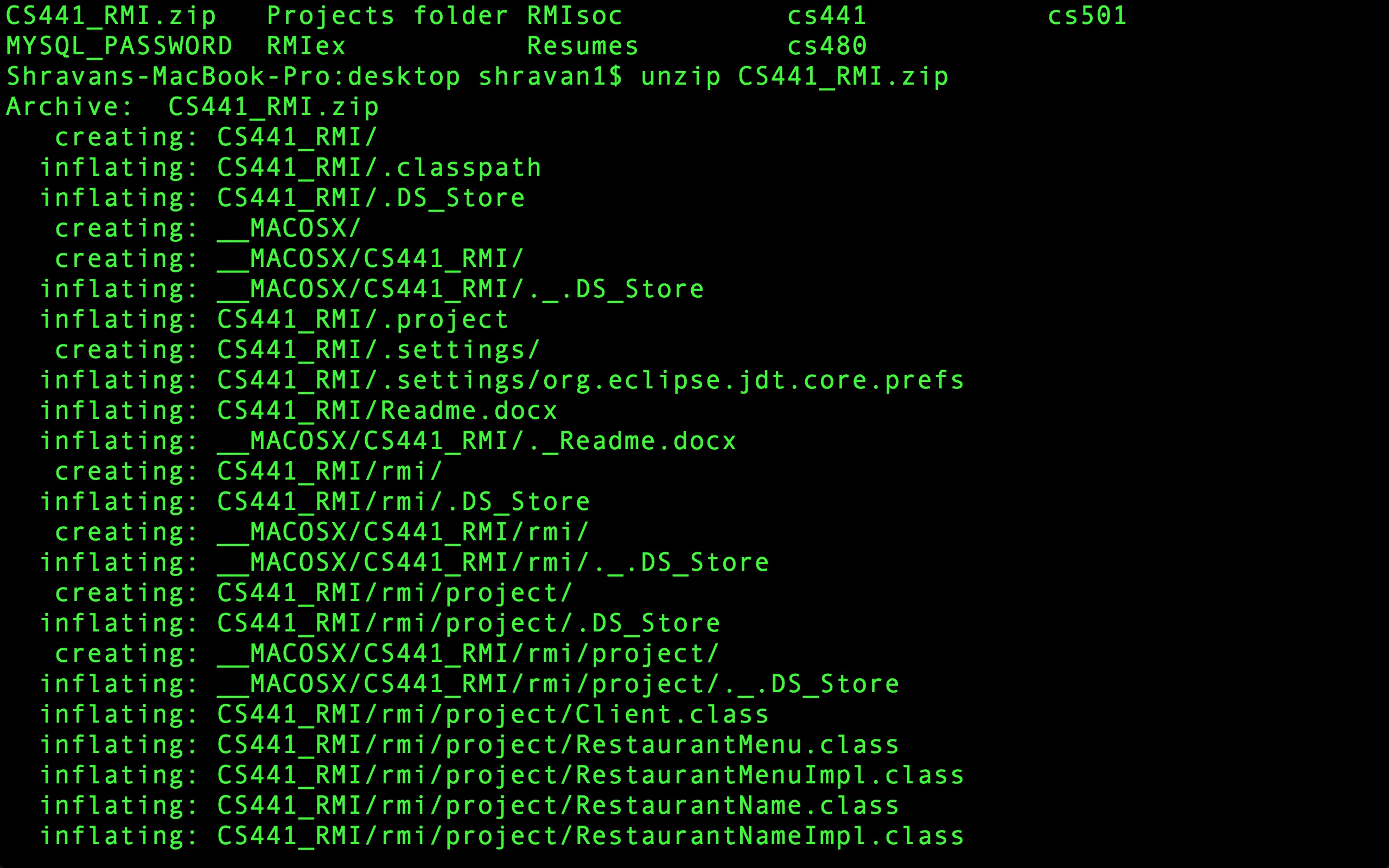
