



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Parking Management System for Hospitals

CSE2006 – Microprocessor and Interfacing

Project Report

Submitted by

Nikhil Kumar Parashar - 19BCE0076

Anmol Bansal - 19BCE0630

Vidushi Gupta - 19BCE0922

Gokul Nair - 19BCE2245

School of Computing Science & Engineering

VIT, Vellore

Under the guidance of

Prof. Ragunath G

VIT, VELLORE

Table of Contents

S. No.		Page No.
1	Abstract	3
2	Introduction	3
3	Objectives	4
4	Literature Survey	4
5	Existing Systems	6
6	Proposed Systems	6
7	Requirement Analysis	6
8	Module List	7
9	Flowchart	8
10	Procedure	9
11	How to use the code?	9
12	Code Implementation	10
13	Output Screenshots	28
14	Conclusion	32
15	Contribution	23
16	References	34

Abstract

In today's world, we often spend at least 5 to 10 minutes trying to find the perfect parking spot. Especially in hospitals, it has been a significant issue for parking operators running the facility within the hospital compound. Moreover, most people prefer to commute using private vehicles which further enhances the need for planning and implementation to maintain the parking area efficiently. The actual parking demand in most areas is higher than the provided parking space. Our project aims to ensure an efficient parking facility by reducing the time wasted and unnecessary space used by the commuters.

In this project, we propose a novel parking management system that takes into account the employees, ambulances and patients. Our proposed system ensures proper parking in public areas where people might be in a hurry, as it displays the number of vacant slots before they enter into the parking place as well as the fee that they will be charged and aims to improve parking regulations. The system will segregate the staff-owned vehicles, visiting vehicles and ambulances to ensure proper allocation of the lanes. This will enable us to provide separate lanes and fees for the vehicles which help in maintaining the flow of entry and exit of vehicles. Surveys tell that 30% of total traffic is developed by people who are in search of vacant parking spaces and slots. With our system, not only the problems revolving around the patients visiting a hospital but also the traffic congestion and parking space availability in the neighbourhood is tackled. This difficulty doesn't arise because of the unavailability of parking spaces, but due to unawareness and inefficiency of the available parking spaces at that point.

Considering the delicacy of the processes in a hospital where time plays an important factor, the patients, the doctors and even the ambulances knowing earlier about their parking spot will surely help them save a precious amount of time. Following a study, the vehicle and transportation authority spend over 70 billion dollars in search of parking spots. A total of 3.6 billion hours and 1.7 billion gallons of fuel are wasted. Successful implementation of our system may increase the chances of this wastage being reduced firstly in hospitals and then in other public areas.

Introduction

Traffic congestion is perhaps the most notable metropolitan transport problem, as it causes high energy utilization and air contamination. The inaccessibility of free parking spots is one of the significant explanations behind traffic congestions. Congestion and parking are interrelated because looking for a free parking space makes extra postponements and increment local circulation. In the focal point of enormous urban communities, 10% of the traffic circulation is expected to cruising, as drivers almost burn through 20 min looking with the expectation of complimentary parking spots.

Within the last 5 years, the increase in the number of cars on the road has led to an increase in the need for parking spaces. Especially in urban cities, the requirement for parking spaces is increasing day by day. When we talk about certain places like hospitals, the increase in car owners have led to an increase in the need for the area for the parking lot. People generally visit hospitals using their private cars, and there are times when parking spaces are irregularly managed which cause delays in the assessment of the patients. Thus, there is a dire need to find a proper solution at such places which can resolve this issue not only for traffic management but also for time-saving and decrease in fuel consumption. Using a proper parking management system can solve this requisite problem. With the use of a proper parking management system, we can tackle this issue by identifying empty spaces and making the parking lots a lot more optimised which will also benefit us to keep a track of the vehicles visiting both for security purposes and covid-19 situation.

Objectives

The objective of our project is to improve the current parking scenario in hospitals by displaying the number of vacant slots before they enter into the parking place as well as the fee that they will be charged and aims to improve parking regulations.

We aim to ensure an efficient parking facility by reducing the time wasted and unnecessary space used by the commuters when trying to find a parking spot.

We plan to implement it by segregating the parking into different lanes for staff-owned vehicles, visiting vehicles and ambulances to ensure proper allocation of the parking space. This will enable us to provide separate lanes and fees for the vehicles which help in maintaining the flow of entry and exit of vehicles.

Literature Survey

1. Parking Management Strategies, Evaluation and Planning

Todd Litman, Transportation Research Board (TRB) in 2021

After a definite analysis of the current parking management strategies, the author concocts significant factors and focuses to remember before conceiving any parking management procedures. Parking management alludes to different policies and arrangements that outcomes a more efficient use. It examines issues with current parking planning, possible costs requirements inside parking facilities and describes how to develop and implement optimal parking management systems.

2. Application for Emu8086 and Proteus in Microcomputer Principal Teaching

Qing WANG and Zhong-nian LI, School of Electrical & Energy Engineering, Nantong Institute of Technology, China in 2018*

The authors set forward a new planning plan through the Emu8086 and Proteus software for assembly program writing. The simulation example used in the project, the frequency divider circuit, is constructed using the 8086 and 8253A. The experimental observations show that the constructed frequency divider circuit can be simulated by the 8086 emulators too.

