# 19AIE105

# OBJECT ORIENTED PROGRAMMING

## END SEMESTER PROJECT REPORT

# **TOPIC:** Event Management


**Submitted By:**
Nandhitha Ravishankar – BL.EN.U4AIE20041
P Sadhana - BL.EN.U4AIE20047

**Submitted to:**
Ms. Nalini Sampath

**AMRITA VISHWA VIDYAPEETHAM**
**BANGALORE – 560035**

**March - 2021**

# Table of Contents

# INTRODUCTION

Procedural Programming language that is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.

Object-oriented programming (OOP) has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the Java code DRY "Don't Repeat Yourself, and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

Here, we have implemented many OOP's concepts in out Program which is "Event Management".

Concepts such as Classes and Objects, Access specifiers, Method overloading and Method overriding, Constructors, Inheritance, Abstract class, Polymorphism, Exceptions and Interfaces have been applied here.

# EXPLAINATION

## 1. Classes and Objects

A class is a Blueprint from which individual objects are created. A class can contain any of the following variable types: Local Variables, Instance Variables and Class Variables.

Objects have states and behavior. A software object's state is stored in fields and behavior is shown via methods. In software development, methods operate on the internal state of an object and the object-to-object communication is done through methods.

## 2. Access Specifiers

Java provides a number of access modifiers to set access levels for classes, variables, methods and constructors. The four access levels are −

- Visible to the package, the default. No modifiers are needed.
- Visible to the class only (private).
- Visible to the world (public).
- Visible to the package and all subclasses (protected).

## 3. Method overloading and Method overriding

In java, it is possible to define two or more methods

within the same class that share the same name, as long as their parameters declaration are different. When this is the case, the methods are said to be overloaded, and the process is known as method overloading.

If a class inherits a method from its superclass, then there is a chance to override the method provided that it is not marked final.
The benefit of overriding is the ability to define a behaviour that's specific to the subclass type, which means a subclass can implement a parent class method based on its requirement. In object-oriented terms, overriding means to override the functionality of an existing method. In method overriding-

- The argument list should be exactly the same as that of the overridden method.
- A method declared final cannot be overridden.
- A method declared static cannot be overridden but can be re-declared.
- If a method cannot be inherited, then it cannot be overridden.

# 4. Constructors

A constructor is a block of codes similar to the method. It is called constructor because it constructs the values at the time of object creation. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory. It is a special type of method which is used to initialize the object. Every time an object is created using the new() keyword, at least one constructor is called. It

calls a default constructor if there is no constructor available in the class. . A Constructor must have the same name as the class name, it shouldn't have any explicit return type and it cannot be abstract, static, final or Synchronized.

# 5. Inheritance

Inheritance is one of the cornerstones of object oriented programming because it allows the creation of hierarchical classification. Using inheritance u can create a general class that defines trait common to a set a related items. This class can be inherited by others, more specific classes , each adding to these things that are unique to it. The class that is inherited is know as the superclass whereas the class that does the inheriting is known as the sub class. To inherit a class, u can incorporate the definition of one class into another by using the extends keyword.

# 6. Abstract class

A class which contains the abstract keyword in its declaration is known as abstract class.

- Abstract classes may or may not contain abstract methods, i.e., methods without body ( public void get(); )
- But, if a class has at least one abstract method, then the class must be declared abstract.
- If a class is declared abstract, it cannot be instantiated.
- To use an abstract class, you have to inherit it from another

class, provide implementations to the abstract methods in it.

- If you inherit an abstract class, you have to provide implementations to all the abstract methods in it.

# 7. Polymorphism

Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object. Because of polymorphism , in java u ca specify a general set of stack routines all all share the same names. In general terms, the concept of polymorphism can be expressed as “one interface and multiple methods. This helps to reduce the complexity by allowing the same interface to be used to specify a general class of action.

# 8. Exceptions

An exception (or exceptional event) is a problem that arises during the execution of a program. When an Exception occurs the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommended, therefore, these exceptions are to be handled. An exception can occur for many different reasons. Following are some scenarios where an exception occurs.

- A user has entered an invalid data.
- A file that needs to be opened cannot be found.
- A network connection has been lost in the middle of

communications or the JVM has run out of memory.

