

Module 27 Mobile Development

| | |
|---|---|
| Module title | Mobile Development |
| Module NFQ level (only if an NFQ level can be demonstrated) | 8 |
| Module number/reference | BSCH-MD |
| Parent programme(s) | Bachelor of Science (Honours) in Computing Science |
| Stage of parent programme | Award stage |
| Semester (semester1/semester2 if applicable) | Semester 1 |
| Module credit units (FET/HET/ECTS) | ECTS |
| Module credit number of units | 10 |
| List the teaching and learning modes | Direct, Blended |
| Entry requirements (statement of knowledge, skill and competence) | Learners must have achieved programme entry requirements. |
| Pre-requisite module titles | None |
| Co-requisite module titles | None |
| Is this a capstone module? (Yes or No) | No |
| Specification of the qualifications (academic, pedagogical and professional/occupational) and experience required of staff (staff includes workplace personnel who are responsible for learners such as apprentices, trainees and learners in clinical placements) | Qualified to as least a Bachelor of Science (Honours) level in Computer Science or equivalent and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent. |
| Maximum number of learners per centre (or instance of the module) | 60 |
| Duration of the module | One Academic Semester, 12 weeks teaching |
| Average (over the duration of the module) of the contact hours per week | 4 |
| Module-specific physical resources and support required per centre (or instance of the module) | One class room with capacity for 60 learners along with one computer lab with capacity for 25 learners for each group of 25 learners |

| Analysis of required learning effort | | |
|---|---------------------------------|-------|
| | Minimum ratio teacher / learner | Hours |
| Effort while in contact with staff | | |
| Classroom and demonstrations | 1:60 | 30 |
| Monitoring and small-group teaching | 1:25 | 18 |
| Other (specify) | | |
| Independent Learning | | |
| Directed e-learning | | |
| Independent Learning | | 102 |
| Other hours (worksheets and assignments) | | 100 |
| Work-based learning – learning effort | | |
| Total Effort | | 250 |

| Allocation of marks (within the module) | | | | | |
|---|-----------------------|--------------------|---------------------------------|-------------------------------|-------|
| | Continuous assessment | Supervised project | Proctored practical examination | Proctored written examination | Total |
| Percentage contribution | 50% | | | 50% | 100% |

Module aims and objectives

The aim of this module is to teach the theoretical and practical foundation of development for mobile based environment (e.g. smartphones). Learners are taught the differences between mobile and desktop based environments and the restrictions that come with development on a mobile based environment (much reduced computation power, touch interaction, limited storage etc.). Learners are exposed to the potential security risks, privacy, and data protection concerns surrounding these devices that contain multiple sensors and information about their users.

The objectives of the module are to give learners the skills to develop applications for a mobile based environment that take into account the limited resource set that is available and also accounting for the application structure prescribed by such environments. UI design is also expanded upon to take account of touch interaction and the UI restrictions of mobile based environments. Sensor use is covered to show what data can be collected along with examples of user information that can be derived from such sources of data.

Minimum intended module learning outcomes

On successful completion of this module, the learner will be able to:

1. Write programs for a mobile based environment
2. Explain how programs interact with the mobile based environment
3. Discuss the key differences between desktop and mobile environments
4. Identify the need for custom controls and demonstrate their implementation
5. Implement interaction with sensors in a mobile based device (e.g. GPS, Accelerometer, Gyroscope)
6. Implement features of mobile programming.
7. Explain the potential risks (both security and privacy) associated with these devices and how to mitigate those risks
8. Discuss how mobile devices fit in the computing ecosystem and the integration of cloud services to backup these devices

Rationale for inclusion of the module in the programme and its contribution to the overall MIPLOs

The module enables learners to understand and program applications for mobile based environments. Learners build applications with well-designed functional UI's built around touch that will make efficient use of limited resources.

These environments are becoming more widespread. Recently mobile based devices shipping volume was over 340 million devices for the first quarter of 2017 alone according to IDC's "Smartphone Vendor Marketshare, 2017 Q1" report.