3. A Novel Parking Management System, for Smart Cities, to Save Fuel, Time, and Money

Siddharth Das, IEEE in 2019

In this paper, the author proposes a new parking management system, that consists of hardware and software modules using Raspberry Pi Device along with a few other parts like sensors and cameras and python modules for efficient data management respectively.

4. IoT Based Smart Parking System Using Deep Long Short Memory Network

Ghulam Ali, Tariq Ali ,Adam Glowacz, Maciej Sulowicz, Ryszard Mielnik, Zaid Bin Faheem and Claudia Martis, Multidisciplinary Digital Publishing Institute(MDPI) in 2020

In this paper, the author proposes a structure dependent on a profound long momentary memory organization to anticipate the accessibility of parking spots with the combination of the Internet of Things (IoT), cloud innovation, and sensor organizations. The test results show that the proposed model outflanks the best in class forecast models. This paper proposes a structure dependent on a profound long transient memory organization to foresee the accessibility of parking spots with the reconciliation of the Internet of Things (IoT), cloud innovation, and sensor organizations.

5. A novel and Secure Smart Parking Management System

Omar Abdulkader, Alwi M. Bamhdi, Vijey Thayanathan, Kamal Jambi, Muasaad Alrasheedi of Institute of Electrical and Electronics Engineers(IEEE) Explore in 2018

This paper aims to design secure and smart parking monitoring, controlling and management solutions based on the integration of Wireless Sensor Network (WSN), Radio Frequency Identification (RFID), Adhoc Network, and Internet of Things (IoT).

6. Design and Implementation of Smart Car Parking System

P B Natarajan, Samit Kumar Ghosh, International Journal of Pure and Applied Mathematics in 2018

The proposed vehicle parking system takes the assistance of IR sensors to track down the vehicle at the entry and the exit area and along these lines assigns just as de-apportions

the accessible leaving openings to the vehicles. The exhibition has shown the capacity of the way to deal with holding the parking garage, increment the passage to the stopping region just as kills the hardships of looking through void parking garages. This paper proposes a smart parking system to solve the problem of unnecessary time consumption in finding parking spots in commercial car park areas using sensors such as IR (infrared) are used to detect the entrance or exit of the car at the parking slot. The existing system gives information about the empty slots

Existing Systems

The software's or the parking management systems used in most of the hospitals are in their preliminary stage currently. The present system only keeps records of the in-time and the exit-time of the vehicles. With such a system, we won't be able to effectively manage parking spaces since once after entry we don't have any record of the number of vehicles currently present inside the parking lot. Therefore, while designing such systems we need to make sure that we can efficiently track them even when they are inside the parking lounge. The main motive here is to know how many spaces are left for parking and which size of vehicles can get in.

Proposed System

The existing system has a few faults that our novel proposed system deliberately solves. With the new system, we can tackle the Parking Space vacancy, Parking Billing and Vehicle Quota for various types of vehicles effectively. In our proposed system, we can classify the vehicle types as ambulance, staff vehicle or a visitor vehicle through which we can change a pre-set amount for each vehicle. The system keeps a record of the number of vehicles parked inside the parking lot, such that we can keep track of vacant spaces for each type of vehicle. The proposed method has a pre-defined limit; which may be different for different vehicle parking spaces; for each vehicle such that there is always the knowledge for the vacancies left for different types of vehicles and if there is any need to convert the spaces between the various options. This would efficiently solve a major problem when there are cases that a certain set of vehicles occupy all spaces.

Requirements Analysis

1. Functional Requirements

- The type of added vehicle should be reflected and the amount of that particular type should be specified.
- The limit of the parking lot must be checked every time a vehicle enters to maintain the flow of traffic.

- The total amount received should be shown to keep a track of the financial status.

2. Non-Functional Requirements

- The system must not take time and respond quickly
- The system must be fault-tolerant
- The system must be maintainable regardless of the environment