Some of these exceptions are caused by user error, others by programmer error, and others by physical resources that have failed in some manner.

# 9. Interfaces

An interface is a reference type in Java. It is similar to a class. It is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

Along with abstract methods, an interface may also contain constants, default methods, static methods, and nested types. Method bodies exist only for default methods and static methods.

Writing an interface is like writing a class. But a class describes the attributes and behaviours of an object. And an interface contains behaviours that a class implements.

Unless the class that implements the interface is abstract, all the methods of the interface need to be defined in the class.

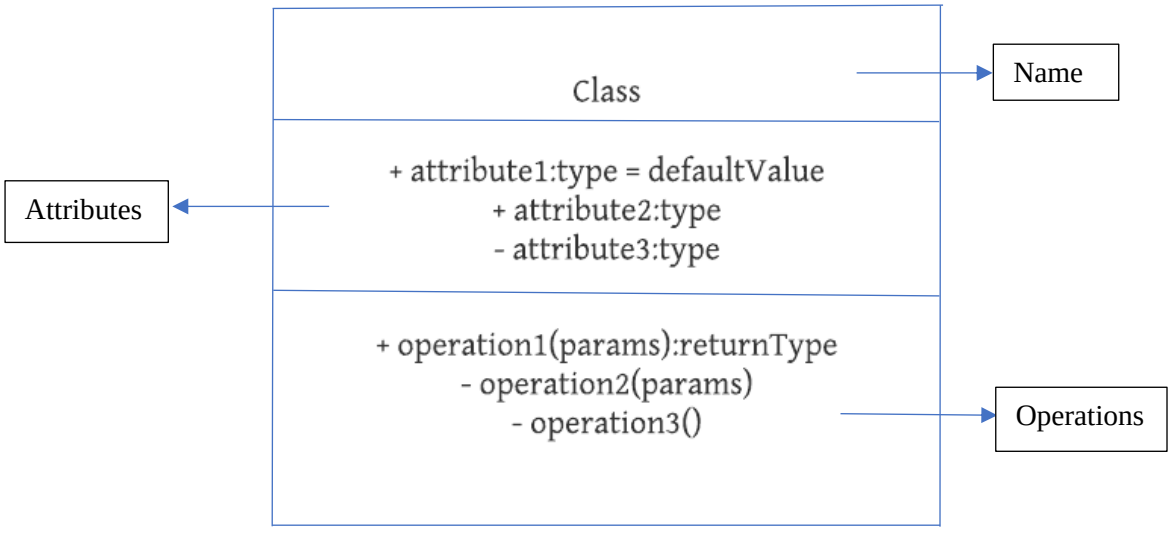
An interface is like a class in the following ways −

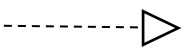
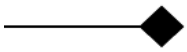
- An interface can contain any number of methods.
- An interface is written in a file with a .java extension, with the name of the interface matching the name of the file.
- The byte code of an interface appears in a .class file.
- Interfaces appear in packages, and their corresponding bytecode file must be in a directory structure that matches the package name.

