Sekolah Pendidikan Profesional dan Pendidikan Berterusan (UTMSPACE)

SCSJ 2154
OBJECT ORIENTED PROGRAMMING

FURNITURE
SEMESTER 1 2015/2016

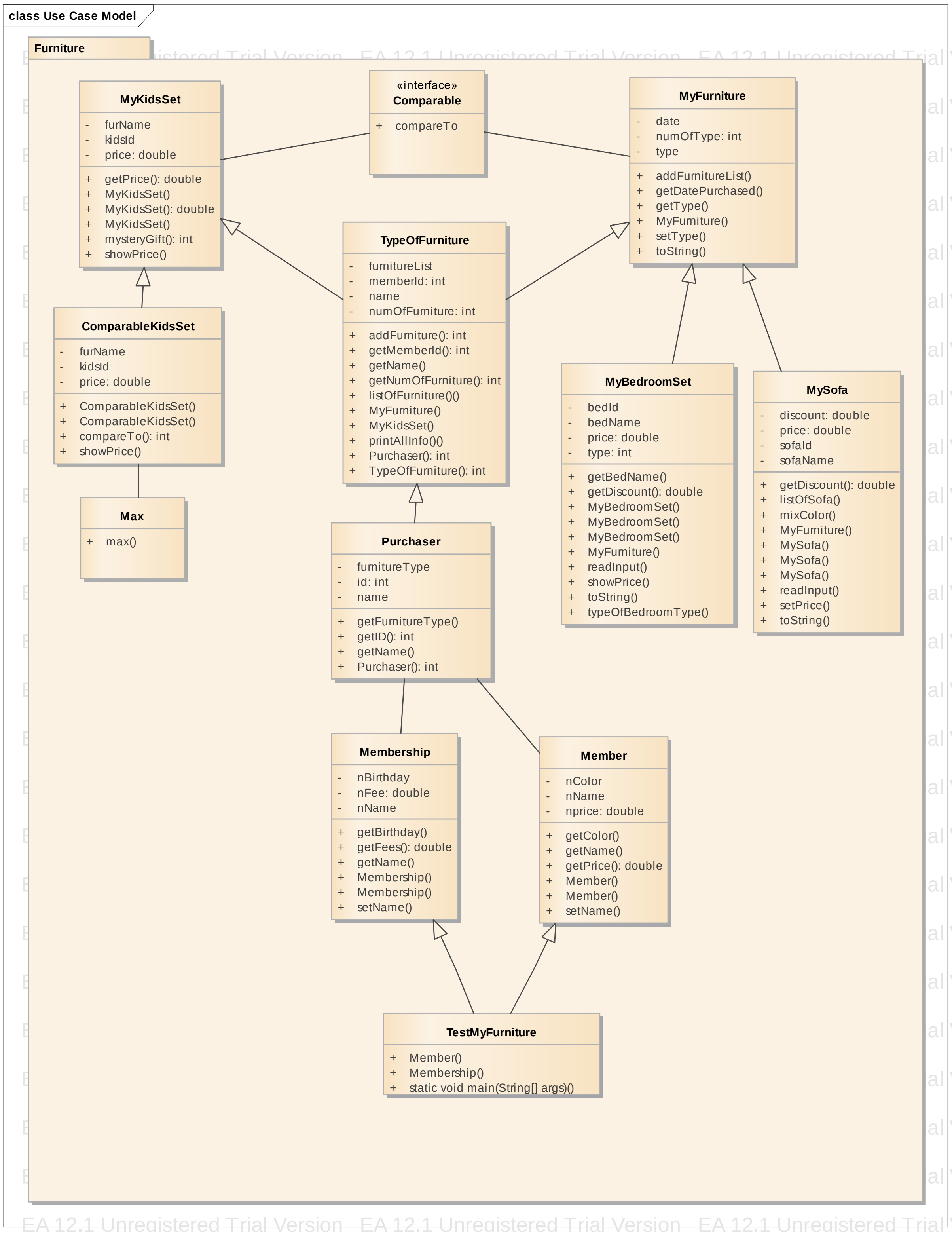
Lecturer Name: DR. ZALMIYAH BINTI ZAKARIA

Prepared by :

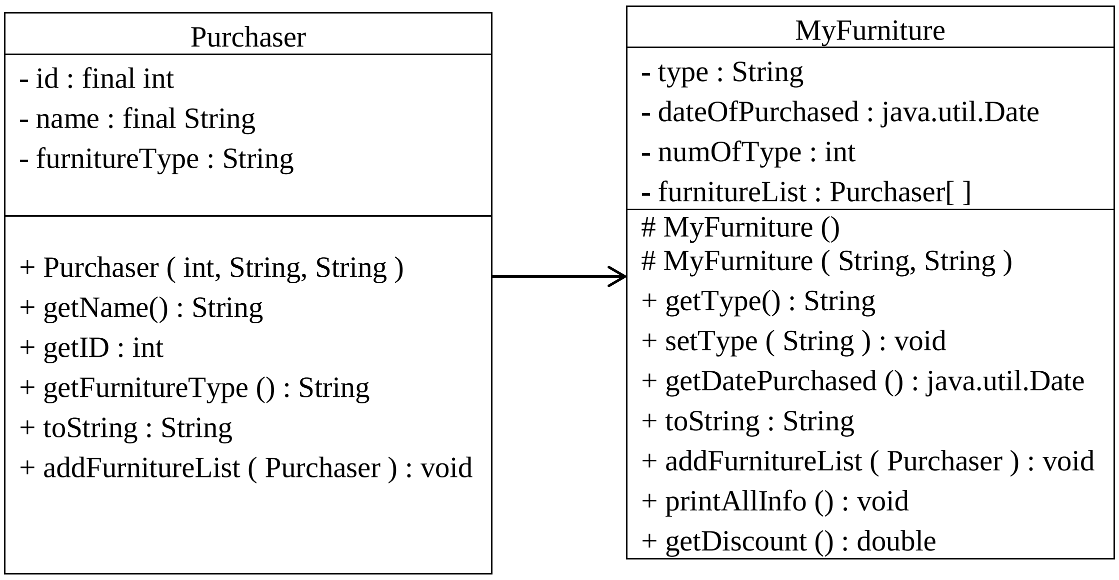
NO	NAME	MATRIC NO
1	Mardiana binti Abu Hassan	SX140047CSJS04
2	Nur Allyyaa binti Abd Halim	SX140055CSJS04
3	Nur Syuhada binti Zulkifli	SX140104SCSV04
4	Nurul Hidayah binti Anuar	SX140056CSJS04