Appendix 1 of the programme document maps MIPLOs to the modules through which they are delivered.

Information provided to learners about the module

Learners receive a programme handbook to include module descriptor, module learning outcomes (MIMLO), class plan, assignment briefs, assessment strategy, and reading materials.

Module content, organisation and structure

Introduction and motivation

- Overview of the module
- What is mobile development?
- Introduction to mobile platforms
- Mobile software ecosystem

- Pervasiveness of mobile environments in today's society

Mobile operating systems and differences between desktop and mobile

- Introduction to Android
- Introduction to iOS
- Differences between desktop and mobile OSes
- Differences and comparison between hardware on desktop and mobile (CPU, GPU, RAM, storage)
- Window system differences between desktop and mobile
- Design issues arising from window system and form factor constraints on mobile.

Application Model and Structure

- Programming model differences between desktop and mobile.
- Handing control over to the mobile OS
- Event driven model
- UI structure of mobile based applications
- Activities and transferring from one to the next (including data transfer)
- Event handling and listeners

Custom Drawing/Widgets

- 2D graphics drawing
- Basic shapes, text, and custom widgets
- Drawing performance and optimisation advice
- Matrix stack and transform operations (Translate, Scale, Rotate) and its use for animation and complex drawing construction

Security

- Permissions: the permissions model (both the old all-or-nothing approach vs the new partial modifiable permissions model) and the consequences for a user and a developer
- Data privacy: the amount of user stored data, potential for malicious applications, where and how should data should be stored
- Service Login/logout: avoid storing passwords, access tokens. Alternative mechanisms
- Security advice for mobile based programs

Sensors

- GPS and location sensors
- Accelerometers

- Gyroscopes
- Magnetic sensors
- Light sensors
- What data can be obtained? And what it can be used for?
- Potential misuse of sensors and data privacy concerns

Multi-threading, Services, and Performance

- The need for multithreading and UI responsiveness
- Multithreading alternatives: single time task, indefinite task, background service and determining which is appropriate to use
- Performance advice: optimising UI layouts for quicker rendering, making use of as much of the SDK as possible, multithreading for parallelism.

Design Patterns

- UI patterns that work well on mobile
- Taking account of touch
- Using gestures to reduce on screen controls
- Organising controls based on Frequency, Importance and Typicalness

Module teaching and learning (including formative assessment) strategy

The module is taught as a combination of lectures and lab sessions. The lecture sessions discuss and explain to learners the differences between desktop based environments and mobile based environments. The programming model, restrictions, and lack of resources are covered in detail. Security concerns are covered and also, best practice for making the most efficient use of the scarce resources is given. In the practical lab sessions learners have the opportunity to interact and develop applications for mobile based environments (most likely smartphones) using the standard SDK, writing custom view functionality, and interaction with sensors.

Assessment is split into two. In terms of continuous assessment there are three major assignments that test the learner's ability to make a working application with standard components before introducing custom controls and sensors. Finally, there is an end of semester exam that tests the learners understanding of the theoretical material.

Timetabling, learner effort and credit

The module is timetabled as one 2.5-hour lecture and one 1.5-hour labs per week.

The number of 10 ECTS credits assigned to this module is our assessment of the amount of learner effort required. Continuous assessment spreads the learner effort to focus on the major components of mobile environment development.

There are 48 contact hours made up of 12 lectures delivered over 12 weeks with classes taking place in a classroom. There are also 12 lab sessions delivered over 12 weeks taking place in a fully equipped computer lab. The learner will need 102 hours of independent effort to further develop the skills and knowledge gained through the contact hours. An additional 100 hours are set aside for learners to work on worksheets and assignments that must be completed for the module.

The team believes that 250 hours of learner effort are required by learners to achieve the MIMLOs and justify the award of 10 ECTS credits at this stage of the programme.