# UML DIAGRAM

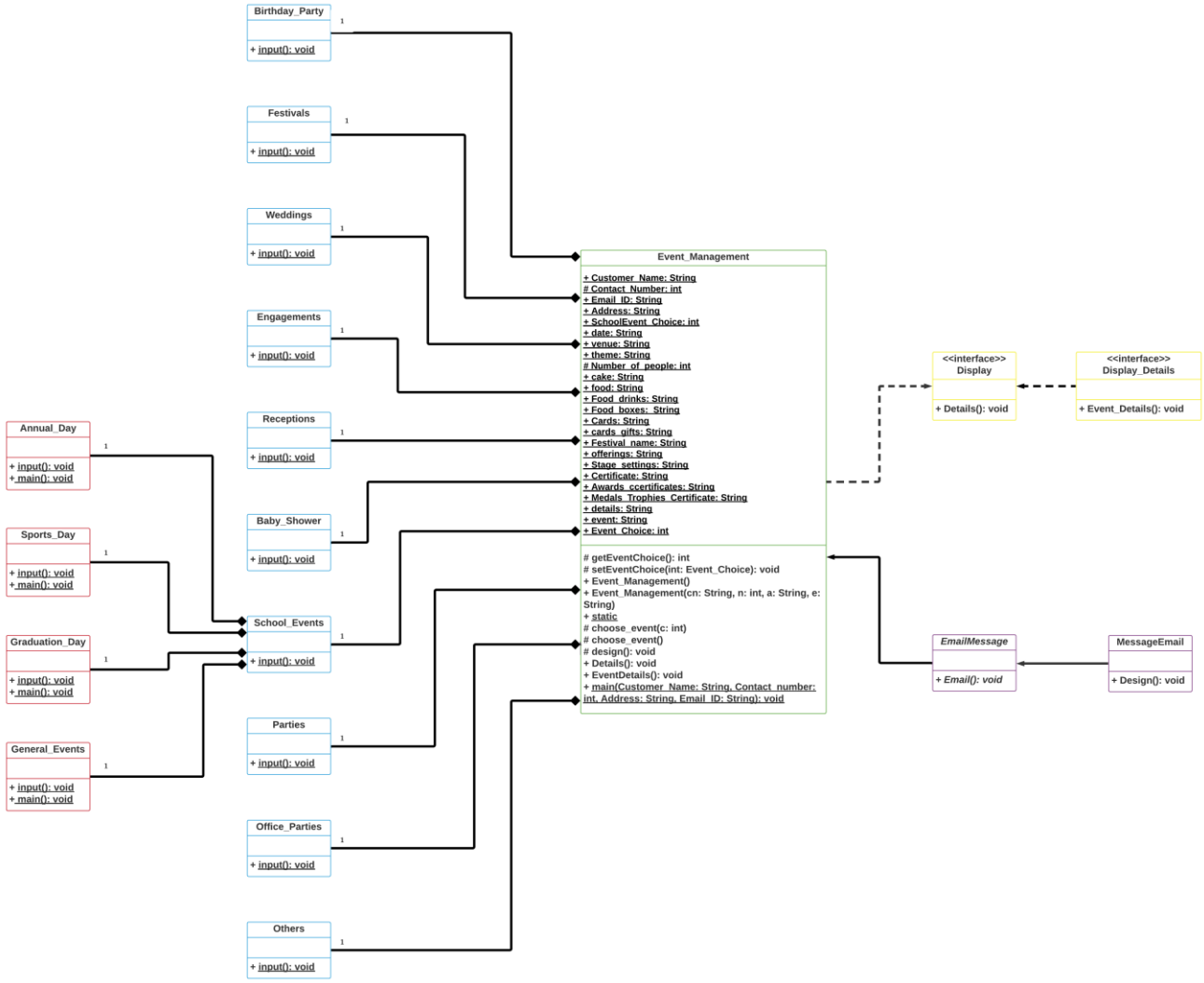
Unified Modeling Language diagram also known as UML diagram is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems. The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