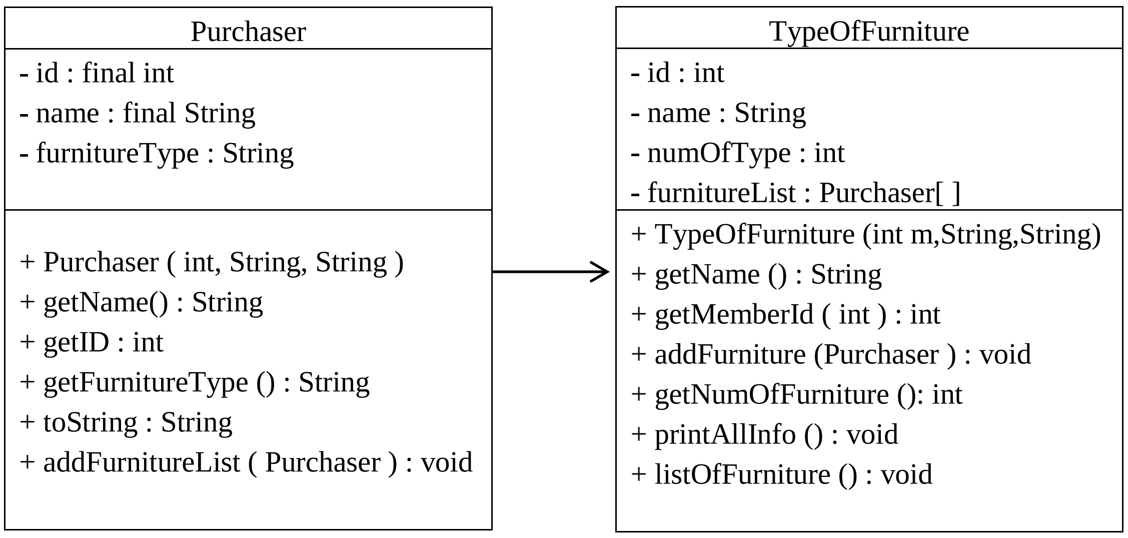
TABLE OF CONTENT

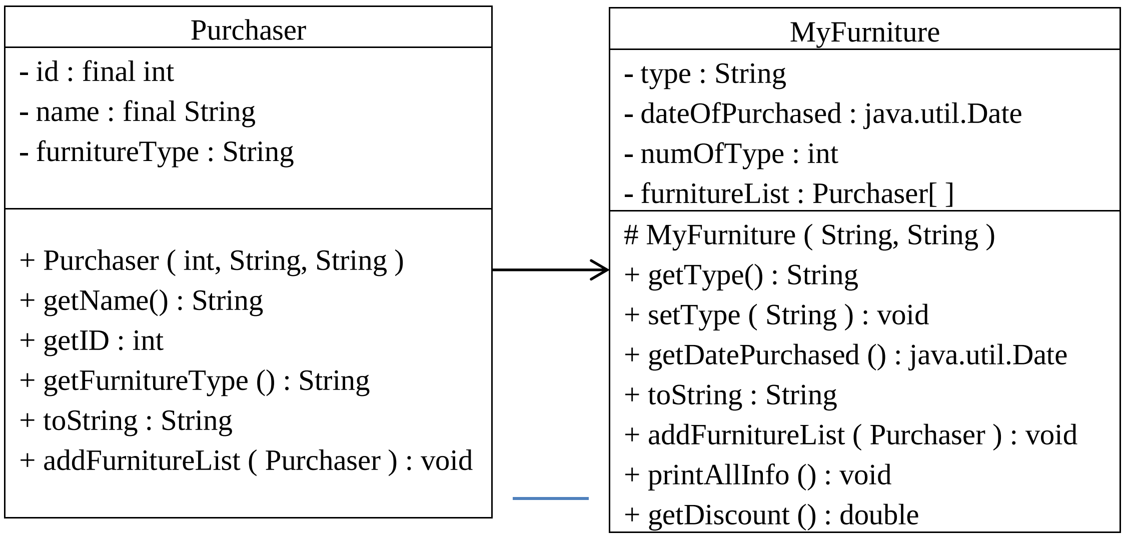
UML CLASS DIAGRAM



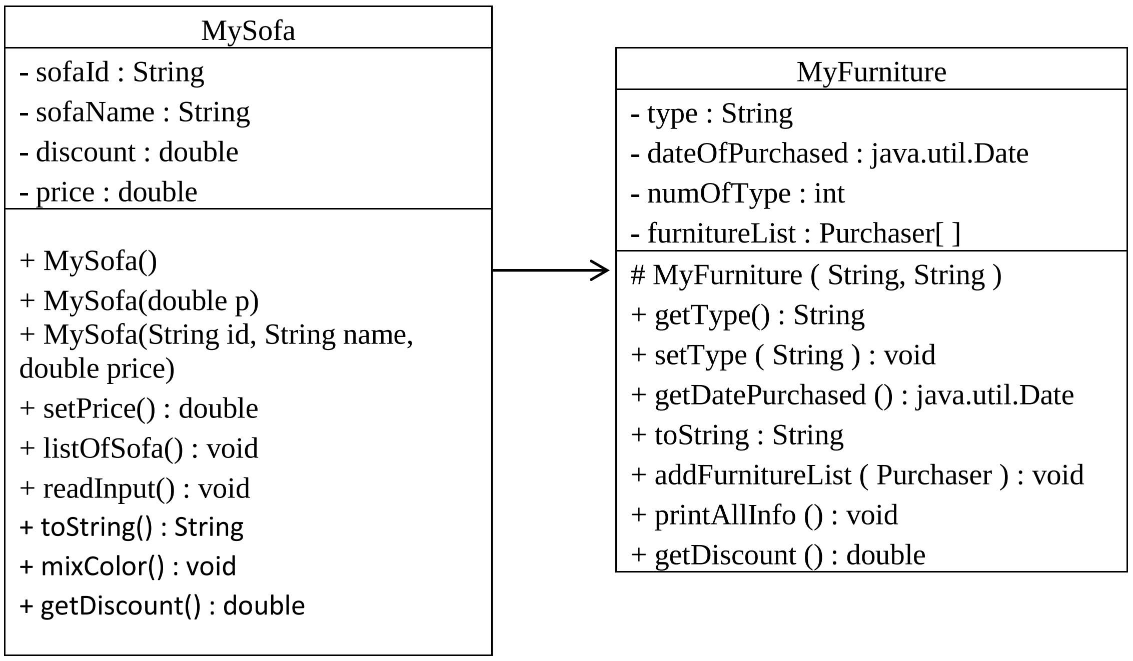
UML CLASS DIAGRAM

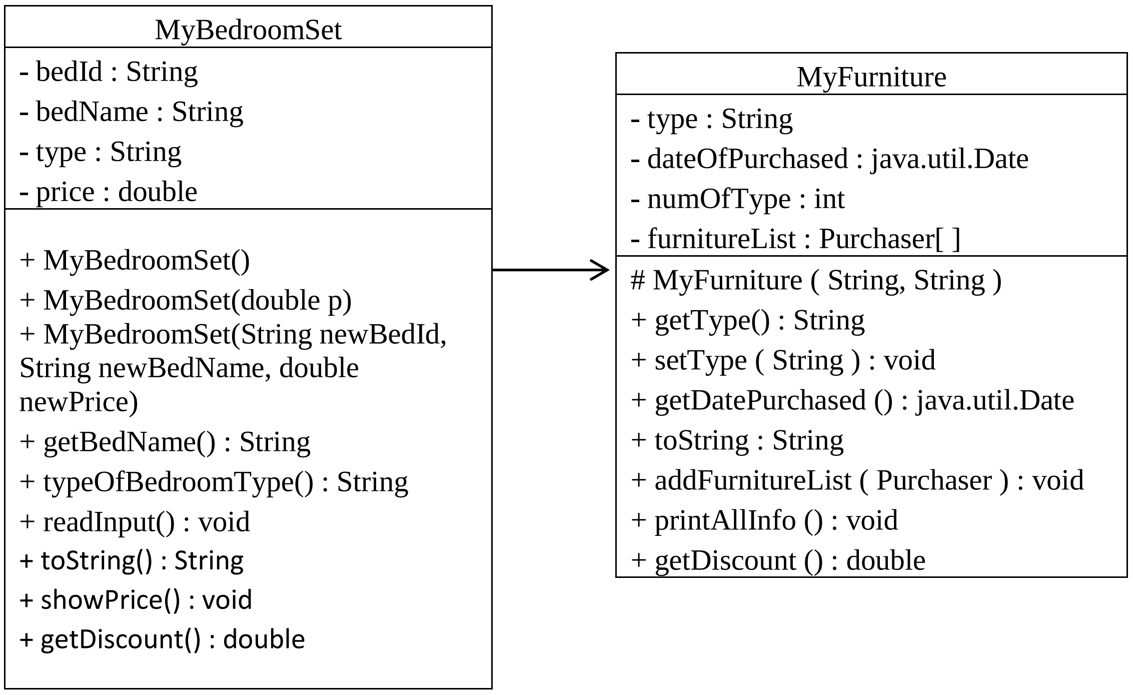

Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
MyFurniture
- type : String
- dateOfPurchased : java.util.Date
- numOfType : int
- furnitureList : Purchaser[ ]
# MyFurniture ()
# MyFurniture (String, String)
+ getType() : String
+ setType (String) : void
+ getDatePurchased () : java.util.Date
+ toString : String
+ addFurnitureList (Purchaser) : void
+ printAllInfo () : void
+ getDiscount () : double


Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
TypeOfFurniture
- id : int
- name : String
- numOfType : int
- furnitureList : Purchaser[ ]
+ TypeOfFurniture (int m,String,String)
+ getName () : String
+ getMemberId (int) : int
+ addFurniture (Purchaser) : void
+ getNumOfFurniture (): int
+ printAllInfo () : void
+ listOfFurniture () : void


Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
MyFurniture
- type : String
- dateOfPurchased : java.util.Date
- numOfType : int
- furnitureList : Purchaser[ ]
# MyFurniture (String, String)
+ getType() : String
+ setType (String) : void
+ getDatePurchased () : java.util.Date
+ toString : String
+ addFurnitureList (Purchaser) : void
+ printAllInfo () : void
+ getDiscount () : double

UML CLASS DIAGRAM





MyKidsSet
- kidsId : String
- furName : String
- price : double
+ MyKidsSet()
+ MyKidsSet(double p)
+ MyKidsSet(String newId, String newFurName, double newPrice)
+ getPrice() : double
+ typeOfBedroomType() : String
+ mysteryGift() : void
+ showPrice() : String

MyKidsSet
- kidsId : String
- furName : String
- price : double
+ MyKidsSet()
+ MyKidsSet(double p)
+ MyKidsSet(String newId, String newFurName, double newPrice)
+ getPrice() : double
+ typeOfBedroomType() : String
+ mysteryGift() : void
+ showPrice() : String

Comparable <interface>
+ compareTo(ComparableKidsSet o)

CLASS EXPLAINATION

1. <u>*MyFurniture Class*</u>

   MyFurniture defined as the superclass for *MySofa* and *MyBedroomSet*. *MyFurniture* models common features of furniture. Both *MySofa* and *MyBedroomSet* contains *getDiscount()* methods to calculate the discount given by the GA Sofa Company. *MyFurniture* is an *abstract* class which the discount can be calculated in both *MySofa* and *MyBedroomSet* class; *getDiscount*() method has been defined in *MyFurniture* class because its implementation depends on the specific price that have been initialized to each furniture type. Such methods are referred to as *abstract* methods and are denoted using abstract modifier in the method header.

   public abstract double getDiscount();

   Once the methods has been defined in *MyFurniture*, this class will become an *abstract* class and will be denoted using the *abstract* modifier in the class header :

   public abstract class MyFurniture { }

   This class declared *type* and *numOfType* as variable which is *type* is from *String* data type while *numOfType* is an *Integer* data type. This class also declare *datePurchased* as variable and used a *Package* from *Date* which it is a class to represent a specific instant in time with millisecond precision when the *Purchaser* bought their furniture. Finally, this class also create an array named *furnitureList* which hold a number of purchaser and there is a one to one relationship (*Association*) between two classes : MyFurniture and Purchaser

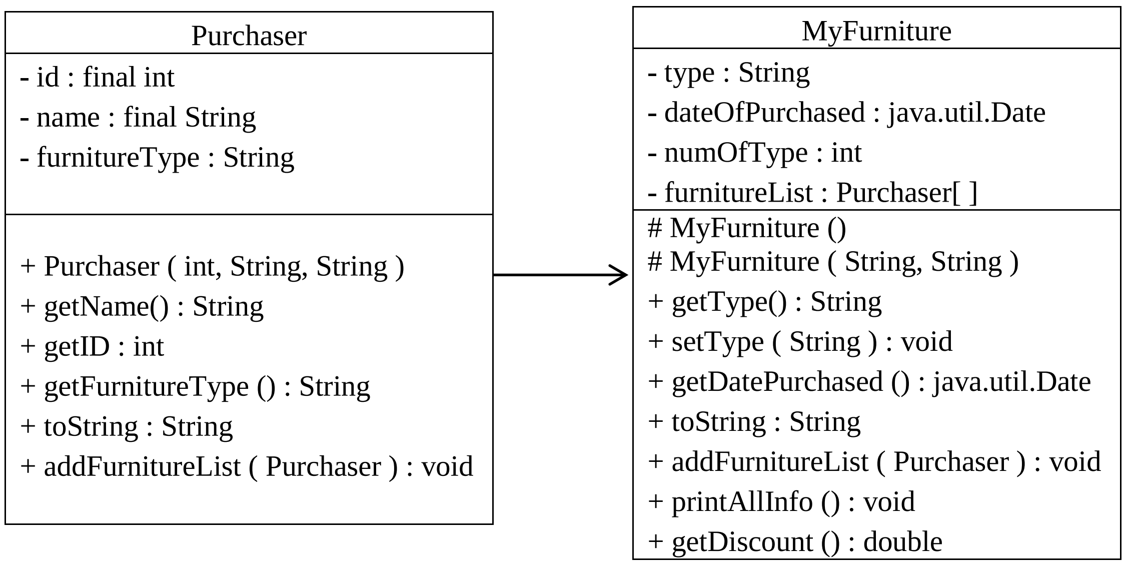

Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
MyFurniture
- type : String
- dateOfPurchased : java.util.Date
- numOfType : int
- furnitureList : Purchaser[ ]
# MyFurniture ()
# MyFurniture (String, String)
+ getType() : String
+ setType (String) : void
+ getDatePurchased () : java.util.Date
+ toString : String
+ addFurnitureList (Purchaser) : void
+ printAllInfo () : void
+ getDiscount () : double

Figure 1.0

MyFurniture class have an overloading constructor which the default is protected constructor and protected constructor with two parameter which both are String data type to construct objects with different initial data values. It also has mutator method *getType ()* with String data type that will return value type of furniture. Accessor, *setType()* is used to set the furniture and cannot return any value. This class is also declare *getDatePurchased ()* method which used a java package called Date which represents a specific instant in time with millisecond precision when is the purchasing is happen. *toString ()* method is used to return a string and display the time of purchase.

addFurnitureList (Purchaser) is showing where the association class relationship between *MyFurniture* and *Purchaser* is happen. This method will add the type of furniture that the purchaser wishes to buy.

```
public class MyFurniture {
public void addFurnitureList(Purchaser p1) {
      furnitureList[numOfType]= p1;
      numOfType++;
```

For *printAllInfo ()* method, this method will list all type of furniture. This method also called *typeOfFurniture ()* method which return String.

```
public void printAllInfo(){
      for(int i = 0;i<numOfType; i++){
            Purchaser s = (Purchaser)furnitureList[i];
            System.out.println((i+1) + ". " +
s.getFurnitureType() );
        }

Output :

1. Sofa Set
2. Bedroom Set          }— List of Furniture
3. Kids Set
```

The last method that has been declare in *MyFurniture* class is a double primitive type named *getDiscount ()* method. This is an abstract method which is we only declare the method in parent and will define the whole process to calculate discount in their subclass which is MySofa and MyBedroomSet

2. <u>*Purchaser Class*</u>

   Purchaser was built with three two primitive data type in three variables which is *id, name and furtnitureType*. This is a concrete class where it has only one constructor three set of assessor method.

