

**ASIA PACIFIC UNIVERSITY OF TECHNOLOGY & INNOVATION**

**CT010-3-1-FSD**

**FUNDEMENTAL OF SOFTWARE DEVELOPMENT**

**GROUP ASSIGNMENT**

**CHAN JIA LE TP049952**

**LEE JIN HENG TP053710**

**MOHAMMED ABDULMAJID**

**MOHAMED H AHMED TP051980**

**GROUP MEMBERS:**

**LECTURER: EN. SURESH A/L S.SAMINATHAN**

**INTAKE CODE: UC1F1809**

Contents

[1.0 | Introduction 3](#_Toc533876285)

[2.0 | Workload Matrix 4](#_Toc533876286)

[3.0 | Pseudocode 5](#_Toc533876287)

[4.0 | Sample Output Screen 10](#_Toc533876288)

[4.0.1 | Main Menu Sample Output 10](#_Toc533876289)

[4.0.2 | Purchasing Module Sample Output 11](#_Toc533876290)

[4.0.3 | Choosing a Ferry after Choosing Business Class or Economic Class seats 12](#_Toc533876291)

[4.0.4 | Selecting Seats after choosing which Ferry to take 13](#_Toc533876292)

[4.0.5 | If Business Class has been fully booked 15](#_Toc533876293)

[4.0.6 | If Economic Class has been fully booked 15](#_Toc533876294)

[4.0.7 | Successfully Chosen A Seat 16](#_Toc533876295)

[4.0.8 | Printing Ticket for users 17](#_Toc533876296)

[4.0.9 | Viewing Seats Sample Output 19](#_Toc533876297)

[4.0.10 | Quit Module Sample Output 20](#_Toc533876298)

[5.0 | Additional features of the system 20](#_Toc533876299)

[5.0.1 | Timetable Displayed 20](#_Toc533876300)

[5.0.2 | Saving Seating Information using Pickle Module 20](#_Toc533876301)

[5.0.3 | Saving Ticket Information 20](#_Toc533876302)

[5.0.4 | Choosing Other Ferries After Economic Class Seats Are Fully Booked 20](#_Toc533876303)

[5.0.5 | Purchasing Another Ticket After The First Purchase 21](#_Toc533876304)

[6.0 | Assumptions 21](#_Toc533876305)

[7.0 | References 21](#_Toc533876306)

# 1.0 | Introduction

This content involves in the design of a ferry ticketing system by using the Python programming language. The system is consists of 8 ferries, having trips from both from Penang to Langkawi and Langkawi to Penang. Timing from 10 a.m. to 5 p.m. daily. Each ferry contains 50 seats, 10 for the business class seats, and 40 seats for the economic class. Each ferry has different IDs. The system allows the consumers to purchase tickets in the ferry and viewing seats that are currently available. The system has a clear layout and instruction with some additional features that allows the system to be more practical and easy to be used by both the programmer and the consumer.

This assignment has contained our effort in designing the system, which includes the source code, pseudocode, workload matrix and the added features and function of the system. Our team will give further explanations of every sample result (output) and additional function.

# 2.0 | Workload Matrix

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Name** | **Design** | **Coding** | **Documentation** | **Signature** |
| 1. | Chan Jia Le | 34% | 40% | 40% |  |
| 2. | Lee Jin Heng | 33% | 30% | 30% |  |
| 3. | Mohammed Abdulmajid Mohamed H Ahmed | 33% | 30% | 30% |  |

# 3.0 | Pseudocode

BEGIN

ferry\_business = [[ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ]

ferry\_economic = [[ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ] , [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ] , [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ], [ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ]]

while True:

Print (“-------------------------------------------------------------------------------------------------”)

Print (“ TIME TABLE: PENANG TO LANGKAWI | LANGKAWI TO PENANG ”)

Print (“ FERRY ID TIME FERRY TIME ”)

Print (“ 001 1000 002 1100 ”)

Print (“ 003 1200 004 1300 ”)

Print (“ 005 1400 006 1500 ”)

Print (“ 007 1600 008 1700 ”)

Print (“TYPE P – TO PURCHASE TICKETS”)

Print (“TYPE V – TO VIEW SEATINGS”)

Print (“TYPE Q – TO QUIT THE SYSTEM”)

Read (user\_input)

IF (user\_input == “P”) :

Print (“ TYPE B – TO PURCHASE BUSINESS CLASS SEATS”)

Print (“ TYPE E – TO PURCHASE ECONOMIC CLASS SEATS”)

Print (“ TYPE M – TO RETURN TO MAIN MENU”)

Read ( user\_input2 )

IF ( user\_input2 == “B” ) :

Read ( ferry\_selection )

Print ( ferry\_business [ferry\_selection - 1][0 : 5] )

Print ( ferry\_business [ferry\_selection - 1][5 : 10] )

Print ( “Please Select A Seat ( 1 – 10 )” )

Read ( seat\_selection )

ferry\_business [ferry\_selection - 1][ seat\_selection -1] = 1

Print (“ Thank You for purchasing the tickets ”)

Time = ferry\_selection + 9

Print (“ Here is you ticket”)

Print ( “TIME: ”, Time, “00” ,“ ”, “FERRY: 00”,ferry\_selection,)

IF ( seat\_selection % 2 == 0):

Destination = “Langkawi to Penang”

ELSE:

Destination = “Penang to Langkawi”

ENDIF

Print ( “SEATING: B”, seat selection, “ DESTINATION: ”,Destination)

Break

ELSE:

