

**WID2001 : KNOWLEDGE REPRESENTATION AND REASONING**

**GROUP 2**

**SEMESTER 2, 2017/2018**

**Instructor:**[**Dr. Siti Soraya Binti Abdul Rahman**](https://spectrum.um.edu.my/user/view.php?id=2870&course=1)

**CAR RECOMMENDATION EXPERT SYSTEM**

CHENG BOON RONG WID160011

IZATUL SYAFIQAH BINTI ZAINAL WID160013

MOHAMAD DANIEL BIN MOHAMAD ROSLAN WID160016

MOHAMAD FAKHRUL RAZI BIN OSMAN WID160017

ALI MAJEDI WID160704

**DATE: 14 MAY 2018**

**INTRODUCTION**

Frame can be defined as a “data structure that represent stereotypical knowledge about a particular object or concept. “ A frame basically consist of three components; object, attribute and value. Frame also could be generic and inherited, where certain value is inherited from class to instances.

In this assignment, we were assigned to focus on cars. There are a few attributes in this frame such as brand, type of car, price and many more. Each attribute has its own values. For example there are two brands of car chosen, Proton and Perodua. These two brands are chosen because they have variety of cars where they can be made as instances. For Perodua cars, we honed down even deeper and each model of car was a subclass in order to showcase the usage of inheritance.

This program works with user need to input a few information about the car that the user wishes to search such as the brand, user’s budget and the type of car. The information from user will be process and the system will find which car is the right match to what the user want.

There are a few of advantages of using frame-based expert system. Firstly, frame is a powerful tool to combine both declarative and procedural knowledge. Besides, frame also can be developed using object-oriented programming languages such as Java.

**AIM AND OBJECTIVES**

1. To apply the knowledge of frame that we learned in class to make the system
2. To apply the concept of frame in a real object
3. To represent the object and its attributes in a user friendly manner

**PROBLEM STATEMENT**

Currently, there are many ways in representing information and knowledge. One of the most powerful and interactive method is by using frame. Frame eases the process in representing the information because the attributes and values of a certain class can be inherited to the instances, which save time and much easier compared to store the same information from the main class or superclass every time a new class is created. Hence, this assignment aimed to apply the concept and knowledge of frame on an object that exist in real life in order to represent the information about the object and its instances in a simple manner.

**SCOPE**

This assignment focuses on two brands of cars, which is Proton and Perodua. For some people, they might face problems in searching their dream car because the search engine might showing results that less relevant and not precise.

The analysis that we made from the user is the brand that the user prefer, the type of car and user’s budget before those information are used to search and match with the cars in our system.

**DESIGN**

|  |
| --- |
| **Class : Car** |
| *Brand*  Proton  Perodua |
| *Car Type*  Sedan  Hatchback  MPV |
| *Name*  Saga  Perdana  Persona  Prevé  Suprima S  Iriz  Exora  Ertiga  Bezza  Alza  Axia  Myvi |
| *Price*  RM24000 - RM68110 |

|  |
| --- |
| **Class: Axia** |
| **Superclass: Car** |
| *Brand*  Perodua |
| *Car Type*  Hatchback |
| *Price*  RM24000-RM41500 |
| *Type*  Standard E  Standard G |
| *Transmission*  Auto  Manual |
| *Fuel consumption*  21.6  22.5 |
| *Audio*  No  Radio & USB  Radio & USB & Bluetooth |
| *Seat material*  Fabric  SE Fabric  Leather |
| *Rear sensor*  No |

|  |
| --- |
| **Class: Myvi** |
| **Superclass: Car** |
| *Brand*  Perodua |
| *Car Type*  Hatchback |
| *Price*  RM44300-RM55300 |
| *Type*  G  H  X  AV |
| *Engine Size*  1.3  1.5 |
| *Transmission*  Auto  Manual |
| *Fuel consumption*  20.1  20.5 |
| *Audio*  Radio & USB  Radio & USB & Microphone  Full with navigation |
| *Eco Idle*  No  Yes |
| *Reverse camera*  No  Yes |

|  |
| --- |
| **Class: Alza** |
| **Superclass: Car** |
| *Brand*  Perodua |
| *Car Type*  MPV |
| *Price*  RM50605-RM62900 |
| *Type*  1.5 S  1.5 SE  1.5 Advance |
| *Transmission*  Auto  Manual |
| *Audio*  Radio & USB & Bluetooth  Full with navigation |
| *Seat Material*  Fabric  SE Fabric  Leather |
| *Shopping Hook*  Without  With |

|  |
| --- |
| **Class: Bezza** |
| **Superclass: Car** |
| *Brand*  Perodua |
| *Car Type*  Sedan |
| *Price*  RM35500-RM49200 |
| *Type*  GXTRA  PREMIUM X  ADVANCE |
| *Engine Size*  1.0  1.3 |
| *Transmission*  Auto  Manual |
| *Fuel Consumption*  21  21.3  21.7  22  22.8 |
| *Audio*  Radio & USB & Bluetooth  Full with navigation |
| *Seat Material*  Fabric  Leather |
| *Rear Headphone Slot*  Yes  Yes + USB |

**PROPOSED RECOMMENDATION ENGINE**

For this assignment, we made an expert system for car recommendation. There are a few algorithms and techniques that have been used.