Work-based learning and practice-placement

There is no work based learning or practice placement involved in the module.

E-learning

The college VLE is used to disseminate notes, advice, and online resources to support the learners. The learners are also given access to Lynda.com as a resource for reference.

Module physical resource requirements

Requirements are for a classroom for 60 learners equipped with a projector, and a 25 seater computer lab for practical sessions with access to a Mobile Development SDK and IDE (Android or iOS but this may change should better technologies arise).

Reading lists and other information resources

Recommended Text

Friesen, J. (2010) *Learn Java for Android development: [get the Java skills and know-how that you'll need to learn and write successful Android apps]*. New York, NY: Apress.

Clayton, C. (2018) *Learn iOS 11 Programming with Swift 4 Learn the fundamentals of iOS app development with Swift 4 and Xcode 9*. Birmingham: Packt Publishing

Thornsby, J. (2016) *Android UI Design*. Birmingham: [Packt Publishing](#)¹

Secondary Reading:

Griffiths, D. (2017) *Head First Android Development*. Boston: O'Reilly Media

Moskala, M. and Wojda, I. (2017) *Android Development with Kotlin: Enhance your skills for Android development using Kotlin*. Birmingham: Packt Publishing.

¹ Last accessed 26/07/2018

Specifications for module staffing requirements

For each instance of the module, one lecturer qualified to at least Bachelor of Science (Honours) in Computer Science or equivalent, and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.. Industry experience would be a benefit but is not a requirement.

Learners also benefit from the support of the programme director, programme administrator, learner representative and the Student Union and Counselling Service.

Module Assessment Strategy

The assignments constitute the overall grade achieved, and are based on each individual learner's work. The continuous assessments provide for ongoing feedback to the learner and relates to the module curriculum.

| No. | Description | MIMLOs | Weighting |
|-----|--|---------|-----------|
| 1 | Practical Assignment that require the learner to build a fully functioning application that only uses the standard components provided by the mobile environment SDK. This introduces the learner to the workings of the mobile based environment and its associated programming model. | 1,2,6 | 12.5% |
| 2 | Practical Assignment that requires the learner to build an application with a custom widget which is drawn purely by mathematics defined by the learner. The SDKs will not provide widgets for every use case thus the learner must be able to derive and construct their own. | 1,2,4,6 | 17.5% |
| 3 | Practical Assignment that requires the learner to build an application that polls sensors and does something useful with it. An example here could be the use of GPS to track a user's run or cycle and provide stats | 1,2,5,6 | 20% |
| 4 | Written exam that tests the theoretical aspects of the module | 1-8 | 50% |

All repeat work is capped at 40%.

Sample assessment materials

Note: All assignment briefs are subject to change in order to maintain current content.

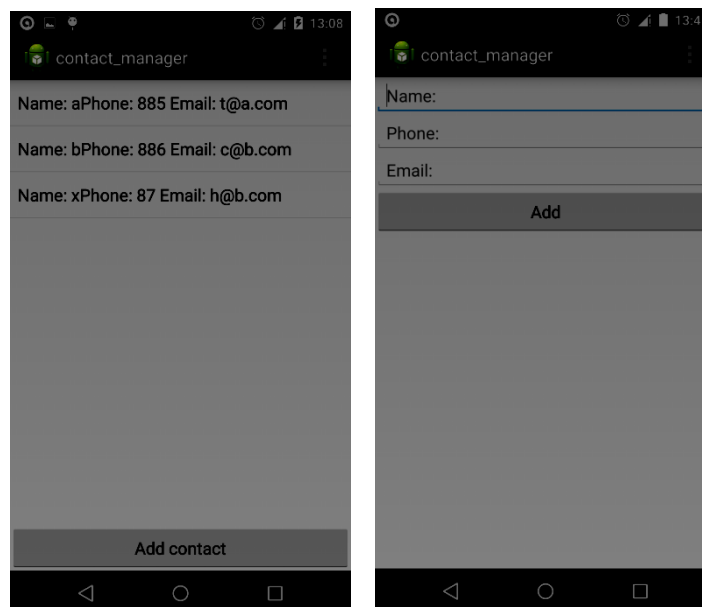
Assignment 01: Building the beginnings of a Contact Manager