IF ( user\_input2 == “E”):

Read ( ferry\_selection )

Print ( ferry\_economic [ferry\_selection - 1][0 : 5] )

Print ( ferry\_economic [ferry\_selection - 1][5 : 10] )

Print ( ferry\_economic [ferry\_selection - 1][10 :15] )

Print ( ferry\_economic [ferry\_selection - 1][15 : 20] )

Print ( ferry\_economic [ferry\_selection - 1][20 : 25] )

Print ( ferry\_economic [ferry\_selection - 1][25 : 30] )

Print ( ferry\_economic [ferry\_selection - 1][30 : 35] )

Print ( ferry\_economic [ferry\_selection - 1][35 : 40] )

Print ( “Please Select A Seat ( 1 – 40 )” )

Read ( seat\_selection )

ferry\_business [ferry\_selection - 1][ seat\_selection -1] = 1

Print (“ Thank You for purchasing the tickets ”)

Time = ferry\_selection + 9

Print (“ Here is you ticket”)

Print ( “TIME: ”, Time, “00” ,“ ”, “FERRY: 00”,ferry\_selection,)

IF ( seat\_selection % 2 == 0):

Destination = “Langkawi to Penang”

ELSE:

Destination = “Penang to Langkawi”

ENDIF

Print ( “SEATING: B”, seat selection, “ DESTINATION: ”,Destination)

Break

ENDIF

ELSE:

IF ( user\_input2 == “M”) :

Continue

ENDIF

ENDIF

ELSE:

Print (“Please choose a ferry to view”)

Read (ferry\_selection)

IF (user\_input == “V”):

Print ( ferry\_business [ferry\_selection - 1][0 : 5] )

Print ( ferry\_business [ferry\_selection - 1][5 : 10] )

Print ( ferry\_economic [ferry\_selection - 1][0 : 5] )

Print ( ferry\_economic [ferry\_selection - 1][5 : 10] )

Print ( ferry\_economic [ferry\_selection - 1][10 :15] )

Print ( ferry\_economic [ferry\_selection - 1][15 : 20] )

Print ( ferry\_economic [ferry\_selection - 1][20 : 25] )

Print ( ferry\_economic [ferry\_selection - 1][25 : 30] )

Print ( ferry\_economic [ferry\_selection - 1][30 : 35] )

Print ( ferry\_economic [ferry\_selection - 1][35 : 40] )

Continue

ELSE:

IF (user\_input == “Q”):

Print (“Please visit again soon”)

Break

ENDIF

ENDIF

ENDIF

# 4.0 | Sample Output Screen

## 4.0.1 | Main Menu Sample Output

Figure 1: Main Menu interface when executing the program

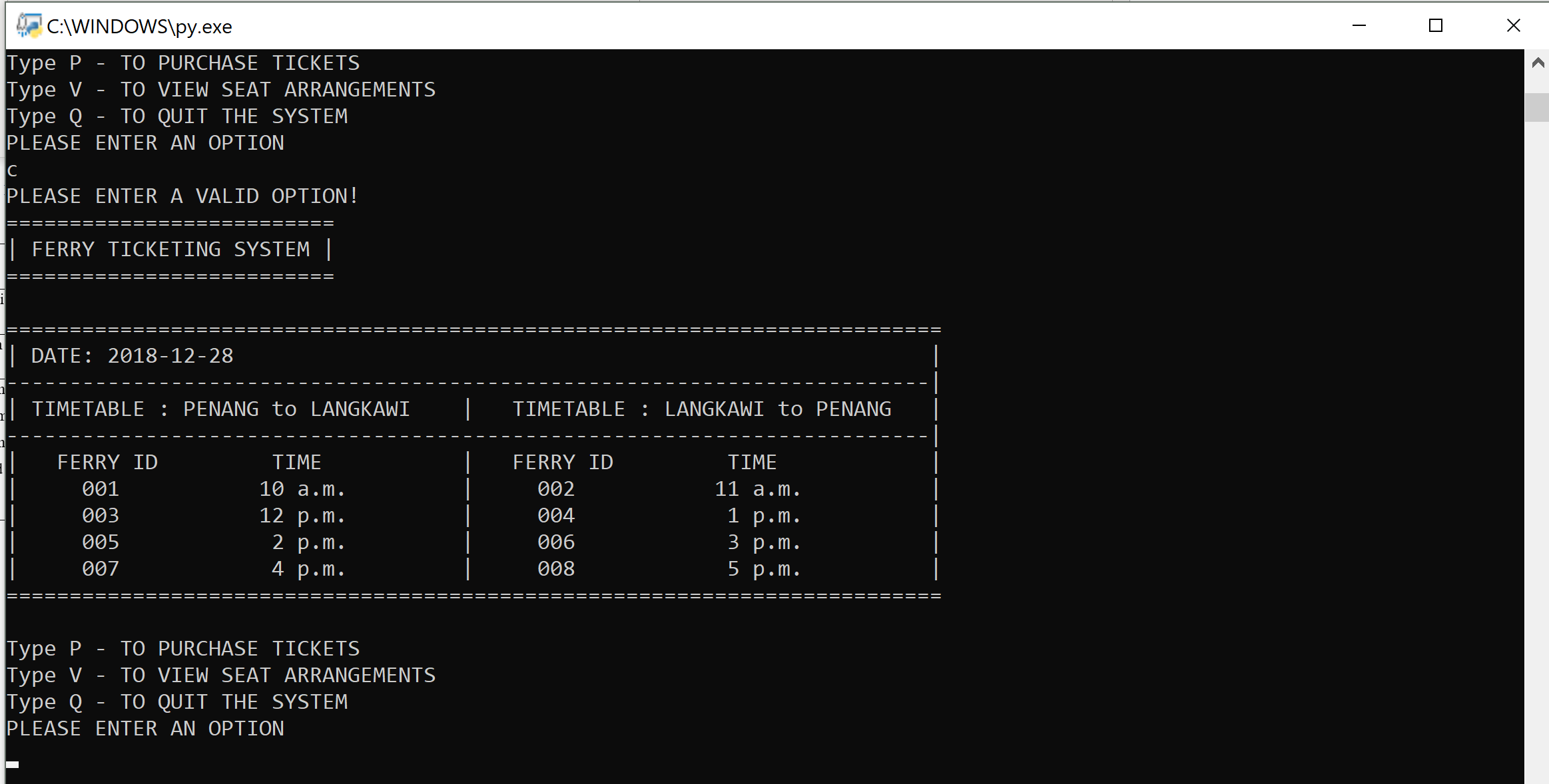
When executing the program, the program will display the main menu, which contains the date, the trip timetable, Ferry IDs along with the departure time of the Ferries. The main function or choices will be stated below as Type P for purchasing tickets, Type V for viewing seat arrangements and Type Q for quitting the system. The system requests for an input for either P,V or Q.

