TICKET RESERVATION SYSTEM USING FCFS AND PRIORITY SCHEDULING

A PROJECT REPORT

Submitted by

BL.EN.U4AIE19041	M. TANUJ
BL.EN.U4AIE19049	P. VENKATA AKHIL
BL.EN.U4AIE19068	V. AISHWARYA

for the course

19AIE202-Operating Systems

Guided and Evaluated by

Ms. Vineetha Jain K. V.

Assistant Professor (Sr. Gr.),

Computer Science and Engineering

AMRITA SCHOOL OF ENGINEERING, BANGALORE

LIST OF CONTENTS

1. **ACKNOWLEDGEMENT ………………………………………………….. 3**
2. **ABSTRACT……………………………………………………………….. 4**
3. **INTRODUCTION…………………………………………………………... 5**
4. **DESIGN………………………………………………………………….. 8**
5. **IMPLEMENTATION…………………………………………………….. 9**
6. **CODE…………………………………………………………………… 10**
7. **RESULTS……………………………………………………………….. 14**
8. **CONCLUSION …………………………………………………………18**

1. ACKNOWLEDGEMENT

The satisfaction that accompanies successful completion of any task would be incomplete without mention of people whom made it possible, and whose constant encouragement and guidance have been source of inspiration throughout the course of this project work.

We offer our sincere pranams at the lotus feet of **"AMMA", MATA AMRITANANDAMAYI DEVI** who showered her blessing upon us throughout the course of this project work.

We owe our gratitude to **Br. Viswamrita Chaitanya Swamiji**, Director, Amrita School of Engineering, Bangalore.

We thank **Dr. Sriram Devanathan**, Principal, Amrita School of Engineering, Bangalore for his support and inspiration.

It is a great pleasure to express our gratitude and indebtedness to our project guide **Mrs. Vineetha Jain K. V.** Assistant Professor, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for her valuable guidance, encouragement, moral support and affection throughout the project work.

We express our Heartfelt thanks to **Dr. Sriram Devanathan,** Chairperson, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for his encouragement and support during the course of work.

Finally, we are forever grateful to our parents, who have loved, supported and encouraged us in all our endeavors.

2. ABSTRACT

In this project we have implemented a ticket reservation system where a passenger can book a ticket for airlines or train. We have used scheduling algorithms like First Come First Serve (FCFS) and Priority scheduling for our implementation. The main aim of this project is to book tickets for the passengers based on their arrival time and age. The user can also opt the method of booking, either FCFS or priority, to book their tickets in reservation system according to their convenience. And when they choose the algorithm, they'll be asked if they want to book for an airline or train. In priority scheduling, the passenger's age is considered as a parameter for priority. Average waiting time and Average wait around time of passengers are displayed after all the passenger's booking.

3. INTRODUCTION

In a single-processor system, only one process could run at a time. Others will have to wait until the CPU is free and can be rescheduled. The objective of multiprogramming is always to think of some process running, to capitalize on CPU utilization. In a simple computer system, the CPU then just sits idle when the running process goes to waiting state during I/O. All this waiting time is wasted; no useful work is accomplished. With multiprogramming, we try to use this time productively. Several processes are kept in memory at one time. When one process must wait, the operating system takes the CPU away from that process and gives the CPU to another process. This pattern continues. Every time one process must wait, another process can take overuse of the CPU.

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy. Process scheduling is an essential part of a Multiprogramming operating systems.

Non Preemptive scheduling:

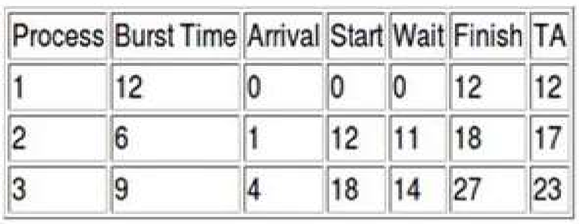
Non preemptive Scheduling is a CPU scheduling technique the process takes the resource (CPU time)
and holds it till the process gets terminated or is pushed to the waiting state.

Preemptive scheduling:

Preemptive Scheduling is a CPU scheduling technique that works by dividing time slots of CPU to a given process.

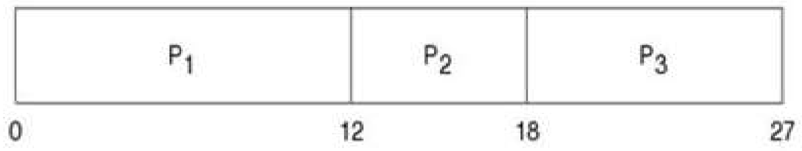
Scheduling algorithms:

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically completes queued calls and processes in order of their arrival. It is the at ease and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU at the outset get the CPU allocation first. In this scheduling algorithm, an individual is served according to the queue manner. The person who arrives first place in the queue first buys the ticket and then the next one. This will continue as long as the last person in the queue purchases the ticket. Utilizing this algorithm, the CPU process works in a similar way. Here is a problem solving example of FCFS scheduling in the fig[3.1]:



Process	Burst Time	Arrival	Start	Wait	Finish	TA
1	12	0	0	0	12	12
2	6	1	12	11	18	17
3	9	4	18	14	27	23

Gantt chart:



average waiting time: (0+11+14)/3 = 8.33

average turnaround time: (12+17+23) = 52/3 = 17.33

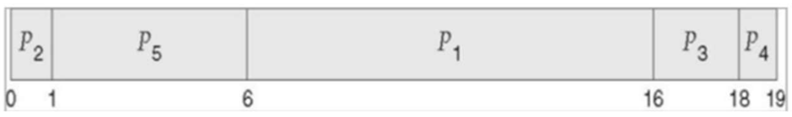
Figure 3.1

Priority Scheduling is a method of scheduling processes which is based on priority. In this algorithm, the scheduler chooses the tasks to work in accordance with the priority. The processes with greater priority should be carried out first. Priority will vary depending on the memory requirements, time requirements, etc. In this scheduling algorithm, a person is not served in compliance with the queue manner.

If two person who arrives first place in the queue buys the ticket as per priority and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works in a similar manner. Here is a problem solving example of Priority scheduling in the fig[3.2]:

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart



- **Average Waiting Time =8.2msec**
- Average turnaround time = (16+1+18+19+6)/5 =60/5=12

Figure 3.2

4. DESIGN

Below flow chart (fig[4.1]) shows the working of our ‘Tyrant Reservation System’ for booking tickets in Airlines and Railways using both FCFS Scheduling and Priority Scheduling algorithms.