   The constructor contains three parameter with String and Integer primitive data type. The three data attributes declare in *Purchaser* class are constants and have been named as *id, name and furnitureType*. This class also have three accessor mothod which is *getName ()* to return String, *getID ()* to return Integer and *get furnitureType ()* to return String.

   Purchaser class has one to one class relationship between *MyFurniture* and *TypeOfFurniture* called *Association*

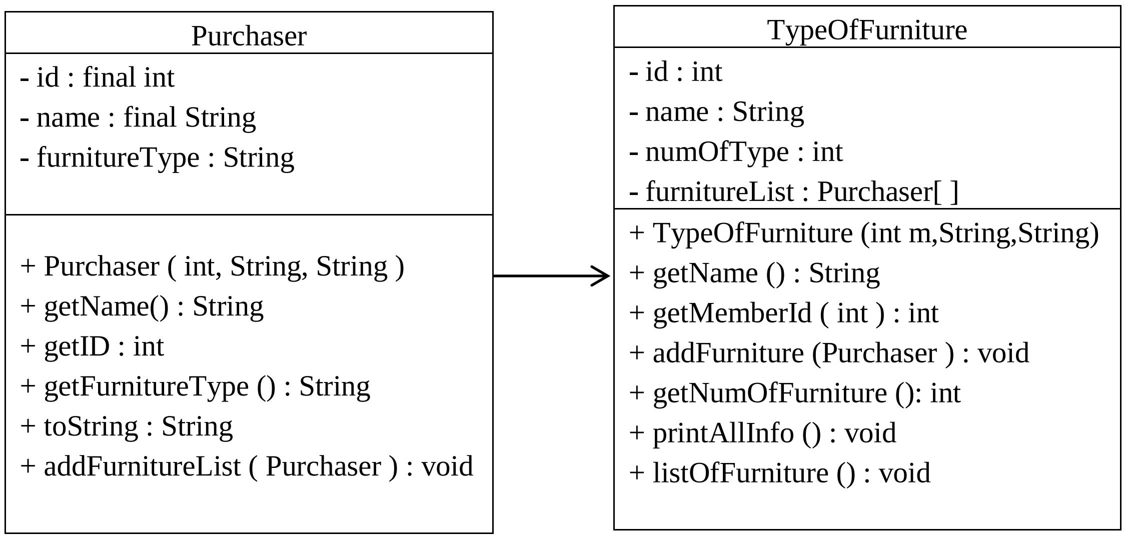

Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
TypeOfFurniture
- id : int
- name : String
- numOfType : int
- furnitureList : Purchaser[ ]
+ TypeOfFurniture (int m,String,String)
+ getName () : String
+ getMemberId (int) : int
+ addFurniture (Purchaser) : void
+ getNumOfFurniture (): int
+ printAllInfo () : void
+ listOfFurniture () : void

Figure 1.

```
public class TypeOfFurniture {

        public void addFurniture(Purchaser p1){
              furnitureList[numOfFurniture]=p1;
              numOfFurniture++;

        public void printAllInfo(){
       .
       .
       for(int i = 0;i<numOfFurniture; i++){
              Purchaser s = (Purchaser)furnitureList[i];
              System.out.println((i+1) + ". " + s.getName());
       }
```

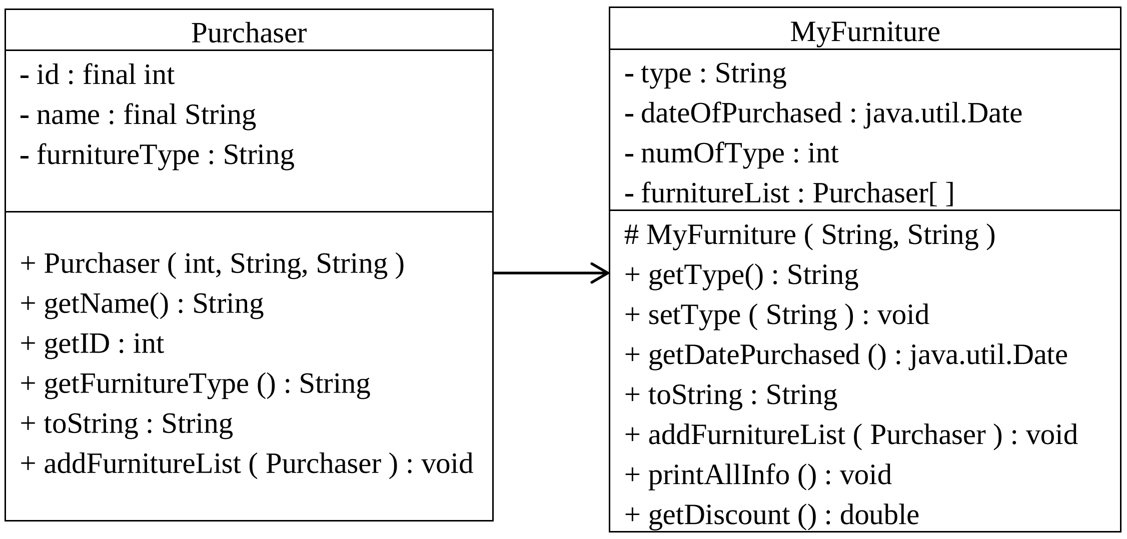

Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
MyFurniture
- type : String
- dateOfPurchased : java.util.Date
- numOfType : int
- furnitureList : Purchaser[ ]
# MyFurniture (String, String)
+ getType() : String
+ setType (String) : void
+ getDatePurchased () : java.util.Date
+ toString : String
+ addFurnitureList (Purchaser) : void
+ printAllInfo () : void
+ getDiscount () : double

Figure 1.2

```
public class MyFurniture {
      public void addFurnitureList(Purchaser p1) {
            furnitureList[numOfType]= p1;
            numOfType++;
      }
 }

public void printAllInfo(){
        for(int i = 0;i<numOfType; i++){
            Purchaser s = (Purchaser)furnitureList[i];
            System.out.println((i+1) + ". " +
               s.getFurnitureType() );
        }
 }
```

3. *TypeOfFurniture*

   *Ty*peOfFurnitures is a solid class written with four types of variables with different type of data type. It use String, integer, java Date package and a constant array named furnitureList which is related with *Purchaser*. This class contains a constructor with parameters, an acsessor and another two methods.

   The constructor with arguments consist an *Integer* and *String* data type known as *name, memberId* and *furnitureList.* Acsessor method which is *getName (), getMemberId* and *getNumOfFurniture* should be able to allows the other class to view the content of an attributes.

   TypeOfFurniture class has one to one class relationship between *MyFurniture* and *Purchaser*. This type of class relationship known as *Association*.

   Method *printAllInfo ()* will display all details of the furniture which are using *for looping* to allows the codes can be repeated as long as the *Boolean Expression* is true.

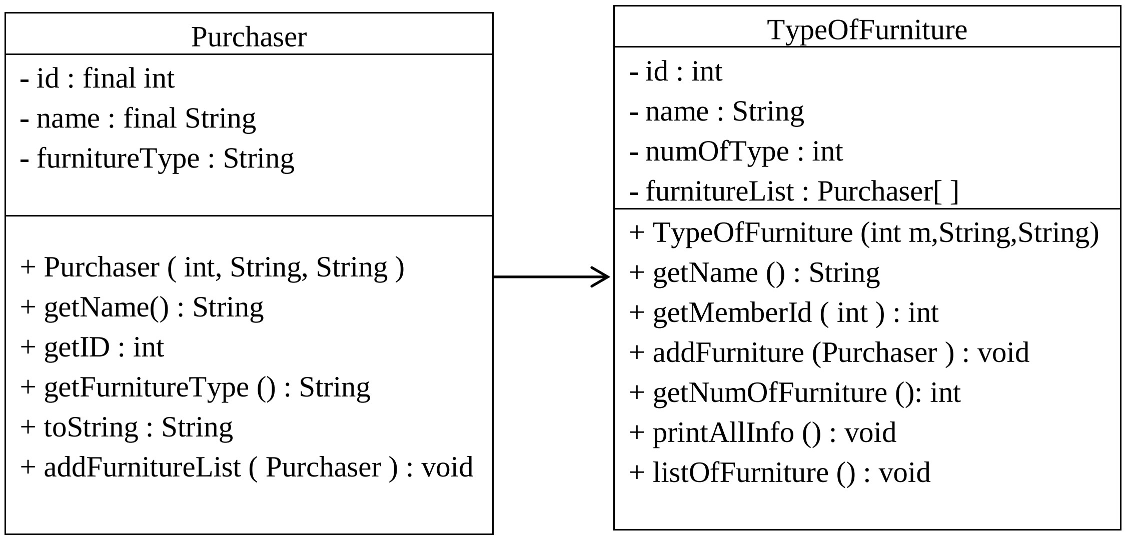

Purchaser
- id : final int
- name : final String
- furnitureType : String
+ Purchaser (int, String, String)
+ getName() : String
+ getID : int
+ getFurnitureType () : String
+ toString : String
+ addFurnitureList (Purchaser) : void
TypeOfFurniture
- id : int
- name : String
- numOfType : int
- furnitureList : Purchaser[ ]
+ TypeOfFurniture (int m,String,String)
+ getName () : String
+ getMemberId (int) : int
+ addFurniture (Purchaser) : void
+ getNumOfFurniture (): int
+ printAllInfo () : void
+ listOfFurniture () : void

Figure 1.3

```
public class TypeOfFurniture {

        public void addFurniture(Purchaser p1){
              furnitureList[numOfFurniture]=p1;
              numOfFurniture++; }

        public void printAllInfo(){
              for(int i = 0;i<numOfFurniture; i++){
```

```
                        Purchaser s = (Purchaser)furnitureList[i];
                        System.out.println((i+1) + ". " + s.getName()); }
```