3. System Requirements

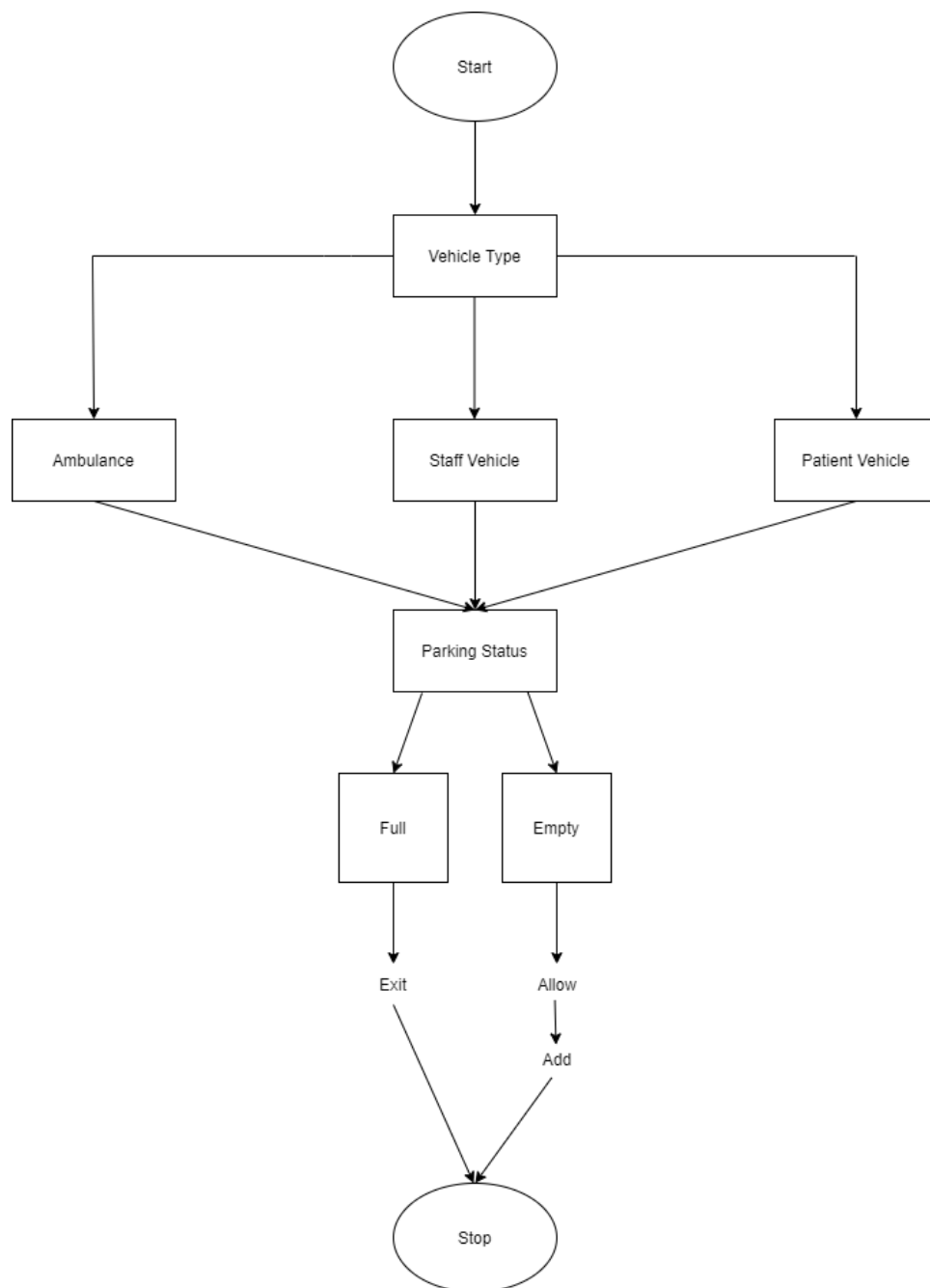
- Any processor capable of running ALP
- RAM: Minimum of 128MB required
- ROM: Minimum of 512MB required

Module List

We have a total of Seven Modules in this system:

1. Main Menu
Displays all the options in the Hospital Parking System, and asks for input from the user.
2. Add/Park Vehicle
Provides options to park an ambulance, staff vehicle or patient vehicle, and then displays the parking fee that will be charged from that type of vehicle.
3. Show Details of the Parking Lot
Displays the total amount to be collected for all the vehicles parked on that day, the total number of vehicles parked, and the number of vehicles parked for each type of vehicle.
4. Delete/Remove Vehicle from the Parking
Deletes all vehicles from the parking space and frees all the space.
5. Set and Check Vehicle Limit
It sets the number of vehicles that are allowed for a certain type of vehicle and when the limit is exceeded, it displays a message for the same.
6. Show the Total Amount to be Received
Shows the total parking fee to be received for that day.
7. Reset Parking Lot Data
Resets the system and all variables to 0 for a new day.

Flowchart



Procedure

Step 1: Define the messages and variables that will be used in the program

Step 2: Display the menu to the user

Step 3: Take input from the user

Step 4: According to the choice entered, compare with the stored value for the respective procedure and call it.

Step 5: If a wrong input is entered, display the message, "Wrong input", and then display a fresh menu.

Step 6: Define the procedure for each option, i.e., Add ambulance, add staff vehicle, add a patient vehicle, show all records, delete a vehicle, Reset Parking Space and Exit the program.

Step 7: Taking the example of the procedure for the ambulance, set the vehicle limit and the further procedure will only run if the lesser than condition is satisfied.

Step 8: Then the amount of parking fee for the respective vehicle is added to the amount, and the same is printed on the output screen.

Step 9: Similar procedure is called for adding a staff vehicle and adding a patient vehicle.

Step 10: To delete a vehicle, a separate procedure is called, in which we decrement the count of total vehicles and that specific vehicle, each by 1.

Step 11: To delete all vehicles, all the vehicle variables, and amount variable is set to 0, and the space has been reset for a new day.

