Android Development

MAD Lab

Lecture-1

- Before writing any program, make sure that the appropriate version of Java Development Kit (JDK) is available on your machines. You can start by saving your source files in bin sub-directory of your jdk directory. Use javac and java to compile and run your program source files and class files, respectively.

First Program

```
class First {
public static void main ( String args[ ]){
System.out.println( “Hello World” );
 }
}
```

Command Line Arguments

- Command line arguments are passed at the time of executing a java program, e.g. if we pass command line argument in the class First in our first problem, we would have done something like this : java First cat Here, cat will be passed as an argument and will be collected at the args[0] array of string
- Attempt Lab Exercise 1 with the ideas discussed

Lecture 2 Inheritance

- What is Inheritance?
- Test for Inheritance Make a class Mother with int x and void show ( ) as its members.
- Now create another class Child that is derived from the Mother class. Create another class Application to test the inheritance.
- class Mother {.....}
- class Child extends Mother {// Empty Body }
- Note: “//” is used for single line comments in java.
- class Application { public static void main (String args[]){
- Mother m= new Mother ( ); m.show( ); // show of Mother is called Child ch=new Child ( );
- ch. show ( ); // show ( ) inherited in Child from Mother is called } }

Overriding

- What is overriding? How is it different from Inheritance?
- Test for Overriding Make a function with the same return type , name of the function, number and type of arguments in the Child class as they are in the Mother class.
- Change the string to be displayed on the screen.
- For example, if Mother class version of show ( ) was displaying “Hello World” then the child class version of show ( ) will display “Hello JUET”.
- Run the application class and discuss the results obtained with your instructor.

Polymorphism

- What are the different kinds of polymorphism?
- Test for Polymorphism
- Write another statement in Application class: Mother m1=new Child ( );
- Now, call show () with reference variable m1 and '.' (dot) operator. Now, make the method show ( ) static in Mother only and check the results after executing Application.
- Repeat this by making show () static in Child only.
- Observe the errors. Test whether the static methods are inheritable or not.
- Now make show () static in both Mother as well as in Child and discuss the results with your instructor.
- Make a table to summarize your results.
- Attempt Lab Exercise 2 with the ideas discussed

Lecture 3 (Summary of Lab Exercise 3)

- Follow the instructions of your instructor to install Android Studio on your machines.
- Get familiar with the Android Studio environment.
- Create an application that displays your name in the center of the emulator screen.
- Create a mobile application having a button with caption ‘Click’.
- On clicking this button a message that is there in a text box should be displayed for a while.
- Create an app containing a picture of Donald Trump on the full mobile screen. On clicking the Trump’s picture, the picture of Indian Prime Minister should appear on the mobile screen.
- On clicking the Indian PM’s picture, picture of Trump should appear again.

Lecture 4

- Views
- View Types: Image View, Recycler View, Button View
- An android application is typically made up of these views
- Views are rendered over an activity
- Multiple Activities are there in an android application
- Follow the instruction carefully as demonstrated and solve the lab exercise 4.

Lecture 5

- Layouts are typed canvases where you lay your activity components.
- GridLayout is a special layout which arranges the components (buttons, image views, etc) in a typical row, column fashion.
- rowCount, colCount are the properties that need to be set for a GridLayout.
- We will learn working on GridLayout using project development.
- We will develop a TicTacToe game.
- We will start with very elementary features and evolve this project with time.
- Follow the instruction carefully as demonstrated and solve the lab exercise 5.

Lecture 6

- Alert Dialogues are used to prompt a user.
- Typically Yes|No|Neutral buttons are placed on an alert dialogue.
- You use AlertDialogue interface to implement Alert Dialogues.
- Alert Dialogue typically consists of:
- Icon: this icon can be inbuilt or customized
- title: title that is shown on the title bar
- message: Message that is to be displayed
- buttons: yes, no, neutral buttons.
- Follow the instruction carefully as demonstrated and solve the lab exercise 6.

Lecture 7

- Creating Menus
- Menu creation is common in web and mobile app development.
- Android Studio framework provides a tool based way to create menus in your app.
- You need to create a menu directory in your app structure, followed by generating a menu resource file.
- Don’t worry, you will see it through demonstration.

Lecture 8

- Shared Preferences
- Shared Preferences is a class, which is used to store the user's credentials or choices in his phone itself.
- When the user opens the app again, these stored credentials are used to customize the app as per his/her choices.
- You need to create an instance of the shared preferences class using:
- SharedPreferences sp=this.getSharedPreferences();
- Now, you can use sp to store and retrieve the credentials as per the requirement.
- Note: You generally store strings as key and value in a shared preferences object. If you want to store some complex objects, then Gson library is to be imported in you android studio's gradle.build file at the module level.

Lecture 9

- Project Arithmetic
- In this session, we will develop a project that kids can use to improve their arithmetic skills.
- This Project is a partially built project on Simple Arithmetic Game for *single player*.
- Most of the coding part is already done, however, some clues are left in between (comments //) where students can write their codes.
- **matchCounter** will keep track of how many matches are played. A match is akin to answer one question.
- If a player correctly answers a match, a one is added in the score array; otherwise, a zero will be added.

Lecture 9 contd..

- When matchCounter reaches a value 3, the sum of last three matches will be added to the **performance** array, which is initially set with -1 at all places.
- For example, the performance array at the start of your app will be like [-1, -1, -1, -1, -1, -1], but after the completion of three matches let the score array be like [1, 0, 1] then the performance array would be like [-1, -1, -1, -1, -1, 2].
- Each time the performance array will be updated, the new value will be added at the last element of the array, and all previous values will be shifted one place left. The first value will be lost, as we have to analyze the performance of the last six gammes only.
- The performance array will be stored in a SharedPreferences object. The SharedPreferences object is fetched at the start of the activity and an alert dialog is presetned to the user suggesting on his or her performance based on the last six games.
- **sumOfScore** method should summing up the values of score array.

Lecture 9 contd…

- A class **LR** is also provided with this project. The LR class has a getSlope method to provide the slope fromo the given data.
- The data must be input to the **getSlope** method as a two-dimensional array (dataFrame). The first dimension will be the predictor values 1, 2, 3, 4, 5, 6 and the corresponding response values will be performance values.
- The getSlope method is a static method, so you won't need to create objects of the LR class to obtain slope. Just use the name of the class LR. For example: LR.getSlope(dataFrame) would return a slope as a double value.
- **dataPrep** method generates a data-frame AKA two-dimensional array to be passed to the getSlope method.
- **getInterpretation** method returns a string based on the analysis of slope obtained from analysis of past six performances.