## UML Notations

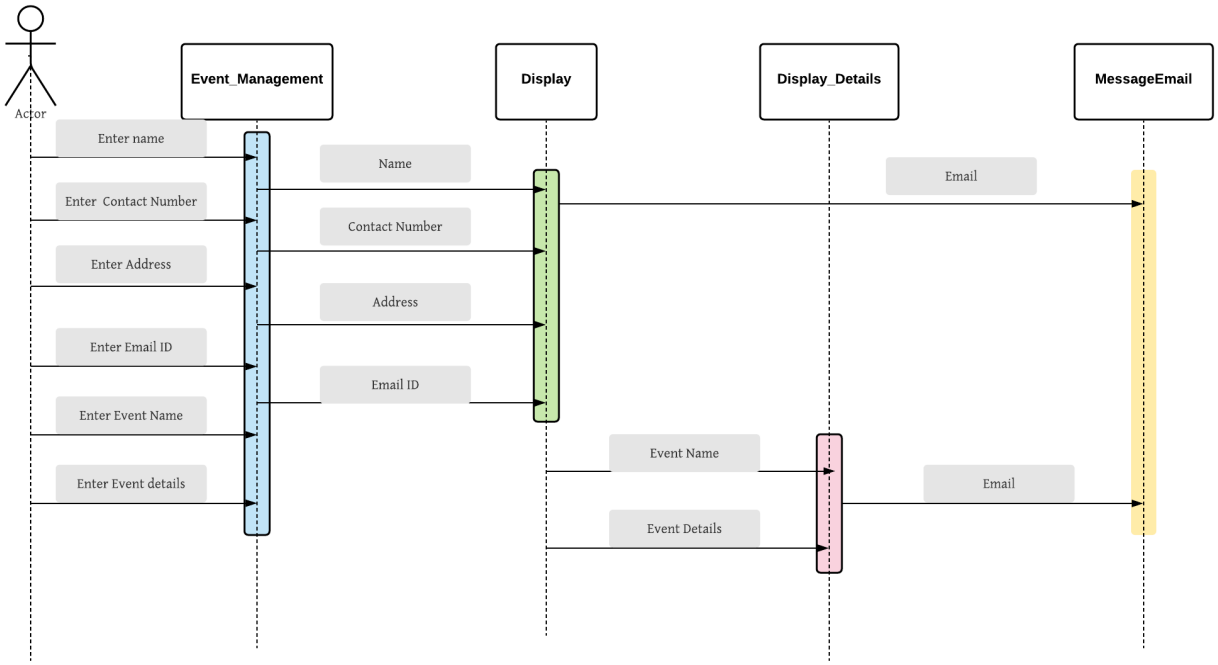



| | |
|---|---|
| + | public access specifier |
| # | protected access specifier |
| underline | static attributes |
| *italics* | abstract methods |
|  | interface |
|  | composition (part – of) |