Introduction:

In this assignment you will be making the beginnings of a simple contact manager application. It will store the contacts in an sqlite database on the device itself with contacts displayed in a simple listview of strings. There will be two activities in this application. The first activity will show the listview with all of the contacts that have been added thus far. There will be a button at the bottom of this activity that will enable the user to add a new contact.

The add contact button will launch an explicit intent that will start a new activity for adding a new contact to the database. It should add in a name a phone number and an email address in a single go. If the contact has been successfully added this second activity should dismiss itself and refresh the list of contacts by sending back an OK message to the main activity. If the user hits the back button it should send back a cancel message and nothing should be done.

This assignment will test your ability to take seven different examples from the android notes and combine them to make a fully working application. A couple of screenshots of the main activity and the add contact activity are seen below as running on a nexus 5.



Notes:

You are required to submit this assignment by 2015-03-08 at 23:55 (8th march) any assignments that are submitted even a second late will have the standard late penalties applied.

For this assignment you may use the following examples from the Android by Example text to help you with this assignment. These examples are

- 011 linear layout example
- 014 relative layout example
- 017 button example
- 018 edittext example
- 024 listview example
- 032 explicit intent with result example
- 041 sqlite database example

You will be required to submit a single archive to the moodle that contains your entire eclipse or android studio project it must be in one of these accepted formats tar.gz/tar.bz2/tar.xz/zip/rar/7z any other compression format will incur a 10% penalty. Code that fails to compile will incur a 30% penalty. The test for compilation is that I should be able to open your project in eclipse or android studio (whichever you use) and have it compile without modification.

Task List:

01) Create a database open helper that will generate a database with four fields. The first is an ID that will be auto generated and unique. The second is the name of the person which is a string, the third is a phone number as a string and finally the email address as a string. (5%)

02) Create a layout that represents the first of the screenshots that you see above. You must use a relative layout, listview and a button (10%)

03) Create a layout that represents the second of the screenshots that you see above. You must use a vertical linear layout, edit texts and a button to generate this layout (15%)

04) In your MainActivity class in the onCreate() method initialise an arraylist of strings that will be attach it to your list view. You should get read only access to your database here and display the list of contacts in the list view. Add a listener to the button that will launch the second activity when it is clicked (15%)

05) in your main activity add in a method that will receive a result from the add contact activity. If the result is good then update the list view (5%)

06) in your main activity add in a method for updating the list view. It should query the full list of contacts from the database and generate a single string for each that contains the name, phone number and email address. Each of these strings should be displayed on the list view. (25%)

07) in the add new contact activity inside your onCreate() method get writeable access to your database. Add a listener to the button that will add a new contact to the database, and return a message to the main activity. (15%)

08) in the add new contact activity add an addContact() method that will add a new contact to the database. (10%)

Assignment 02: building a custom view to implement the game of minesweeper

introduction:

Note: read the whole assignment brief first before implementing it contains very important information.

In this assignment you will be tasked with building a custom view that will render and implement the rules of minesweeper. Assuming you are not already familiar with minesweeper the premise of the game is simple. You are given a two dimensional grid representing a minefield. However all of the cells are covered so you cannot see what is underneath. To sweep the minefield you must uncover cells one after the other until all non-mined cells are uncovered. Non-mined cells will include a number indicating how many adjacent cells contain a mine relative to that cell. If there are no adjacent mines a number will not be displayed.

Players should be able to use a combination of a logical reasoning and a little bit of guesswork to determine what cells contain mines. If a cell contains a mine they should be able to mark it as such. Any time a player clicks on a marked cell (usually by accident) it will not uncover. Unless the user explicitly unmarks it.