1. Relationship of the frame

* A Kind Of (AKO): Class and subclasses
* Example: Car (superclass) has a kind of relationship with classes Axia, Myvi, Alza, Bezza

1. Procedural Information or Demons

* IF-NEEDED: A procedure which is executed when the slot value is needed
* Example: Price has IF-NEEDED procedure because it is determined based on the type of car

1. Iteration

* to ensure the inference engine keep running

Figures below are showing how the system is running.

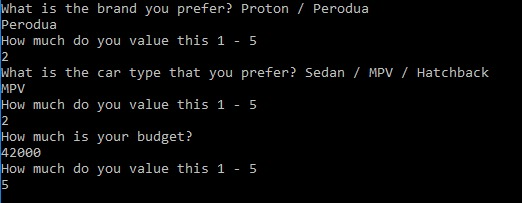


Figure 1: The system ask the appropriate questions and how important each attribute is

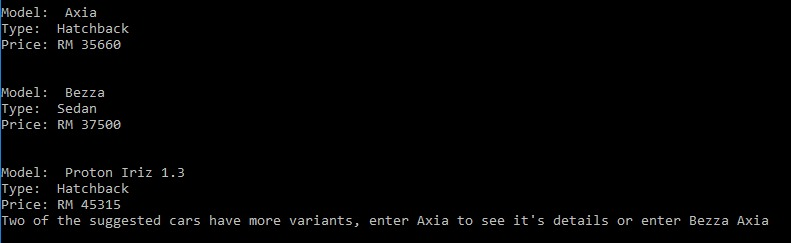


Figure 2: The system displays cars sorted by which fits the description the most. If any of the cars have more variants, user is asked if they would like to see more details

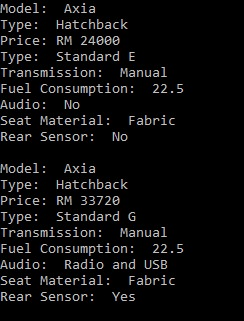


Figure 3: If user opted to view more details, every variant of the selected car is displayed

Source code:

class Car:

def \_\_init\_\_(self, brand, car\_type, name, price):

self.brand = brand

self.car\_type = car\_type

self.name = name

self.price = price

def show(self):

print("\nModel: ", self.name)

print("Type: ", self.car\_type)

print("Price: RM", self.price)

class Axia(Car):

def \_\_init\_\_(self, brand, car\_type, name, price, type, Transmission, Fuel\_C, Audio, Seat\_S, Rear\_S):

Car.\_\_init\_\_(self, brand, car\_type, name, price)

self.type= type

self.Transmission= Transmission

self.Fuel\_C= Fuel\_C

self.Audio= Audio

self.Sear\_S= Seat\_S

self.Rear\_S= Rear\_S

def show(self):

print("\nModel: ", self.name)

print("Type: ", self.car\_type)

print("Price: RM", self.price)

print("Type: ", self.type)

print("Transmission: ", self.Transmission)

print("Fuel Consumption: ", self.Fuel\_C)

print("Audio: ", self.Audio)

print("Seat Material: ", self.Sear\_S)

print("Rear Sensor: ", self.Rear\_S)

class Bezza(Car):

def \_\_init\_\_(self, brand, car\_type, name, price, type, engine\_S, Transmission, Fuel\_C, Audio, Seat\_S, Rear\_H):

Car.\_\_init\_\_(self, brand, car\_type, name, price)

self.type= type

self.engine\_S= engine\_S

self.Transmission= Transmission

self.Fuel\_C= Fuel\_C

self.Audio= Audio

self.Sear\_S= Seat\_S

self.Rear\_H= Rear\_H

def show(self):

print("\nModel: ", self.name)

print("Type: ", self.car\_type)

print("Price: RM", self.price)

print("Type: ", self.type)

print("Engine Size: ", self.engine\_S)

print("Transmission: ", self.Transmission)

print("Fuel Consumption: ", self.Fuel\_C)

print("Audio: ", self.Audio)

print("Seat Material: ", self.Sear\_S)

print("Rear Headphone Slot: ", self.Rear\_H)

class Alza(Car):

def \_\_init\_\_(self, brand, car\_type, name, price, type, Transmission, Audio, Seat\_S, Shoppin\_H):