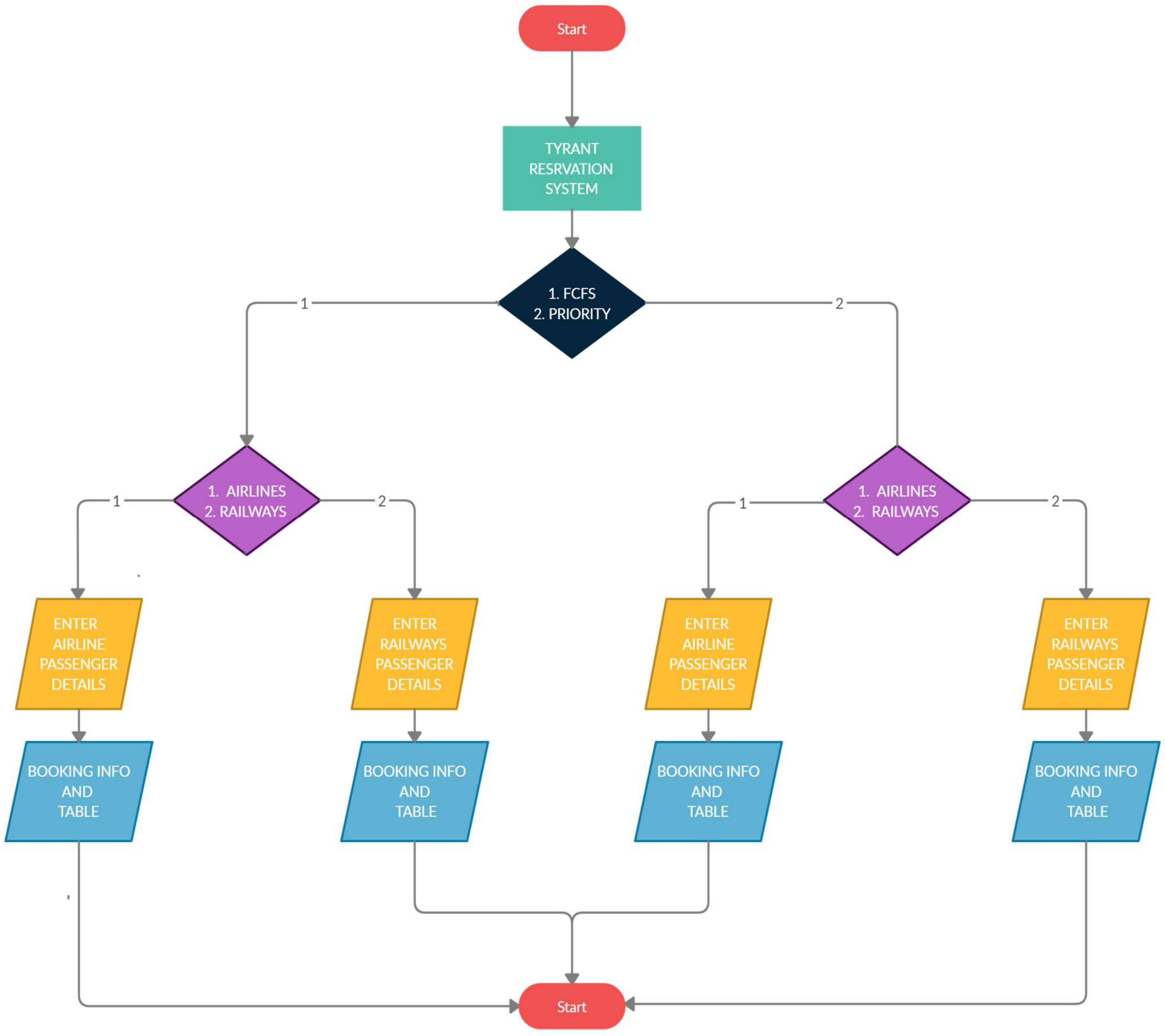


Figure 4.1

5. IMPLEMENTATION

1. We have implemented our code in java.
2. Our code contains the following methods:

   - book_fcfs( ) - This method consists of the code where the passengers details are asked and allots a seat to the passenger. This method books ticket for FCFS scheduling.

   - book_priority( ) - This method consists of the code where the passengers details are asked and allots a seat to the passenger. This method books ticket for priority scheduling.

   - status_display( ) - This method consists of the code where the status of seats are displayed.

   - findWaitingTime( ) - Finds the waiting time of the passenger who is going to book the ticket.

   - findTurnAroundTime( ) - Finds the turn around time of the passenger who is going to book the ticket.

