*PARKING-LOT PROJECT*

**Team Members Name:**

| AKSHARA | DESIGNING THE PARKING LOT. |
| --- | --- |
| AISHWARYA | DIFFERENT IDEAS FOR IMPLEMENTING THE LOT. |
| ROSHAN | FINDING BUGS AND FIXING IT. |
| PRATHIK | WRITING THE CODE AND DESIGN. |

**FINAL SOLUTION:**

**Person**:

It takes care of all the information about the person, this information will be collected by the attendant at the parking lot.

**class Person{**

**Person(String name,Vehicle v,Boolean x){**

**this.name = name;**

**this.v = v;**

**this.Handicapped = x;**

**}**

**private String name;**

**private Vehicle v;**

**public Boolean Handicapped;**

**}**

**Parking Attendant:**

This person takes care of all the information about the parking lot, he/she will be allocating parking spots in the lot, and all the transaction takes place.

class Parking\_Attendant{

private String name;

Parking\_Attendant(String name){

this.name = name;

}

void Welcome(){

System.out.println("Welcome to the lot");

}

void find\_the\_spot(Parking\_floor f1,Vehicle v){

if(v.Vehicle\_type.equals("car")){

if(!f1.bike\_capacity\_is\_full()){

System.out.println("You can park in the floor no: "+f1.get\_floor\_no());

}

}

else if(v.Vehicle\_type.equals("bike")){

if(!f1.bike\_capacity\_is\_full()){

System.out.println("You can park in the floor no: " + f1.get\_floor\_no());

}

}

else{

System.out.println("No spots");

}

}

void dept\_msg(){

System.out.println("Have a safe ride");

}

}

**Money:**

This class is used to pay the money to the parking attendant and all types of transactions.

class Money{

private int money;

private String mode\_of\_payement;

Money(int money,String mode\_of\_payement){

this.money = money;

this.mode\_of\_payement = mode\_of\_payement;

}

int get\_money(){

return this.money;

}

void pay\_money(int money){

System.out.println("Payed Rs: "+money);

}

}

**Ticket:**

This will given by parking attendant after you are given the spot in the parking lot which includes various information about the person and his vehicle.

class Ticket{

private Person p;

private Vehicle v;

private int arrival\_time\_h;

private int arrival\_time\_m;

Ticket(Person p,Vehicle v,int arrival\_time\_h,int arrival\_time\_m){

this.p = p;

this.v=v;

this.arrival\_time\_h = arrival\_time\_h;

this.arrival\_time\_m = arrival\_time\_m;

}

int money\_to\_be\_paid(int departure\_time\_h,int departure\_time\_m){

int h = departure\_time\_h-arrival\_time\_h;

if(v.Vehicle\_fuel.equals("EV")){

if((0<h && h<1)){

return 30;

}

else if(1<h && h<3)return 20;

else return 10;

}

else{

if ((0 < h && h < 1)) {

return 20;

} else if (1 < h && h < 3)

return 10;

else

return 5;

}

}

void Transcation(int x){

System.out.println("Pay the amount: "+x);

}

}

**Parking Lot:**

This is the starting block of the program which includes details about the parking lot.

class Parking\_lot {

private String name;

private String area;

private String city;

Parking\_lot(String name, String area, String city){

this.name = name;

this.area = area;

this.city = city;

}

void Welcome\_To\_Lot() {

System.out.println("Welcome to the Amazing Parking lot");

System.out.println("Name of Parking Lot: " + this.name);

System.out.println("area: " + this.area);

System.out.println("city: " + this.city);

}

}

**Parking Floor:**

This is the key body of the entire project which includes several information of each floor. It has floor no,capacity of vehicles and also capacities of each type of parking spot, entry and exit points of the floor.