Step 12: Finally, to terminate the program, the particular DOS Function is called, and the program is halted.

How to use the Code?

1. Open the .asm file on a device that has emu8086 installed.
2. Emulate the program and click on run.
3. An output screen will be displayed with the menu options for the user to choose from.
4. The menu will be repeated until the user chooses to exit the program.

Code Implementation

```
;PARKING MANAGEMENT SYSTEM FOR HOSPITALS
.model small

.stack 100h

.data

menu db '*****MENU*****$'
menu1 db 'Enter 1 to park an ambulance$'
menu2 db 'Enter 2 to park a staff vehicle$'
menu3 db 'Enter 3 to park a patient vehicle$'
menu4 db 'Enter 4 to show the parked vehicles$'

menudelete1 db 'Enter 5 to remove ambulance$'
menudelete2 db 'Enter 6 to remove staff vehicle$'
menudelete3 db 'Enter 7 to remove patient vehicle$'
menudelete4 db 'Enter 8 to remove all vehicles$'

menu6 db 'Enter 9 to exit the program$'

msg db 'Enter your choice: $'

msg1 db 'Parking is full$'
msg2 db 'Wrong input$'
msg3 db 'Staff Vehicle$'
msg4 db 'Patient Vehicle$'
msg5 db 'Records$'
msg6 db 'There is more space$'
msg7 db 'Total amount is = $'
msg8 db 'Total numbers of vehicles parked = $'
msg9 db 'Total number of ambulances parked = $'
```

```
msg10 db 'Total number of staff vehicles parked = $'
msg11 db 'Total number of patient vehicles parked = $'
msg12 db '***Records deleted successfully***$'
msg13 db '***Vehicle removed successfully***$'
```

```
fee db 'Parking Fee: Rs. $'
```

```
amount dw 0
count dw '0'
```

```
a dw '0'    ;ambulances
s dw '0'    ;staff vehicles
p dw '0'    ;patient vehicles
```

```
.code
```

```
main proc
```

```
    mov ax,@data
```

```
    mov ds,ax
```

```
while_:
```

```
    ;Menu to be displayed
```

```
    mov dx,10
```

```
    mov ah,2
```

```
    int 21h
```

```
    mov dx,13
```

```
    mov ah,2
```

```
    int 21h
```

```

mov dx,offset menu    ;MENU
mov ah,9              ;Display a character string
int 21h               ;Call DOS Function
mov dx,10
mov ah,2              ;Write to a standard output device
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menu1    ;OPTION1
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menu2    ;OPTION2
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13

```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset menu3      ;OPTION3
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset menu4      ;OPTION4
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset menudelete1 ;OPTION5
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menudelete2    ;OPTION6
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menudelete3    ;OPTION7
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menudelete4    ;OPTION8
mov ah,9
```

```
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset menu6          ;OPTION9
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

mov dx,offset msg           ;ENTER CHOICE MESSAGE
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h
```

```

;Taking input from user
mov ah,1          ;Read input from keyboard
int 21h           ;Call DOS Function
mov bl,al
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h

;Now comparing the option entered to the cases
mov al,bl
cmp al,'1'
je amb            ;Jump if equal to 1
cmp al,'2'
je sv
cmp al,'3'
je pv
cmp al,'4'
je rec
cmp al,'5'
je del1
cmp al, '6'
je del2
cmp al, '7'
je del3
cmp al, '8'
je del

```



```
cmp al, '9'  
je end_
```

```
mov dx,offset msg2  
mov ah,9  
int 21h
```

```
mov dx,10  
mov ah,2  
int 21h  
mov dx,13  
mov ah,2  
int 21h
```

```
jmp while_
```

```
amb:  
call ambulance
```

```
sv:  
call staffvehicle
```

```
pv:  
call patientvehicle
```

```
rec:  
call recrd
```

```

del1:
call dela

del2:
call dels

del3:
call delp

del:
call delt

end_:
mov ah,4ch          ;Exit the program
int 21h             ;Call DOS Function

main endp

ambulance proc
    cmp a,'4'        ;upto 4 ambulances can be parked
    jl amb1          ;Jump if lesser than
    mov dx,offset msg1
    mov ah,9
    int 21h
    jmp while_
    jmp end_

amb1:

```

```

    mov ax,0          ;Parking Fee for Ambulance is 0
    add amount, ax
    mov dx,0
    mov bx,10
    mov cx,0