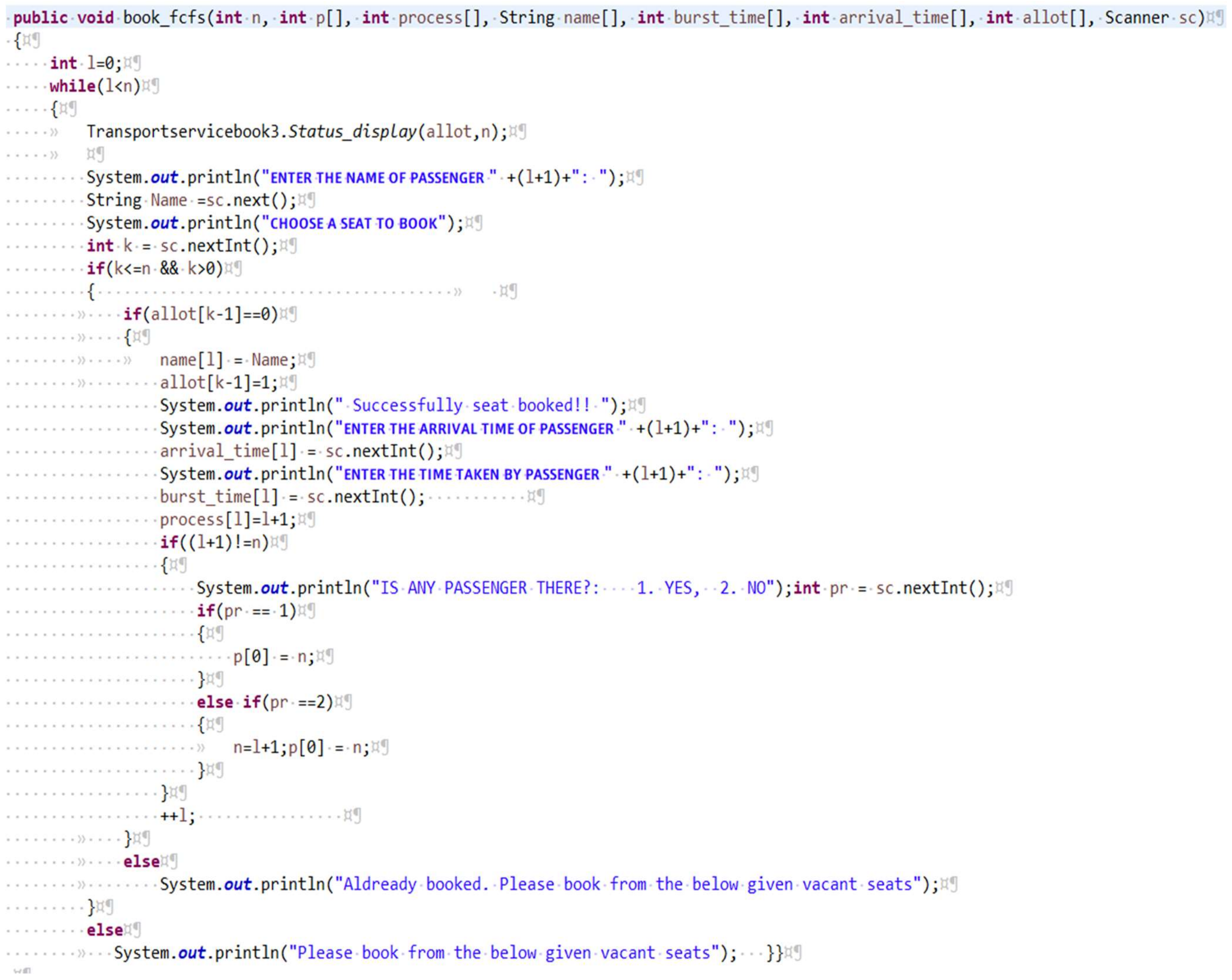
   - findavgTime_fcfs( ) – This method prints the process scheduling table and the average waiting time and the average turn around time in fcfs scheduling algorithm.

   - findavgTime_priority( ) - This method prints the process scheduling table and the average waiting time and the average turn around time in priority scheduling algorithm.

3. Main class consists of the code where the passenger can choose the transportation method either airlines or railways, fill their details and confirm their booking and the analysis of passengers data will be shown.

6. CODE

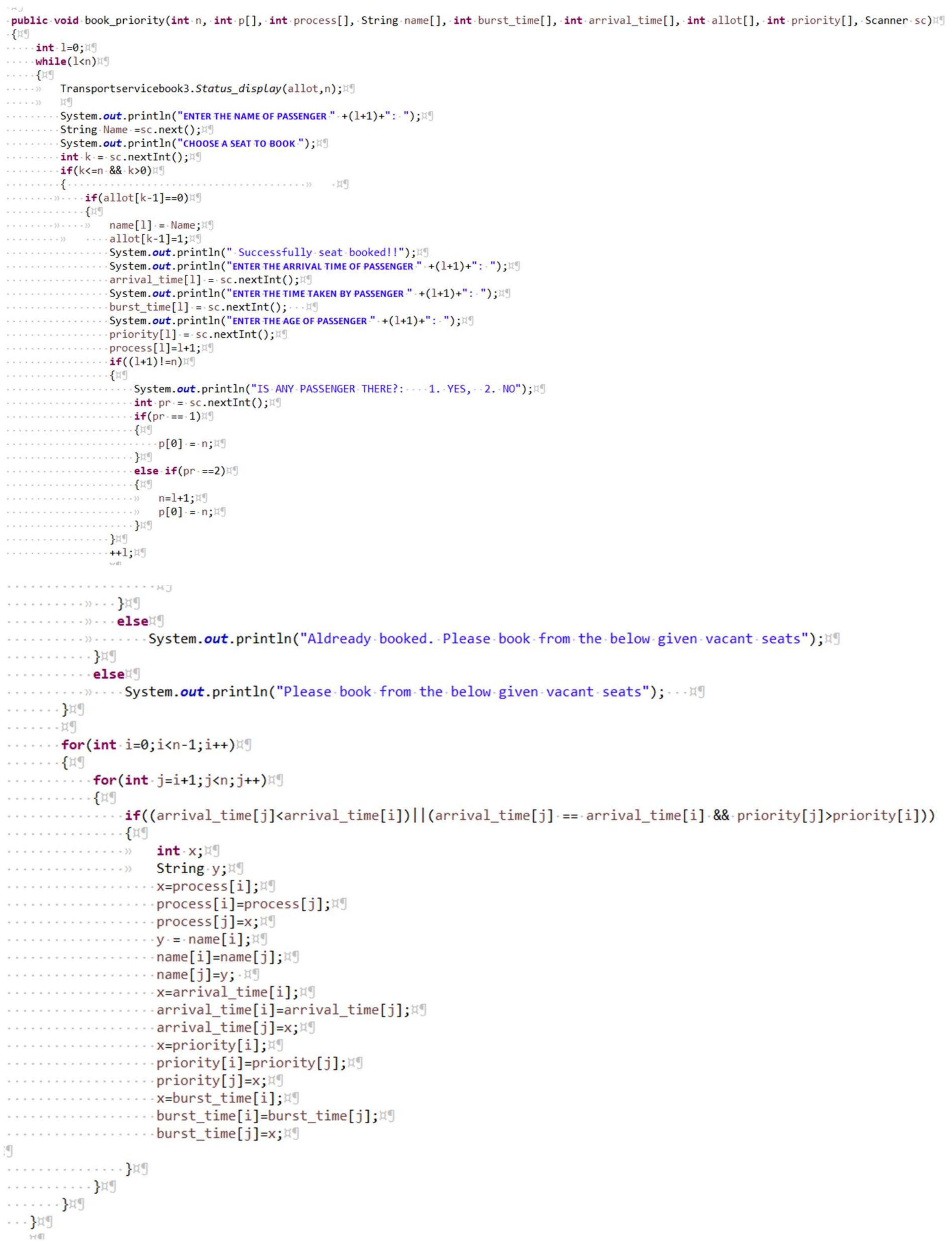
- **book_fcfs( ):**



```
public void book_fcfs(int n, int p[], int process[], String name[], int burst_time[], int arrival_time[], int allot[], Scanner sc)
{
    int l=0;
    while(l<n)
    {
        Transportservicebook3.Status_display(allot,n);
        
        System.out.println("ENTER THE NAME OF PASSENGER " +(l+1)+": ");
        String Name =sc.next();
        System.out.println("CHOOSE A SEAT TO BOOK");
        int k = sc.nextInt();
        if(k<=n && k>0)
        {
            if(allot[k-1]==0)
            {
                name[l] = Name;
                allot[k-1]=1;
                System.out.println(" Successfully seat booked!! ");
                System.out.println("ENTER THE ARRIVAL TIME OF PASSENGER " +(l+1)+": ");
                arrival_time[l] = sc.nextInt();
                System.out.println("ENTER THE TIME TAKEN BY PASSENGER " +(l+1)+": ");
                burst_time[l] = sc.nextInt();
                process[l]=l+1;
                if((l+1)!=n)
                {
                    System.out.println("IS ANY PASSENGER THERE?:    1. YES,  2. NO");int pr = sc.nextInt();
                    if(pr == 1)
                    {
                        p[0] = n;
                    }
                    else if(pr ==2)
                    {
                        n=l+1;p[0] = n;
                    }
                }
                ++l;
            }
            else
                System.out.println("Aldready booked. Please book from the below given vacant seats");
        }
        else
           System.out.println("Please book from the below given vacant seats");   }}
```

