PBD
Spring 2020
Final Exam
03/06/2020
Time Limit: 105 Minutes

Name: _____

By signing my name above,
I agree to the University of Ljubljana
academic honesty policies.

This exam contains 2 pages and 4 questions.
Total of points is 50.

1. (13 points) The code shows an Arduino programme for reading temperature from a car engine's digital temperature sensor and printing the temperature on the serial output.
   (a) (5 points) Expand the code so that when the temperature goes above 110 degrees Celsius an alarm light (LED connected to pin 13) goes on. It should go off again, if the temperature drops below 100 degrees, but not before that.
   (b) (5 points) A crash sensor is connected to pin 3. This is a binary accelerometer that reports HIGH if the acceleration values are above the threshold that indicates a very sudden change (most likely a crash). Expand the code so that airbags (connected to pin 4) are deployed (by setting pin 4 to HIGH) if a crash is detected. Crash detection should be handled by an interrupt service routine connected to sensorPin and airbag deployment instruction(s) should be in the main loop;
   (c) (3 points) Explain the purpose of "volatile" keyword in line 8.

   ANSWER: The volatile keyword is intended to **prevent the compiler from applying any optimizations because the value of the variable can be changed by code outside the scope of the current code** at any time. The system always reads the current value of a volatile variable **form the memory location**, rather than keeping its value in a register. In this example, volatile ensures that in every iteration "if (crashState)" gets evaluated with a fresh value of the crashState, allowing for this variable to be changed within an interrupt service routine.

```
1  #include <OneWire.h>
2  #include <DallasTemperature.h>
3  #define ONE_WIRE_BUS 2
4
5  OneWire oneWire(ONE_WIRE_BUS);
6  DallasTemperature sensors(&oneWire);
7
8  volatile int crashState = 0;  // a variable for keeping the crash status
9  const int sensorPin = 3;      // the number of the crash sensor pin
10 const int airbagPin = 4;      // the number of the airbag release pin
11 const int ledPin =  13;       // the number of the LED pin
12
13 // ADDED (a):
14 bool highTemp = false;
15 float tempC;
16
17 void setup(void)
18 {
19   pinMode(sensorPin, INPUT);
20   pinMode(airbagPin, OUTPUT);
21   pinMode(ledPin, OUTPUT);
```

```
22
23    // ADDED (b):
24    attachInterrupt(digitalPinToInterrupt(sensorPin), crashDetect, RISING);
25
26    Serial.begin(9600);
27    sensors.begin();
28  }
29
30  void loop(void)
31  {
32    sensors.requestTemperatures(); // Send the command to get temperatures
33    Serial.print("Engine temperature is: ");
34
35    // ADDED (a):
36    tempC = sensors.getTempCByIndex(0));
37
38    Serial.print(tempC);
39
40     // ADDED (a):
41     if(tempC > 110) {
42         highTemp = true;
43     } else if (tempC < 100) {
44         highTemp = false;
45     }
46     digitalWrite(ledPin, highTemp);
47
48     // ADDED (b):
49     if (crashState==1) {
50         digitalWrite(airbagPin, HIGH);
51     }
52  }
53
54  // ADDED (b):
55  void crashDetect(void) {
56      crashState = 1;
57  }
```

2. (8 points) WWW Platform.
   (a) (4 points) In his 1996 article the author of WWW Tim Berners Lee ponders on the problem of verifying the correctness of the documents on the Web. He gives an example of buying a house online and checking whether "the owner has not sold off half of the back garden 10 years ago and haven't told you". Briefly explain how this could be solved today.

   ANSWER: Documents on the Web can be verified through **digital signatures**, where a trusted third party confirms that a document is indeed produced by the signed creator. An alternative to such a centralized verification is the **blockchain**, which distributes the verification process over a large number of miners.
   NOTE: mentioning either digital signatures or the blockchain is sufficient

   (b) (4 points) Tim Berners Lee also discusses the problem of "how do you get copies of heavily used documents out to as many places as you can?" How is this problem solved nowadays?

   ANSWER: Nowadays, copies of popular Web documents are stored closer to the users by relying on **Content Distribution Networks (CDNs)**, such as Akamai. CDN servers are deployed in networks of Internet Service Providers (ISPs) delivering end-user connectivity.

