**Assignment-7**

**1.OOPS concept**

* Object

Any entity that has state and behavior is known as an object. For example, a chair, pen,

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory.

## Class

Collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

### Inheritance

When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

### Polymorphism

If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.

* Overriding and Overloading are two very important concepts in Java.
* Overloading occurs when two or more methods in one class have the same method name but different parameters.
* Overriding means having two methods with the same method name and parameters (i.e., method signature). One of the methods is in the parent class and the other is in the child class. Overriding allows a child class to provide a specific implementation of a method that is already provided its parent class.
* Overriding is a run-time concept while overloading is a compile-time concept.

#### Abstraction

Hiding internal details and showing functionality is known as abstraction. For example phone call, we don't know the internal processing.

In Java, we use abstract class and interface to achieve abstraction.

### Encapsulation

Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule, it is wrapped with different medicines.

A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

# 2.Structures

A structure, is basically a collection of elements of **Same/Different** data types.Keyword is used for structure is struct.

**3.Type casting**

Type casting is when you assign a value of one primitive data type to another type.  When you assign the value of one data type to another, you should be aware of the compatibility of the data type. If they are compatible, then [Java](https://www.edureka.co/blog/what-is-java/) will perform the conversion automatically known as Automatic Type Conversion and if not, then they need to be casted or converted explicitly.

In Java, there are two types of casting:

* Widening Casting (automatically) - converting a smaller type to a larger type size  
  byte -> short -> char -> int -> long -> float -> double
* Narrowing Casting (manually) - converting a larger type to a smaller size type  
  double -> float -> long -> int -> char -> short -> byte

**4.Json**

JSON stands for JavaScript Object. It is an independent data exchange format and is the best alternative for XML. This chapter explains how to parse the JSON file and extract necessary information from it.

Android provides four different classes to manipulate JSON data. These classes are JSONArray,JSONObject,JSONStringer and JSONTokenizer.

**5.RecyclerView**

Android RecyclerView is a more advanced, powerful and flexible version of the [ListView](https://www.journaldev.com/9247/android-listview-example-tutorial). Android RecyclerView is similar to ListView except that it forces us to use **RecyclerView.ViewHolder** class to hold the elements which is not a compulsion in ListView

As the name suggests, Android RecyclerView is used to reuse cells when scrolling up and down by recycling the items in the list. Another improvement in RecyclerView is that it allows us to set the [LayoutManagers](https://www.journaldev.com/9495/android-layout-linearlayout-relativelayout) dynamically at runtime, unlike the ListView which was only available in a Vertical scrolling List. RecyclerView allows us to set the following types of Layouts at runtime.

Classes

* The RecyclerView.ItemAnimator class provides better support to animating the views unlike the ListViews
* The RecyclerView.ItemDecorator class provides better support when it comes to adding borders and dividers thereby giving huge control to us

**6.Cardview**

Android CardView UI component shows information inside cards. This component is generally used to show contact information. This component is available in another support library so we have to add its dependency too.

Android CardView widget allows us to control the background color, shadow, corner radius, elevation etc. For using the custom attributes in XML, we need to add the following namespace declaration to the parent layout. Following is the namespace declaration with some attributes from our project.

**7.Registering.**

In android **Event Registration** is the process by which an Event Handler gets registered with an **Event Listener** so that the handler is called when the Event Listener fires the event.

The different ways to register event listeners in our android applications.

* By specifying an event handlers directly in **activity\_main.xml** file, we can register event listeners.
* By using [Activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle) class that implements a listener interface, we can register event listeners.
* By using an anonymous class.

**8. Event Handler.**

In android, **Event Handlers** are useful to define a several callback methods when we are building a custom components from view.