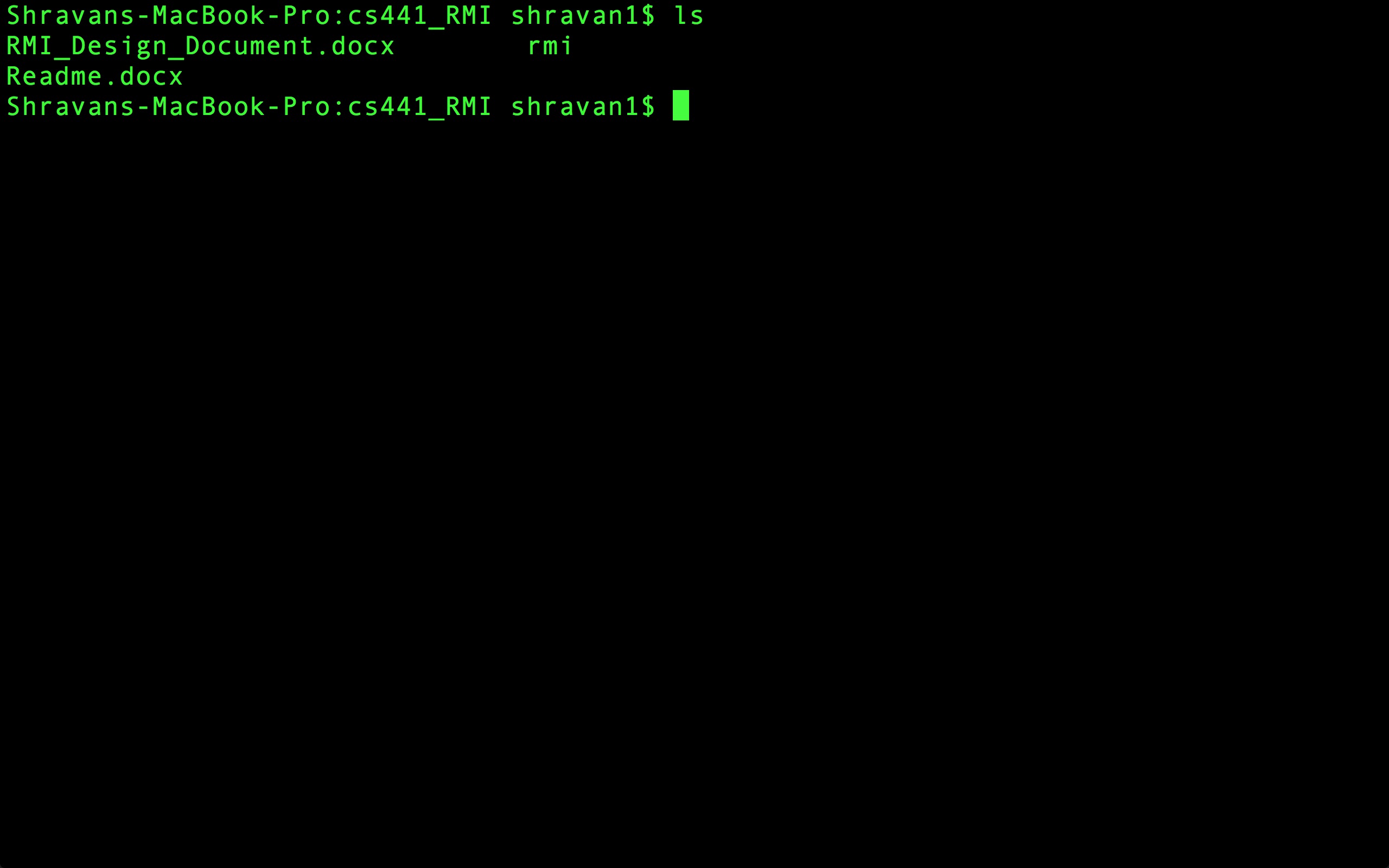
Running the program