# UML Class Diagram



# UML Sequence Diagram



# CODE

```
import java.util.*;
class Event_Management implements Display, Display_Details
{
   //Declaring static variables
   static public String Customer_Name;
   static protected int Contact_Number;
   static public String Email_ID;
   static public String Address;
   static public int SchoolEvent_Choice;
   static public String date, venue, theme;
   static protected int Number_of_people;
   static public String cake, food, Food_drinks, Food_boxes;
   static public String Cards, cards_gifts;
   static public String Festival_name;
   static public String offerings;
   static public String Stage_settings;
   static public String Certificate, Awards_certificates,
Medals_Trophies_Certificate;
   static public String details;
   static public String event;
   static public int Event_Choice;
   protected int getEventChoice()
   {
      return Event_Choice;
   }
   protected void setEventChoice(int Event_Choice)
   {
      this.Event_Choice = Event_Choice;
   }
   //Constructor
   Event_Management()
   {
      Customer_Name = "";
      Contact_Number = 0;
      Address = "";
      Email_ID = "";
   }
   //Constructor Overloading
   Event_Management(String cn, int n, String a, String e)
```

```
{
   Customer_Name = cn;
   Contact_Number = n;
   Address = a;
   Email_ID = e;
}
//static block
static
{
   System.out.println("\f");
   System.out.println("Welcome to Creative corner");
}
//Method Overloading
protected final void choose_event(int c)
{
   Event_Choice = c;
}
//Method Overloading
protected final void choose_event()
{
   Scanner sc = new Scanner(System.in);
   System.out.println("Choose the event: ");
   System.out.println("1. Birthday Party");
   System.out.println("2. Festivals");
   System.out.println("3. Weddings");
   System.out.println("4. Engagements");
   System.out.println("5. Receptions");
   System.out.println("6. Baby Shower");
   System.out.println("7. School Events");
   System.out.println("8. Parties");
   System.out.println("9. Office Parties");
   System.out.println("10. Others");
   Event_Choice = sc.nextInt();
   //switch case
   switch(Event_Choice)
   {
      case 1 :
            Birthday_Party obj1 = new Birthday_Party();
            obj1.input();
            System.out.println("");
            break;
      case 2 :
```

```
        Festivals obj2 = new Festivals();
        obj2.input();
        System.out.println("");
        break;
    case 3 :
        Weddings obj3 = new Weddings();
        obj3.input();
        System.out.println("");
        break;
    case 4 :
        Engagements obj4 = new Engagements();
        obj4.input();
        System.out.println("");
        break;
    case 5 :
        Receptions obj5 = new Receptions();
        obj5.input();
        System.out.println("");
        break;
    case 6 :
        Baby_Shower obj6 = new Baby_Shower();
        obj6.input();
        System.out.println("");
        break;
    case 7 :
        School_Events obj7 = new School_Events();
        obj7.input();
        System.out.println("");
        break;
    case 8 :
        Parties obj8 = new Parties();
        obj8.input();
        System.out.println("");
        break;
    case 9 :
        Office_Parties obj9 = new Office_Parties();
        obj9.input();
        System.out.println("");
        break;
    case 10 :
        Others obj10 = new Others();
        obj10.input();
```

```
                System.out.println("");
                break;
            default:
                System.out.println("");
        }
    }
    protected void design()
    {
        for (int i=1; i<=30 ; i++)
        {
            System.out.print(" *  ");
        }
        System.out.println("");
        for(int j=1; j<=30; j++)
        {
            System.out.print("*** ");
        }
        System.out.println("");
        System.out.println("");
    }
    public void Details()
    {
        System.out.println("****************** CUSTOMER DETAILS
******************");
        System.out.println("Name: "+Customer_Name);
        System.out.println("Contact Number: "+Contact_Number);
        System.out.println("Address: "+Address);
        System.out.println("E-Mail ID: "+Email_ID);
    }
    public void EventDetails()
    {
        System.out.println("******************* EVENT DETAILS
*******************");
        if(Event_Choice == 1)
        {
            System.out.println("Date of the Party: "+date);
            System.out.println("Venue: "+venue);
            System.out.println("Theme: "+theme);
            System.out.println("Number of people: "+Number_of_people);
            System.out.println("Cake flavours and description: "+cake);
            System.out.println("Food Items: "+food);
            System.out.println("Cards and Gifts: "+cards_gifts);
```

```
    System.out.println("Additional Details: "+details);
}
else if(Event_Choice == 2)
{
    System.out.println("Festival's name: "+Festival_name);
    System.out.println("Date of the Festival: "+date);
    System.out.println("Venue: "+venue);
    System.out.println("Theme: "+theme);
    System.out.println("Number of people: "+Number_of_people);
    System.out.println("Offerings: "+offerings);
    System.out.println("Additional Details: "+details);
}
else if(Event_Choice == 3)
{
    System.out.println("Date of the Wedding: "+date);
    System.out.println("Venue: "+venue);
    System.out.println("Theme: "+theme);
    System.out.println("Number of people: "+Number_of_people);
    System.out.println("Food Items: "+food);
    System.out.println("Cards: "+Cards);
    System.out.println("Additional Details: "+details);
}
else if(Event_Choice == 4)
{
    System.out.println("Date of the Engagement: "+date);
    System.out.println("Venue: "+venue);
    System.out.println("Theme: "+theme);
    System.out.println("Number of people: "+Number_of_people);
    System.out.println("Stage settings: "+Stage_settings);
    System.out.println("Food Items: "+food);
    System.out.println("Additional Details: "+details);
}
else if(Event_Choice == 5)
{
    System.out.println("Date of the Reception: "+date);
    System.out.println("Venue: "+venue);
    System.out.println("Theme: "+theme);
    System.out.println("Number of people: "+Number_of_people);
    System.out.println("Stage settings: "+Stage_settings);
    System.out.println("Cake flavours and description: "+cake);
    System.out.println("Food Items: "+food);
    System.out.println("Cards: "+Cards);
```

```
            System.out.println("Additional Details: "+details);
        }
        else if(Event_Choice == 6)
        {
            System.out.println("Date of the Baby Shower: "+date);
            System.out.println("Venue: "+venue);
            System.out.println("Theme: "+theme);
