College of Informatics and Virtual Education

Department of Computer Science and Engineering

CP 313: MOBILE APPLICATIONS DEVELOPMENT

Test One (50 Marks)

QUESTION ONE

a. Write True or False for the following statements: - **(1 Mark Each)**

i. A fragment has its own layout and its own behavior with its own life cycle callbacks methods.

ii. You can add or remove fragments in an activity while the activity is running.

iii. You can combine multiple fragments in a single activity to build a multi-pane user interface (UI).

iv. A fragment can be used in multiple activities.

v. Fragment life cycle is closely related to the life cycle of its host activity.

vi. A fragment can implement a behavior that has no user interface component.

vii. Fragments were added to the Android API in Honeycomb version of Android which API version 11.

b. To add a fragment dynamically to an activity, you first need to define the container view in xml file where the fragment will be hosted. Through the instances of **FragmentManager** and **FrgamentTransaction** you can add, remove or replace fragment in an activity. Use the FragmentTransaction methods to accomplish each of the following: -

i. To create the object of the FragmentManager. **(3 Marks)**

ii. To create the object of the fragmentTransaction. **(3 Marks)**

iii. Use the object of the FragmentTransaction in (ii) to add a fragment called **fragmentOne** to a fragment container called **fragment_container_view.** **(5 Marks)**

QUESTION TWO

a. Give the Outline of the contents of an XML file, called **main.xml**, that would generate the layout shown in Figure 1. **(5 Marks)**

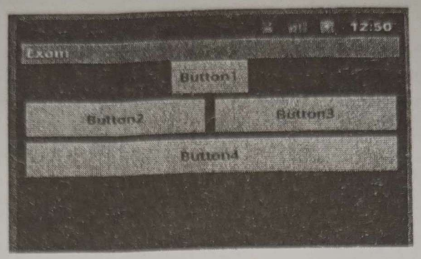

Figure 1: Layout

b. Below is a part of activity java class named MainActivity. Ids for the given widgets are btnSubmit, rgGender for radio group, rbFemale and rbMale.

```
public class MainActivity extends AppCompatActivity {
   Button submit;
   RadioGroup gender;
   RadioButton Female, Male;
   EditText username, email;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView();

        submit.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
            }
        });
        }}
```

i. Write code statement(s) to capture the ids of the given widgets and store them in a respective variables. **(3 Marks)**

ii. Write a code statement(s) within **setContentView** method to link the given activity (**MainActivity**) with its XML file named **activity_main.xml.** **(2 Marks)**

iii. Use Android Bundle class to send the information (username, email, gender) to another activity called **Activity2.java** (NB: Android Bundle is used to pass data between activities). **(8 Marks)**

iv. Use TOAST to display a message **"You have clicked Me"** when the submit button is clicked. **(2 Marks)**

QUESTION THREE

Figure 2, illustrates the paths an activity might take between states. The rectangles represent the callback methods you can implement to perform operations when the activity transitions between states. Fill the rectangles numbered 46, 47,48,49,50 and 51with the name of callback methods.

(1 Mark1 Each)

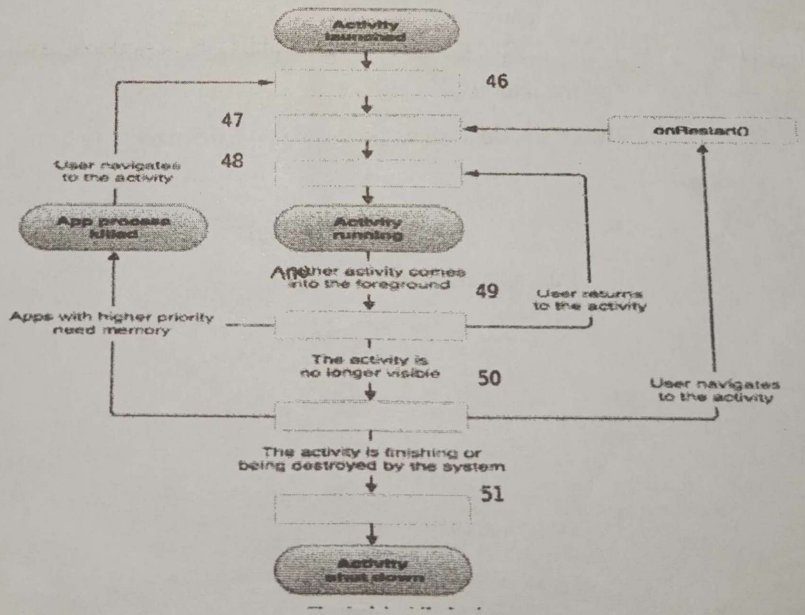

Figure 2: Activity Life Cycle

QUESTION FOUR

Below is a part of androidManifest.xml file.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.myapplication18">
<application
  android:allowBackup="true"
  android:icon="@mipmap/ic_launcher"
  android:label="@string/app_name"
  android:roundIcon="@mipmap/ic_launcher_round"
  android:supportsRtl="true"
  android:theme="@style/Theme.MyApplication">
  <activity
    .....
   </activity>
    </application>

  </manifest>
```

i. Write a code to show how you add a permission to allow this app to connect to the internet. **(2 Marks)**

ii. Write a code to show how you make the activity to be the launcher activity. **(2 Marks)**

iii. Write a code to show how you register the activity called MyActivity in an Android manifest.xml. **(2 Marks)**