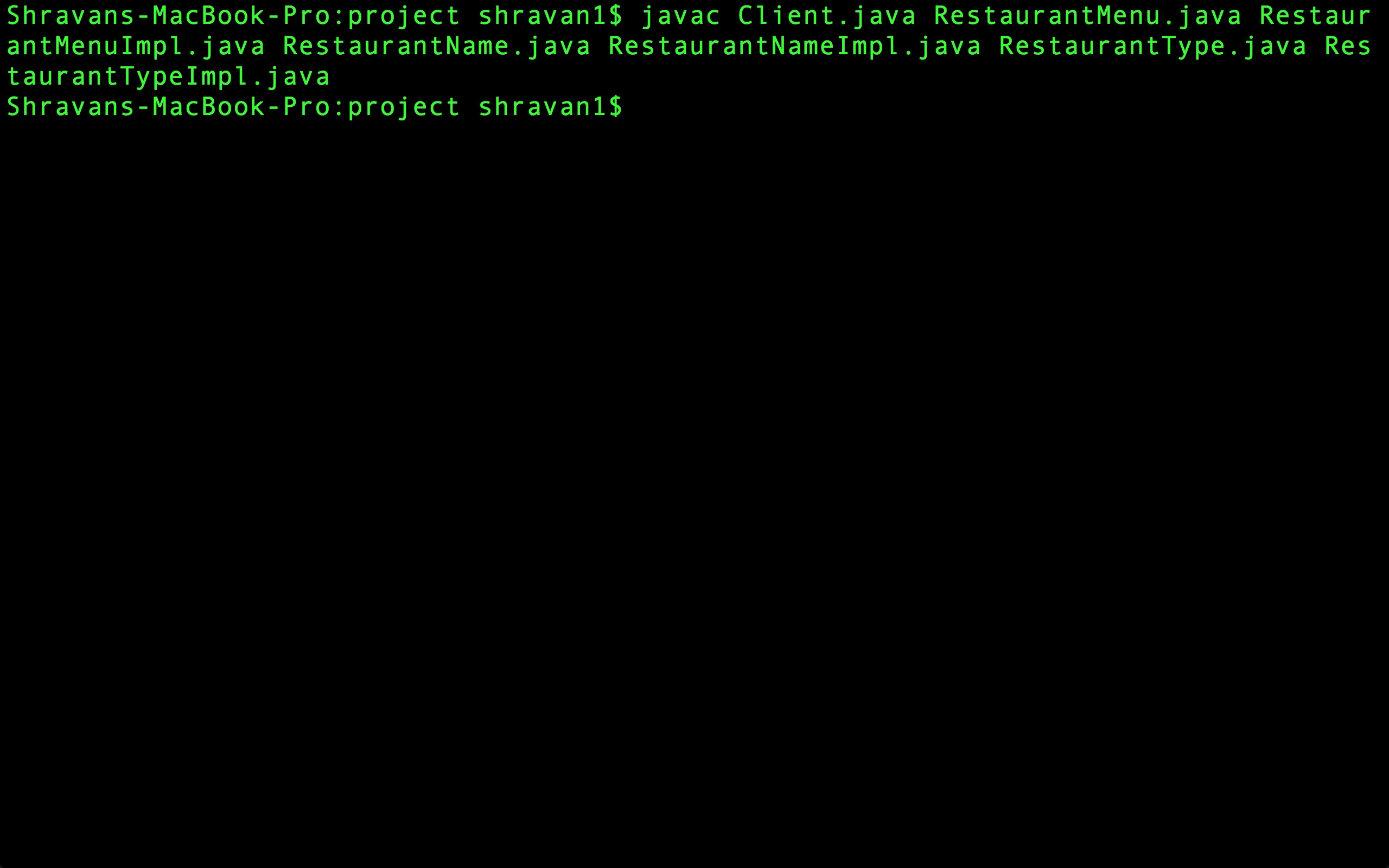
Unzip the zip folder cs441\_RMI.zip

This folder contains rmi, readme file and design document





Compile the classes (javac Client.java RestaurantMenu.java RestaurantMenuImpl.java RestaurantName.java RestaurantNameImpl.java RestaurantType.java RestaurantTypeImpl.java)



Now run the RestaurantMenu server (java -classpath /Users/shravan1/Desktop/CS441\_RMI -Djava.rmi.server.codebase=file:/Users/shravan1/Desktop/CS441\_RMI/ rmi.project.RestaurantMenuImpl)

The output would be

Restaurant Menu Server Ready

Open a new instance of the terminal. Go to the folder which contains the class files.

Now start the RestaurantName server(java -classpath /Users/shravan1/Desktop/CS441\_RMI -Djava.rmi.server.codebase=file:/Users/shravan1/Desktop/CS441\_RMI/ rmi.project.RestaurantNameImpl)

The output would be

Restaurant Name server bound

Open a new instance of the terminal. Go to the folder which contains the class files.