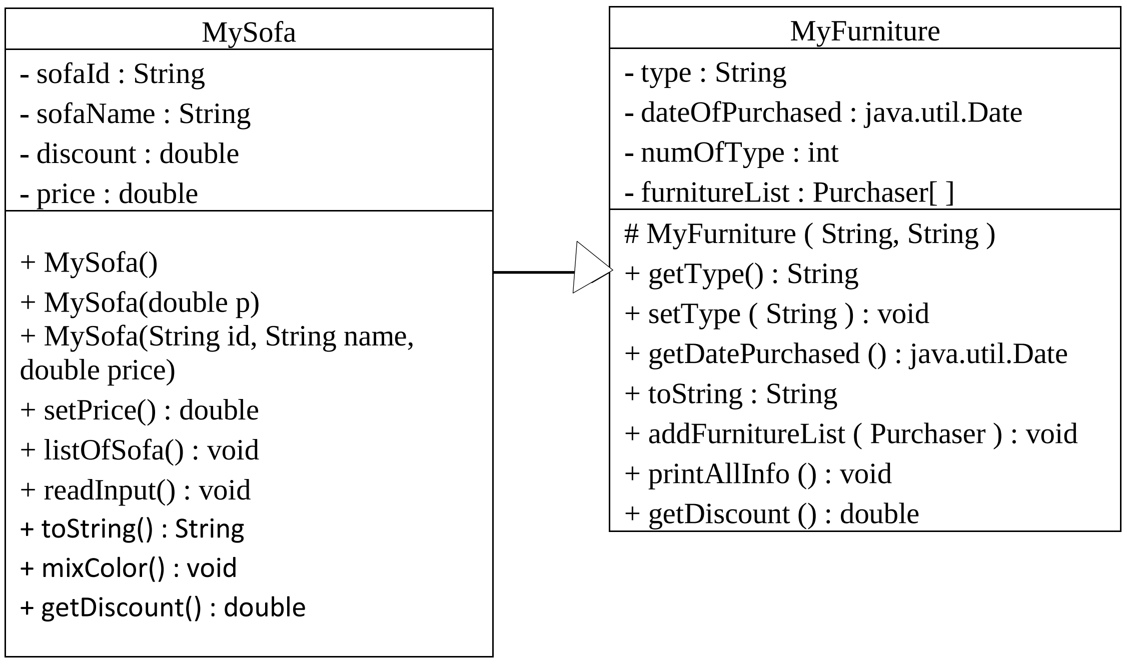
4. <u>*MySofa Class*</u>

   MySofa was defined as the subclass for *MyFurniture* written with four types of variables with different type of data type. It contains *getDiscount()* methods for calculate discount given by the GA Sofa Company. *MySofa* was an *child* class of *MyFurniture* which is since the discount can be calculated in its class; *getDiscount*() method has been define in *MyFurniture* class because its implementation depends on the specific price that have been initialize to every furniture type. Such methods are referred to as *abstract* methods from superclass and are denoted using abstract modifier in the method header.

```
public double getDiscount()      //Abstract Method
{
//double newDiscount;

//System.out.println("\nYou get 10% discount");
//newDiscount = price - ( price * 0.05 );
//System.out.println("You have to paid : RM " + newDiscount);

return price -(price*0.10);
}
```

MySofa class have default constructor and constructor overloading as below:

```
public MySofa()

    {
        sofaId = "";
        sofaName = "";
        price = 0.0;
    }

    public MySofa(double p)

    {
        price = p;
    }


    public MySofa(String id, String name, double price)

    {
        sofaId = id;
        sofaName = name;
        price = price;
    }
```



The method *listOfSofa() is used to display the sofa list name that have been chosen by the buyer.*

```
public void listOfSofa()

    {
        JDialog.setDefaultLookAndFeelDecorated(true);
        System.out.print("\n");
        Object[] selectionValues = { "GA233 Lousiana", "GA235 
Gorgia", "GA236 Ottava" };
        String initialSelection = "GA233 Louisiana";
        Object selection = JOptionPane.showInputDialog(null, 
"Your Favourite Sofa Set?",
                "Your Favourite Sofa Set", 
JOptionPane.QUESTION_MESSAGE, null, selectionValues, 
initialSelection);
        System.out.println("Your choosen Sofa Set is : " + 
selection);
}
```

Method *readInput()* is used to accept user input.

```
public void readInput()

{
    Scanner input = new Scanner(System.in);

    System.out.print("Enter sofa's price : ");
    price = input.nextDouble();

}
```

Method *toString()* is used to print the output by returning the string.

```
public String toString() {
  return "Purchased on " + datePurchased;}
```

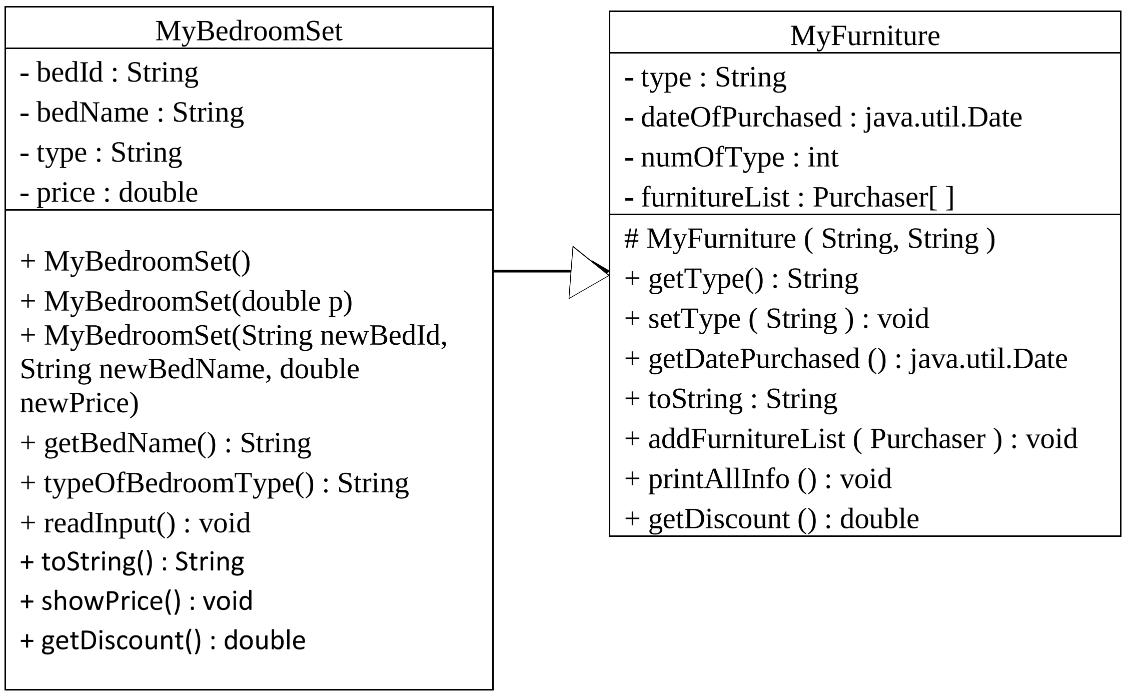
5. *MyBedroomSet Class*

MyBedroomSet was defined as the subclass for *MyFurniture* written with four types of variables with different type of data type. It contains *getDiscount()* methods for calculate discount given by the GA Sofa Company. *MyBedroomSet* was an *child* class of *MyFurniture* which is since the discount can be calculated in its class; *getDiscount*() method has been define in *MyFurniture* class because its implementation depends on the specific price that have been initialize to every furniture type. Such methods are referred to as *abstract* methods from superclass and are denoted using abstract modifier in the method header.

```
public double getDiscount() {
            return price -(price*0.05);
        }
}
```

MyBedroomSet class have default constructor and constructor overloading as below:

```
public MyBedroomSet()
       {
             bedId = "";
             bedName = "";
             price = 0.0;
       }


        public MyBedroomSet(double p) {
            price = p;
        }

       public MyBedroomSet(String newBedId, String newBedName,
double newPrice)
       {
            bedId = newBedId;
            bedName = newBedName;
            price = newPrice;
       }
```


MyBedroomSet
- bedId : String
- bedName : String
- type : String
- price : double
+ MyBedroomSet()
+ MyBedroomSet(double p)
+ MyBedroomSet(String newBedId, String newBedName, double newPrice)
+ getBedName() : String
+ typeOfBedroomType() : String
+ readInput() : void
+ toString() : String
+ showPrice() : void
+ getDiscount() : double
MyFurniture
- type : String
- dateOfPurchased : java.util.Date
- numOfType : int
- furnitureList : Purchaser[ ]
# MyFurniture (String, String)
+ getType() : String
+ setType (String) : void
+ getDatePurchased () : java.util.Date
+ toString : String
+ addFurnitureList (Purchaser) : void
+ printAllInfo () : void
+ getDiscount () : double

6. *MyKidsSet Class*

MyKidsSet is an abstract class and was a superclass to *ComparableKidsSet* class. This class was declare three set of data attribute which is *kidsId, furName* and *price* where *kidsId* and *furName* is a String while *price* is double.

MyKidsSet class have a few list of methods that have been used such as a constructors, acsessor and the other method that will discuss later. This class also has an implementation by using an *Interface*.

MyKidsSet () is no-argument constructor where all values of the three variables that have been declared is set to default, in which Java will set all of the object's numeric variables to 0 and reference variable to the special value null. This class used constructor with one argument and three argument. This type of constructors known as an *Overloading Constructor.*

- public MyKidsSet()
- public MyKidsSet(double p)
- public MyKidsSet(String newId, String newFurName, double newPrice)

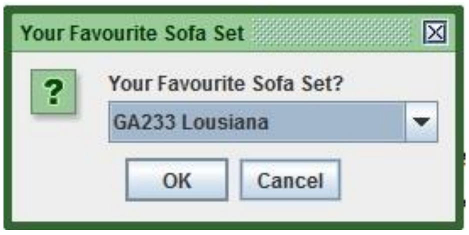
getPrice () was declared as *double* return type and also used keyword *this* which is referred to the current instance of the method.

showPrice () method is able to request input from keyboard by using *Scanner* class that have been provided by Java. Types of choosen kids set and prices will be displayed based on the kid's set id entered using *switch case*.

```
public String showPrice()
{
    Scanner input = new Scanner(System.in);
         :
         :
    switch(kidsId)
    {
        case("IK001"):
                price = 45.00;
                System.out.print("Price is RM :" + price);
                break;
         :
         :
}
```

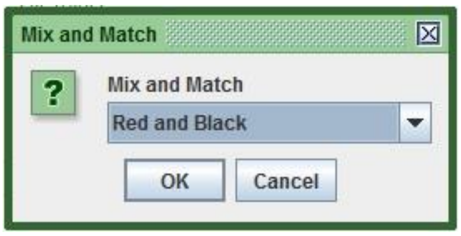
The *mysteryGift ()* method is using a package that have been provided by Java which is JDialog and JOpitonPane. User can choose their sofa set and colors. This method also use constant array to display both sofa and colors.

```
public void mysteryGift()
    {
        JDialog.setDefaultLookAndFeelDecorated(true);
        System.out.print("\n");
        System.out.print("Congrat's. You're entitled to get a
             mystery gift from us!!!");

        Object [] selectionValues = {"Little Tayo", "Barbie",
"Thomas & Friends", "Roborcar Poli"};
        String initialSelection = "Key Chain";
        Object selection = JOptionPane.showInputDialog(null,
"Mystery Gift",
            "Choose Your Gift", JOptionPane.QUESTION_MESSAGE,
null, selectionValues, initialSelection);
        System.out.println("\nYou have choose : " +
selection);
    }
```


Your Favourite Sofa Set

Your Favourite Sofa Set?