Figure 6.1

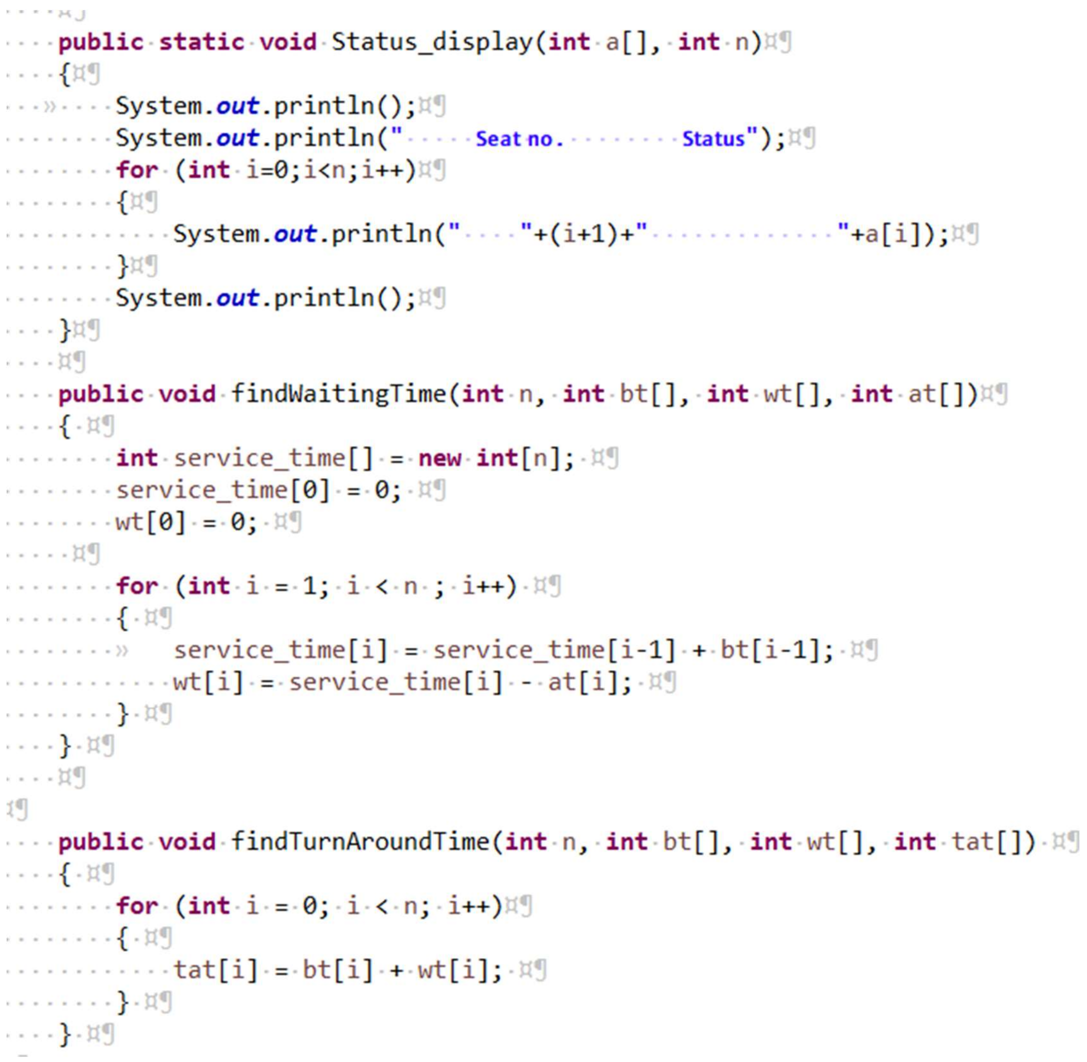
- **book_priority( ):**



```
public void book_priority(int n, int p[], int process[], String name[], int burst_time[], int arrival_time[], int allot[], int priority[], Scanner sc)
{
    int l=0;
    while(l<n)
    {
        Transportservicebook3.Status_display(allot,n);

        System.out.println("ENTER THE NAME OF PASSENGER " +(l+1)+": ");
        String Name =sc.next();
        System.out.println("CHOOSE A SEAT TO BOOK ");
        int k = sc.nextInt();
        if(k<=n && k>0)
        {
            if(allot[k-1]==0)
            {
                name[l] = Name;
                allot[k-1]=1;
                System.out.println(" Successfully seat booked!!");
                System.out.println("ENTER THE ARRIVAL TIME OF PASSENGER " +(l+1)+": ");
                arrival_time[l] = sc.nextInt();
                System.out.println("ENTER THE TIME TAKEN BY PASSENGER " +(l+1)+": ");
                burst_time[l] = sc.nextInt();
                System.out.println("ENTER THE AGE OF PASSENGER " +(l+1)+": ");
                priority[l] = sc.nextInt();
                process[l]=l+1;
                if((l+1)!=n)
                {
                    System.out.println("IS ANY PASSENGER THERE?:    1. YES,  2. NO");
                    int pr = sc.nextInt();
                    if(pr == 1)
                    {
                        p[0] = n;
                    }
                    else if(pr ==2)
                    {
                        n=l+1;
                        p[0] = n;
                    }
                }
                ++l;

            }
            else
                System.out.println("Aldready booked. Please book from the below given vacant seats");
        }
        else
            System.out.println("Please book from the below given vacant seats");
    }

    for(int i=0;i<n-1;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            if((arrival_time[j]<arrival_time[i])||(arrival_time[j] == arrival_time[i] && priority[j]>priority[i]))
            {
                int x;
                String y;
                x=process[i];
                process[i]=process[j];
                process[j]=x;
                y = name[i];
                name[i]=name[j];
                name[j]=y;
                x=arrival_time[i];
                arrival_time[i]=arrival_time[j];
                arrival_time[j]=x;
                x=priority[i];
                priority[i]=priority[j];
                priority[j]=x;
                x=burst_time[i];
                burst_time[i]=burst_time[j];
                burst_time[j]=x;

            }
        }
    }
}
```

