Distributed Systems

Mini-Test

1 Sensor Framework

A

a)

Given that this is written inside a function in a class, which extends AppCompatActivity and imports the necessary classes

```
SensorManager sensorMgr = (SensorManager) getSystemService(SENSOR_SERVICE);
List<Sensor> sensors = sensorMgr.getSensorList(Sensor.TYPE_ALL);
for(Sensor currSens : sensors){
        System.out.println(currSens.getName());
}
```

In sensors all sensors that are available will be stored in a list. To get one or print it's name value, one must only iterate through the list.

b)

Given a Sensor that got retrieved called "sensor"

```
sensor.getMaximumRange()
```

will give the Maximum Range back (though depending on the Manufacturer the returnValues may vary or be higher than actual measurable values)

c)

Given a SensorManager "mSensorManager" and a Sensor "Sensor" and that the class, in which this function is defined, implements SensorEventListener

```
@Override
protected void onResume() {
        super.onResume();
        if(Sensor != null && mSensorManager != null) {
        mSensorManager.registerListener(this, Sensor, SensorManager.SENSOR_DELAY_FASTEST);
}
```

Will register the Sensor "Sensor" with the maximum available rate (Ideally 0ms delay between 2 samples)

B

If the log really takes a large amount of time it is bad to call log inside the "onSensorChanged" function, because it will be called quite often and the log will block further actions.

2 Activity Lifecycle

The foreground activity (the activity at the top of the screen that the user is currently interacting with) is considered the most important. Its process will only be killed as a last resort, if it uses more memory than is available on the device. Generally at this point the device has reached a memory paging state, so this is required in order to keep the user interface responsive.

A visible activity (an activity that is visible to the user but not in the foreground, such as one sitting behind a foreground dialog) is considered extremely important and will not be killed unless that is required to keep the foreground activity running.

A background activity (an activity that is not visible to the user and has been paused) is no longer critical, so the system may safely kill its process to reclaim memory for other foreground or visible processes. If its process needs to be killed, when the user navigates back to the activity (making it visible on the screen again), its onCreate(Bundle) method will be called with the savedInstanceState it had previously supplied in onSaveInstanceState(Bundle) so that it can restart itself in the same state as the user last left it.

source: https://developer.android.com/reference/android/app/Activity.html

onResume() -> foreground activity
onPause() -> visible activity
onStop() -> background activity

3 Resources
It is easier to rename it if the string is used often and should be renamed due to design choices.
If you do this for all important literal strings, you have an overview over all in-use strings.
If you were to translate your app to different languages, it would be easy to support multiple languages that way.

4 Intent
An intent is an abstract description of an operation to be performed. It can be used with startActivity to launch an Activity, broadcastIntent to send it to any interested BroadcastReceiver components, and startService(Intent) or bindService(Intent, ServiceConnection, int) to communicate with a background Service.
An Intent provides a facility for performing late runtime binding between the code in different applications. Its most significant use is in the launching of activities, where it can be thought of as the glue between activities. It is basically a passive data structure holding an abstract description of an action to be performed.

Explicit Intents have specified a component (via setComponent(ComponentName) or setClass(Context, Class)), which provides the exact class to be run. Often these will not include any other information, simply being a way for an application to launch various internal activities it has as the user interacts with the application.

Implicit Intents have not specified a component; instead, they must include enough information for the system to determine which of the available components is best to run for that intent.

source: https://developer.android.com/reference/android/content/Intent.html

5 Service lifestyle
a) false
You can call stopService(intent) on the intent of the service or stopSelf() from within the service

b) false
Started services cannot return results/values or interact with it's starting component. Bound services on the other hand can send data to the launching component (client). So for example a bound service might be playing an audio file and sending data regarding audio start/pause/stop and the time elapsed to the launching Activity component so that the UI can be updated accordingly.

source: http://codetheory.in/understanding-android-started-bound-services/

c) true

d) false

6 Android Manifest file
following lines are missing :
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<service android:name"ch.ethz.inf.vs.android.nethz.locationsender.LocationService"/>