GA233 Lousiana


Mix and Match

Mix and Match

Red and Black

compareTo () method is an abstract method and was implemented using *Comparable Inteface*

public abstract int compareTo(ComparableKidsSet o);

7. *Comparable Interface*

   Comparable Interface is an interface for comparing object that have been defined in java.lang. This is accomplished by letting the class for the object implement this interface using the *implements* keyword. The relationship between the class and the interface is known as *interface inheritance*.

```
package furniture;

public interface Comparable {
    public int compareTo(ComparableKidsSet o);

}
```

   The *compareTo ()* method determines the order of this object with the specified object *o* and returns a negative integer, zero or positive integer if this object is less than, equal or more greater than *o*.

8. *ComparableKidsSet Class*

   Comparable extends *MyKidSet* implements *Comparable* as shown below inherits all the contants from the *Comparable* interface and implements the methods in the interface.

   The *compareTo ()* method compares the price of the two kids set. An instance of *ComparabeKidsSet* is also an instance of *MyKidsSet and Comparable.*

```
public class ComparableKidsSet extends MyKidsSet implements
Comparable {
    :
    :
    public ComparableKidsSet() {}

    public ComparableKidsSet(ComparableKidsSet o1) {

    }
    public int compareTo(ComparableKidsSet o) {
        if(getPrice() > ((ComparableKidsSet)o).getPrice())
            return 1;
        else if(getPrice() <
             ((ComparableKidsSet)o).getPrice())
            return -1;
        else
            return 0;
    }
```

showPrice () method is able to request input from keyboard by using *Scanner* class that have been provided by Java. Types of choosen kids set and prices will be displayed based on the kid's set id entered using *switch case*.

9. <u>*Max Class*</u>

In *Max* class, *o1* is declared as *ComparableKidsSet* and *(Comparable)o1* tells the compiler to cast *o1* into *Comparable* so that the *compareT*o method can be invoked from o*1*.

```
public static ComparableKidsSet max(ComparableKidsSet
o1,ComparableKidsSet o2){
        if(((ComparableKidsSet)o1).compareTo(o2) > 0){
             System.out.println("\nYour 1st item is more
             expensive then 2nd item");
             return o1;
         }
     else{
         System.out.print("\nYour 2nd item is more cheaper.
             Enjoy!!");
         return o2;
       }
    }
```

10. <u>*Member Class*</u>

Member is a solid class written with three types of variables with different type of data type. It use String, double and related to a constant *ArrayList* named stud in *TestMyFurniture*. This class contains a constructor with parameters, an accessor and another two methods.

The constructor with arguments consist an *String* and *double* data type known as *name, color* and *price.* Accessor method which is *getName (), getColor* and *getPrice* should be able to allows the other class to view the content of an attributes.

11. <u>*Membership Class*</u>

    Membership is a solid class written with three types of variables with different type of data type. It use String, double and related to a constant *ArrayList* named ship in *TestMyFurniture*. This class contains a constructor with parameters, an accessor and another two methods.

    The constructor with arguments consist a *String* and *double* data type known as *nName, nBirth* and *nFee.* Accessor method which is *getName (), getBirthday* and *getFees* should be able to allows the other class to view the content of an attributes.

12. <u>*TestMyFurniture Class*</u>

    TestMyFurniture class is the driver application for Furniture Application System. All of methods that have written from all different classes will invoke here.

    The *ArrayList* is used in here to store unlimited number of furniture and membership for this system.

```
ArrayList<Member> stud = new ArrayList<>();
ArrayList<Membership> ship = new ArrayList<>();
```

Boolean control is also been used in this system to let the system to called *displayMenu ()* method in both for Member's and Furniture's in which the system will display menu first to let the user choose the menu as long as the *boolean control* is *true.* Once the user has chosen their menu, then the chosen menu number will display.

```
public static int displayMenu(){
        Scanner selection = new Scanner(System.in);
        System.out.println("=========================");
        System.out.println("GA Sofa Furniture Details");
        System.out.println("=========================");
        System.out.println("1 - Add New Furniture");
        System.out.println("2 - Remove Furniture");
        System.out.println("3 - Add Furniture to Specific
             Index");
        System.out.println("4 - Display Furniture
             Information");
        System.out.println("0 - Exit");
        System.out.println("=========================");
        System.out.print("Your Selection : ");
        int select = selection.nextInt();
        System.out.println();

        return select;
    }
```

The *if – else statement* below will execute one group of statement based on user selection. The methods in each *if – else statement* will be called and the values of the methods will be assigned to the new *Member* object which is *mem1*. Finally the *ArrayList* for *Member* class will append new furniture in the array. The process is all same for creating new furniture into the specific index and to remove specific index funriture's and finally will displayed it on the table.

```
boolean control = true;
while(control){
      if(userSelection==1){
            System.out.println("*******************************");
            System.out.println("\t\tEnter New Data ");
            System.out.println("*******************************");
            Scanner option = new Scanner(System.in);
            System.out.print("How many set of furniture you want to
                  add:");
            int op = option.nextInt();
            for(int i = 0;i<op;i++){
                  Member mem1 = input(); //Create new object
                  stud.add(mem1); //ArrayList | Append new elements
            }
       }}
       :
       :

      private static Member input() {
            Scanner input = new Scanner(System.in);
            System.out.println("*******************************");
            System.out.println("\t\tEnter your details ");
            System.out.println("*******************************");
            System.out.print("Enter furniture name :");
            String name = input.nextLine();
            System.out.print("Enter color :");
            String color = input.nextLine();
            System.out.print("Enter price :");
            double price = input.nextDouble();
            Member mem = new Member(name, color, price);
            System.out.println();
            return mem;

      }
```

The *if – else statement* below will execute one group of statement based on user selection. The methods in each *if – else statement* will be called and the values of the methods will be assigned to the new *Membership* object which is *ship1*. Finally the *ArrayList* for *Membership* class will append new member in the array. The process is all same for creating new member into the specific index and to remove specific index member's and finally will displayed it on the table.

```
boolean control = true;
while(control1){
      if(userSelection1==1){
            System.out.println("********************************");
            System.out.println("\t\tEnter New Data ");
            System.out.println("********************************");
            Scanner option = new Scanner(System.in);
            System.out.print("How many set of furniture you want to
                  add:");
            int op = option.nextInt();
            for(int i = 0;i<op;i++){
                  Memberhip ship1 = input1(); //Create new object
                  stud.add(mem1); //ArrayList | Append new elements
            }
       }}
       :
       :

      private static Membership input1() {
            Scanner input = new Scanner(System.in);
            System.out.println("********************************");
            System.out.println("\t\tEnter your details ");
            System.out.println("********************************");
            System.out.print("Enter member name :");
            String name = input.nextLine();
            System.out.print("Enter Member Birthday :");
            String color = input.nextLine();
            System.out.print("Enter Member Fees Registration :");
            double price = input.nextDouble();
            Member sh = new Membership(name, birth, fees);
            System.out.println();
            return sh;

      }
           for(int i = 0;i<a;i++){

               Membership ship1 = input1();
               ship.add(ship1);

           }

       }
```

This explanation is about how to manage the chosen furniture by members. *MySofa* and *MyBedroom* will be creating new object while *MyFurniture* will creating new object and pass it by value to *MyBedroomSet* and *MySofa*.

```
MySofa ms1 = new MySofa();
MyBedroomSet b1 = new MyBedroomSet();

MyFurniture object2 = new MyBedroomSet(5000);
MyFurniture object1 = new MySofa(10000);
```

The codes below explain how an association relationship between *Purchaser* and *TypeOfFurniture* is implemented.

```
Purchaser p1 = new Purchaser(001, "Mardiana", "Kids Set");
Purchaser p2 = new Purchaser(002, "Syuhada", "Bedroom Set");
Purchaser p3 = new Purchaser(003, "Allyyaa", "Sofa Set");

TypeOfFurniture s1 = new TypeOfFurniture(001,"Emelda", "Sofa
      Set");
s1.addFurniture(p3);
s1.addFurniture(p2);
s1.printAllInfo();
```

When invoking *displayMyFurnitureObject ()* the method *getDiscount () defined* in *MySofa* and *MyBedroomSet* class are used. Method *Overriding* also implemented to override from *MyFurniture* class.

```
public static void displayMyFurnitureObject(MyFurniture
object) {
    System.out.printf("Total Price After Discount Is RM %.2f
             ", object.getDiscount() );
}

System.out.println("\n" +ms1.toString());
```

When invoking *equalDiscount (object1,object2)* the *getDiscount ()* method defined in *MySofa and MyBedroomSet* class is used for *object1.getDiscount.* Since the discount given by the company was different, the *getDiscount()* defined in *MyBedroomSet* class is used for *object2.getDiscount ()* .

```
equalDiscount(object1,object2));
```

The *totalPaid (object1,object2)* was defined in main used to calculate total between the two price

```
public static double totalPaid(MyFurniture object1,
      MyFurniture object2) {
            return object1.getDiscount() +
            object2.getDiscount();
}
```

The *equalDiscount (MyFurniture object1, MyFurniture object2)* was defined in main and was implemented to get the differentiate between the two price and will return bolean true or false.

```
public static boolean equalDiscount(MyFurniture object1,
      MyFurniture object2) {
            return object1.getDiscount() ==
            object2.getDiscount();
}
```

This application was used *Comparable Interface* to compare price between to items in *KidsSet* class. In order to accomplish this, the *Comparable Interface* was defined in the *Comparable* class. The *compareTo ()* method determines the order of this object with the specified object *o* and return a negative interger, zero or positive integer if this object is less then, equal or greater than *o*.

```
public interface Comparable {
    public int compareTo(ComparableKidsSet o);

}
```

A generic *max* method was defined in *Max* class to tells the compiler to cast *o1* into *Comparable* so that the *compareTo ()* method can be invoked from *o1*.

```
public class TestMyFurniture {
      public static void main (String[] args) {
            :
            :
            System.out.println(Max.max(kids1, kids2));
      }
}
```

```
public class Max {
        public static ComparableKidsSet max(ComparableKidsSet
             o1,ComparableKidsSet      o2){
        if(((ComparableKidsSet)o1).compareTo(o2) > 0){
             System.out.println("\nYour 1st item is more
             expensive then 2nd item");
             return o1;
        }
        else{
            System.out.print("\nYour 2nd item is more cheaper.
             Enjoy!!");
            return o2;
        }
    }

}
```

SOURCE CODES

MyFurniture Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.text.DecimalFormat;

public abstract class MyFurniture {
    private String type;
    final java.util.Date datePurchased;
    private int numOfType;
    private final Purchaser[] furnitureList;

    /** Construct a default furniture object */
    protected MyFurniture() {
        datePurchased = new java.util.Date();
        furnitureList = new Purchaser[5];