Figure 6.2

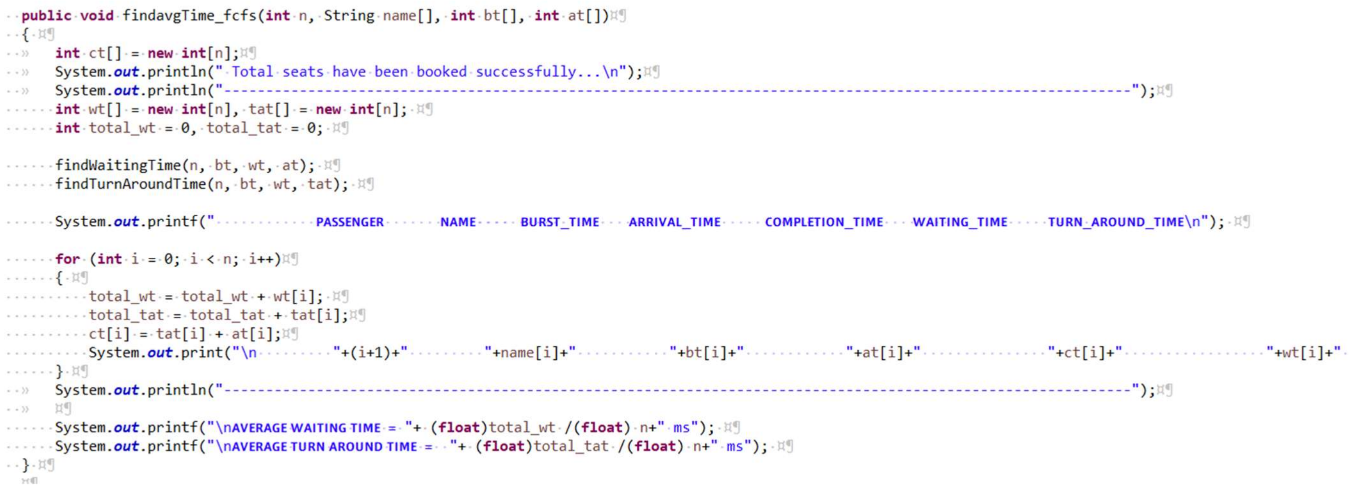
- **Status_display( ), findWaitingTime( ), findTurnAroundTime( )**



```
    public static void Status_display(int a[], int n)
    {
        System.out.println();
        System.out.println("    Seat no.        Status");
        for (int i=0;i<n;i++)
        {
            System.out.println("    "+(i+1)+"             "+a[i]);
        }
        System.out.println();
    }

    public void findWaitingTime(int n, int bt[], int wt[], int at[])
    {
        int service_time[] = new int[n];
        service_time[0] = 0;
        wt[0] = 0;

        for (int i = 1; i < n ; i++)
        {
            service_time[i] = service_time[i-1] + bt[i-1];
            wt[i] = service_time[i] - at[i];
        }
    }


    public void findTurnAroundTime(int n, int bt[], int wt[], int tat[])
    {
        for (int i = 0; i < n; i++)
        {
            tat[i] = bt[i] + wt[i];
        }
    }
```

Figure 6.3

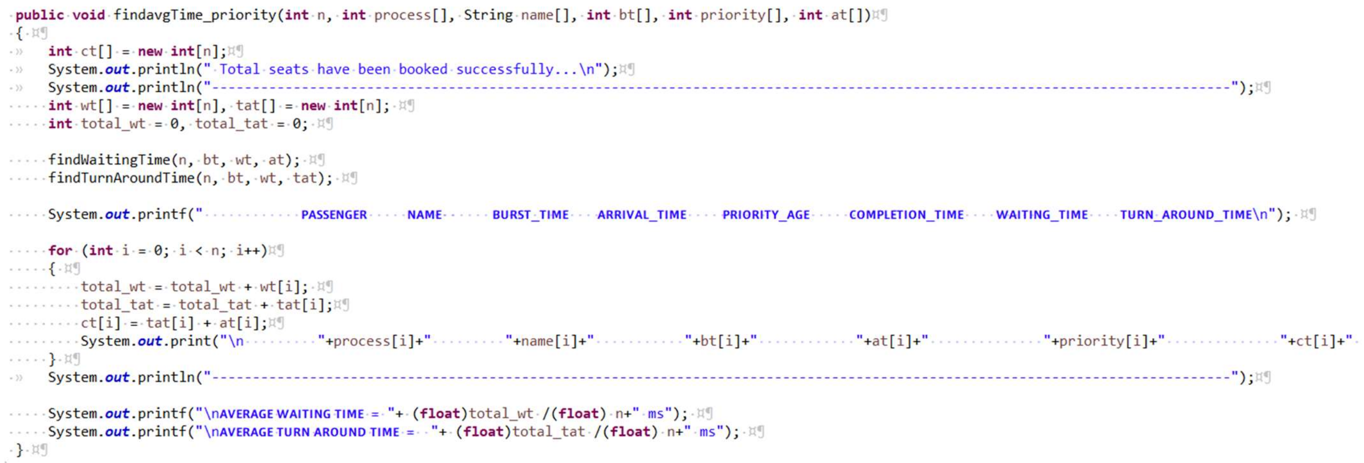
- **findavgTime_fcfs( )**



```
  public void findavgTime_fcfs(int n, String name[], int bt[], int at[])
  {
      int ct[] = new int[n];
      System.out.println(" Total seats have been booked successfully...\n");
      System.out.println("--------------------------------------------------------------------------------------------------------");
      int wt[] = new int[n], tat[] = new int[n];
      int total_wt = 0, total_tat = 0;

      findWaitingTime(n, bt, wt, at);
      findTurnAroundTime(n, bt, wt, tat);

      System.out.printf("            PASSENGER      NAME     BURST_TIME   ARRIVAL_TIME     COMPLETION_TIME   WAITING_TIME     TURN_AROUND_TIME\n");

      for (int i = 0; i < n; i++)
      {
          total_wt = total_wt + wt[i];
          total_tat = total_tat + tat[i];
          ct[i] = tat[i] + at[i];
          System.out.print("\n         "+(i+1)+"         "+name[i]+"           "+bt[i]+"            "+at[i]+"               "+ct[i]+"                 "+wt[i]+"
      }
      System.out.println("--------------------------------------------------------------------------------------------------------");

      System.out.printf("\nAVERAGE WAITING TIME = "+ (float)total_wt /(float) n+" ms");
      System.out.printf("\nAVERAGE TURN AROUND TIME =  "+ (float)total_tat /(float) n+" ms");
  }
```

