Project 1 Task 4 Distributed Systems

1)A)a)

```
listView = (ListView) findViewById(R.id.listView);
sManager = (SensorManager) this.getSystemService(this.SENSOR_SERVICE);
sList = sManager.getSensorList(Sensor.TYPE_ALL);
List<String> sNames = new ArrayList();
for (int i = 0; i <sList.size(); i++){
    sNames.add(((Sensor)sList.get(i)).getName());
}
listView.setAdapter(new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, sNames));
```

b)

```
SensorManager sManager = (SensorManager) getSystemService(SENSOR_SERVICE);
Sensor accel = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
float range = accel.getMaximumRange();
```

c)

(in a class that implements SenesorEventListener)

```
SensorManager sManager = (SensorManager) getSystemService(SENSOR_SERVICE);
Sensor accel = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
sManager.registerListener(this, accel, SensorManager.SENSOR_DELAY_FASTEST);
```

B)

The values of the sensors are never read (with event.values()) and hence never end up in the respective arrays. Hence there is no data read to display.

2.)

a)

An event to start B is dispatched. When B is started, activity A is paused – onPause() is called, and, if activity B completely covers activity A, onStop() is called. The system might now kill the activity at any time to gain free space.

When activity B receives the Intent, onCreate, onStart and onResume are called, and then the activity is running and in the foreground.

b)

As described above, onStop() is called, and the activity is suspended and might be killed by the system. nDestroy() would then be called.

c)

onPause() and onStop() are called for activity B, for activity A it depends – if it was destroyed by the system beforehand, it will be restarted and onCreate(), onStart() and onResume() will be called. Otherwise, onRestart(), onStart() and onResume are called.

3.)

There are different ways to support different screen sizes. The maybe simplest one is to declare sizes with attributes like match-parent or wrap-content, and pixel sizes in dp, which scale up according to screen sizes, to retain the pixel density. Additionally one can provide different layouts and bitmaps for different screen sizes in the resources.

4.)

An intent describes an intended operation, which is normally passed to the system by some other function, and the system normally performs it asynchronously. The action mostly consists of starting or communicating with an activity or service, broadcasting something or binding a service.

An explicit intent already specifies, which class or component will handle the intent, while an implicit intent does not specify this. Instead, it specifies which action should be performed (and additional criteria), and the system then chooses an activity which is able to handle this task for you.

5.)

a) wrong. Can be stopped with stopService() simply.

b) true. First of all, services started with startService() can still be bound to interact with other services/processes. Secondly, of course there are other ways (e.g. over a memory object) to communicate between the service and another process.

c) true. As soon as all clients are unbound, the service is killed by the system.

d) false. For this, one should use an IntentService.

6.)

firstly, a service tag for the location service is missing.

```
<service android:name="ch.ethz.inf.vs.android.netz.antitheft.LocationService"
    android:exported="false"/>
```

secondly, we need permissions to send sms and access the fine grained location:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.SEND_SMS" />
```

Thats all :) I hope you enjoyed the extraordinary insights and discoveries experssed in this document.