The important method we applied is an emergency exit for the floors such that it will be useful for the people who are in an emergency situation.

class Parking\_floor {

private int floor\_no;

private int capacity\_of\_vehicles;

private int vehicle\_count = 0;

private int Compact\_Spot = 0;

private int Large\_Spot = 0;

private int Handicapped\_Spot = 0;

private int Bike\_Spot = 0;

void set\_floor\_no(int floor\_no){

this.floor\_no = floor\_no;

}

int get\_floor\_no() {

return floor\_no;

}

void set\_capacity(int cap) {

this.capacity\_of\_vehicles = cap;

}

void enter\_the\_floor(Vehicle v,Person p) {

System.out.println("Enter the vehicle to the floor " + this.floor\_no);

vehicle\_count++;

if(v.Vehicle\_type.equals("van") || v.Vehicle\_type.equals("car"))Compact\_Spot++;

else if(v.Vehicle\_type.equals("Truck"))Large\_Spot++;

else if(p.Handicapped==true)Handicapped\_Spot++;

else Bike\_Spot++;

}

void exit\_the\_floor() {

System.out.println("exit the vehicle from the floor " + this.floor\_no);

vehicle\_count--;

}

void emergency\_exit(){

System.out.println("Moving to the ground floor");

}

boolean capacity\_is\_full() {

if (vehicle\_count == capacity\_of\_vehicles)

return true;

return false;

}

boolean compact\_capacity\_is\_full() {

if (Compact\_Spot == capacity\_of\_vehicles/4)

return true;

return false;

}

boolean large\_capacity\_is\_full() {

if (Large\_Spot == capacity\_of\_vehicles / 4)

return true;

return false;

}

boolean handicapped\_capacity\_is\_full() {

if (Handicapped\_Spot == capacity\_of\_vehicles / 4)

return true;

return false;

}

boolean bike\_capacity\_is\_full() {

if (Bike\_Spot== capacity\_of\_vehicles / 4)

return true;

return false;

}

void Display\_board() {

System.out.println("Floor No: " + this.floor\_no);

System.out.println("Empty Spots: " + (this.capacity\_of\_vehicles - this.vehicle\_count));

System.out.println("Compact Spots: "+(capacity\_of\_vehicles/4 - Compact\_Spot));

System.out.println("Large Spots: " + (capacity\_of\_vehicles / 4 - Large\_Spot));

System.out.println("Handicapped Spots: " + (capacity\_of\_vehicles / 4 - Handicapped\_Spot));

System.out.println("Bike Spots: " + (capacity\_of\_vehicles / 4 - Bike\_Spot));

}

}

**Parking Spot:**

We have chosen the abstract method for defining the parking spot as there were different types of parking spots mentioned in the task.

i) Compact Spot - Van/Car

ii) Large Spot - Truck

iii)Handicapped Spot- For handicapped people

iv)Bike spot-For Bike.

abstract class Parking\_Spot {

private String type\_of\_parking\_spot;

protected int Spot\_number;

public boolean Is\_spot\_free;

abstract void assign\_spot\_to\_vehicle(Vehicle v,Person p);

abstract void remove\_vehicle\_from\_spot();

abstract void set\_EV(Vehicle v);

}

class Compact\_Spot extends Parking\_Spot{

void assign\_spot\_to\_vehicle(Vehicle v,Person p){

if(v.Vehicle\_type.equals("van") || v.Vehicle\_type.equals("car") ){

Is\_spot\_free = false;

}

}

void set\_EV(Vehicle v){

if(v.Vehicle\_fuel.equals("EV")){

System.out.println("Set a Electric Charger At the Spot");

}

}

void remove\_vehicle\_from\_spot(){

Is\_spot\_free = false;

}

}

class Large\_Spot extends Parking\_Spot {

void assign\_spot\_to\_vehicle(Vehicle v,Person p) {

if (v.Vehicle\_type.equals("Truck")) {

Is\_spot\_free = false;

}

}

void set\_EV(Vehicle v) {

if (v.Vehicle\_fuel.equals("EV")) {

System.out.println("Set a Electric Charger At the Spot");

}

}

void remove\_vehicle\_from\_spot() {

Is\_spot\_free = false;

}

}

class Handicapped\_Spot extends Parking\_Spot {

void assign\_spot\_to\_vehicle(Vehicle v,Person p) {

if (p.Handicapped) {

Is\_spot\_free = false;

}

}

void set\_EV(Vehicle v) {

if (v.Vehicle\_fuel.equals("EV")) {

System.out.println("Set a Electric Charger At the Spot");

}

}

void remove\_vehicle\_from\_spot() {

Is\_spot\_free = false;

}

}class Bike\_Spot extends Parking\_Spot {

void assign\_spot\_to\_vehicle(Vehicle v,Person p) {

if (v.Vehicle\_type.equals("Bike")) {

Is\_spot\_free = false;