Car.\_\_init\_\_(self, brand, car\_type, name, price)

self.type = type

self.Transmission = Transmission

self.Audio = Audio

self.Seat\_S = Seat\_S

self.Shoppin\_H = Shoppin\_H

def show(self):

print("\nModel: ", self.name)

print("Type: ", self.car\_type)

print("Price: RM", self.price)

print("Type: ", self.type)

print("Transmission: ", self.Transmission)

print("Audio: ", self.Audio)

print("Seat Material: ", self.Seat\_S)

print("Shopping Hook: ", self.Shoppin\_H)

class MyVi(Car):

def \_\_init\_\_(self, brand, car\_type, name, price, type, Engine\_S, Transmission, Fuel\_C, Audio, Eco\_I, Reverse\_C):

Car.\_\_init\_\_(self, brand, car\_type, name, price)

self.type = type

self.Engine\_S= Engine\_S

self.Transmission = Transmission

self.Fuel\_C = Fuel\_C

self.Audio = Audio

self.Eco\_I = Eco\_I

self.Reverse\_C = Reverse\_C

def show(self):

print("\nModel: ", self.name)

print("Type: ", self.car\_type)

print("Price: RM", self.price)

print("Type: ", self.type)

print("Engine size: ", self.Engine\_S)

print("Transmission: ", self.Transmission)

print("Fuel Consumption: ", self.Fuel\_C)

print("Audio: ", self.Audio)

print("Eco Idle: ", self.Eco\_I)

print("Reverse Camera: ", self.Reverse\_C)

from Car import Car, Axia, Bezza, Alza, MyVi

carList = []

peroduaList= []

def add(list, x):

list.append(x)

#### PROTON CARS

# Brand, Type, Model, Price

add(carList, Car("Proton","Sedan","Proton Saga 1.3", 41425))

add(carList, Car("Proton","Sedan","Proton Perdana 2.0", 110540))

add(carList, Car("Proton","Sedan","Proton Perdana 2.4", 134820))

add(carList, Car("Proton","Sedan","Proton Persona 1.6", 48200))

add(carList, Car("Proton","Sedan","Proton Prevé 1.6", 60610))

add(carList, Car("Proton","Hatchback","Proton Suprima S 1.6", 68110))

add(carList, Car("Proton","Hatchback","Proton Iriz 1.3", 45315))

add(carList, Car("Proton","MPV","Proton Exora 1.6", 65700))

add(carList, Car("Proton","MPV","Proton Ertiga 1.4", 62770))

### PERODUA

add(carList, Car("Perodua","Sedan","Bezza", 37500))

add(carList, Car("Perodua","MPV","Alza", 53565))

add(carList, Car("Perodua","Hatchback","Axia", 35660))

add(carList, Car("Perodua","Hatchback","Myvi", 46300))

### more details

add(peroduaList, Axia("Perodua","Hatchback","Axia", 24000, "Standard E", "Manual", 22.5, "No", "Fabric", "No"))

add(peroduaList, Axia("Perodua","Hatchback","Axia", 33720, "Standard G", "Manual", 22.5, "Radio and USB", "Fabric", "Yes"))

add(peroduaList, Axia("Perodua","Hatchback","Axia", 35660, "Standard G", "Auto", 21.6, "Radio and USB", "Fabric", "Yes"))

add(peroduaList, Axia("Perodua","Hatchback","Axia", 36635, "Standard E", "Manual", 22.5, "Radio and USB and Bluetooth", "SE Fabric", "Yes"))

add(peroduaList, Axia("Perodua","Hatchback","Axia", 38580, "Standard E", "Auto", 21.6, "Radio and USB and Bluetooth", "SE Fabric", "Yes"))

add(peroduaList, Axia("Perodua","Hatchback","Axia", 41500, "Standard E", "Auto", 21.6, "No", "Leather", "Yes"))

add(peroduaList, Bezza("Perodua","Sedan","Bezza", 35500, "GXTRA", 1.0, "Manual", 22.8, "Radio and USB and Bluetooth", "Fabric", "Yes"))

add(peroduaList, Bezza("Perodua","Sedan","Bezza", 37500, "GXTRA", 1.0, "Auto", 21.3, "Radio and USB and Bluetooth", "Fabric", "Yes + USB"))

add(peroduaList, Bezza("Perodua","Sedan","Bezza", 41400, "PREMIUM X", 1.3, "Manual", 21.7, "Radio and USB and Bluetooth", "Fabric", "Yes + USB"))

add(peroduaList, Bezza("Perodua","Sedan","Bezza", 43350, "PREMIUM X", 1.3, "Auto", 21, "Radio and USB and Bluetooth", "Fabric", "Yes + USB"))

add(peroduaList, Bezza("Perodua","Sedan","Bezza", 49200, "ADVANCE", 1.3, "Auto", 22, "Full with navigation", "Leather", "Yes + USB"))