You will be tasked with building a fully working game of minesweeper running on a 10x10 board with 20 mines. Players should be able to reset the game, switch between marking and uncovering modes and have some information displayed to them about how many mines they have marked and how many mines there are in total.

If you are not familiar with Minesweeper I would strongly suggest that you play it for about 20 to 30 minutes to make yourself familiar and then get to working on the application.

Please read through the rest of this assignment before you start coding straight away. In the words of Scott Meyers “Weeks of coding saves hours of planning”.

Notes:

Take note with the milestones. If you skip a milestone or have serious errors with a milestone, further milestones will not be considered for correction. e.g. if you skip milestone 03 or have bad errors with it then milestones 04,05,06,... will not be considered for correction. With regards to errors, depending on their severity I will either deduct 5, 10, or 15% for each bug. Thus it's possible that if you implement all milestones but have 20 small bugs you can end up with a score of 0%. This is to encourage you to build robust software free from bugs.

No structure will be provided for the application so it is up to you to determine what datastructures and algorithms you will need to complete this assignment. You will be expected to use the user login service, JDO objects, and JSPs in order to complete this assignment

For submission you are required to upload an archive of your complete eclipse project to the moodle by 2016-xx-xx at 23:55:00. The accepted archive formats are the following six: 7z/zip/rar/tar.gz/tar.bz2/tar.xz.

You are also expected to have a git repository with your work. To use git install the Eggit package in eclipse. Git is a source code manager that will serve as a backup of

your work and also a rollback facility if you mess up your source. You are expected to make an initial commit immediately after the project is created and at least 1 commit for each milestone (when you finish a milestone at the beginning of your commit comment write "Milestone X:" where X is the number of the milestone.

Penalties:

There will be penalties applied to your score if any of the following conditions are not met. These are silly things that slow down my correction of your work. Thus if you want your result back in the quickest possible time make sure you do not fall foul of these penalties.

- -40% for a missing git repository. Make sure when you go to submit your work that there is a `.git/` directory in your project before archiving. I like to see development history on assignments to see how it was made. You will also lose marks for missing commits.
- -30% for non compiling code. I should be able to open your project have it compile and run immediately. You are final year learners, non compiling code is a grave sin at your level.
- -10% for a non supported archive format. I have a python script that can unpack and sort all projects downloaded from moodle. Anything not in those archive formats I have to unpack and sort by hand.
- Standard late penalties: submission even a second late after 23:55:00 on submission day will have the late submission penalties applied.

Milestones:

As noted before make sure that you complete milestones properly before moving onto the next one. The percentage in brackets indicates the maximum score you can achieve if you pass that milestone.

01) Define the shell of a custom view class and generate a layout for the main activity consisting of the custom view, two buttons, and two textviews. (5%)

02) Force the custom view to restrict itself to a square size (10%)

03) Draw the initial state of the game board with all cells covered. Black should be used as the fill colour and white lines should separate the cells (20%)

04) Implement basic touch behaviour that will uncover a cell. When a cell is uncovered it should change from a black colour to a grey colour. (35%)

05) Implement methods to place 20 mines randomly in the minefield and render the mines when a cell is uncovered. A cell containing a mine should have a red background with a black M in the foreground. You will fail this milestone if: (50%)

- less than 20 mines are placed (it's possible random number generation will give you the same coordinates)
- the mines are not rendered

06) Implement methods to render the individual numbers that may appear in a cell. 1 should be blue, 2 should be green, 3 should be yellow, and four and above should be

red. Also write code to determine the number that should be displayed in each cell when the game is initialised for the first time. (75%)

07) modify the touch behaviour to stop accepting input when a mine is uncovered. It should only reenable input when the user has clicked the reset button. (80%)

08) The other button should switch between displaying “uncover mode” or “marking mode” each time it is clicked. In uncover mode each touch should result in a cell being uncovered. In marking mode each time a covered cell is touched it should change to yellow to denote it is marked, or back to black to denote it is unmarked. Use the two text fields to denote the total number of mines and the number of mines marked. In uncover mode touching a marked cell should do nothing.(100%)