3. (15 points) Android.
   (a) (4 points) Briefly describe MVC programming pattern in Android; draw a diagram;

   ANSWER: MVC stands for Model-View-Controller. Model (often an SQLite database) manages the data, controller (often within an Activity or Fragment class) controller responds to the user input and performs interactions on the data model objects, while the View (often an XML layout) displays the data to the user. [Diagram in the lecture slides - Android Architecture Components].
   NOTE: general MVC (without the details on the Android implementation) is not sufficient for all points. A diagram without labels is not sufficient.

   (b) (3 points) What are the drawbacks of MVC in Android?

   ANSWER: Controller and View are tightly connected, thus if we want to change the View we often have to modify the Controller as well. In addition, the controller (Activity, Fragment) heavily depends on user interactions.

   (c) (4 points) Briefly describe MVVM programming pattern in Android; draw a diagram;

   ANSWER: MVVM stands for Model-View-ViewModel. The new component, the ViewModel, is responsible for wrapping the model and preparing observable data. The ViewModel does not know who is observing, can be more than one View; View binds to observable data invokes actions exposed by the ViewModel. [Diagram in

the lecture slides - Android Architecture Components].
NOTE: general MVVM (without the details on the Android implementation) is not sufficient for all points. A diagram without labels is not sufficient.

(d) (2 points) Explain the purpose of the ViewModel class.

ANSWER: The ViewModel class in Android provides the data, while not being aware of the views using the model (it can service multiple views), and it survives the Activity/Fragment lifecycle changes.

(e) (2 points) Explain the purpose of the LiveData class.

ANSWER: The LiveData class in Android holds the data and allows for it to be observed; the class **notifies** the **observers** when the data changes; **the class is lifecycle-aware**, thus it updates the observer when its lifecycle change requires so.

4. (14 points) You are designing an exercise monitoring Android smartphone application.

(a) (5 points) The application should track the time a user spends running and walking throughout the day. Describe in detail how you would implement this with the minimum use of energy. (NOTE: aim for API 26 or higher, your app can get destroyed when in the background)

ANSWER: We can detect running and walking activities using Google Activity Transition API. We create ActivityTransition objects for detecting when a user *enters* the *running* state, *exits* the *running* state, *enters* the *walking* state, and *exits* the *walking* state. We create an ActivityRecognitionRequest and supply the above transitions as well as a PendingIntent that will start an IntentService when a desired transition happens. Within the IntentService, we keep track of times when each of the transition happens and by summing the subtractions of exit and enter times calculate the total time spent in running and/or walking. Keeping the recorded times in persistent storage (e.g. SQLite DB) ensures that we don't lose the track of activity times in case an app is restarted. Further, the app should use a Foreground Service to minimize the chance of it being destroyed by the OS.
NOTE: Answers that don't use the Google Activity Transition API are fine if you clearly specify how periodic sensing is going to be performed (e.g. instantiate an AlarmManager in the Foreground Service to spawn an IntentService to sample sensors), describe which sensors are going to be used (e.g. accelerometer sensed for 10 seconds every two minutes, etc.), and how the machine learning part is implemented (e.g. a Markov Chain to track transitions in activities, while mean, variance, and mean crossing rate of accelerometer sensor data is extracted from the samples and pushed through a pre-trained activity recognition classifier from a third party library), etc.

(b) (5 points) About twenty minutes after a user is done "running" the app should send a notification with the summary of the run. Describe in detail how you would implement this – aim for accuracy of the notification delay.

ANSWER: In the IntentService that is called when an activity transition is detected, we check whether the transition is exit from the running state, and if so, calculate the summary of the last run. We then schedule an AlarmManager to fire an alarm at current time + twenty minutes. We provide another PendingIntent that starts an IntentService in charge of posting a notification to the user. To achieve the highest accuracy, we should use setExactAndAllowWhileIdle function when setting the alarm.
NOTE: Relying on WorkManager is inaccurate. Relying on Handlers works only for short periods of time (e.g. seconds) when a user is actively using the app.

(c) (4 points) When the phone is connected to a WiFi network and a charger, the app should send a notification with cumulative details of all exercise activity completed in the current day. Describe in detail how you would implement this.

ANSWER: We use WorkManager to create a (Periodic)WorkRequest that has Constraints set to setRequiresCharging(true) and setRequiredNetworkType(NetworkType.UNMETERED). We attach a Worker to the request and in it use NotificationManager and Builder to craft and sending a notification with details of all exercise activity completed in the current day.
NOTE: Relying on AlarmManager, WakeLock, and BroadcastListeners requires the app to be active at all times and is extremely energy inefficient.