Figure 6.4

- **findavgTime_priority( )**

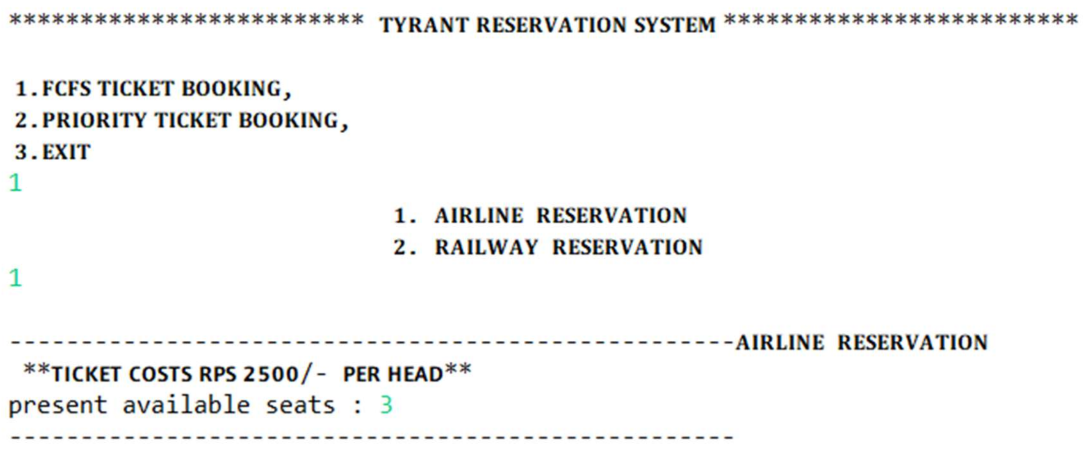


```
 public void findavgTime_priority(int n, int process[], String name[], int bt[], int priority[], int at[])
 {
     int ct[] = new int[n];
     System.out.println(" Total seats have been booked successfully...\n");
     System.out.println("--------------------------------------------------------------------------------------------------------");
     int wt[] = new int[n], tat[] = new int[n];
     int total_wt = 0, total_tat = 0;

     findWaitingTime(n, bt, wt, at);
     findTurnAroundTime(n, bt, wt, tat);

     System.out.printf("            PASSENGER    NAME      BURST_TIME   ARRIVAL_TIME    PRIORITY_AGE    COMPLETION_TIME   WAITING_TIME   TURN_AROUND_TIME\n");

     for (int i = 0; i < n; i++)
     {
         total_wt = total_wt + wt[i];
         total_tat = total_tat + tat[i];
         ct[i] = tat[i] + at[i];
         System.out.print("\n         "+process[i]+"         "+name[i]+"           "+bt[i]+"            "+at[i]+"              "+priority[i]+"              "+ct[i]+"
     }
     System.out.println("--------------------------------------------------------------------------------------------------------");

     System.out.printf("\nAVERAGE WAITING TIME = "+ (float)total_wt /(float) n+" ms");
     System.out.printf("\nAVERAGE TURN AROUND TIME =  "+ (float)total_tat /(float) n+" ms");
 }
```

Figure 6.5

7. RESULTS

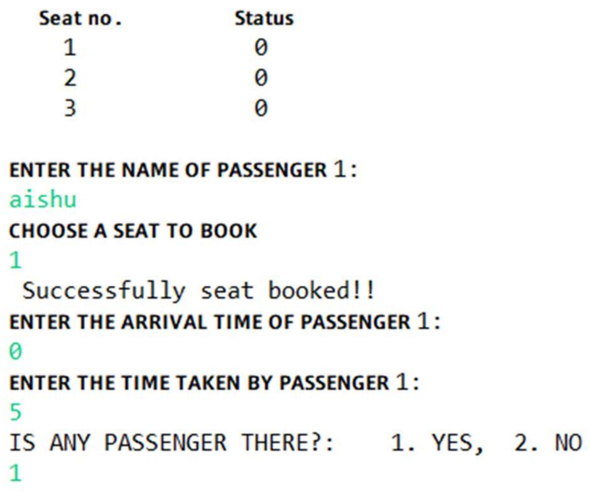
- **USING FCFS:**



```
**************************  TYRANT RESERVATION SYSTEM **************************

 1.FCFS TICKET BOOKING,
 2.PRIORITY TICKET BOOKING,
 3.EXIT
1
                            1.  AIRLINE  RESERVATION
                            2.  RAILWAY  RESERVATION
1

-----------------------------------------------------AIRLINE  RESERVATION
 **TICKET COSTS RPS 2500/-  PER HEAD**
present available seats : 3
-----------------------------------------------------
```

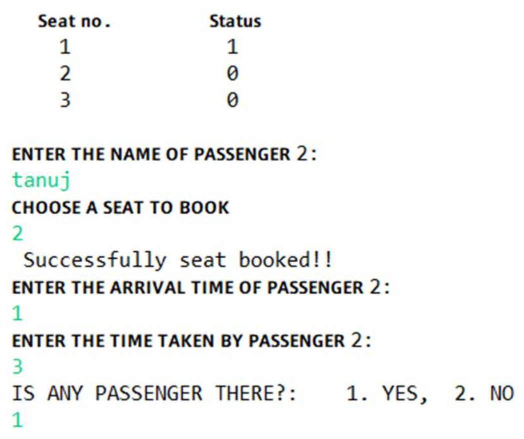
Figure 7.1

Entry of details of passengers are taken in [Fig 7.2, 7.3, 7.4]