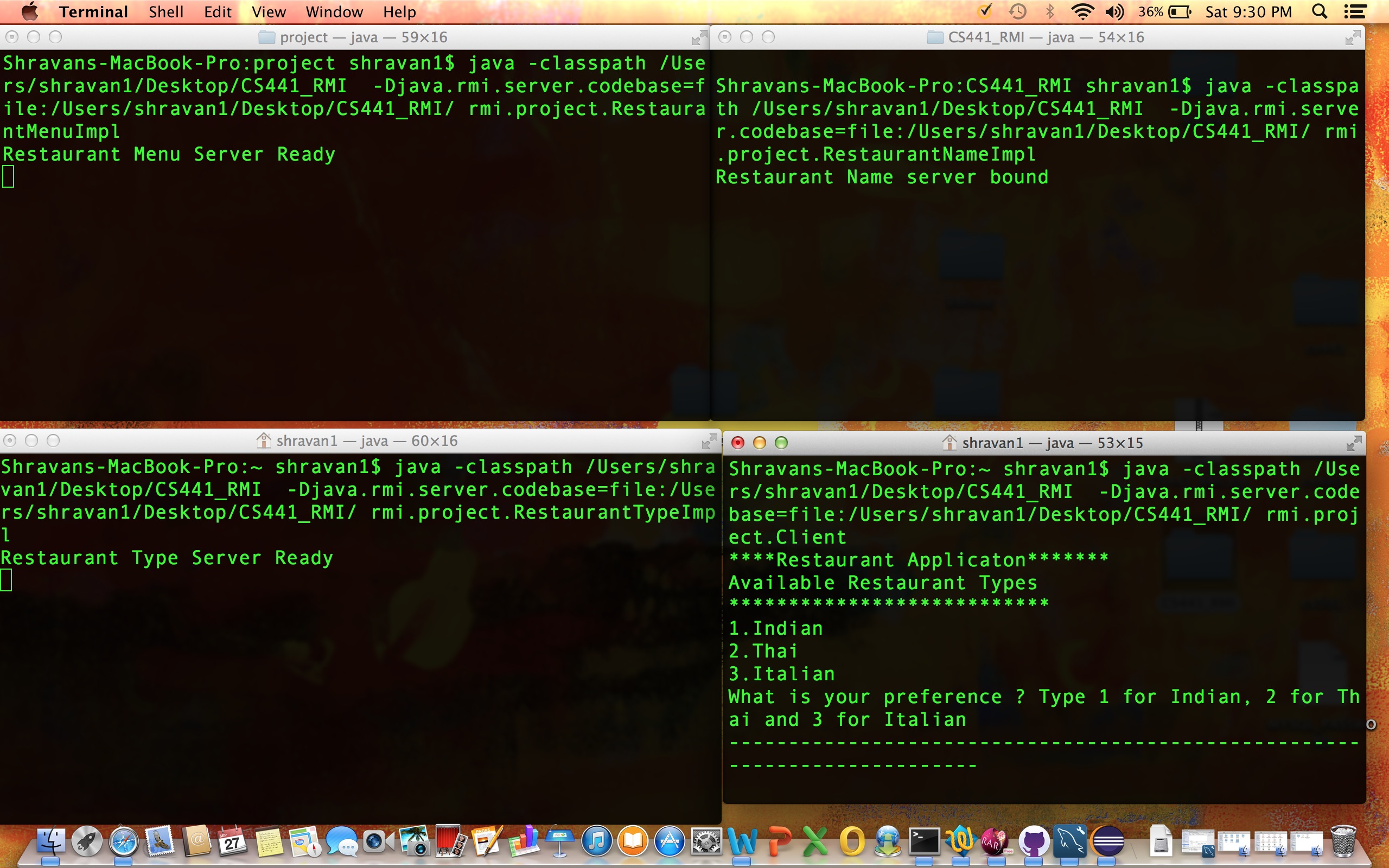
Now start the RestaurantType server(java -classpath /Users/shravan1/Desktop/CS441\_RMI -Djava.rmi.server.codebase=file:/Users/shravan1/Desktop/CS441\_RMI/ rmi.project.RestaurantTypeImpl)

The output would be

Restaurant Type server Ready

Open a new instance of the terminal. Go to the folder which contains the class files.

Now start the Client server(java -classpath /Users/shravan1/Desktop/CS441\_RMI -Djava.rmi.server.codebase=file:/Users/shravan1/Desktop/CS441\_RMI/ rmi.project.Client)



The output would be

\*\*\*\*Restaurant Applicaton\*\*\*\*\*\*\*

Available Restaurant Types

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1.Indian

2.Thai

3.Italian

What is your preference ? Type 1 for Indian, 2 for Thai and 3 for Italian

--------------------------------------------------------------------------

Type the restaurant number of your preference

The output should be like this if you select 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Available restaurants under your preference

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1.Patio

2.Come Italy

Enter the number of your preferred restaurant to get the corresponding Menu

--------------------------------------------------------------------------

Type the number of the restaurant of your preference.