}

}

void set\_EV(Vehicle v) {

if (v.Vehicle\_fuel.equals("EV")) {

System.out.println("Set a Electric Charger At the Spot");

}

}

void remove\_vehicle\_from\_spot() {

Is\_spot\_free = false;

}

}

**Vehicle:**

This is a class which takes care of various types of vehicle and fuel,vehicle no etc.

class Vehicle {

public String Vehicle\_type;

public String Vehicle\_fuel; //EV or Normal vehicle

public String Vehicle\_no;

Vehicle(String p) {

this.Vehicle\_type = p;

}

public void set\_vehicle\_no(String Vehicle\_no) {

this.Vehicle\_no = Vehicle\_no;

}

void set\_vehicle\_fuel(String Vehicle\_fuel){

this.Vehicle\_fuel = Vehicle\_fuel;

}

public void get\_vehicle\_type() {

System.out.println(Vehicle\_type);

}

}

class Car extends Vehicle {

Car(String p) {

super(p);

}

}

class Truck extends Vehicle {

Truck(String p) {

super(p);

}

}

class Van extends Vehicle {

Van(String p) {

super(p);

}

}

class Bike extends Vehicle {

Bike(String p) {

super(p);

}

}

**Main Function:**

We have created a working model using all the classes and we have more focussed on maintaining the different types of class so that working becomes easier.

public class Main{

public static void main(String[] args){

Parking\_lot p1 = new Parking\_lot("Xeno\_lot","RajNagar","Hubli");

p1.Welcome\_To\_Lot();

Parking\_floor f1 = new Parking\_floor();

f1.set\_capacity(4);

f1.set\_floor\_no(1);

Parking\_floor f2 = new Parking\_floor();

f2.set\_capacity(4);

f2.set\_floor\_no(2);

Parking\_floor f3 = new Parking\_floor();

f3.set\_capacity(4);

f3.set\_floor\_no(3);

Vehicle v = new Vehicle("bike");

v.set\_vehicle\_fuel("NV");

v.set\_vehicle\_no("5456");

Person prathik = new Person("prathik",v,false);

/\* For floor 1\*/

Compact\_Spot f1c1 = new Compact\_Spot();

Large\_Spot f1c2 = new Large\_Spot();

Handicapped\_Spot f1c3 = new Handicapped\_Spot();

Bike\_Spot f1c4 = new Bike\_Spot();

/\* For floor 2 \*/

Compact\_Spot f2c1 = new Compact\_Spot();

Large\_Spot f2c2 = new Large\_Spot();

Handicapped\_Spot fc3 = new Handicapped\_Spot();

Bike\_Spot f2c4 = new Bike\_Spot();

/\* For floor 3 \*/

Compact\_Spot f3c1 = new Compact\_Spot();

Large\_Spot f3c2 = new Large\_Spot();

Handicapped\_Spot f3c3 = new Handicapped\_Spot();

Bike\_Spot f3c4 = new Bike\_Spot();

Parking\_Attendant xyz = new Parking\_Attendant("Roshan");

xyz.Welcome();

if(f1c4.Is\_spot\_free){

f1c4.assign\_spot\_to\_vehicle(v,prathik);

}

else{

System.out.println("you cant park");

}

Ticket t1 = new Ticket(prathik,v,2,30);

f1c4.remove\_vehicle\_from\_spot();

int money = t1.money\_to\_be\_paid(4, 50);

Money m = new Money(money,"UPI");

t1.Transcation(money);

m.pay\_money(money);

xyz.dept\_msg();

}

}

**Assumptions:**

* Customers can pay through UPI (different types of currencies are accepted).
* Each floor should contain a display board which will display the number of spots that are available.
* Having lifts for each floor.(1 lift for person and another lift for emergency exit)
* Total number of floors : 3
* All floors have equal no of parking spots for all types of vehicles.
* Providing nearest parking spots for handicapped drivers on each floor.
* Building a fuel machine which provides fuel to EV.(The working procedure is same as how the electric bill of a house is generated)
* EV → 50% of NV(time for which they left the vehicle)+cost of charging
* NV (Bicycle → 10),(bikes → 20 Rs),(Cars → 50),(Heavy vehicles → 100),(3 wheeler → 30).
* All Electric Vehicles will definitely use the charging spots.

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