            System.out.println("Number of people: "+Number_of_people);
            System.out.println("Cake flavours and description: "+cake);
            System.out.println("Food Items: "+food);
            System.out.println("Cards: "+Cards);
            System.out.println("Additional Details: "+details);
        }
        else if(Event_Choice == 7)
        {
            if(SchoolEvent_Choice == 1)
            {
                System.out.println("Date of the Annual Day: "+date);
                System.out.println("Venue: "+venue);
                System.out.println("Theme: "+theme);
                System.out.println("Number of people: "+Number_of_people);
                System.out.println("Food Boxes: "+Food_boxes);
                System.out.println("Stage settings: "+Stage_settings);
                System.out.println("Awards and Certificates: "+Awards_certificates);
                System.out.println("Additional Details: "+details);
            }
            else if(SchoolEvent_Choice == 2)
            {
                System.out.println("Date of the Sports Day: "+date);
                System.out.println("Venue: "+venue);
                System.out.println("Theme: "+theme);
                System.out.println("Number of people: "+Number_of_people);
                System.out.println("Food Boxes: "+Food_boxes);
                System.out.println("Medals, Trophies and and Certificates:
"+Medals_Trophies_Certificate);
                System.out.println("Additional Details: "+details);
            }
            else if(SchoolEvent_Choice == 3)
            {
                System.out.println("Date of the Graduation Day: "+date);
                System.out.println("Venue: "+venue);
                System.out.println("Theme: "+theme);
```

```
            System.out.println("Number of people: "+Number_of_people);
            System.out.println("Food Boxes: "+Food_boxes);
            System.out.println("Stage settings: "+Stage_settings);
            System.out.println("Certificates: "+Certificate);
            System.out.println("Additional Details: "+details);
        }
        else if(SchoolEvent_Choice == 4)
        {
            System.out.println("Date of General Events: "+date);
            System.out.println("Venue: "+venue);
            System.out.println("Theme: "+theme);
            System.out.println("Number of people: "+Number_of_people);
            System.out.println("Food Boxes: "+Food_boxes);
            System.out.println("Stage settings: "+Stage_settings);
            System.out.println("Additional Details: "+details);
        }
    }
    else if(Event_Choice == 8)
    {
        System.out.println("Date of the Party: "+date);
        System.out.println("Venue: "+venue);
        System.out.println("Theme: "+theme);
        System.out.println("Number of people: "+Number_of_people);
        System.out.println("Food Items and Drinks: "+Food_drinks);
        System.out.println("Additional Details: "+details);
    }
    else if(Event_Choice == 9)
    {
        System.out.println("Date of the Office Party: "+date);
        System.out.println("Venue: "+venue);
        System.out.println("Theme: "+theme);
        System.out.println("Number of people: "+Number_of_people);
        System.out.println("Food Items and Drinks: "+Food_drinks);
        System.out.println("Additional Details: "+details);
    }
    else if(Event_Choice == 10)
    {
        System.out.println("Date of the Event: "+event);
    }
}
public static void main(String []args)
{
```

```
        //creating an object for the Event_Management class
        Event_Management obj = new Event_Management();

        Scanner sc = new Scanner(System.in);
        System.out.println("To start of, please enter the following details");
        System.out.println("Name: ");
        String Customer_Name = sc.next();
        System.out.println("Contact Number: ");
        int Contact_number = sc.nextInt();
        System.out.println("Address: ");
        String Address = sc.next();
        System.out.println("Email ID: ");
        String Email_ID = sc.next();

        obj.choose_event();
        obj.design();
        //creating object for the interface class
        Display obj1 = new Event_Management(Customer_Name,
Contact_number, Address, Email_ID);
        obj1.Details();
        //creating object for the interface class
        Display_Details obj2 = new Event_Management();
        obj2.EventDetails();
        //creating object for the abstract class
        MessageEmail obj3 = new MessageEmail();
        obj3.Email();
        obj3.Design();

    }
}
//inheritance
class Birthday_Party extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Birthday Party");
        System.out.println("Date of the Party: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
```

```
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Cake flavors and description: ");
        cake = sc.next();
        System.out.println("Food Items: ");
        food = sc.next();
        System.out.println("Cards and Gifts: ");
        cards_gifts = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Festivals extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Festival");
        System.out.println("Festival's Name: ");
        Festival_name = sc.next();
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Offerings: ");
        offerings = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Weddings extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Wedding");
        System.out.println("Date: ");
```

```
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food items: ");
        food = sc.next();
        System.out.println("Cards: ");
        Cards = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Engagements extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Engagement");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Stage Settings: ");
        Stage_settings = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food items: ");
        food = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Receptions extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
```

```
        System.out.println("Enter the details for the Engagement");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Stage Settings: ");
        Stage_settings = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Cake flavors and description: ");
        cake = sc.next();
        System.out.println("Food: ");
        food = sc.next();
        System.out.println("Cards: ");
        Cards = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Baby_Shower extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Baby Shower");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Cake flavors and description: ");