Figure 2: Invalid Input from User

If user enters an invalid input for this instance is c, the system will ask the User to “PLEASE ENTER A VALID OPTION!”, and display the Main Menu and requesting an input from the user again.

## 4.0.2 | Purchasing Module Sample Output

Figure 3: Purchasing Module after entering P

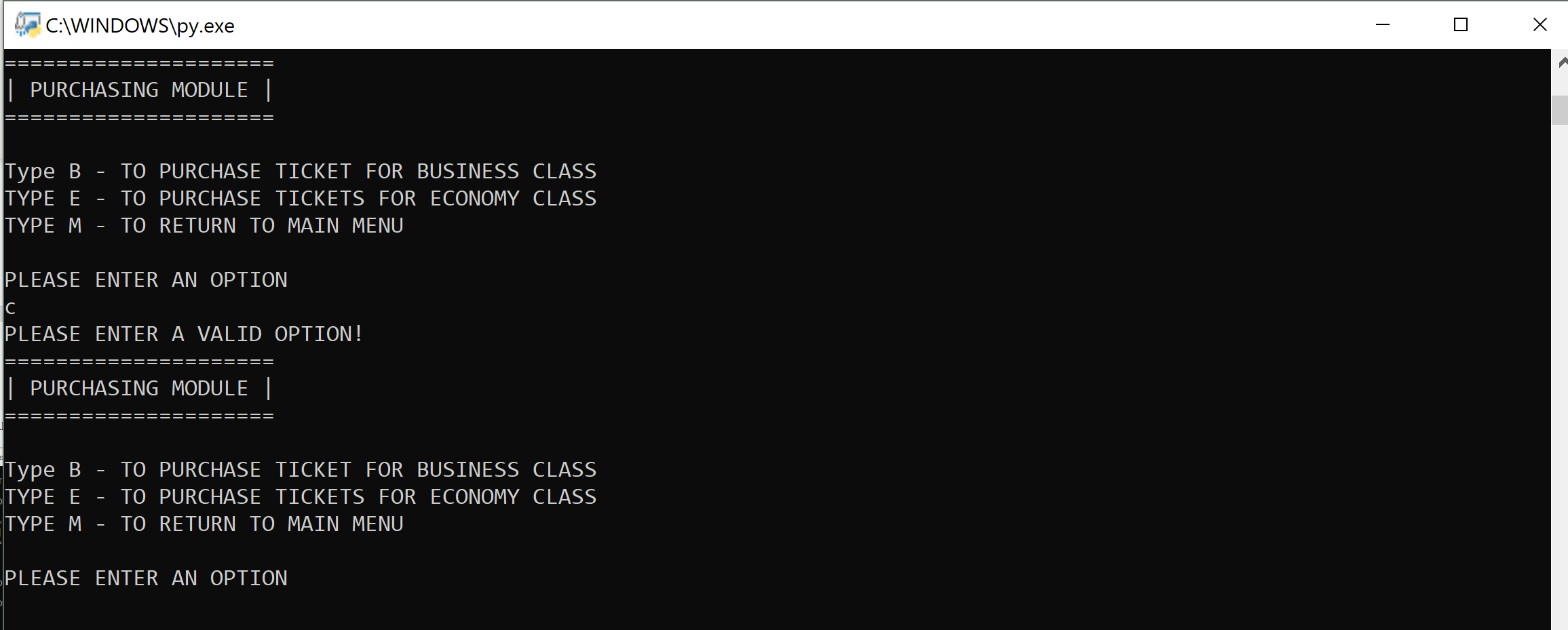
After entering P, It will proceed to the Purchasing module as shown above, It also states out the additional option that allows consumers to pick from either purchasing Business Class seats by Entering B, purchasing Economic Class seats by Entering E or returning back to the Main Menu by Entering M.

Figure 4: Invalid Input from User

If User inserts an input that is not request by the system, the system will again display “PLEASE ENTER A VALID OPTION” and continues to display the purchase module requesting for a valid input.