    }

    /** Construct a furniture object with the specified type and choice
     * @return
     * @param type
     * @param c
     * @param numOfType
     */
    protected MyFurniture(String type, String c) {
        datePurchased = new java.util.Date();
        this.type = type;
        furnitureList = new Purchaser[5];
    }

    /**Return type
     * @return */
    public String getType() {
        return type;
    }

    /** Set a new type of furniture
     * @param type */
    public void setType(String type) {
        this.type = type;
    }

    /** Get dateCreated
 * @return  */
    public java.util.Date getDatePurchased() {
        return datePurchased;
    }
```

```
   /** Return a string representation of this object
   @Overidde Annotation*/
    public String toString() {
        return "Purchased on " + datePurchased;
    }

    /** Showing association
    @param p1 */
    public void addFurnitureList(Purchaser p1) {
        furnitureList[numOfType]= p1;
        numOfType++;
    }

    public void printAllInfo(){
        for(int i = 0;i<numOfType; i++){
            Purchaser s = (Purchaser)furnitureList[i];
            System.out.println((i+1) + ". " +  s.getFurnitureType() );
        }
    }

    /** Abstract method getDiscount */
    public abstract double getDiscount();

}
```

SOURCE CODES

Purchaser Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;

public class Purchaser {
    private final int id;
    private final String name;
    private String furnitureType;

    public Purchaser (int i, String s, String n){
        id = i;
        name = n;
        furnitureType = s;
    }

    public String getName() {
        return name;
    }

    public int getID() {
        return id;
    }

    public String getFurnitureType(){
        return furnitureType;
    }
}
```

SOURCE CODES

TypeOfFurniture Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;

public class TypeOfFurniture {

    private final String name;
    private final int memberId;
    private final Purchaser[] furnitureList;
    private int numOfFurniture;

    public TypeOfFurniture(int m,String n, String j){
        name = n;
        memberId = m;
        furnitureList = new Purchaser[10];
    }

    public String getName(){
        return name;
    }

    public int getMemberId(){
        return memberId;
    }

    public void addFurniture(Purchaser p1){
        furnitureList[numOfFurniture]=p1;
        numOfFurniture++;
    }

    public int getNumOfFurniture(){
        return numOfFurniture;
    }

    public void printAllInfo(){
        System.out.println("\nThis furniture set was taken by : " + name);
        System.out.println(name + " was taken " + numOfFurniture + " set (s).");
        System.out.println("List of furniture(s) taken : ");

        for(int i = 0;i<numOfFurniture; i++){
            Purchaser s = (Purchaser)furnitureList[i];
            System.out.println((i+1) + ". " + s.getName());
        }
    }
    public void listOfFurniture()
    {
        String [] list = {"1. IK001 Sundvik Rocking Chair","2. IK002 Kritter
                          Children Chair","3. IK003 Utter Children Table",
                          "4. IK004 Stuva Children Table","5. IK005
                          Poang Children Arm Chair"};

        System.out.println("NOTES : Choose 2 type of Kids Set ------>");
        System.out.print("\nHi! Here is the list of furniture that you can choose
             for your kiddos!! =)");
        System.out.print("\n\n");

        for(int i=0; i<5; i++)
                System.out.println(list[i]);

    }
}
```

SOURCE CODES

MySofa Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.util.Scanner;
import javax.swing.JDialog;
import javax.swing.JOptionPane;

/** Extends MyFurniture Object */
public  class MySofa extends MyFurniture {
    private String sofaId;
    private String sofaName;
    private double discount;
    private double price;


    public MySofa()

    {
        sofaId = "";
        sofaName = "";
        price = 0.0;
    }

    public MySofa(double p)

    {
        price = p;
    }

    public MySofa(String id, String name, double price)
    {
        sofaId = id;
        sofaName = name;
        price = price;
    }

    public void setPrice(double newPrice)

    {
        price = newPrice;
    }
    public void listOfSofa()

    {
        JDialog.setDefaultLookAndFeelDecorated(true);
        System.out.print("\n");
        Object[] selectionValues = { "GA233 Lousiana", "GA235 Gorgia", "GA236
             Ottava" };
        String initialSelection = "GA233 Louisiana";
        Object selection = JOptionPane.showInputDialog(null, "Your Favourite Sofa
             Set?", "Your Favourite Sofa Set", JOptionPane.QUESTION_MESSAGE,
             null, selectionValues, initialSelection);
        System.out.println("Your choosen Sofa Set is : " + selection);
    }

    public void readInput()

    {
        Scanner input = new Scanner(System.in);

        System.out.print("\nEnter your favourite sofa code : ");
        sofaId = input.nextLine();

        System.out.print("Enter your favourite sofa name : ");
        sofaName = input.nextLine();

        System.out.print("Enter sofa's price : ");
        price = input.nextDouble();

    }

    @Override
      public String toString() {
        return "Thank You!. Your Purchased Date is on " + datePurchased ;
    }

    public void mixColor()

    {
        JDialog.setDefaultLookAndFeelDecorated(true);
        System.out.print("\n");
        System.out.print("Ohh wait!!!You can choose your favourite color too!!!");
        Object [] selectionValues = {"Red and Black", "Pink and Turqoise", "Light
             Green and Cream", "Orrange and Light Green"};
        String initialSelection = "Red and Black";
        Object selection = JOptionPane.showInputDialog(null, "Mix and Match",
            "Mix and Match", JOptionPane.QUESTION_MESSAGE, null, selectionValues,
             initialSelection);
        System.out.println("Colour : " + selection);
    }

    //Abstract
    public double getDiscount()

    {
        return price -(price*0.10);
    }
}
```

SOURCE CODES

MyBedroomSet Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.util.Scanner;

/** Extends abstract MyFurniture Object */
public class MyBedroomSet extends MyFurniture {
	private String bedId, bedName;
	private String type;
	private double price;

	public MyBedroomSet()
	{
		bedId = "";
		bedName = "";
		price = 0.0;
	}

	 public String getBedName()
	{
		return bedName;
	}

	 public MyBedroomSet(double p) {
	     price = p;
	 }

	public MyBedroomSet(String newBedId, String newBedName, double newPrice)
	{
	     bedId = newBedId;
	     bedName = newBedName;
	     price = newPrice;
	}

	 //to filter type of bedroom based on user choice
	public String typeOfBedroomType()
	{
	     String newType;

	     Scanner input = new Scanner(System.in);

	     System.out.print("\nEnter your preferred size of bedroom set : SINGLE
	     | QUEEN | KING : ");

	     newType = input.nextLine();

	     switch (newType) //filtering type
	     {
	             case "SINGLE":
	                     newType = "G045 Zella Charcoal Set";
```

```
                                System.out.println("G045 Zella Charcoal Set");
                                break;

                        case "QUEEN":
                                newType = "G046 Exquisite Twin Set";
                                System.out.println("G046 Exquisite Twin Set");
                                break;

                        case "KING":
                                newType = "G047 Chatham King Set";
                                System.out.println("G047 Chatham King Set");
                                break;

                        default:
                                System.out.println("Invalid choice!!");
                                System.exit(0);
                }
                return newType;

            }

            public void readInput()

           {
                Scanner input = new Scanner(System.in);

                System.out.print("\nEnter your favourite bedroom code : ");
                bedId = input.nextLine();

                System.out.print("Enter your favourite bedroom set name : ");
                bedName = input.nextLine();
           }

           @Override
           public String toString() {
                return super.toString() + "\nPurchased on :" + datePurchased;
           }


            //@Abstract Method
           public double getDiscount() {
                return price -(price*0.05);
           }
}
```

SOURCE CODES

MyKidsSet Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.util.Scanner;
import javax.swing.JOptionPane;
import javax.swing.JDialog;

public abstract class MyKidsSet {
    private String kidsId, furName;
    private double price;

    public MyKidsSet()
    {
        kidsId = "";
        furName = "";
        price = 0.0;
    }

    public MyKidsSet(double p) {
        price = p;
    }

    public MyKidsSet(String newId, String newFurName, double newPrice)
    {
        kidsId = newId;
        furName = newFurName;
        price = 0.0;
    }

    public double getPrice() {
        return this.price;
    }

    public String showPrice()
    {
        Scanner input = new Scanner(System.in);

        System.out.print("\nPlease enter the kid's set Id : ");
        kidsId = input.nextLine();

        System.out.print("Pease enter the kid's furniture name :");
        furName = input.nextLine();

        switch(kidsId)
        {
            case("IK001"):
                    price = 45.00;
                    System.out.print("Price is RM :" + price);
                    break;
            case("IK002"):
                    price = 100.00;
                    System.out.print("Price is RM :" + price);
                    break;

            case("IK003"):
                    price = 250.00;
                    System.out.print("Price is RM :" + price);
                    break;

            case("IK004"):
                    price = 150.00;
                    System.out.print("Price is RM :" + price);
                    break;

            case("IK005"):
                    price = 450.00;
                    System.out.print("Price is RM :" + price);
                    break;

            default:
                    System.out.println("Invalid choice!!");
                    System.exit(0);
        }
        return kidsId;
    }

    public void mysteryGift()
    {
        JDialog.setDefaultLookAndFeelDecorated(true);      //using JOptionPane
        System.out.print("\n");
        System.out.print("Congrat's. You're entitled to get a mystery gift from
             us!!!");

        Object [] selectionValues = {"Little Tayo", "Barbie", "Thomas & Friends",
             "Roborcar Poli"};
        String initialSelection = "Key Chain";
        Object selection = JOptionPane.showInputDialog(null, "Mystery Gift",
            "Choose Your Gift", JOptionPane.QUESTION_MESSAGE, null,
             selectionValues, initialSelection);
        System.out.println("\nYou have choose : " + selection);
    }

    /**Has Implementations Using Interface*/
    public abstract int compareTo(ComparableKidsSet o);


}
```

SOURCE CODES

Comparable Interface

```
/**
 *
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;

public interface Comparable {
    public int compareTo(ComparableKidsSet o);

}
```

SOURCE CODES

ComparableKidsSet Class

```
/**
 *
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.util.Scanner;

public class ComparableKidsSet extends MyKidsSet implements Comparable {

    private String kidsId, furName;
    private double price;

    public ComparableKidsSet() {}

    /** Construct a ComparableKidsSet with specified properties */
    public ComparableKidsSet(ComparableKidsSet o1) {

    }

    /** Implement the compareTo method to defined in Comparable */
    @Override Annotation
     */
    public int compareTo(ComparableKidsSet o) {
        if(getPrice() > ((ComparableKidsSet)o).getPrice())
            return 1;
        else if(getPrice() < ((ComparableKidsSet)o).getPrice())
            return -1;
        else
            return 0;
    }

    public String showPrice()
    {
        Scanner input = new Scanner(System.in);

        System.out.print("\nPlease enter the kid's set Id : ");
        kidsId = input.nextLine();

        System.out.print("Pease enter the kid's furniture name :");
        furName = input.nextLine();

        switch(kidsId)
        {
            case("IK001"):
                    price = 45.00;
                    System.out.printf("Price is RM%.2f ", price);
                    break;

            case("IK002"):
                    price = 100.00;
                    System.out.printf("Price is RM%.2f ",price);
                    break;

            case("IK003"):
                    price = 250.00;
                    System.out.printf("Price is RM%.2f ",price);
                    break;

            case("IK004"):
                    price = 150.00;
                    System.out.printf("Price is RM%.2f ", price);
                    break;

            case("IK005"):
                    price = 450.00;
                    System.out.printf("Price is RM%.2f ",price);
                    break;

            default:
                    System.out.println("Invalid choice!!");
                    System.exit(0);
        }
        return kidsId;
    }

}
```

