

| **TITLE :Case Study (for Class Diagram)** |
| --- |

**AIM:** Draw class Diagram for the chosen Case Study . Clearly show

* + Attributes
  + Multiplicities between classes
  + Aggregations/compositions/Association between classes
  + Generalization between classes in the class diagram.

And show the implementation of aggregation, association, composition and generalization between the classes.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO1:** Understand the features of object oriented programming compared with procedural approach with C++ and Java.

**CO2**: Explore arrays, vectors, classes and objects in C++ and Java.

**CO3:** Implement scenarios using object oriented concepts (Drawing class diagram, relationship between classes, sequence diagram)

**CO4**: Explore the interface, exceptions, multithreading, packages

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1.Ralph Bravaco , Shai Simoson , “Java Programing From the Group Up” Tata McGraw-Hill.

2.Grady Booch, Object Oriented Analysis and Design .

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

DefineClass, Methods, Object.

Understanding of Aggregation, Association, Composition and Generalization between classes

**List Of Classes:**

**Identify Attributes for each class:**

**Identify List of Methods in each classes:**

MenuItem Class:

Attributes:

name: String

description: String

price: double

Methods:

getName(): String

getDescription(): String

getPrice(): double

Restaurant Class:

Attributes:

name: String

menu: List<MenuItem>

Methods:

None in the simplified version. In a real-world application, you would have methods to add and manage menu items.

RestaurantCatalog Class:

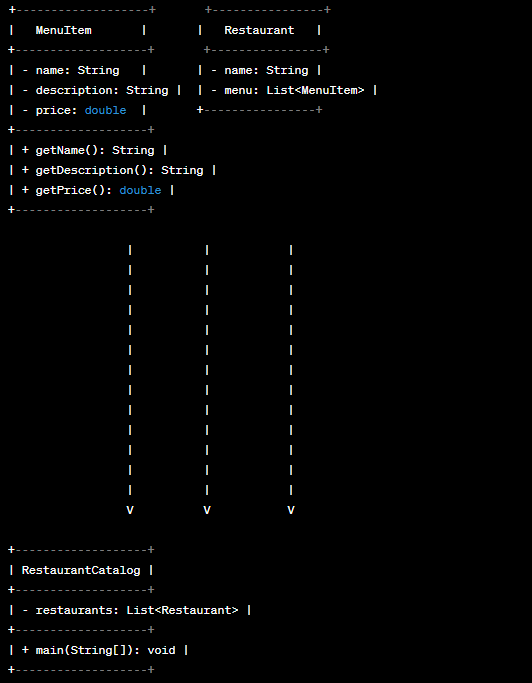
Attributes:

restaurants: List<Restaurant>

Methods:

main(String[]): void: Entry point of the program, responsible for managing user input and displaying restaurant and menu information.

**Class Diagram:**



**Algorithm:**

Algorithm RestaurantCatalog:

1. Initialize an empty list restaurants (List<Restaurant>).

2. Create multiple restaurants and add menu items to them, e.g.:

- Create restaurant1 (Restaurant).

- Create menu items (MenuItem) and add them to restaurant1.

- Create restaurant2 (Restaurant).

- Create menu items and add them to restaurant2.

- Add restaurant1 and restaurant2 to the restaurants list.

3. Display a welcome message.

4. Start an input loop:

a. Display restaurant options and a 'q' option to quit.

b. Read the user's choice.

5. If the user enters 'q', exit the loop and display a goodbye message.

6. If the user enters a number (restaurant index):

a. Check if the entered index is valid.

b. If valid, retrieve the selected restaurant from the list.

c. Display the menu for the selected restaurant.