## **4.0.3 | Choosing a Ferry after Choosing Business Class or Economic Class seats**

Figure 5: Choosing a Ferry after picking Business Class

Figure 6: Choosing a Ferry after picking Economic Class

After Typing B to pick Business Class or Typing E to pick Economic Class , System will request an input from the user to choose a Ferry from 1 to 8.

## 4.0.4 | Selecting Seats after choosing which Ferry to take

Figure 7: Seating for Business Class

Figure 8: Seating for Economic Class

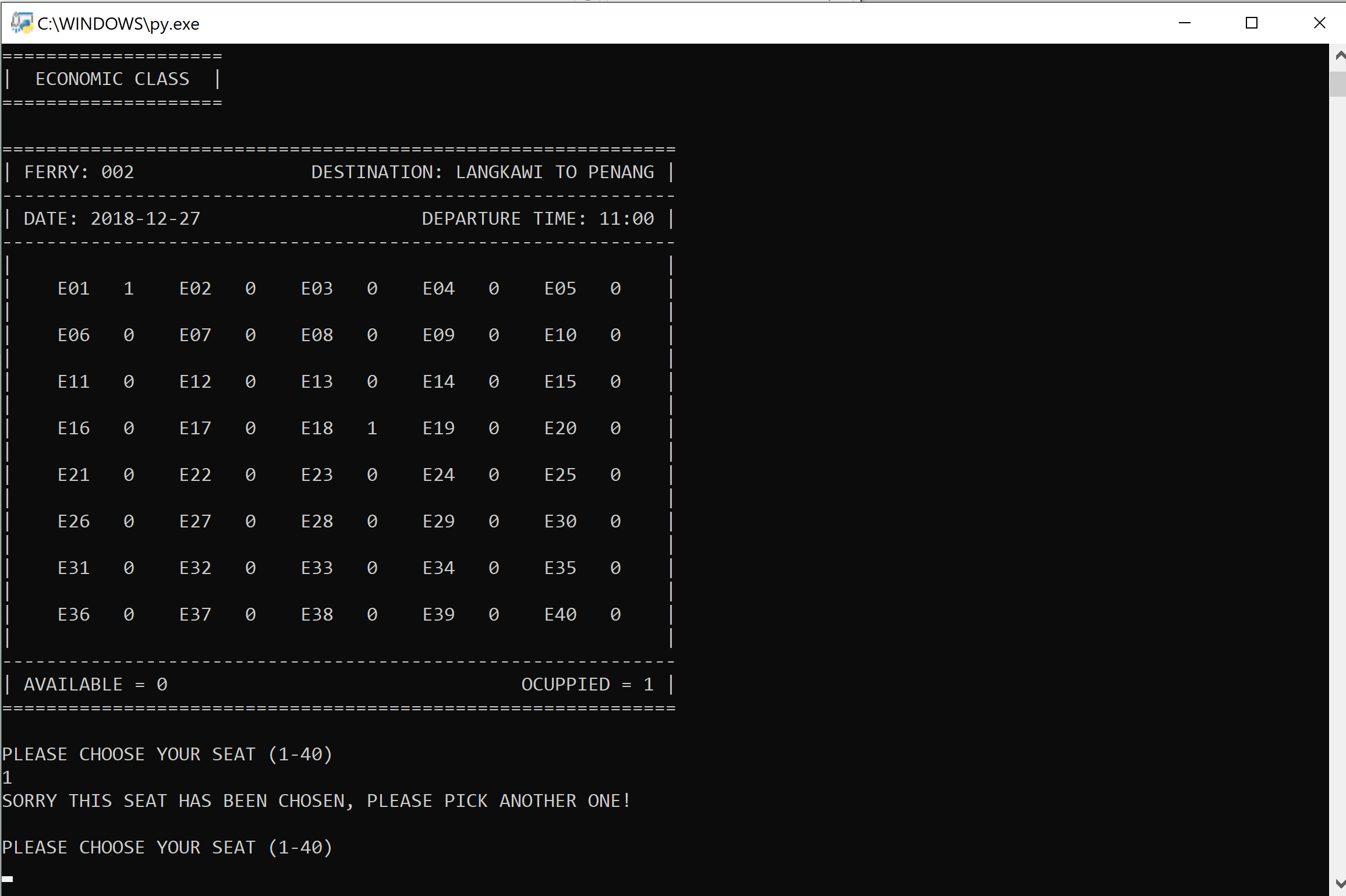
After choosing a Ferry, for this instance is Ferry number 2, the system will display either the Business Class or Economic Class Seating dependent on what the user has chosen on the past selection. The system will display every seat in the ferries that are available or occupied, the Ferry ID, destination, date and Departure time. The 0 indicates that the seat is available, 1 indicates the seats has been occupied and taken by other users. The system now will request the user to select the seating from either 1 to 10 for Business Class or 1 to 40 for Economic Class seats.

Figure 9: Selecting Chosen Seats

If user has chosen a seat that is occupied, the system will then display “SORRY THIS SEAT HAS BEEN CHOSEN, PLEASE PICK ANOTHER ONE”, and request for users selection again.

## 4.0.5 | If Business Class has been fully booked

Figure 10: Business Class seats fully booked

If one of the Ferry’s Business Class seats are fully booked, user are given a choice to choose another Ferry.

## 4.0.6 | If Economic Class has been fully booked

Figure 11: Economic Class seats fully booked

If one of the Ferry’s Economic Class seats are fully booked, user are given a choice to choose another Ferry.