SOURCE CODES

Max Class

```
/**
 *
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;

/** Max.java : Find a maximum object */
public class Max {

    /**Return the maximum between two object*/
    public static ComparableKidsSet max(ComparableKidsSet o1,ComparableKidsSet
      o2){
        if(((ComparableKidsSet)o1).compareTo(o2) > 0){
            System.out.println("\nYour 1st item is more expensive then 2nd item");
            return o1;
        }
        else{
            System.out.print("\nYour 2nd item is more cheaper. Enjoy!!");
            return o2;
        }
    }


}
```

SOURCE CODES

Member Class

```
package furniture;

/**
 *@author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

public class Member {

    private String nName;
    private String nColor;
    private double nprice;

    public Member()
    {
        nName = "";
        nColor = "";
        nprice = 0.0;
    }

    public Member(String name,String color,double price)
    {
        nName = name;
        nColor = color;
        nprice = price;
    }
    public void setName(String n){
        nName = n;
    }

    public String getName()
    {
        return this.nName;
    }
    public String getColor()
    {
        return this.nColor;
    }

     public double getPrice()
    {
        return this.nprice;
    }

}
```

SOURCE CODES

Membership Class

```java
/**
 *@author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;

public class Membership {

    private String nName;
    private String nBirthday;
    private double nFee;

    public Membership()
    {
        nName = "";
        nBirthday = "";
        nFee = 0.0;
    }

    public Membership(String name,String birth,double fees)
    {
        nName = name;
        nBirthday = birth;
        nFee = fees;
    }
    public void setName(String n){
        nName = n;
    }

    public String getName()
    {
        return this.nName;
    }
    public String getBirthday()
    {
        return this.nBirthday;
    }

     public double getFees()
    {
        return this.nFee;
    }

}
```

SOURCE CODES

TestMyFurniture Class

```
/**
 * @author myTeam --> MARDIANA | NUR ALLYYAA | NUR SYUHADA | NURUL HIDAYAH
 */

package furniture;
import java.util.ArrayList;
import java.util.Scanner;
import java.text.DecimalFormat;


public class TestMyFurniture {
public static void main(String[] args) {


System.out.print("\n/////////////////////////////////////////////////////////////");

System.out.print("\n//Mardiana binti Abu
      Hassan\tSX140047CSJS04/////////////////");
    System.out.print("\n//Nur Allyyaa binti Abdul
      Halim\tSX140055CSJS04/////////////////");
    System.out.print("\n//Nur Syuhadah binti
      Zulkifli\tSX1401047SCSV04/////////////////");
    System.out.print("\n//Nurul Hidayah binti
      Anuar\tSX140056CSJS04/////////////////");
    System.out.print("\n//Application Name :
      Furniture/////////////////////////////////");

System.out.print("\n/////////////////////////////////////////////////////////////");
    System.out.print("\n\n\n");


//*********************************************************************************
    //To Display Furniture Menu
//*********************************************************************************
    ArrayList<Member> stud = new ArrayList<>();

    boolean control = true;

    while(control){
        int displayMenu = displayMenu();
        int userSelection = userSelection(displayMenu);
        System.out.println("User selects : " + userSelection);

        if(userSelection==1){

            System.out.println("******************************************");
            System.out.println("\t\tEnter New Data ");
            System.out.println("******************************************");
            Scanner option = new Scanner(System.in);
            System.out.print("How many set of furniture you want to add:  ");
            int op = option.nextInt();
            for(int i = 0;i<op;i++){
                Member mem1 = input();
                stud.add(mem1);
            }

        }
        else if (userSelection==2){

            Scanner remove = new Scanner(System.in);
            System.out.println("Insert index to remove");
            int rem = remove.nextInt();
            if(rem>stud.size())
            {
                System.out.println("Index out of bound");
                control = false;
            }
            stud.remove(rem);
        }

        else if (userSelection==0){
            control = false;
        }

        else if (userSelection==3){
            Scanner inp = new Scanner(System.in);
            System.out.println("Enter index number: ");
            int index = inp.nextInt();
            Member std = option3();
            stud.add(index, std);
        }

        else if (userSelection==4){
            System.out.println("\nThanks for adding to the directory\n");

            displayTable();

            for (Member stud1 : stud) {
            String a=((Member) stud1).getName();
            String b =((Member) stud1).getColor();
            double c =((Member) stud1).getPrice();
            String format = "%1$-15s%2$-25s%3$-10s\n";
            DecimalFormat deci = new DecimalFormat("0.00");
            System.out.format(format, a, b,  deci.format(c));

            }

        }


System.out.println("===========================================================");
        System.out.println("\n");

}
//End Of Furniture Array List
```

```
//*******************************************************************************
    //To Display Membership Menu
//*******************************************************************************

    ArrayList<Membership> ship = new ArrayList<>();

    boolean control1 = true;

    while(control1){
        int displayMenu1 = displayMenu1();
        int userSelection1 = userSelection1(displayMenu1);
        System.out.println("User selects : " + userSelection1);

        if(userSelection1==1){

            System.out.println("********************************************");
            System.out.println("\t\tEnter New Membership Details ");
            System.out.println("********************************************");
            Scanner opt = new Scanner(System.in);
            System.out.print("How many members you want to register today:  ");
            int a = opt.nextInt();

            for(int i = 0;i<a;i++){

                Membership ship1 = input1();
                ship.add(ship1);

            }

        }
        else if (userSelection1==2){

            Scanner remove = new Scanner(System.in);
            System.out.println("Insert index to remove");
            int rem = remove.nextInt();
            if(rem>ship.size())
            {
                System.out.println("Index out of bound");
                control1 = false;
            }
            ship.remove(rem);
        }

        else if (userSelection1==0){
            control1 = false;
        }

        else if (userSelection1==3){
            Scanner in = new Scanner(System.in);
            System.out.println("Enter index number: ");
            int ind = in.nextInt();
            Membership st = option5();
            ship.add(ind, st);
        }

        else if (userSelection1==4){
            System.out.println("\nThanks for adding to the directory\n");

            displayTable1();
            for (Membership sh1 : ship) {

                String am=((Membership) sh1).getName();
                String bm =((Membership) sh1).getBirthday();
                double cm =((Membership) sh1).getFees();
                String format = "%1$-15s%2$-25s%3$-10s\n";
                DecimalFormat deci = new DecimalFormat("0.00");
                System.out.format(format, am, bm,  deci.format(cm));

            }
        }

System.out.println("===========================================================");
    System.out.println("\n");

}
//End Of Membership Array

//********************************************************************************
    //To Manage Purchaser Details of Purchased Item
//********************************************************************************

    MySofa ms1 = new MySofa();
    MyBedroomSet b1 = new MyBedroomSet();

    /*Abstract*/
    MyFurniture object2 = new MyBedroomSet(5000);
    MyFurniture object1 = new MySofa(10000);

     /** Association*/
    System.out.println();
    System.out.println("TO MANAGE PURCHASER DETAILS OF PURCHASED ITEM");
    System.out.println("Here Is Your Purchaser Details | " + object1.toString());
    System.out.println();

    Purchaser p1 = new Purchaser(001, "Mardiana", "Kids Set");
    Purchaser p2 = new Purchaser(002, "Syuhada", "Bedroom Set");
    Purchaser p3 = new Purchaser(003, "Allyyaa", "Sofa Set");

    TypeOfFurniture s1 = new TypeOfFurniture(001,"Emelda", "Sofa Set");
    s1.addFurniture(p3);
    s1.addFurniture(p2);
    s1.printAllInfo();

    System.out.println("\n1st Customer | Sofa Set Descriptions");
    ms1.listOfSofa();
    ms1.mixColor();

    /** Abstract Class | To Display Sofa Price After Discount */
    displayMyFurnitureObject(object1);

    System.out.println("\n" +ms1.toString());

    System.out.println();
    System.out.println("\nBedroom Descriptions");
    b1.typeOfBedroomType();

     /** Display Sofa Discount */
    displayMyFurnitureObject(object2);

    System.out.println();

    /**To Check If Both Price Are Same Using Abstract Class */
    System.out.println("\nThe two type of furniture have the same price? " +
      equalDiscount(object1,object2));
    System.out.println();

    System.out.println("*******************************************");
    System.out.printf("Total Amount Paid is : RM %.2f",
      totalPaid(object1,object2));
    System.out.println("\n*******************************************");

    System.out.println("\n2nd Customer | Bedroom Set Descriptions");
    TypeOfFurniture s2 = new TypeOfFurniture(002,"Emir Zafran","Bedroom Set");
    s2.addFurniture(p1);
    s2.printAllInfo();

    s2.listOfFurniture();

    //TODO Interface Implementation | To Find Maximum Price between 2 types of
      kids furniture
    ComparableKidsSet myKids1 = new ComparableKidsSet();
    ComparableKidsSet myKids2 = new ComparableKidsSet();
    myKids1.showPrice();
    System.out.println();
    myKids2.showPrice();

    ComparableKidsSet kids1 = new ComparableKidsSet(myKids1) ;
    ComparableKidsSet kids2 = new ComparableKidsSet(myKids2) ;

    System.out.println();
    System.out.println(Max.max(kids1, kids2));

}

/** A method for comparing the discount of the two MyFurniture Object | Abstract
Class
 *
 * @param object1
 * @param object2
 * @return
 */
public static boolean equalDiscount(MyFurniture object1, MyFurniture object2) {
    return object1.getDiscount() == object2.getDiscount();
}
```

```
public static double totalPaid(MyFurniture object1, MyFurniture object2) {
    return object1.getDiscount() + object2.getDiscount();
}

/** A method for displaying  the discount of the two MyFurniture Object */
public static void displayMyFurnitureObject(MyFurniture object) {
    System.out.printf("Total Price After Discount Is RM %.2f ",
      object.getDiscount() );
}

/** Polymorphism through overriding methods : Display Properties of Furniture */
public static void displayObject(MyFurniture object) {
    System.out.println("test " + object.getDatePurchased() + "Type of furniture "
+ object.getType());
}

//*******************************************************************************
//To Display Menu Of The Application
//*******************************************************************************

 public static int displayMenu(){
        Scanner selection = new Scanner(System.in);
        System.out.println("==========================");
        System.out.println("GA Sofa Furniture Details");
        System.out.println("==========================");
        System.out.println("1 - Add New Furniture");
        System.out.println("2 - Remove Furniture");
        System.out.println("3 - Add Furniture to Specific Index");
        System.out.println("4 - Display Furniture Information");
        //System.out.println("5 - Search Data");
        System.out.println("0 - Exit");
        System.out.println("==========================");
        System.out.print("Your Selection : ");
        int select = selection.nextInt();
        System.out.println();

        return select;
    }

 public static int userSelection(int displayMenu){
        int a;
        int b = 0;
        a=displayMenu;
        if (a==1){
            System.out.println("1 selected");
            b=1;
        }
        else if (a==2){
            System.out.println("2 selected");
            b=2;
        }
        else if (a==3){
            System.out.println("3 selected");
            b=3;
        }
        else if (a==4){
            System.out.println("4 selected");
            b=4;
        }
        else if (a==5){
            System.out.println("5 selected");
            b=5;
        }
        else if (a==0){
            System.out.println("Exit");
            b=0;
        }
        else
            System.out.println("Error");

        return b;

    }
    //End Furniture Menu

  //To Display Membership Menu
  public static int displayMenu1(){
        Scanner selection = new Scanner(System.in);
        System.out.println("==========================");
        System.out.println("GA Sofa Membership Details");
        System.out.println("==========================");
        System.out.println("1 - Add New Member");
        System.out.println("2 - Remove Member");
        System.out.println("3 - Add Member to Specific Index");
        System.out.println("4 - Display Member's Information");
        System.out.println("0 - Exit");
        System.out.println("==========================");
        System.out.print("Your Selection : ");
        int select = selection.nextInt();
        System.out.println();

        return select;
    }

  public static int userSelection1(int displayMenu1){
        int a;
        int b = 0;
        a=displayMenu1;
        if (a==1){
            System.out.println("1 selected");
            b=1;
        }
        else if (a==2){
            System.out.println("2 selected");
            b=2;
        }
        else if (a==3){
            System.out.println("3 selected");
            b=3;
        }
        else if (a==4){
            System.out.println("4 selected");
            b=4;
        }
        else if (a==5){
            System.out.println("5 selected");
            b=5;
        }
        else if (a==0){
            System.out.println("Exit");
            b=0;
        }
        else
            System.out.println("Error");

        return b;

    }
    //End Of Membership Menu

//Display Table For Furniture Type Menu
public static void displayTable(){

System.out.println("=======================================================");
    String format = "%1$-15s%2$-25s%3$-10s\n";
    System.out.format(format, "Furniture's Name", "Furniture's Color", "Furniture
      Price");

System.out.println("=======================================================");
}

//Display Table For Member's Menu
public static void displayTable1(){

System.out.println("=======================================================");
    String format = "%1$-15s%2$-25s%3$-10s\n";
    System.out.format(format, "Member's Name", "Member's Birthday", "Member's
Fee");

System.out.println("=======================================================");
}

//To Add New Furniture
private static Member input() {
    Scanner input = new Scanner(System.in);
    System.out.println("****************************************");
    System.out.println("\t\tEnter your details ");
    System.out.println("****************************************");
    System.out.print("Enter furniture name :");
    String name = input.nextLine();
    System.out.print("Enter color :");
    String color = input.nextLine();
    System.out.print("Enter price :");
    double price = input.nextDouble();
    Member mem = new Member(name, color, price);
    System.out.println();
    return mem;

}

//To Add New Furniture To Specific Index
private static Member option3() {
    Scanner index = new Scanner(System.in);
    System.out.print("Enter furniture name :");
    String name = index.nextLine();
    System.out.print("Enter color :");
    String color = index.nextLine();
    System.out.print("Enter price :");
    double price = index.nextDouble();
    Member mem2 = new Member(name, color, price);
    System.out.println();
    return mem2;
}

//To Add New Membership
private static Membership input1() {
    Scanner input = new Scanner(System.in);
    System.out.println("*******************************************");
    System.out.println("\t\tEnter your details ");
    System.out.println("*******************************************");
    System.out.print("Enter Member name :");
    String name = input.nextLine();
    System.out.print("Enter Member Birthday :");
    String birth = input.nextLine();
    System.out.print("Enter Member Fees Registration :");
    double fees = input.nextDouble();
    Membership sh = new Membership(name, birth, fees);
    //stud.add(new Member(name,color,price));
    System.out.println();
    return sh;
}

//To Add New Membership To Specific Index
private static Membership option5() {
    Scanner index = new Scanner(System.in);
    System.out.print("Enter Member's name :");
    String name = index.nextLine();
    System.out.print("Enter Member's Birthday :");
    String birth = index.nextLine();
    System.out.print("Enter Member's fee Registration :");
    double fees = index.nextDouble();
    Membership shp = new Membership(name, birth, fees);
    System.out.println();
    return shp;
    }

}
```