12:                ;To push the amount into the stack
    div bx
    push dx
    mov dx,0
    mov ah,0
    inc cx
    cmp ax,0
    jne 12          ;jump if not equal to

    mov dx,offset fee
    mov ah,9
    int 21h

13:                ;To print the amount from the stack digit by
digit
    pop dx
    add dx,48
    mov ah,2
    int 21h
    loop 13

    inc count
    inc a

```

```
    jmp while_
```

```
    jmp end_
```

```
staffvehicle proc
```

```
    cmp s,'25'
```

```
    jl staffvehicle1
```

```
    mov dx,offset msg1
```

```
    mov ah,9
```

```
    int 21h
```

```
    jmp while_
```

```
    jmp end_
```

```
staffvehicle1:
```

```
    mov ax,100
```

```
    add amount, ax
```

```
    mov dx,0
```

```
    mov bx,10
```

```
    mov cx,0
```

```
122:
```

```
    div bx
```

```
    push dx
```

```
    mov dx,0
```

```
    mov ah,0
```

```
    inc cx
```

```
    cmp ax,0
```

```
    jne 122
```

```
mov dx,offset fee
```

```
mov ah,9
```

```
int 21h
```

```
133:
```

```
    pop dx
```

```
    add dx,48
```

```
    mov ah,2
```

```
    int 21h
```

```
loop 133
```

```
inc count
```

```
inc s
```

```
jmp while_
```

```
jmp end_
```

```
patientvehicle proc
```

```
    cmp p,'50'
```

```
    jl patientvehicle1
```

```
    mov dx,offset msg1
```

```
    mov ah,9
```

```
    int 21h
```

```
    jmp while_
```

```
    jmp end_
```

```
patientvehicle1:
```

```
    mov ax, 100
```

```
add amount, ax
mov dx,0
mov bx,10
mov cx,0
1222:
    div bx
    push dx
    mov dx,0
    mov ah,0
    inc cx
    cmp ax,0
    jne 1222

mov dx,offset fee
mov ah,9
int 21h

1333:
    pop dx
    add dx,48
    mov ah,2
    int 21h
loop 1333

inc count
inc p
jmp while_
jmp end_
```

```

recrd proc
    mov dx,offset msg7
    mov ah,9
    int 21h

    ;Print the total amount
    mov ax, amount

    mov dx,0
    mov bx,10
    mov cx,0
totalpush:
    div bx
    push dx
    mov dx,0
    inc cx
    cmp ax,0
    jne totalpush

totalprint:
    pop dx
    add dx,48
    mov ah,2
    int 21h
loop totalprint

    mov dx,10
    mov ah,2

```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset msg8
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,count
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset msg9
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,a
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,10
```



```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset msg10
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,s
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,offset msg11
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,p
```

```
mov ah,2
```

```
int 21h
```

```
jmp while_
```

```
jmp end_
```

```
dela proc
```

```
dec a
```

```
dec count
```

```
mov dx,offset msg13
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
```

```
mov dx,13
```

```
mov ah,2
```

```
int 21h
```

```
jmp while_
```

```
jmp end_
```

```
dels proc
```

```
dec s
```

```
dec count
```

```
mov dx,offset msg13
```

```
mov ah,9
```

```
int 21h
```

```
mov dx,10
```

```
mov ah,2
```

```
int 21h
mov dx,13
mov ah,2
int 21h
jmp while_
jmp end_
```

delp proc

```
dec p
dec count
mov dx,offset msg13
mov ah,9
int 21h
mov dx,10
mov ah,2
int 21h
mov dx,13
mov ah,2
int 21h
jmp while_
jmp end_
```

delt proc

```
mov a,'0'
mov s,'0'
mov p,'0'
mov amount,0
mov count,'0'
```

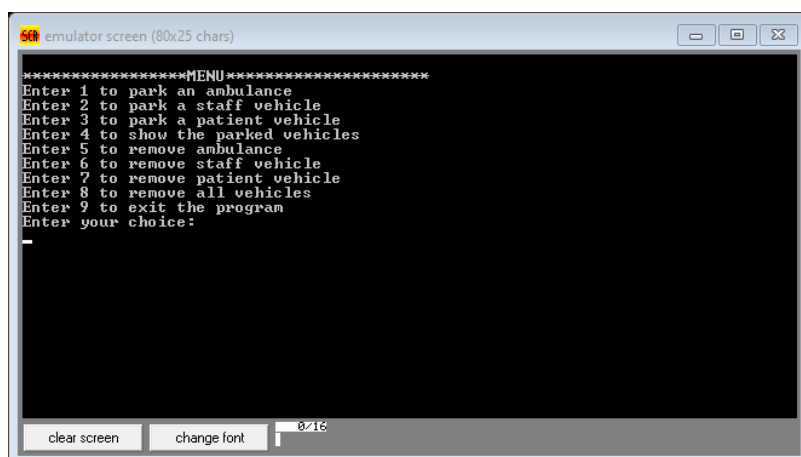
```
    mov dx,offset msg12
    mov ah,9
    int 21h
    mov dx,10
    mov ah,2
    int 21h
    mov dx,13
    mov ah,2
    int 21h

    jmp while_
    jmp end_

end main
```