## 4.0.7 | Successfully Chosen A Seat

Figure 12: After successfully chosen a seat

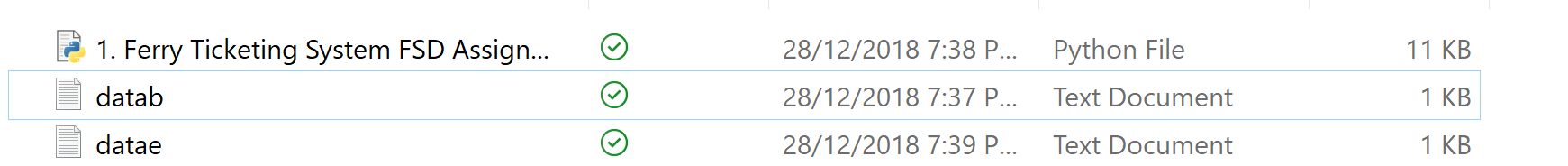
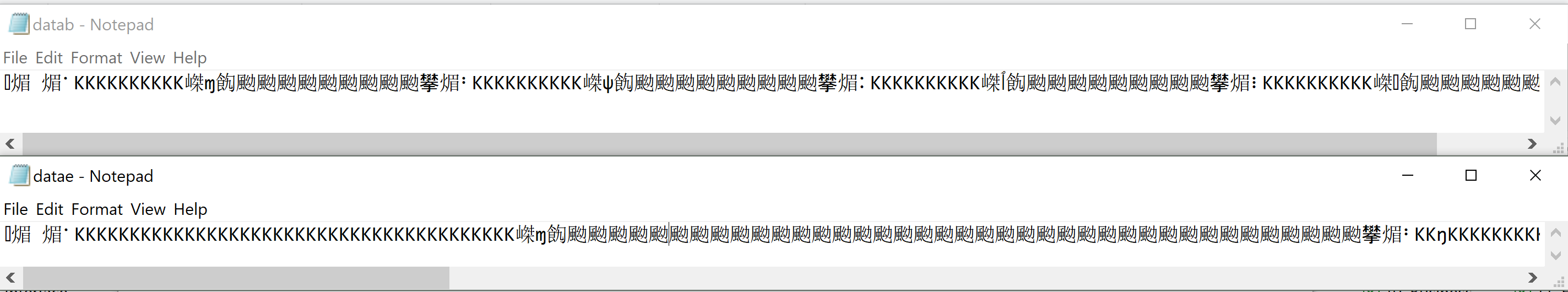
After user has chosen an available seat, for this instance it is seat number 3 for Economic Class. the system will then request user to enter their names.

Figure 13: Information Inside the Text Files

Figure 14: Saving Seating Information in Text File

The ferry and seating information will then be saved inside files, so after the program is closed the seating details will still remain. As shown above it will be saved to datab and datae for both business class seats and economic class seats. The information will be saved in binary so the content in the files are not readable for people but only readable for the program.

## 4.0.8 | Printing Ticket for users

Figure 15: Printing Ticket for users

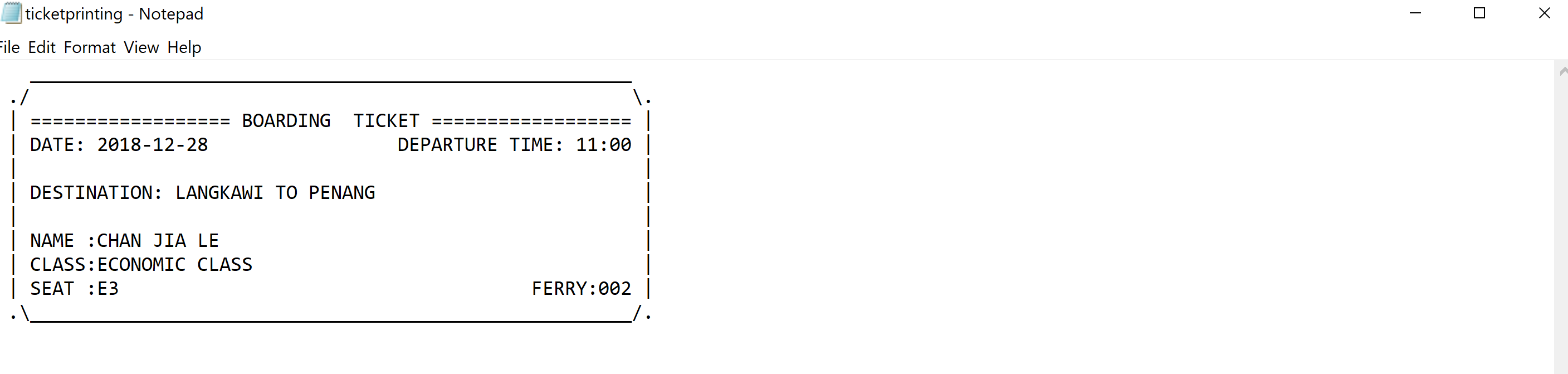
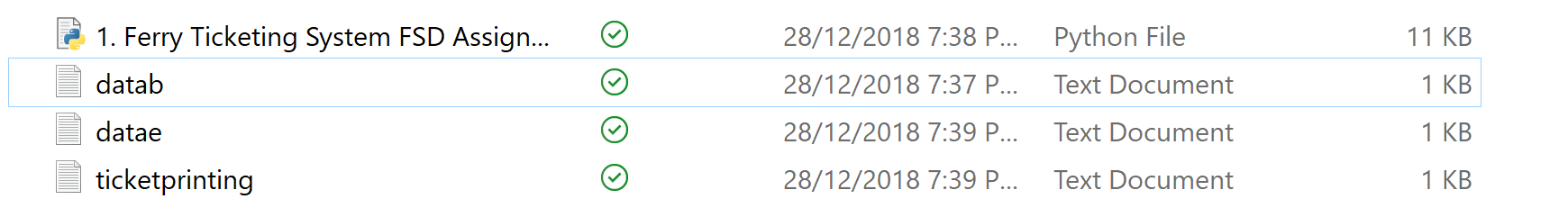
After user entered a name, for this instance it is Chan Jia Le. The system will print a ticket with all the details including the date, departure time, destination, class, seat and the Ferry ID to ensure that the ticket will contain every detail that the user needs in order to have a great experience. The system will then ask if the users wanted to purchase another ticket.