add(peroduaList, Alza("Perodua","MPV","Alza", 50650, "1.5 S", "Manual", "Radio and USB and Bluetooth", "Fabric", "Without"))

add(peroduaList, Alza("Perodua","MPV","Alza", 53565, "1.5 S", "Auto", "Radio and USB and Bluetooth", "Fabric", "Without"))

add(peroduaList, Alza("Perodua","MPV","Alza", 54540, "1.5 SE", "Manual", "Radio and USB and Bluetooth", "SE Fabric", "With"))

add(peroduaList, Alza("Perodua","MPV","Alza", 57455, "1.5 SE", "Auto", "Radio and USB and Bluetooth", "SE Fabric", "With"))

add(peroduaList, Alza("Perodua","MPV","Alza", 62900, "1.5 Advance", "Auto", "Full with navigation", "Leather", "With"))

add(peroduaList, MyVi("Perodua","Hatchback","Myvi", 44300, "G", 1.3, "Manual", 20.5, "Radio and USB", "No", "No"))

add(peroduaList, MyVi("Perodua","Hatchback","Myvi", 46300, "G", 1.3, "Auto", 20.1, "Radio and USB", "No", "No"))

add(peroduaList, MyVi("Perodua","Hatchback","Myvi", 48300, "X", 1.3, "Auto", 21.1, "Radio and USB and Microphone", "Yes", "No"))

add(peroduaList, MyVi("Perodua","Hatchback","Myvi", 51800, "H", 1.5, "Auto", 20.1, "Radio and USB and Microphone", "Yes", "No"))

add(peroduaList, MyVi("Perodua","Hatchback","Myvi", 55300, "H", 1.5, "Auto", 20.1, "Full with navigation", "Yes", "Yes"))

def show():

for car in carList:

print("\nModel: ",car.name)

print("Type: ", car.car\_type)

print("Price: RM", car.price)

def run():

a = input("What is the brand you prefer? Proton / Perodua\n")

av = input("How much do you value this 1 - 5\n")

b = input("What is the car type that you prefer? Sedan / SUV / Hatchback\n")

bv = input("How much do you value this 1 - 5\n")

c = input("How much is your budget?\n")

cv = input("How much do you value this 1 - 5\n")

scores= []

for car in carList:

x = 0

if car.brand == a:

x += 1\* int(av)

if car.car\_type == b:

x += 1\* int(bv)

if car.price < int(c):

x += 1\* int(cv)

scores.append([car, x])

scores\_sorted= []

for i in range(len(scores)):

max= scores[len(scores)-1]

for j in range(len(scores)):

if max[1]< scores[j][1]:

max= scores[j]

scores\_sorted.append(max)

scores.remove(max)

names= []

for i in range(len(scores\_sorted)):

if i< 3:

scores\_sorted[i][0].show()

for j in range(len(peroduaList)- 1):

if peroduaList[j].name== scores\_sorted[i][0].name:

check= True

for s in names:

if s== peroduaList[j].name: check= False

if check:

names.append(peroduaList[j].name)

print("\*Adding")

if len(names)== 1:

check= input("The car "+ names[0]+ " Has more variants, would you like to see it ?\t")

if check== "yes":

for i in peroduaList:

if i.name == names[0]:

i.show()

elif len(names)== 2:

check= input("Two of the suggested cars have more variants, enter "+ names[0]+ " to see it's details or enter "+ names[1]+" ")

for i in peroduaList:

if i.name== check:

i.show()

elif len(names)== 3:

check= input("Three of the suggested cars have more variants, enter "+ names[0]+ " to see it's details or enter "+ names[1]+ " or enter "+ names[2]+" ")

for i in peroduaList:

if i.name== check:

i.show()

run()

wait = input("\n\nPress enter to exit")

**CONCLUSION**

In conclusion, frame is suitable to be the base of an expert system because it is much easier to store the information in the system with the implementation of inference engine. Besides, it can be developed by any of object oriented programming language such as Java.

Despite the advantages of using frame as a base of expert system, there is also the limitation. Since there are different slots and demons used, it will be difficult to define the hierarchical system and find the inheritance path.

**REFERENCES**

1. *The AI Dictionary.* Retrieved May 13, 2018 from<http://www.cse.unsw.edu.au/~billw/aidict.html>
2. *NA.* Retrieved April 18, 2018 from<http://www.proton-edar.com.my/en/models.aspx>
3. *Perodua Car - Bezza, Axia, Myvi, Alza Price Promotion Malaysia.* Retrieved April 18, 2018 from  
   http://www.perodua-car.com/v1/?gclid=Cj0KCQjwxN\_XBRCFARIsAIufy1ZLbfeOSh\_eqSsFoHIrbieW7kimJxE0gw-X\_LTU5MFJRwMd2hIa-eAaAjmBEALw\_wcB