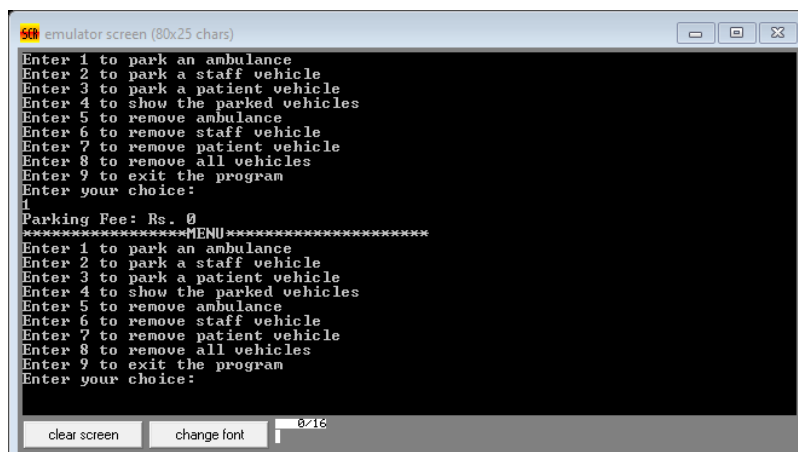
Output Screenshots

1. Menu



The prompt shows the menu with various options as mentioned in the image. All we need to do is type the desired number corresponding to the operation we want to perform.

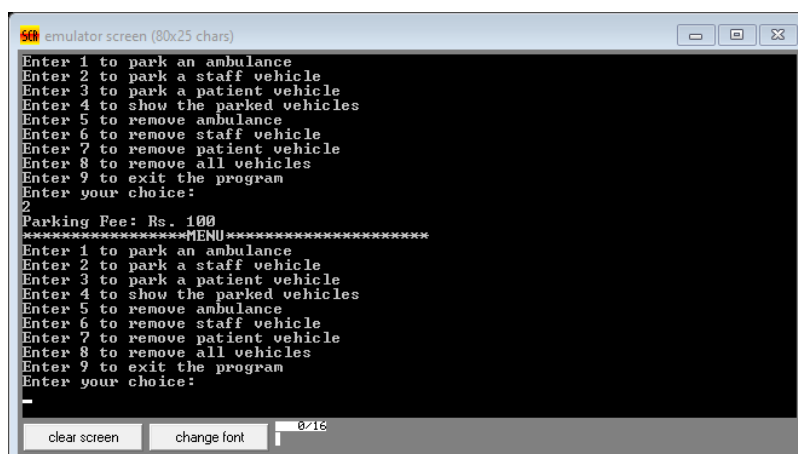
2. Adding an ambulance



```
emulator screen (80x25 chars)
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
1
Parking Fee: Rs. 0
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
clear screen change font 0/16
```

If we choose 1, the parking space is filled with an ambulance and the parking fee is shown on the screen.

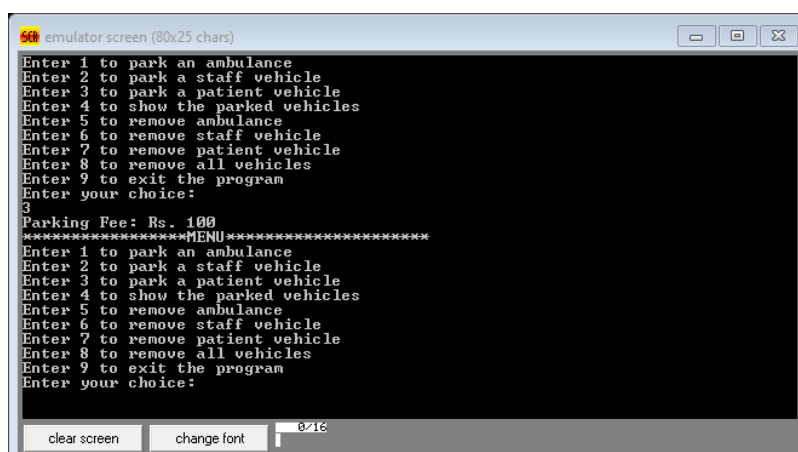
3. Adding a staff vehicle



```
emulator screen (80x25 chars)
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
2
Parking Fee: Rs. 100
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
clear screen change font 0/16
```

If we choose 2, the parking space is filled with a private vehicle and the parking fee is shown on the screen.

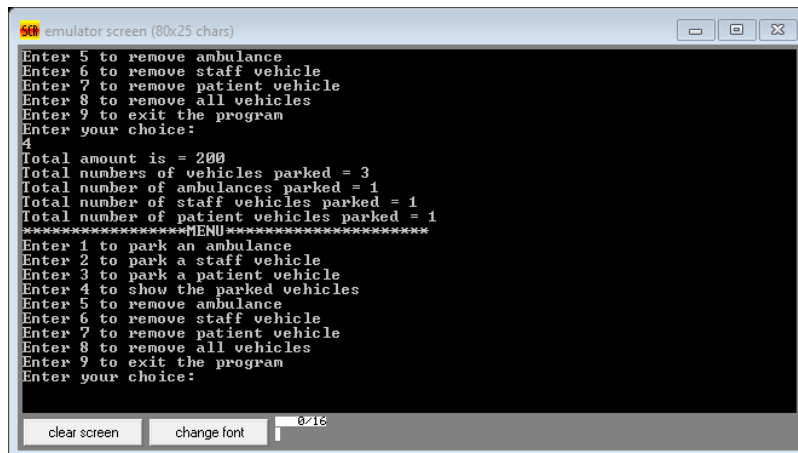
4. Adding a patient vehicle



```
emulator screen (80x25 chars)
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
3
Parking Fee: Rs. 100
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
clear screen change font 0/16
```

If we choose 3, the parking space is filled with a patient vehicle and the parking fee is shown on the screen.