        cake = sc.next();
        System.out.println("Food: ");
        food = sc.next();
        System.out.println("Cards: ");
        Cards = sc.next();
        System.out.println("Additional details: ");
```

```
        details = sc.next();
    }
}
class School_Events extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("School events");
        System.out.println("1. Annual day");
        System.out.println("2. Sports Day");
        System.out.println("3. Graduation day");
        System.out.println("4. General Event");
        System.out.println("5. Others");
        System.out.println("Choose the School Event");
        SchoolEvent_Choice = sc.nextInt();
        switch(SchoolEvent_Choice)
        {
            case 1 :
                Annual_Day obj1 = new Annual_Day();
                obj1.input();
                System.out.println("");
                break;
            case 2 :
                Sports_Day obj2 = new Sports_Day();
                obj2.input();
                System.out.println("");
                break;
            case 3 :
                Graduation_Day obj3 = new Graduation_Day();
                obj3.input();
                System.out.println("");
                break;
            case 4 :
                General_Events obj4 = new General_Events();
                obj4.input();
                System.out.println("");
                break;

            default:
                System.out.println("Please enter a school event and its details");
        }
```

```
    }
}
class Parties extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Party");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food and drinks: ");
        Food_drinks = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
class Office_Parties extends Event_Management
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Party");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food and drinks: ");
        Food_drinks = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
}
```

```
class Others extends Event_Management
{
   static void input()
   {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the details for the Event");
      event = sc.nextLine();
   }
}
class Annual_Day extends School_Events
{
   static void input()
   {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the details for the Annual Day");
      System.out.println("Date: ");
      date = sc.next();
      System.out.println("Venue: ");
      venue = sc.next();
      System.out.println("Theme: ");
      theme = sc.next();
      System.out.println("Number of people: ");
      Number_of_people = sc.nextInt();
      System.out.println("Food boxes: ");
      Food_boxes = sc.next();
      System.out.println("Awards and Certificates: ");
      Awards_certificates = sc.next();
      System.out.println("Stage Settings: ");
      Stage_settings = sc.next();
      System.out.println("Additional details: ");
      details= sc.next();
   }
   public static void main(String []args)
   {
      Annual_Day obj = new Annual_Day();
      obj.input();
   }
}
class Sports_Day extends School_Events
{
   static void input()
   {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Sports Day");
        System.out.println("Date: ");
        date= sc.next();
        System.out.println("Venue: ");
        venue= sc.next();
        System.out.println("Theme: ");
        theme= sc.next();
        System.out.println("Number of people: ");
        Number_of_people= sc.nextInt();
        System.out.println("Food boxes: ");
        Food_boxes = sc.next();
        System.out.println("Medals, Trophies and Certificates: ");
        Medals_Trophies_Certificate= sc.next();
        System.out.println("Additional details: ");
        details= sc.next();
    }
    public static void main(String []args)
    {
        Sports_Day obj = new Sports_Day();
        obj.input();
    }
}
class Graduation_Day extends School_Events
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Graduation Day");
        System.out.println("Date: ");
        date = sc.next();
        System.out.println("Venue: ");
        venue = sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food boxes: ");
        Food_boxes = sc.next();
        System.out.println("Certificates: ");
        Certificate = sc.next();
        System.out.println("Stage Settings: ");
```

```
        Stage_settings = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
    public static void main(String []args)
    {
        Graduation_Day obj = new Graduation_Day();
        obj.input();
    }
}
class General_Events extends School_Events
{
    static void input()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the details for the Event");
        System.out.println("Date: ");
        date= sc.next();
        System.out.println("Venue: ");
        venue= sc.next();
        System.out.println("Theme: ");
        theme = sc.next();
        System.out.println("Number of people: ");
        Number_of_people = sc.nextInt();
        System.out.println("Food boxes: ");
        Food_boxes = sc.next();
        System.out.println("Stage Settings: ");
        Stage_settings = sc.next();
        System.out.println("Additional details: ");
        details = sc.next();
    }
    public static void main(String []args)
    {
        General_Events obj = new General_Events();
        obj.input();
    }
}
//interface
interface Display
{
    public void Details();
}
```

```
interface Display_Details extends Display
{
   public void EventDetails();
}
//abstract
public abstract class EmailMessage extends Event_Management
{
   void Email()
   {
      //exception
      try
      {
         if(Event_Choice == 1 || Event_Choice == 2 || Event_Choice == 3 ||
Event_Choice == 4 || Event_Choice == 5
         || Event_Choice == 6 || Event_Choice == 7 || Event_Choice == 8 ||
Event_Choice == 9 || Event_Choice == 10)
         {
            System.out.println("");
            System.out.println("");
            throw new Message("The details of your choices have been sent to
your registered Email ID");
         }
         else
         {
            System.out.println("Invalid Input");
         }
      }
      catch(Message n)
      {
         System.out.println(n);
      }
   }
}
class MessageEmail extends EmailMessage
{
   void Design()
   {
      System.out.println("");
      System.out.println("");
      for (int i=1; i<=30 ; i++)
      {
         System.out.print(" *  ");
```

```
        }
        System.out.println("");
        for(int j=1; j<=30; j++)
        {
           System.out.print("*** ");
        }
        System.out.println("");
        System.out.println("");
    }
}
class Message extends Exception
{
    public Message(String n)
    {
        super(n);
    }
}
```