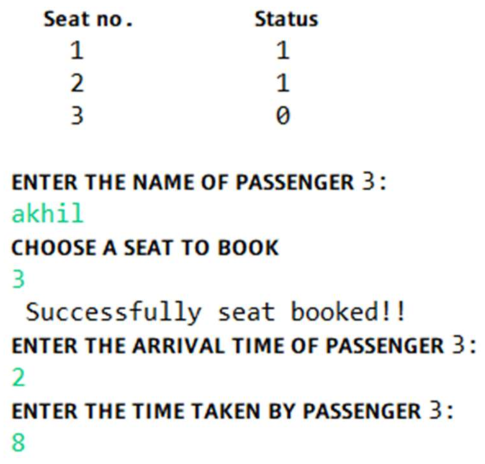
```
  Seat no.        Status
    1               0
    2               0
    3               0

ENTER THE NAME OF PASSENGER 1:
aishu
CHOOSE A SEAT TO BOOK
1
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 1:
0
ENTER THE TIME TAKEN BY PASSENGER 1:
5
IS ANY PASSENGER THERE?:    1. YES,  2. NO
1
```

Figure 7.2



```
 Seat no.          Status
   1                 1
   2                 0
   3                 0

ENTER THE NAME OF PASSENGER 2:
tanuj
CHOOSE A SEAT TO BOOK
2
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 2:
1
ENTER THE TIME TAKEN BY PASSENGER 2:
3
IS ANY PASSENGER THERE?:    1. YES,  2. NO
1
```

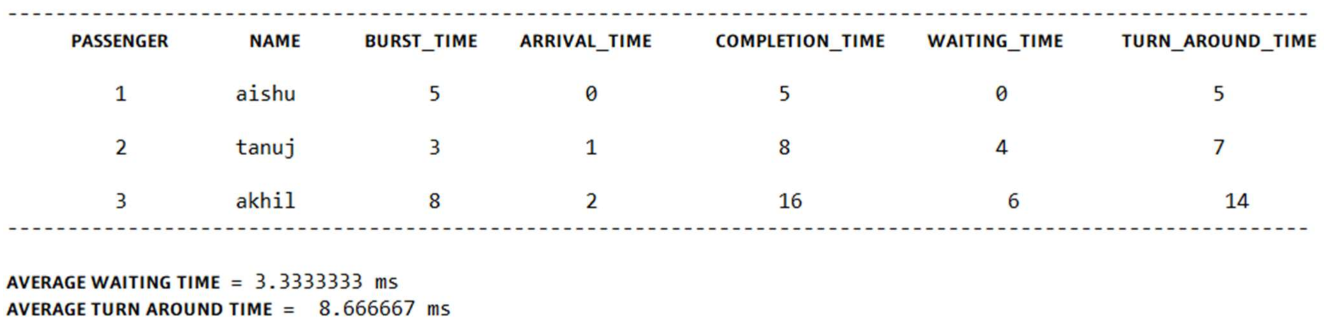
Figure 7.3



```
 Seat no.          Status
   1                 1
   2                 1
   3                 0

ENTER THE NAME OF PASSENGER 3:
akhil
CHOOSE A SEAT TO BOOK
3
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 3:
2
ENTER THE TIME TAKEN BY PASSENGER 3:
8
```

Figure 7.4

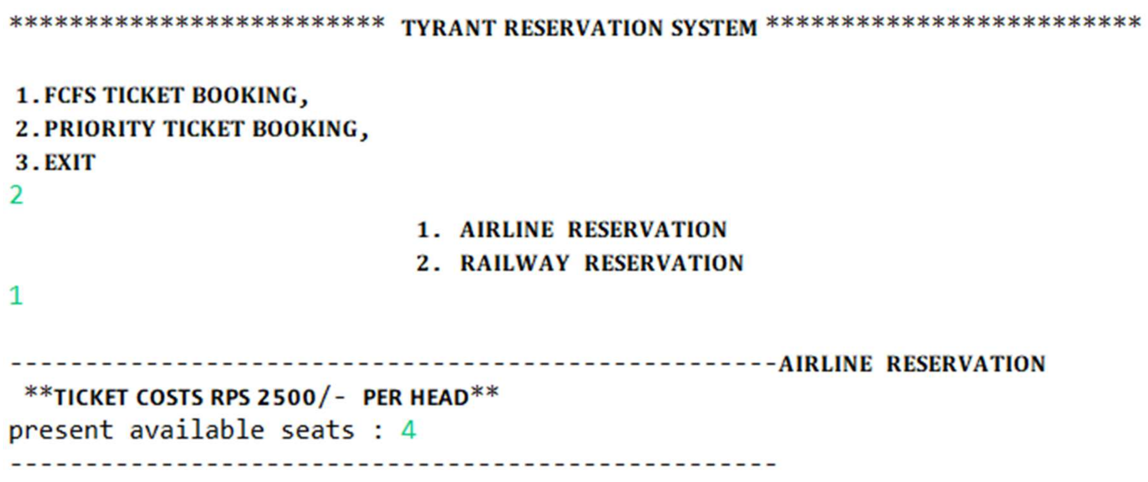
Analysis of passengers data is shown in scheduling table in [fig 7.5]:



```
-----------------------------------------------------------------------------------------------
   PASSENGER     NAME    BURST_TIME   ARRIVAL_TIME   COMPLETION_TIME   WAITING_TIME   TURN_AROUND_TIME

      1          aishu       5             0                5                0                5

      2          tanuj       3             1                8                4                7

      3          akhil       8             2               16                6               14
-----------------------------------------------------------------------------------------------

AVERAGE WAITING TIME  = 3.3333333 ms
AVERAGE TURN AROUND TIME =  8.666667 ms
```