Figure 17: Ticket in Text File

Figure 16: Saving Ticket to Text File

The ticket will then be saved inside a text file in this case is “ticketprinting”, so after the program has finish its purpose the ticket for still remain in the text file for the convenience of User printing it.

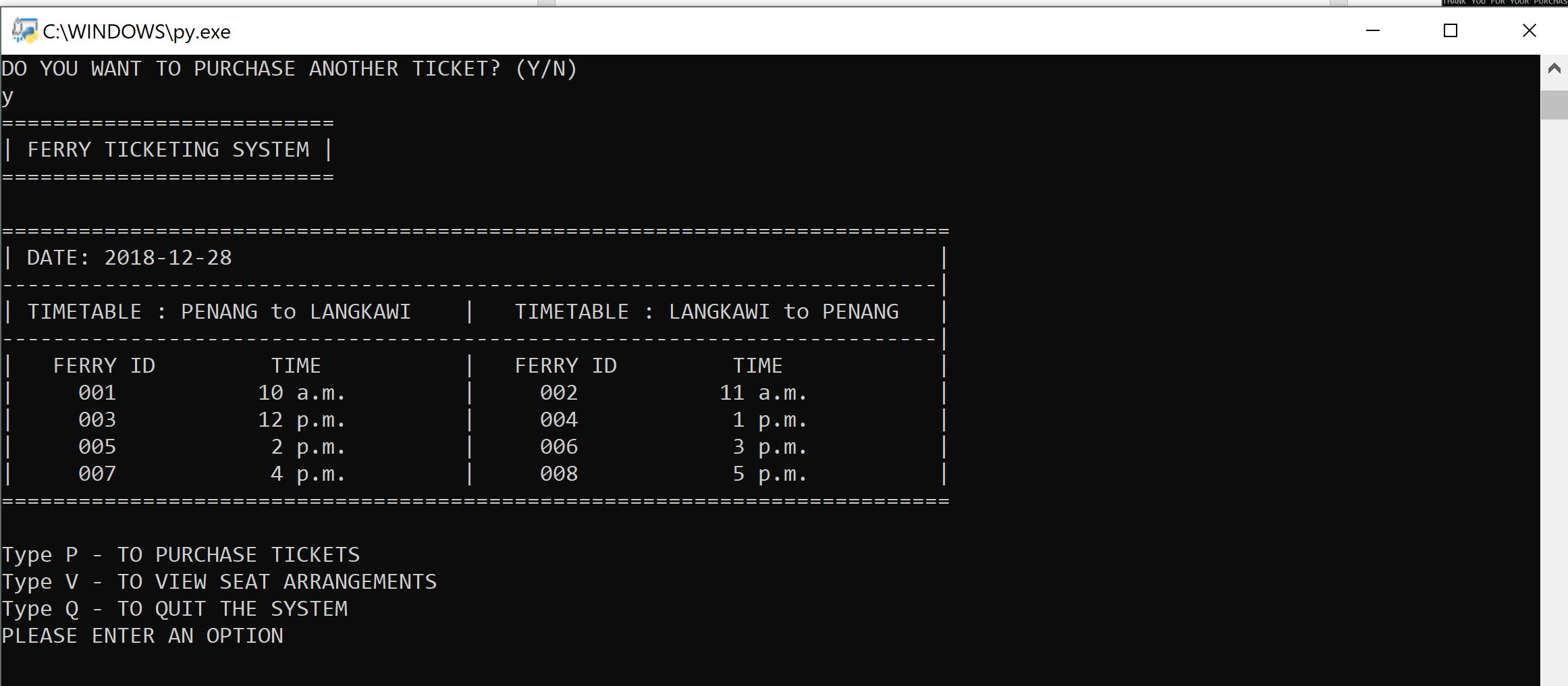
.

Figure 18: If Users agree on purchasing another ticket

If the User enters Y which is Yes, the system will again return back to the Main Menu for user to again purchase another ticket.