# OUTPUT

## Output 1:

```
Welcome to Creative corner
To start of, please enter the following details
Name:
Naina
Contact Number:
1234567890
Address:
Marathalli,Bangalore
Email ID:
Naina123@gmail.com
Choose the event:
1. Birthday Party
2. Festivals
3. Weddings
4. Engagements
5. Receptions
6. Baby Shower
7. School Events
8. Parties
9. Office Parties
10. Others
1
Enter the details for the Birthday Party
Date of the Party:
15-03-2021
Venue:
Koramangla
Theme:
Black&White
Number of people:
15
Cake flavors and description:
Chocolate,Layered
Food Items:
Chips,Drinks,Fries,Burger,Pizza
Cards and Gifts:
InvitaionCards,Flowers
Additional details:
ChairsAndTables

 *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***

****************** CUSTOMER DETAILS ******************
Name: Naina
Contact Number: 1234567890
Address: Marathalli,Bangalore
E-Mail ID: Naina123@gmail.com
******************* EVENT DETAILS ******************
Date of the Party: 15-03-2021
Venue: Koramangla
Theme: Black&White
Number of people: 15
Cake flavours and description: Chocolate,Layered
Food Items: Chips,Drinks,Fries,Burger,Pizza
Cards and Gifts: InvitaionCards,Flowers
Additional Details: ChairsAndTables


Message: The details of your choices have been sent to your registered Email ID


 *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
```

# Output 2:

```
Welcome to Creative corner
To start of, please enter the following details
Name:
Ravi
Contact Number:
23456786
Address:
Koramangla
Email ID:
ravidesai12345@yahoo.com
Choose the event:
1. Birthday Party
2. Festivals
3. Weddings
4. Engagements
5. Receptions
6. Baby Shower
7. School Events
8. Parties
9. Office Parties
10. Others
12

 *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***

****************** CUSTOMER DETAILS ******************
Name: Ravi
Contact Number: 23456786
Address: Koramangla
E-Mail ID: ravidesai12345@yahoo.com
******************* EVENT DETAILS *******************
Invalid Input


 *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *   *
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
```

# CONCLUSION

Java is one of the world's most widely used computer language and Object oriented programming is a part of Java wherein we create objects which contain both data as well as methods. It is faster as well as easy to execute and provides a clear structure for the programs.

In this program of ours, we have used nine object oriented concepts and have implemented these on a BlueJ software. We have also implemented a UML Diagram which shows the basic structure of our program.