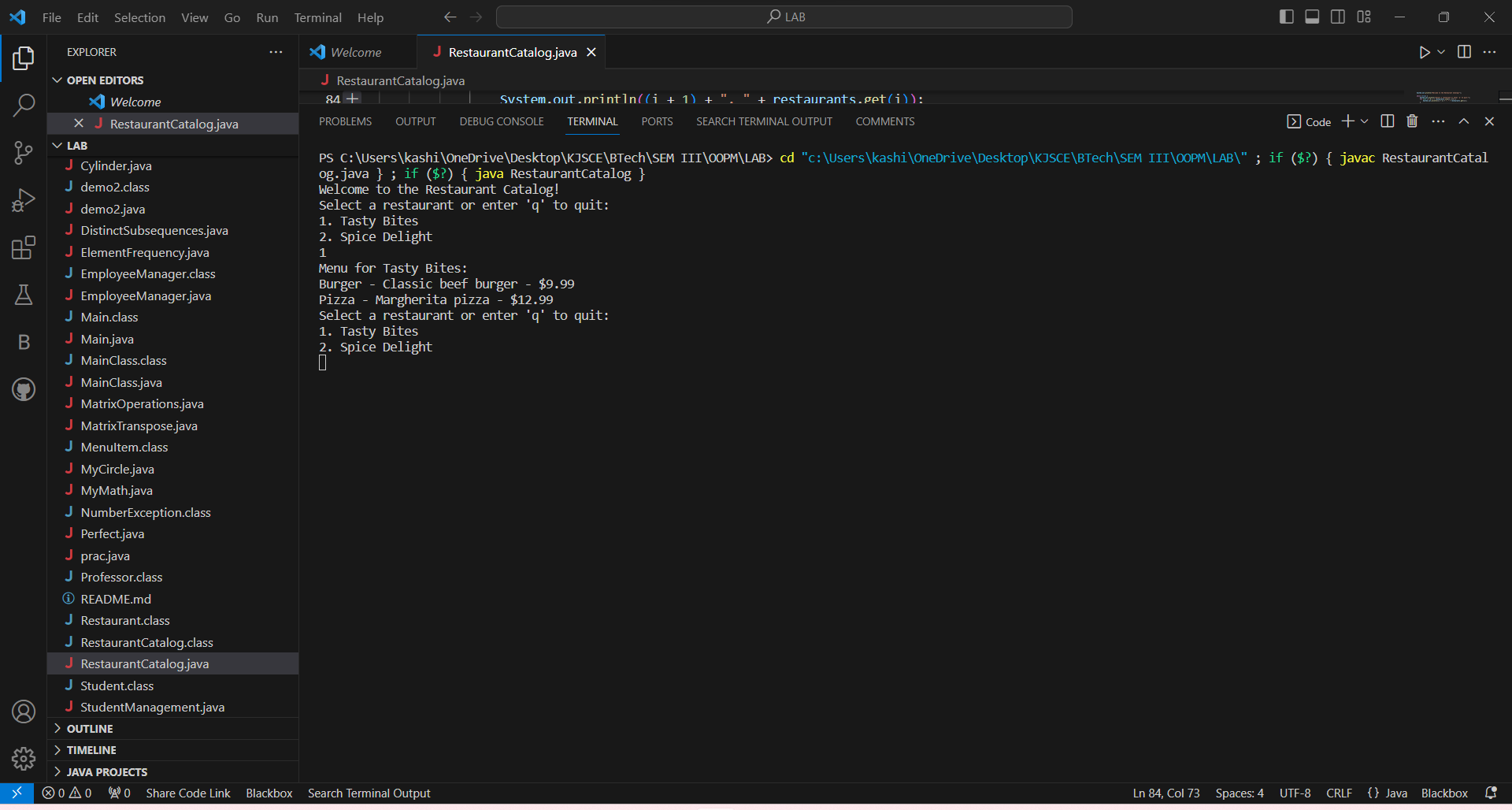
7. If the user enters an invalid input, display an error message.

8. Repeat the loop until the user decides to quit.

End of Algorithm.

**Implementation details**: (Class Diagram and Code)

**Output:**



**Conclusion**

We learned about class diagram in java.

**Date: 23/10/2023 Signature of faculty in-charge**

**Post Lab Descriptive Questions**

**1.** **Consider the following class:**

public class TypeOfVariable{

public static int a;

int b,c;

public void printValue(){

int x = 10;

}

public static void main(String args[]){

TypeOfVariable object=new TypeOfVariable();

object.printValue();

}

}

a). What are the class/static variables?

b). What are the instance variables?

c.)What are local variables?

**Ans:**

In the provided class `TypeOfVariable`, we can identify the following types of variables:

a) Class/Static Variables:

- `public static int a`

Class variables are declared using the `static` keyword. In this class, `a` is a class variable because it is declared as `public static int a`. Class variables are shared among all instances of the class.

b) Instance Variables:

- `int b`

- `int c`

Instance variables are not declared with the `static` keyword and are associated with an instance of the class. In this class, `b` and `c` are instance variables. Each instance of the class (object) will have its own set of these variables.

c) Local Variables:

- `int x` (inside the `printValue` method)

Local variables are declared within methods, constructors, or blocks. In this class, `x` is a local variable because it is declared within the `printValue` method. Local variables have a limited scope and are only accessible within the method or block in which they are declared.

**2.What is the output from the following code:**

public class Test

{

    static int x = 11;

    private int y = 33;

    public void method1(int x)

    {

        Test t = new Test();

        this.x = 22;

        y = 44;

        System.out.println("Test.x: " + Test.x);

        System.out.println("t.x: " + t.x);

        System.out.println("t.y: " + t.y);

        System.out.println("y: " + y);

    }

    public static void main(String args[])

    {

        Test t = new Test();

        t.method1(5);

    }

}

**Ans:**

Let's analyze the code step by step to understand the output:

1. The `Test` class is defined with class variables `static int x` and instance variable `private int y`.

2. Inside the `method1` method:

- A new `Test` object `t` is created.

- `this.x = 22;` sets the class variable `x` to 22.

- `y = 44;` sets the instance variable `y` of the `t` object to 44.

3. The `System.out.println` statements print the following values:

- `Test.x: 22` because `this.x = 22` changed the class variable.

- `t.x: 22` because `t` is an object of the `Test` class and shares the class variable.

- `t.y: 44` because `y = 44` sets the instance variable `y` for the `t` object.

- `y: 44` because `y` refers to the instance variable of the current object (`this.y`), which was set to 44 inside the method.

4. In the `main` method, a `Test` object `t` is created, and the `method1(5)` is called on it. This method modifies the class and instance variables as described above.

Therefore, the output of the code will be:

```

Test.x: 22

t.x: 22

t.y: 44

y: 44

```

The key point to understand here is that the `this` keyword refers to the current object (in this case, `t`), so when you set `this.x = 22`, it affects the class variable `x` for all instances of the class.