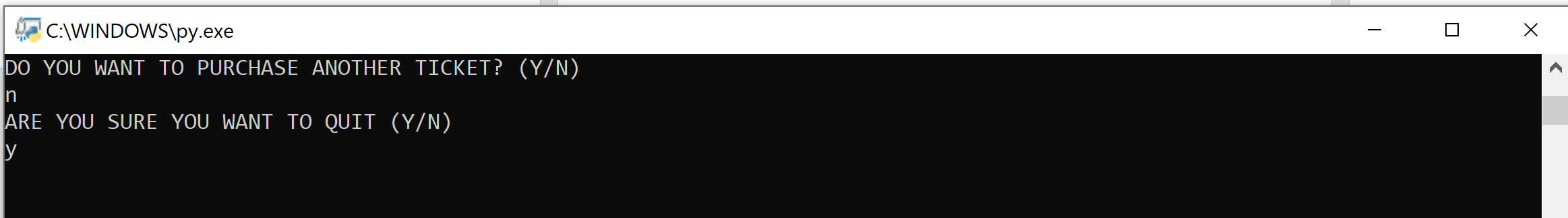
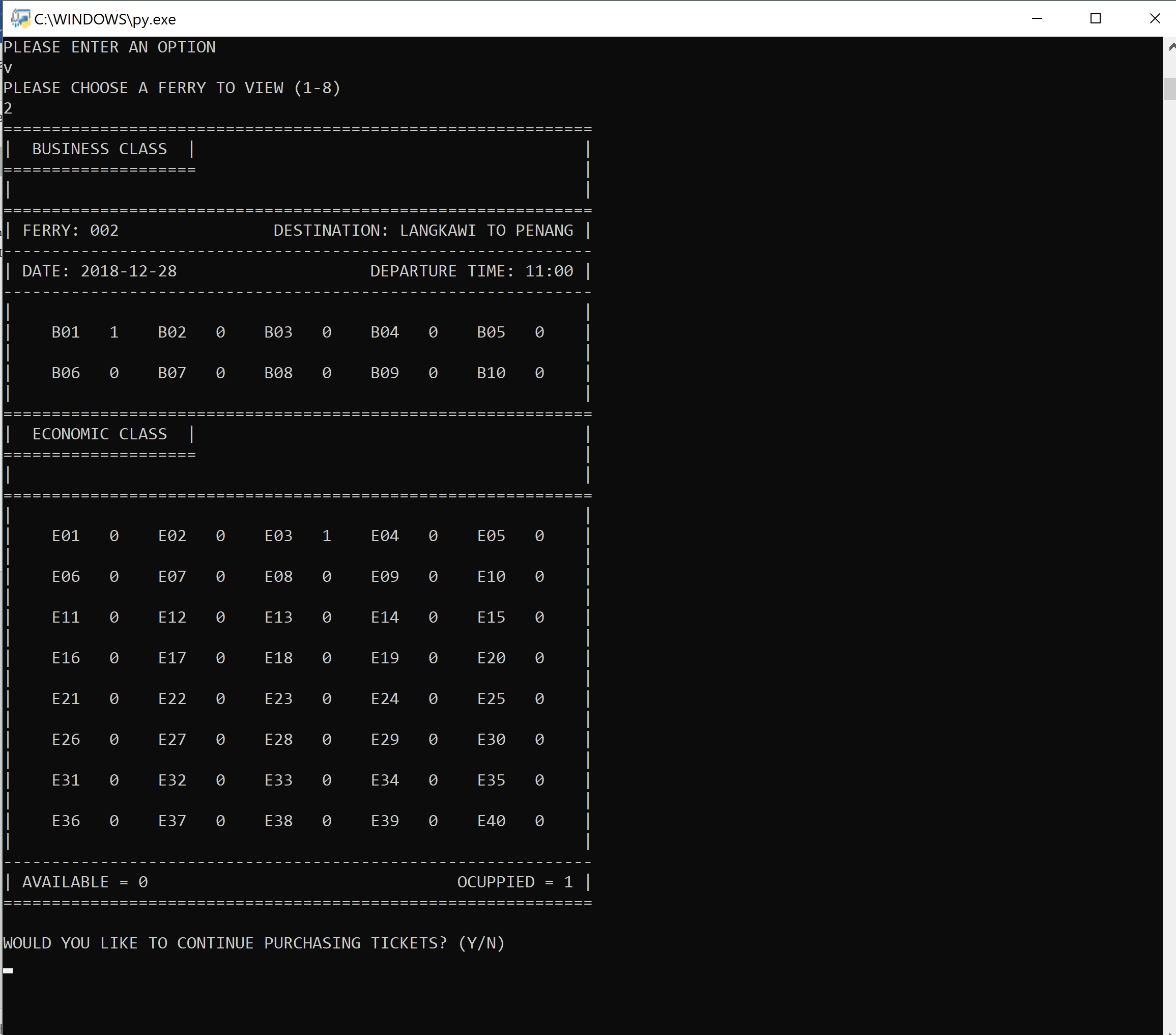
If the User enters N which is No, the system will again ask the User to reinsure that they wanted to quit, If user enters Y, the program will then shut down. If it is a N that is inserted by the User the system will return back to the Main Menu.