OUTPUT

```
////////////////////////////////////////////////////////////////
//Mardiana binti Abu Hassan      SX140047CSJS04//////////////////
//Nur Allyyaa binti Abdul Halim  SX140055CSJS04//////////////////
//Nur Syuhadah binti Zulkifli    SX1401047SCSV04//////////////////
//Nurul Hidayah binti Anuar      SX140056CSJS04//////////////////
//Application Name : Furniture//////////////////////////////////
////////////////////////////////////////////////////////////////


==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 1

1 selected
User selects : 1
*******************************************
             Enter New Data
*******************************************
How many set of furniture you want to add:  3
*******************************************
             Enter your details
*******************************************
Enter furniture name :Lousiana
Enter color :Black and Red
Enter price :2500

*******************************************
             Enter your details
*******************************************
Enter furniture name :Catham
Enter color :White
Enter price :5500

*******************************************
             Enter your details
*******************************************
Enter furniture name :Rocking Chair
Enter color :White
Enter price :45

====================================================================
==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 4

4 selected
User selects : 4

Thanks for adding to the directory

====================================================================
Furniture's NameFurniture's Color        Furniture Price
====================================================================
Lousiana       Black and Red           2500.00
Catham         White                   5500.00
Rocking Chair  White                   45.00
====================================================================


==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 2

2 selected
User selects : 2
Insert index to remove
1
====================================================================


==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 4

4 selected
User selects : 4
```

```
Thanks for adding to the directory

====================================================================
Furniture's NameFurniture's Color        Furniture Price
====================================================================
Lousiana       Black and Red            2500.00
Rocking Chair  White                    45.00
====================================================================



==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 3

3 selected
User selects : 3
Enter index number:
1
Enter furniture name :Ottava
Enter color :White
Enter price :4500

====================================================================



==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 4

4 selected
User selects : 4

Thanks for adding to the directory

====================================================================
Furniture's NameFurniture's Color        Furniture Price
====================================================================
Lousiana       Black and Red            2500.00
Ottava         White                    4500.00
Rocking Chair  White                    45.00
====================================================================
==========================
GA Sofa Furniture Details
==========================
1 - Add New Furniture
2 - Remove Furniture
3 - Add Furniture to Specific Index
4 - Display Furniture Information
0 - Exit
==========================
Your Selection : 0

Exit
User selects : 0
====================================================================



==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 1

1 selected
User selects : 1
*******************************************
             Enter New Membership Details
*******************************************
How many members you want to register today:  4
*******************************************
             Enter your details
*******************************************
Enter Member name :Mardiana
Enter Member Birthday :Jan 82
Enter Member Fees Registration :100

*******************************************
             Enter your details
*******************************************
Enter Member name :Allyyaa
Enter Member Birthday :Sept 92
Enter Member Fees Registration :120

*******************************************
             Enter your details
*******************************************
Enter Member name :Syu
Enter Member Birthday :Jun 90
Enter Member Fees Registration :130

*******************************************
             Enter your details
*******************************************
Enter Member name :Hidayah
Enter Member Birthday :Sept 92
Enter Member Fees Registration :120

====================================================================

==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 4

4 selected
User selects : 4

Thanks for adding to the directory

====================================================================
Member's Name  Member's Birthday        Member's Fee
====================================================================
Mardiana       Jan 82                   100.00
Allyyaa        Sept 92                  120.00
Syu            Jun 90                   130.00
Hidayah        Sept 92                  120.00
====================================================================


==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 2

2 selected
User selects : 2
Insert index to remove
2
====================================================================


==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 4
```

```
4 selected
User selects : 4

Thanks for adding to the directory

====================================================================
Member's Name  Member's Birthday        Member's Fee
====================================================================
Mardiana       Jan 82                   100.00
Allyyaa        Sept 92                  120.00
Hidayah        Sept 92                  120.00
====================================================================



==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 3

3 selected
User selects : 3
Enter index number:
3
Enter Member's name :Syuhada
Enter Member's Birthday :Jun 92
Enter Member's fee Registration :120

====================================================================



==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 4

4 selected
User selects : 4
```

```
Thanks for adding to the directory

====================================================================
Member's Name  Member's Birthday        Member's Fee
====================================================================
Mardiana       Jan 82                   100.00
Allyyaa        Sept 92                  120.00
Hidayah        Sept 92                  120.00
Syuhada        Jun 92                   120.00
====================================================================


==========================
GA Sofa Membership Details
==========================
1 - Add New Member
2 - Remove Member
3 - Add Member to Specific Index
4 - Display Member's Information
0 - Exit
==========================
Your Selection : 0

Exit
User selects : 0
====================================================================


TO MANAGE PURCHASER DETAILS OF PURCHASED ITEM
Here Is Your Purchaser Details | Thank You!. Your Purchased Date is on Wed Dec 23
23:04:52 SGT 2015


This furniture set was taken by : Emelda
Emelda was taken 2 set (s).
List of furniture(s) taken :
1. Sofa Set
2. Bedroom Set

1st Customer | Sofa Set Descriptions

Your choosen Sofa Set is : GA233 Lousiana

Ohh wait!!!You can choose your favourite color too!!!Colour : Red and Black
Total Price After Discount Is RM 9000.00
Thank You!. Your Purchased Date is on Wed Dec 23 23:04:52 SGT 2015


Bedroom Descriptions

Enter your preferred size of bedroom set : SINGLE | QUEEN | KING : QUEEN
G046 Exquisite Twin Set
Total Price After Discount Is RM 4750.00

The two type of furniture have the same price? false

*******************************************
Total Amount Paid is : RM 13750.00
*******************************************
```

```
2nd Customer | Bedroom Set Descriptions

This furniture set was taken by : Emir Zafran
Emir Zafran was taken 1 set (s).
List of furniture(s) taken :
1. Kids Set
NOTES : Choose 2 type of Kids Set ------>

Hi! Here is the list of furniture that you can choose for your kiddos!! =)

1. IK001 Sundvik Rocking Chair
2. IK002 Kritter Children Chair
3. IK003 Utter Children Table
4. IK004 Stuva Children Table
5. IK005 Poang Children Arm Chair

Please enter the kid's set Id : IK005
Pease enter the kid's furniture name :Poang Children Arm Chair
Price is RM450.00

Please enter the kid's set Id : IK001
Pease enter the kid's furniture name :Sundvik Rocking Chair
Price is RM45.00

Your 2nd item is more cheaper. Enjoy!!furniture.ComparableKidsSet@3249256e
```