**1.**

1. Object

Object is an instance of a class. In each object is the behavior of a class described individually. For example, in class dog, one dog can have black hair and the other one brown.

Example var;

2. Class

A class defines common behavior and state for objects.

Class example{

int number = 0;

public void increment(int numberToIncrement){

numberToIncrement ++;

}

}

3. Instantiation of object (creating an object)

Instance of a class is a variable and reference of an object to specific part of memory.

Example var = new Example();

4. Visibility (public / private / protected)

Public - accessibility from all classes within the project

Private - accessible only within the class where is defined

Protected - accessible from the class where is defined and subclasses

5. Member datas / methods

Members can be described as characteristics of a class. In a programming it refers to a variable or a function which can be used by a specific object.

number = 0;

var.increment(number);

number = var.number;

6. Inheritance

Subclass inherits all behavior and state from its super class. The subclass can add its own state and behavior.

Class Animal {  
 public void animalGreeting(Sytem.out.println(“Hi, I am animal”);  
}  
  
class Pig extends Animal {  
 String name;  
}

class MainActivity {  
 onCreate {  
 Pig myPig = new Pig();    
    System.out.println(myPig.animalGreeting() + “ My name is: “ +

myPig.name);  
 }  
}

7. Interface

Interface is similar to a class but can only contain method signatures.

Interface cannot be instantiated, does not contain any constructors and all methods are abstract.

interface Animal {  
 public void animalSound();  
 public void sleep();   
}  
  
class Pig implements Animal {  
 public void animalSound() {   
 System.out.println("The pig says: wee wee");  
}  
 public void sleep() {   
 System.out.println("Zzz");  
 }  
}

class MainActivity {  
 onCreate {  
 Pig myPig = new Pig();  // Create a Pig object  
    myPig.animalSound();  
     myPig.sleep();  
 }  
}

8. Polymorphism

Polymorphism occurs when there is a lot of instances of Super class and they have all common certain features. Is supported with overriding rather than abstract classes and interfaces in case of Abstraction.

class Animal {  
  public void animalSound() {  
    System.out.println("The animal makes a sound");  
  }  
}  
  
class Pig extends Animal {  
  public void animalSound() {  
    System.out.println("The pig says: wee wee");  
  }  
}  
  
class Dog extends Animal {  
  public void animalSound() {  
    System.out.println("The dog says: bow wow");  
  }  
}

9. Overriding

Defining a behavior that is specific to a subclass type and it can implement parent class based on its requirements

Final methods cannot be overridden

Class Animal {  
 public void animalGreeting(Sytem.out.println(“Hi, I am animal”);  
}  
  
class Pig extends Animal {  
 String name;

@Overrides

public void animalGreeting(Sytem.out.println(“Overriden greeting”);

}

class MainActivity {  
 onCreate {  
 Pig myPig = new Pig();    
    System.out.println(myPig.animalGreeting() + “ My name is: “

+ myPig.name);  
 }  
}

10.Abstract classes

Cannot be instantiated and it needs to be extended. Allows to view things in more general terms and enables easier dealing with the changes that may come.

abstract class Animal{

abstract void run();

}

Class Dog extends Animal{

void run() { Sytem.out.println(“I am running”);

}

Class MainActivity{

OnCreate{

Dog obj = new Animal();

obj.run();

}

}

**2.**

<?xml version="1.0" encoding="utf-8"?>  
<android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"  
xmlns:app="http://schemas.android.com/apk/res-auto"  
xmlns:tools="http://schemas.android.com/tools"  
android:layout\_width="match\_parent"  
android:layout\_height="match\_parent"  
tools:context=".MainActivity">  
  
 <android.support.constraint.ConstraintLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 tools:layout\_editor\_absoluteX="77dp">  
  
 <EditText  
 android:id="@+id/editText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="Time input"  
 android:inputType="time" />  
  
 <Switch  
 android:id="@+id/switch1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Switch" />  
  
 <Button  
 android:id="@+id/button7"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Button" />  
</LinearLayout>  
  
<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginBottom="32dp"  
 android:orientation="horizontal"  
 app:layout\_constraintBottom\_toTopOf="@+id/textView"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:layout\_editor\_absoluteX="0dp">  
  
 <CheckBox  
 android:id="@+id/checkBox3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="ChceckBox3" />  
  
 <CheckBox  
 android:id="@+id/checkBox2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="CheckBox2" />  
  
 <CheckBox  
 android:id="@+id/checkBox"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="CheckBox1" />  
  
 </LinearLayout>  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 </android.support.constraint.ConstraintLayout>  
   
</android.support.design.widget.CoordinatorLayout>

**3.**

1. What programming languages you can use for Android app development?

Java, Kotlin, C++

2. What is .apk file?

.apk file is a file which contains build of a project and it is a file Android device use to install the app.

3. How android runs the app.

Linux user app, each app has different ID. Each app runs in separate virtual machine.

4. Components of an Android app

Activities: Activity is content what users can see. It is a view. An app may consist of more activities, usually every view of an app is a different activity. Example in an email app, one activity serves as a mail view and other as creating a new mail.

Services: Operates running of an app and all processes that are needed to be carried out.

Broadcast receivers: Receives broadcast messages. Listens to an event, ex. Whether the device is being charged. If this condition happens, something reacts to it.

Content providers: for an app, content providers are fundamental to query data

5. What is manifest file and what is its purpose?

Information for an Android system about app existence and all its components, which much be declared in the manifest file.

Declares API level, required hardware/ software features, and libraries that needs to be linked.

6. What are resources? Why they are needed?

Resources can be images, audio files or anything related to the visual concept on an app.