Assignment 03: Building a journey tracker by combining custom views, the GPS sensor and general layouts

Introduction:

In this assignment you will be tasked with building a simple GPS based journey tracker that generates and displays statistics about how fast you are going your average speed and the total distance you have travelled. A custom view will show a line graph depicting the historical speed over the journey. There are four different line colours in this graph that you will see in the screenshots below. Green is the actual trace of speed as obtained from the GPS sensor. Grey are lines that mark certian speeds. Red is the overall average for the journey while Orange is the average for the last 20 samples.

With respect to the grey lines you can choose to set them for every 2km/h or 10km/h. It is recommended that you use the 2km/h version for walking or if you wish to test by driving use the 10km/h lines instead. Either will be accepted.

This assignment will combine everything that you have learned in the previous two assignments and on top of this will ask you to integrate a single sensor. If you are using an android device to do this application you may find yourself walking outside a few times to test if the GPS tracking is working correctly. It is also possible to simultate this by using GenyMotion as it provides facilities to pass GPS values to the virtual device itself. Note that the speed measured in the GPS is in meters per second thus to get km/h you must multiply it by 3.6

Notes:

You are required to submit this assignment by 2015-04-24 at 23:55 (24th April) any assignments that are submitted even a second late will have the standard late penalties applied.

For this assignment you will may use the following examples from the Android by Example text to help you with this assignment. These examples are

- 028 creating a full custom view
- examples on the text view, and buttons and the relative layout
- example on using the GPS sensor

You will be required to submit a single archive to the moodle that contains your entire eclipse or android studio project it must be in one of these accepted formats tar.gz/tar.bz2/tar.xz/zip/rar/7z any other compression format will incur a 10% penalty.

Code that fails to compile will incur a 30% penalty. The test for compilation is that I should be able to open your project in eclipse or android studio (whichever you use) and have it compile without modification.

Task List:

01) Create a layout for your main activity using the relative layout that shows your custom view, text views for the time, speed, current average and overall average and a button for stopping and starting the tracker (10%)

02) implement a custom control called TraceView that will implement the following methods (50%)

- all three standard constructors with a shared init method
- setTrace() a method for attaching an array list of GPS trace data
- updateOverallAverage() updates the overall average speed to be shown on the trace (computed in MainActivity)
- updateCurrentAverage() updates the current average speed to be shown on the trace (computed in MainActivity)
- update() determines the minimum and maximum speeds and recalculates how much space in pixels there should be for a single second and a single km/h, horizontally and vertically respectively. It should force an update of the display
- onDraw() draws a black background and draws the necessary 2km/h or (10km/h) lines and the current and average speeds. It should also draw the main trace with a small green circle denoting each datapoint.
- OnMeasure() to force the view to be drawn square

03) implement a MainActivity class that will implement the following methods (40%)

- onCreate() pulls all the components from the xml file, initialises the array list and attaches it to the TraceView. Adds a listener to the button that will start and stop tracking (should change text to reflect state). Add in a location listener that listens for updates every second. It should update the array list, overall average, and current average. It should update the UI
 - **Note marks will be awarded for an efficient O(1) implementation for updating the averages on each update.**
- startTracking() responsible for starting the tracking of the user. Should reset all values and clear out the previous location objects
- stopTracking() responsible for stopping user tracking
- updateUI() responsible for updating the TraceView and the text fields

GRIFFITH COLLEGE DUBLIN

**QUALITY AND QUALIFICATIONS IRELAND
EXAMINATION**

MOBILE DEVELOPMENT

Lecturer(s):

External Examiner(s):

Date: XXXXXXXX

Time:XXXXXXXX

**THIS PAPER CONSISTS OF SIX QUESTIONS
FIVE QUESTIONS TO BE ATTEMPTED
ALL QUESTIONS CARRY EQUAL MARKS**

**IN ALL CASES, CANDIDATES SHOULD *READ THE ENTIRE QUESTION*, BEFORE
ANSWERING ANY PART**

QUESTION 1 (Custom Views and Multi Touch)

Answer both (a) and (b)

- a) “When rendering a custom view it is advised that memory allocation and deallocation should never be performed in drawing methods”. Justify why this advice should be followed. Explain two consequences if this advice is ignored.