The output should be like this if you type 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Menu list of Selected Restaurant

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Fudge - $3.99

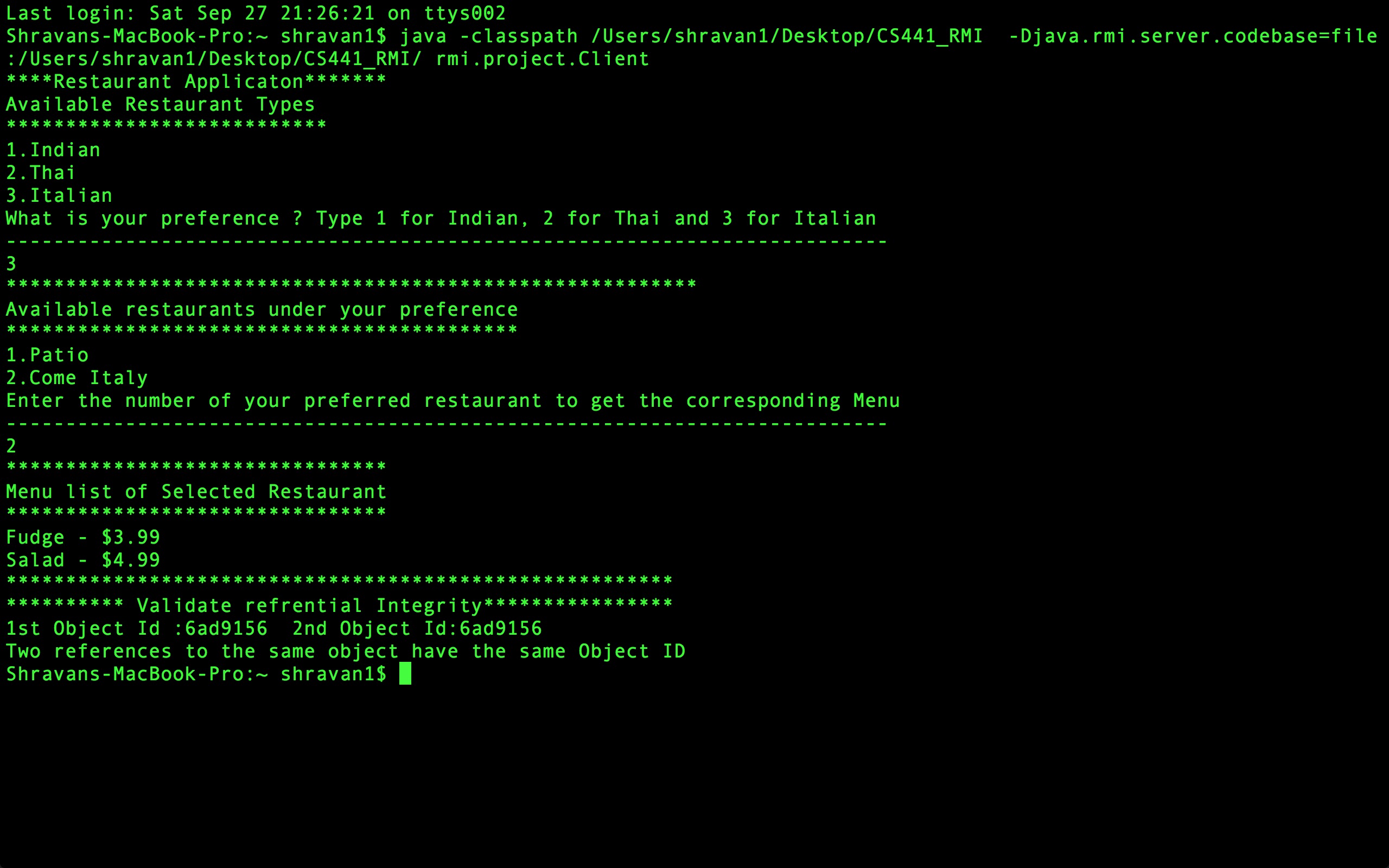
Salad - $4.99

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\* Validate refrential Integrity\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1st Object Id :6ad9156 2nd Object Id:6ad9156

Two references to the same object have the same Object ID



Summary: After the menu is fetched, a client object is created with two references to it. The references are then passed to the RestaurantType server method as two parameters. Making the client object to implement serializable makes this possible. The two references are then passed as parameters to RestaurantMenu server and the object ids of these respective references are extracted and the text is passed as a string to client object, which prints the text on its console.

As the two references have the same object id, Referential integrity is maintained.