**National University of Computer & Emerging Sciences**

**Karachi Campus**



**Project Proposal**

**Object Oriented Programming**

**Section: J**

**[Car Management System]**

**Group Members**

**19K-0289 Ali Hasnain**

**19K-0209 Syed Saad**

**19K-0330 Ziyaan Ali**

**Problem:**

As seen nowadays due to increase in crime rate, cars are snatched or stole, so to inquire about that to whom it belongs it may lack the management or proper data handling.

**Introduction:**

We are developing a program in which we would be handling all the necessary details about cars in a store / garage / showroom from the phase of data to be entered by the owner till the buying process of the costumer and would maintain it at a particular place.

So, mainly there would be 2 different setups /mode ie owner/customer.

Owner would be accessing all the day and filling all the information about the car from buying to selling phase, and Our customer mode would be that he can view all the cars available by their kind of specifications they provide, and while selling phase owner would be marking or tracking /filling all the necessary information he must have to maintain a record of the car and customer too in a legal manner.

**Methodology of OOP:**

**Encapsulation (data Hiding) + Inheritance:**

The use of Accessibility modes that are public, private and protected ensures the role of encapsulation.

In Our project these features are used as we are making classes and Our classes , the member data s and functions like entering the data of customer while buying can only be filled by the owner and owner can view anything he want , so here it means that all the data of a car is inherited and is managed by the owner of the store and he is the only accesses of everything and while calling anything directly from the main body costumer cannot access or change any data without owner’s permission which means data has been hidden or in safe mode.

**Polymorphism:**

The use of function, operator overloading and binding is included our program.

In our program we are using hierarchical as well as multiple inheritance.

**Class diagrams: +private –public \* protected**

|  |
| --- |
| **CAR** |
| + type string |
| +tax float |
| +price float |
| +registration no |
| \*assigntype() void |
| \*assignprice()void |
| \*assignregno()void |
| -displaytype()void |
| -displayprice()void |
| -displayregno()void |
| -displaytax()void |
|  |

|  |
| --- |
| **MODEL** |
| + year int |
| + engineno long |
| \*assignyear() void |
| \*assignengno()void |
| -displayyear()void |
| -displayengno()void |

|  |
| --- |
| **CUSTOMER** |
| +cnic long |
| +address string |
| +paymenttype char |
| +license long |
| \*assigncnic() void |
| \*assignaddress() void |
| \*assignpmtype()void |
| \*assignlicense() void |
| -displayall() void |
| Displaycnic() void |

|  |
| --- |
| **TYRES** |
| +nooftyres int |
| +warrantyoftyres float |
| +tyre company |
| \*assignnot()void |
| \*assignwarranty()void |
| \*assigncompany()void |
| -displaynot()void |
| -displaycompany()void |
| -displaywarranty()void |

|  |
| --- |
| **OWNER** |
| +nameofowner const string |
| +carstatue string |
| +noofcars int |
| -assignname() void |
| -assignstatus()void |
| -assignnoc()void |
| -assigneverything() void  <here it contains every function of assign of every class in switch statements> |
| -displayer () void  <here it contains every function of display of every class in switch statements> |
| +mutators ()void <here it contains mutators of every memberdata in switch cases> |

|  |
| --- |
| **BRAND** |
| +brandname string |
| +yearsince int |
| +range float |
| \*assignbrand()void |
| \*assignyrsnce()void |
| \*assignrange()void |
| -displaybrand()void |
| -displayys()void |
| -displayrange()void |

Our owner is the child class which will be accessing all the data,

Whereas costumer will be only there to set all details regarding him and can only view anything he want.

Whereas other classes are there which can be wholely have been giving its access to owner to assign and change any value.

THANKYOU.