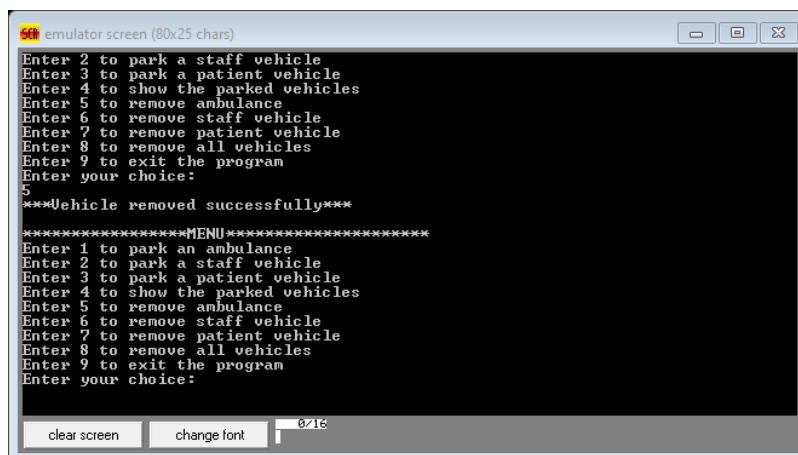
5. Show all the parked vehicles



```
emulator screen (80x25 chars)
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
4
Total amount is = 200
Total numbers of vehicles parked = 3
Total number of ambulances parked = 1
Total number of staff vehicles parked = 1
Total number of patient vehicles parked = 1
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
```

If we choose 4, the program shows the details of the parking lot, with the total amount and the numbers of vehicles parked with type.

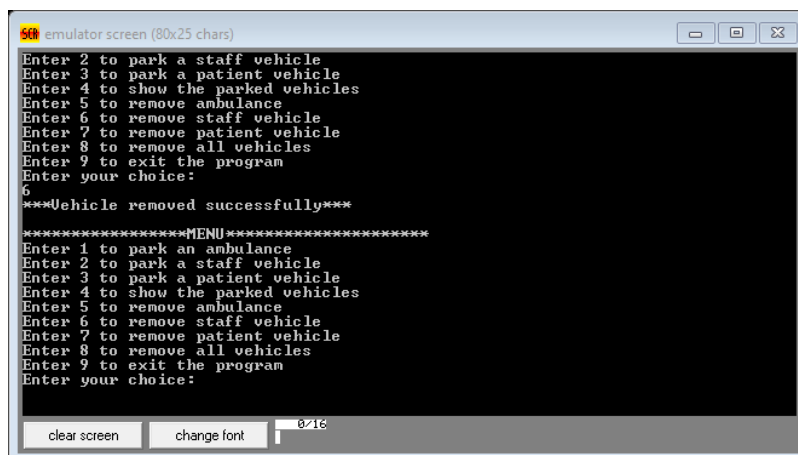
6. Delete an ambulance



```
emulator screen (80x25 chars)
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
5
***Vehicle removed successfully***
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
```

If we choose 5, one ambulance is deleted from the parking system.

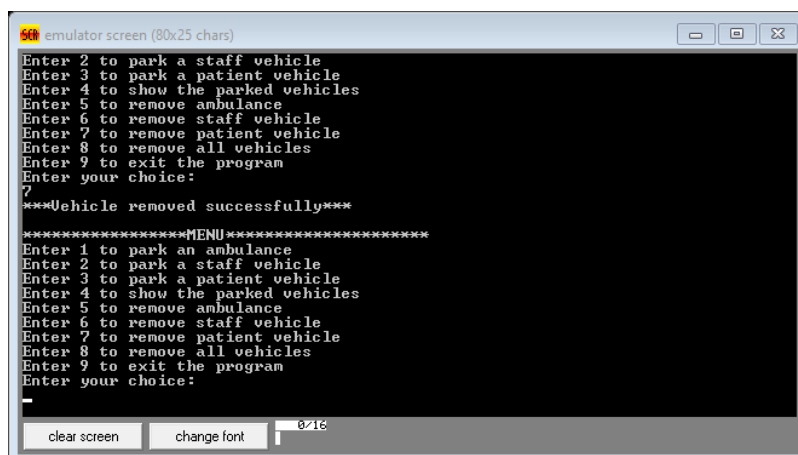
7. Delete a staff vehicle



```
emulator screen (80x25 chars)
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
6
***Vehicle removed successfully***
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
```

If we choose 6, one staff vehicle is deleted from the parking system.