Figure 7.5

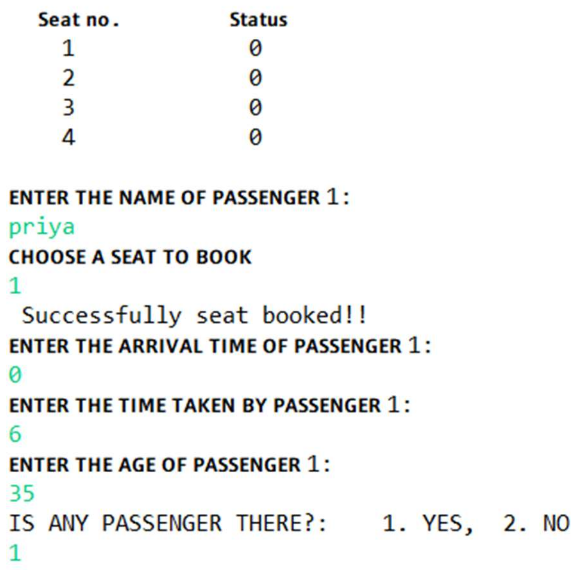
- **USING PRIORITY:**



```
************************* TYRANT RESERVATION SYSTEM *************************

 1.FCFS TICKET BOOKING,
 2.PRIORITY TICKET BOOKING,
 3.EXIT
2
                              1. AIRLINE RESERVATION
                              2. RAILWAY RESERVATION
1

---------------------------------------------------AIRLINE RESERVATION
 **TICKET COSTS RPS 2500/- PER HEAD**
present available seats : 4
---------------------------------------------------
```

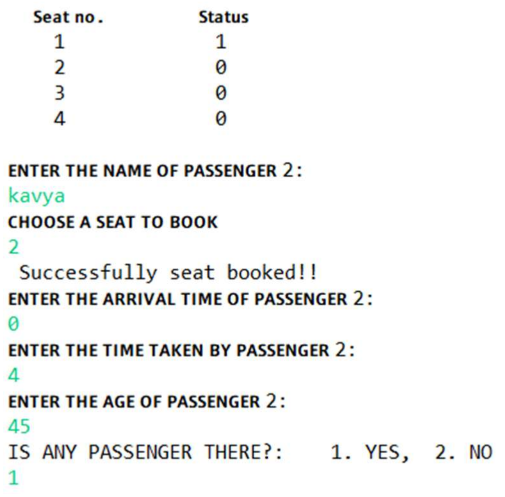
Figure 7.6

Entry of details of passengers are taken in [Fig 7.7, 7.8, 7.9]



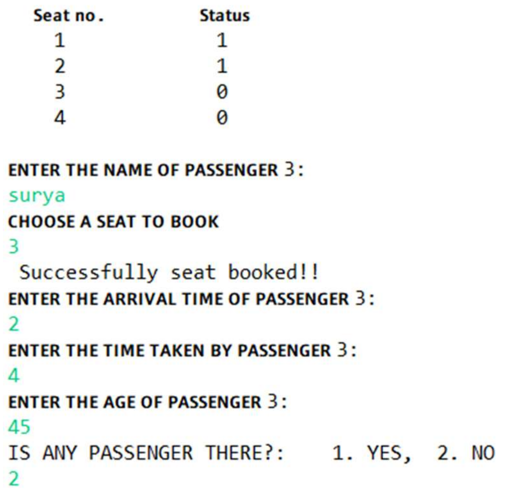
```
  Seat no.        Status
     1              0
     2              0
     3              0
     4              0

ENTER THE NAME OF PASSENGER 1:
priya
CHOOSE A SEAT TO BOOK
1
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 1:
0
ENTER THE TIME TAKEN BY PASSENGER 1:
6
ENTER THE AGE OF PASSENGER 1:
35
IS ANY PASSENGER THERE?:    1. YES,  2. NO
1
```

Figure 7.7



```
  Seat no.          Status
     1                1
     2                0
     3                0
     4                0

ENTER THE NAME OF PASSENGER 2:
kavya
CHOOSE A SEAT TO BOOK
2
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 2:
0
ENTER THE TIME TAKEN BY PASSENGER 2:
4
ENTER THE AGE OF PASSENGER 2:
45
IS ANY PASSENGER THERE?:    1. YES,  2. NO
1
```

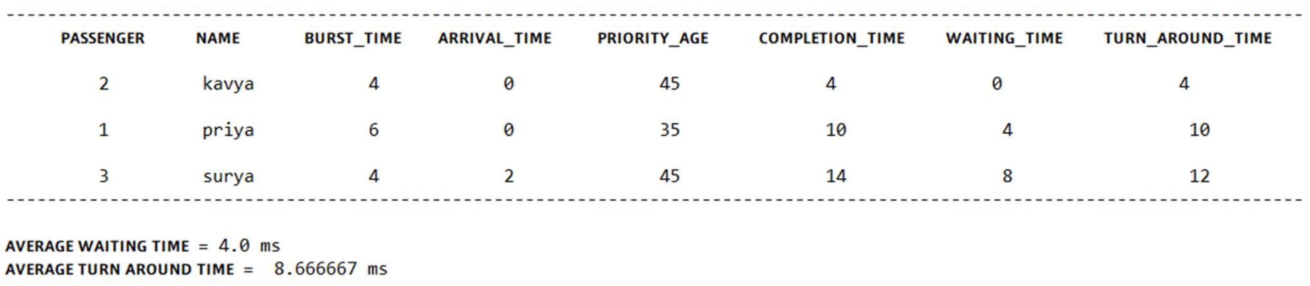
Figure 7.8



```
  Seat no.          Status
     1                1
     2                1
     3                0
     4                0

ENTER THE NAME OF PASSENGER 3:
surya
CHOOSE A SEAT TO BOOK
3
 Successfully seat booked!!
ENTER THE ARRIVAL TIME OF PASSENGER 3:
2
ENTER THE TIME TAKEN BY PASSENGER 3:
4
ENTER THE AGE OF PASSENGER 3:
45
IS ANY PASSENGER THERE?:    1. YES,  2. NO
2
```

Figure 7.9

Analysis of passengers data is shown in scheduling table in [fig 7.10]:



```
--------------------------------------------------------------------------------------------------------------------
     PASSENGER      NAME      BURST_TIME    ARRIVAL_TIME    PRIORITY_AGE    COMPLETION_TIME    WAITING_TIME    TURN_AROUND_TIME

        2          kavya          4             0               45                4                 0                 4

        1          priya          6             0               35               10                 4                10

        3          surya          4             2               45               14                 8                12
--------------------------------------------------------------------------------------------------------------------

AVERAGE WAITING TIME  = 4.0 ms
AVERAGE TURN AROUND TIME =   8.666667 ms
```

Figure 7.10

8. CONCLUSION

By using FCFS and Priority Scheduling algorithms, we have created a ticket reservation system where the user can opt the method of booking, either FCFS or priority, to book their tickets in reservation system according to their convenience. We have improved our concept of scheduling algorithms by implementing it in a real time application.