(10 marks)

- b) Explain the importance of using algebraic expressions rather than fixed values when writing drawing operations for a custom view. Determine which values the algebraic expressions must be based on for this to work.

(10 marks)

Total (20 marks)

QUESTION 2 (Activity Lifecycle and Task Management)

Answer (a), (b) and (c)

- a) Construct a diagram showing all states an activity and all lifecycle handlers that are called on transition from one state to another.

(8 marks)

- b) Explain why the Android OS implements the Activity Lifecycle.

(4 marks)

- c) “The Task list plays an important role in freeing memory from applications in Android”. Justify this statement by explaining what the task list represents and also how it influences the decisions made in freeing memory.

(8 marks)

Total (20 marks)

QUESTION 3 (Sensors)

Answer (a), (b) and (c)

- (a) Different android devices will have different sensor capabilities, in terms of availability and spec of sensors. In addition, an android device may have more than one sensor of a particular type. Outline how Android implements the identification of sensors and sensor capabilities. Your answer should clearly indicate how Android handles the variations of sensors among devices.

(8 marks)

- (b) Gyroscopes, proximity sensors and barometric sensors are all types of sensors which may be found on different devices. Explain what each of these sensors are.

(9 marks)

- (c) Indicate if permissions required for sensors? Justify your answer.

(3 marks)

Total (20 marks)

QUESTION 4 (Multi-Threading)

Answer (a), (b) and (c)

- (a) "A good practice in creating responsive applications is to make sure your main UI thread does the minimum amount of work and doesn't contain any long running tasks." What are the rules imposed by android regarding responsiveness? Justify their existence.

(5 marks)

- (b) Consider the following tasks and indicate which type of multi-threading would be most suitable. Justify your choices.

- (a) Downloading a file where the internet connection will not be needed once the file is downloaded.
- (b) You require a network connection to remain open during the lifetime of the application.

(4 marks)

- (c) Services are one method of multi-threading that can be used in android programming. Outline under which circumstances services are the appropriate multi-threading option, give examples and explain how services are implemented in code.

(11 marks)

Total (20 marks)

QUESTION 5 (Marshmallow Changes)

Answer (a), (b) and (c)

"A battery that works smarter, not harder"

The Android.com website made the above statement about the changes introduced in Android's 6.0 version - Marshmallow.

- (a) Identify and explain the changes introduced by marshmallow that this statement is referring to.

(5 marks)

- (b) Indicate the criteria which would see these changes take effect on an android device.

(6 marks)

- (c) Outline the restrictions that would be placed on an application as a result of these changes

(9 marks)

Total (20 marks)

QUESTION 6 (Android Programming)

Answer both (a) and (b)

- a) Write an `init()` method that initialises a set of four colours (red, green, blue, and white) and an `onDraw(Canvas c)` method that draws four non overlapping squares that take up the entire space of the view. Each square should be a different colour, you are required to make your custom view scale up and down for all devices, and you are also required to use the `save()`, `translate()` and `restore()` methods for drawing each square. You may assume your view is square shaped (width and height equal).

(10 marks)

- b) Assuming you have an `ArrayList` of `Location` objects (GPS latitude, longitude objects, called `locations`) and a float (called `total_distance`, and already initialised to zero) write an `OnLocationChanged(Location l)` method that will keep track of the last 100 `Location` objects, as each new location is added make sure to update the total distance covered (The `Location.distanceTo(other Location)` method will be useful here)

(10 marks)

Total (20 marks)