Figure 19: If Users don't want to purchase another ticket

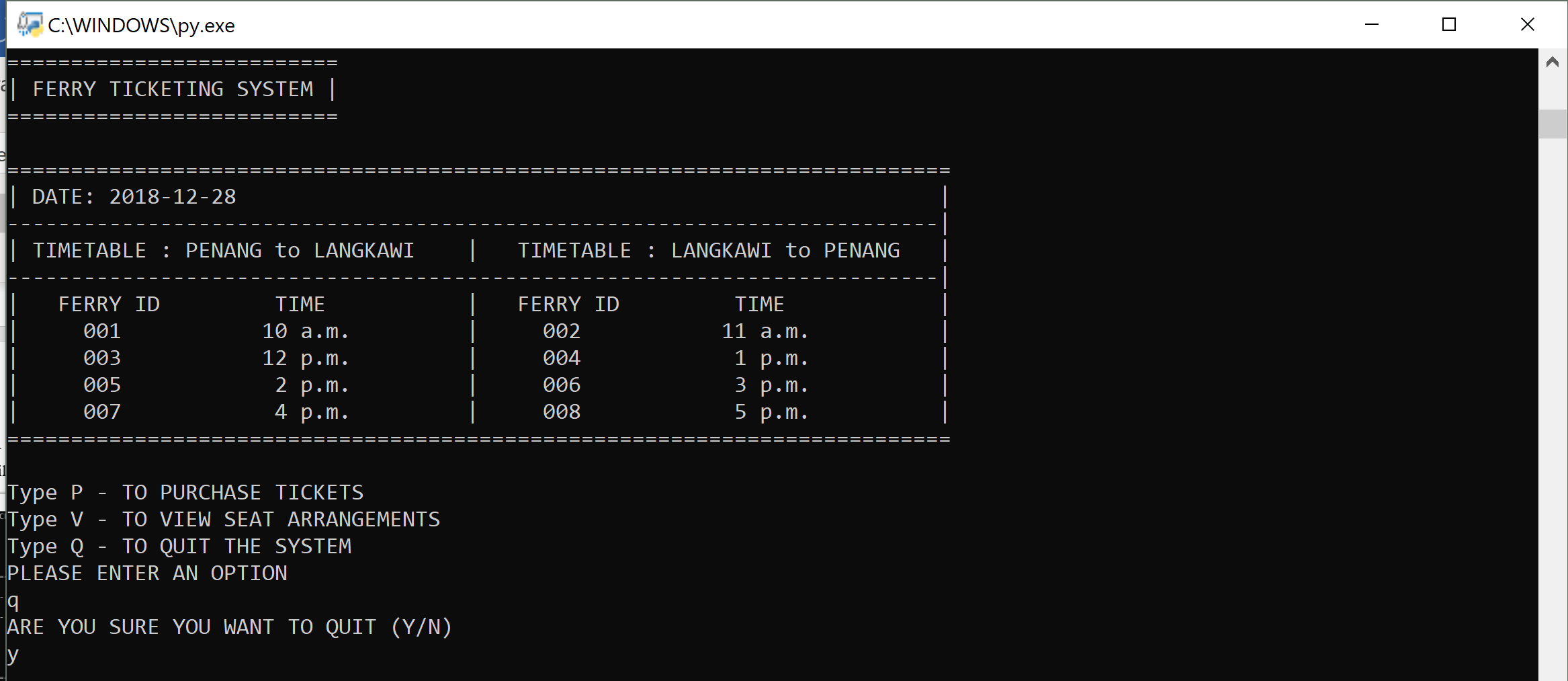
## 4.0.9 | Viewing Seats Sample Output

Figure 20: Viewing Seats Interface

After Inserting V on Main Menu to view the seats, the system will ask the user to choose a Ferry of choice from 1 to 8, for this instance it is a two, the system will then display all the seating to the user including the business class and economic class seats. The seats with a 0 is available and the seats with a 1 is occupied. The system will then ask if the User wants to purchase the ticket.

## 4.0.10 | Quit Module Sample Output

Figure 21: Quit Module

After typing in Q, the system will then ask if the user is sure to exit the system, If the user types Y (Yes), the system will then shut down. If user’s input is a N (No), the system will then display the Main Menu again.

# 5.0 | Additional features of the system

## 5.0.1 | Timetable Displayed

This is the additional design of the system that displays all the information for the user straight away for the user, so user has a better idea of which ferry to choose and the trip time without confusing the consumers.

## 5.0.2 | Saving Seating Information using Pickle Module

This additional feature allows the system to keep record of every seating after the tickets has been purchased even if the program has finished executing and closed down. After successfully selecting a seat, we used the pickle module to write the data into a text file in binary form, the system will then read from the data that is created and continue to display the information.

## 5.0.3 | Saving Ticket Information

This additional feature allows the user to save a copy of the ticket after the program has finish its job and closed down. The system will write a text file in order to allow users to revisit the tickets after the system is closed.

## 5.0.4 | Choosing Other Ferries After Economic Class Seats Are Fully Booked

The system has an added function to allow users to repick the ferry after the ferry’s economic seats are fully booked.

## 5.0.5 | Purchasing Another Ticket After The First Purchase

After successfully purchasing a ticket, User are given a choice to purchase another ticket for added flexibility.

# 6.0 | Assumptions

The system is capable of letting users purchasing Tickets over 8 Ferries, they can choose either Business Class or Economic Class seats for only that day, each ferry contains a total of 50 seats, 10 being Business Class seats and the remaining ones being Economic Class seats . The trip time and location will be as stated on the timetable in the Main Menu , if the ferries are all fully booked, users can only wait until the another day to purchase the tickets again. Users are not allowed to select seats that are occupied.

# 7.0 | References

Programiz, n.d. *Python Global Keyword.* [Online]   
Available at: https://www.programiz.com/python-programming/global-keyword  
[Accessed 25 December 2018].

Programiz, n.d. *Python String ljust().* [Online]   
Available at: https://www.programiz.com/python-programming/methods/string/ljust  
[Accessed 25 December 2018].

Python, n.d. *pickle — Python object serialization¶.* [Online]   
Available at: https://docs.python.org/3/library/pickle.html  
[Accessed 25 December 2018].

Python, n.d. *time — Time access and conversions.* [Online]   
Available at: https://docs.python.org/3/library/time.html  
[Accessed 25 December 2018].

thefoutheye, 2014. *Add leading Zero Python [duplicate].* [Online]   
Available at: https://stackoverflow.com/questions/21620602/add-leading-zero-python/21620624#21620624  
[Accessed 10 December 2018].