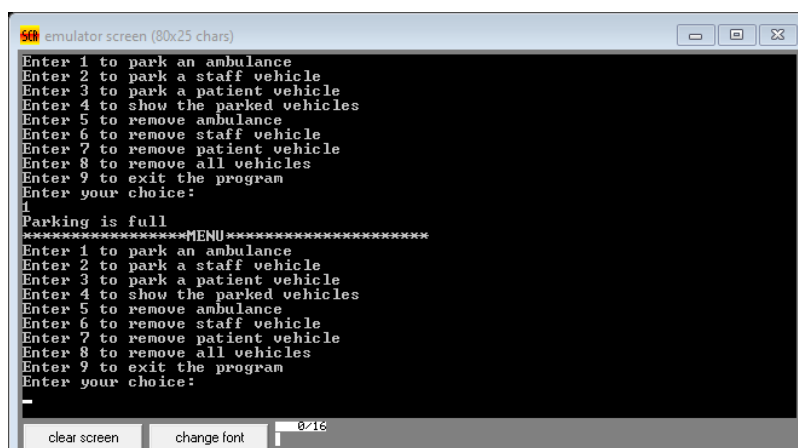
8. Delete a patient vehicle



```
emulator screen (80x25 chars)
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
7
***Vehicle removed successfully***
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
```

If we choose 7, one patient vehicle is deleted from the parking system.

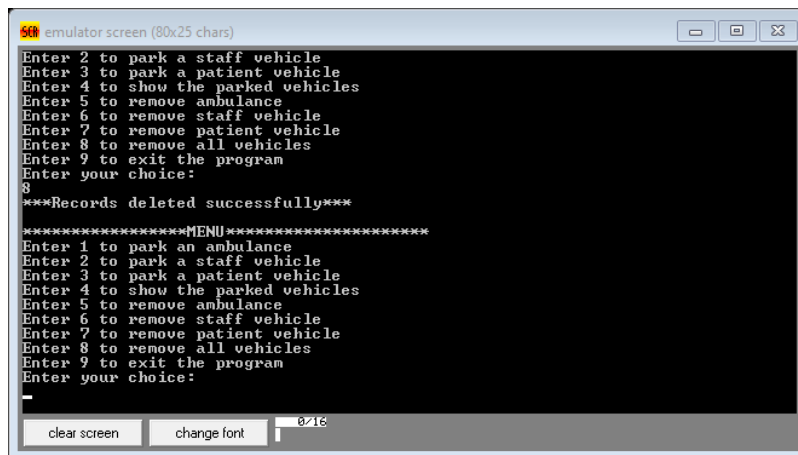
9. Limit has been exceeded (Parking Full)



```
emulator screen (80x25 chars)
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
1
Parking is full
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
```

After 4 ambulances have been parked, there isn't space for any more ambulances to be parked.

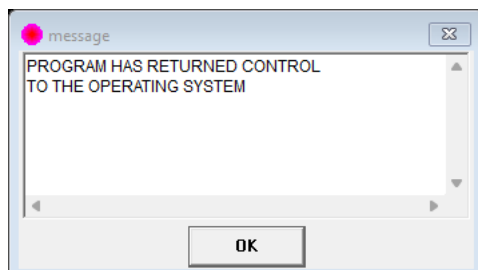
10. Reset Parking Space



```
emulator screen (80x25 chars)
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
8
***Records deleted successfully***
*****MENU*****
Enter 1 to park an ambulance
Enter 2 to park a staff vehicle
Enter 3 to park a patient vehicle
Enter 4 to show the parked vehicles
Enter 5 to remove ambulance
Enter 6 to remove staff vehicle
Enter 7 to remove patient vehicle
Enter 8 to remove all vehicles
Enter 9 to exit the program
Enter your choice:
_
clear screen  change font  0/10
```

If we choose 8, the parking lot is reset.

11. Terminate the program



Conclusion

With the proposed method, we have successfully implemented all the modules for a hospital parking system, by using commands like CMP, JE, JLE, MOV and LOOPS. The program successfully compiles all the information needed to manage a parking system. The admin using the program gets the total fee calculated, and a proper record of all vehicles that are currently parked. The program can keep a track record of the parking limit however the data of each day's parking doesn't get stored anywhere for future reference, as we would need to make a database connection to do that.

Contribution

Nikhil Kumar Parashar - 19BCE0076

Literature Survey and Documentation with Printing Menu and Comparing the User Input

Anmol Bansal - 19BCE0630

Literature Survey and Documentation with Add Vehicle and Calculate Total Amount

Vidushi Gupta - 19BCE0922

Literature Survey and Documentation with Delete Vehicle and Show All Records

Gokul Nair - 19BCE2245

Literature Survey and Documentation with Setting the Limit for Parking Lot and Reset the Parking Data

References

[1] https://www.vtpi.org/park_man.pdf

[2] <https://www.dpi-proceedings.com/index.php/dtcse/article/view/24791>

[3] <https://ieeexplore.ieee.org/document/8666537>

[4] <https://www.mdpi.com/2079-9292/9/10/1696>

[5]

https://www.researchgate.net/publication/327107437_Design_and_Implementation_of_Smart_Car_Parking_System_Using_LabVIEW

<more resources related to ASM and EMU8086>

Plagiarism Report

