University of Westminster
School of Computer Science & Engineering

5COSC023W Coursework 1 (Semester 2)

Module leader	Dr D. Dracopoulos
Unit	Coursework 1
Weighting:	50%
Qualifying mark	30%
Description	
Learning Outcomes Covered in this Assignment:	LO1, LO4, LO5
Handed Out:	February 2025
Due Date	24/3/2025 13:00
Expected deliverables	Source code/XML files/Resources (images, etc)/Video
Method of Submission:	Online via Blackboard
Type of Feedback and Due Date:	Individual feedback via Blackboard within 3 weeks of submission **All marks will remain provisional until formally agreed by an Assessment Board.**

Assessment regulations

Refer to section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website :http://www.westminster.ac.uk/study/current-students/resources/academic-regulations

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether

the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

5COSC023W MOBILE APPLICATION DEVELOPMENT - Assignment 1
Deadline 24/3/2025, 13:00

Dr Dimitris C. Dracopoulos
Email: d.dracopoulos@westminster.ac.uk

Description

You are required to implement an Android application using Kotlin and Jetpack Compose described by the specifications below.

You are not allowed to use third-party libraries. The only libraries that you can use are the standard Android API libraries found in the following URL (with the exception of Views that you should NOT use):

`https://developer.android.com/reference/`

Your are not allowed to use View classes. You should use Jetpack Compose instead.

You should NOT disable `Activity` recreation (e.g. when screen orientation changes) for any parts of your work. You should also not disable re-orientation of the device.

Your task is to implement a dice game Android application and according to the game rules and the implementation details below.

Your program will be displaying images of the dice thrown by the two players: the human and the computer. You can use the images found in the following URL:

`https://freesvg.org/search/`

or any other dice images that you prefer.

The rules of the game

A human player competes against the computer. Both players throw 5 dice at the same time. The score of each throw for each player is the sum of the numbers shown on the faces of the dice. The objective of the game is to reach a score of 101 or more (instead of 101 another target can be set by the human before play starts) by throwing 5 dice as many times as necessary.

After a roll, each player may either score it or take up to two optional rerolls. For each reroll, they may reroll all of the dice or select any of the dice to keep and only reroll the remainder.

A player may score at any time, thus ending their current throw; after the second reroll (three rolls in total) they must score it.

After both players score their rolls, the procedure is repeated until one of the players reach 101 or more points by summing all of their scores. If both players reach 101 or more with the same number of attempts (a single attempt is considered as one roll followed by 2 optional rerolls) the player with the highest score wins.

As an example, assume that the human player scores 30 points in the first attempt (a roll followed by 2 optional rerolls), 25 in the second, 11 in the third, 28 in the fourth and 15 in the fifth achieving a total score of 109 in five attempts. If the computer did not score 101 in four attempts or more than 109 in five attempts, the human wins. In the case that both the computer and the human achieved the same score in the five attempts (i.e. 109 in the example), each player throws for a last time all five dice and the player with the highest sum in that roll wins (no optional rerolls are allowed in this case). In the case of a tie again, both players keep rethrowing all five dice until one of them wins.

Specification

1. When the application starts, it presents the user with 2 buttons labelled *New Game*, and *About*. *(2 marks)*

2. Clicking on the *About* button should present the user with a popup window which describes the author (student id and name) and the message:

   ```
   I confirm that I understand what plagiarism is and have read and
   understood the section on Assessment Offences in the Essential
   Information for Students.  The work that I have submitted is
   entirely my own. Any work from other authors is duly referenced
   and acknowledged.
   ```

   (2 marks)

3. Clicking on the *New Game* button, the user will be presented with the game screen which they interact with.

   The screen should contain 2 buttons labelled `Throw` and `Score` (additional buttons could be included for the additional tasks described below).

   Each time the `Throw` button is pressed, a simulation of throwing 5 dice by both the human player and the computer is performed simultaneously:

   The images of the five dice rolled by the human player and the five dice rolled by the computer are displayed.

   Each of the dice images should be selected randomly with an outcome of a number between 1 to 6. Not all of them need be unique, as one roll of 5 dice may result in 6, 1, 4, 4, 2, i.e. the same outcome representing 4 on the face of a die could be generated for two different dice.

   (10 marks)

   - 2 marks for displaying the images

- 3 marks for displaying the same random images every time
- 5 marks for displaying different random images every time

4. The human player may choose to score by pressing a button `Score` or take up to two optional rerolls (see below). As soon as the player clicks on `Score` the total score for the current game should be updated for both the human player and the computer player (also the dice for the computer player should be updated according to the computer player strategy, see below).

   The current total score for the game (both human and computer) is displayed on the top right of the screen.

   If the user performs the maximum of 3 rolls for that turn, the score is updated automatically without the need to press the `Score` button (see the rules of the games described above).

   (10 marks)

   - 3 marks for score updating both the human and the computer
   - 3 marks for updating the score after 3 rolls
   - 2 marks for disabling the human to re-roll after 3 rolls
   - 2 marks for correct calculation of the score

5. For each of the 2 optional rerolls, the human player should be able to select (it is left up to you to design the appropriate user interface for this) which dice (if any) he would like to keep for that roll. After selecting this, the human player should press the `Throw` button again and the dice which have not been selected for keeping should be rerolled.

   (10 marks).

   - Partial implementation: 1-5 marks
   - Minor omissions: 6-10 marks

6. The computer player follows a random strategy. I.e. first it decides randomly whether it would like to reroll (up to a maximum of 3 rolls per time) and if this is the case it decides randomly which dice to keep. A single (first) roll for the computer player occurs and it is displayed only after the human player clicks on the `Throw` button.

   If the human player clicks on the `Score` button, the computer player uses all of its remaining rolls for that turn according to the random strategy, i.e. the final result of the five dice is displayed after the computer has used (optionally based on the random strategy) the 2 rerolls (for a total maximum of 3 rolls).

   (10 marks).

   - 5 marks: the strategy of the computer is not fixed for whether to reroll
   - 5 marks: random action for which dice to keep

7. When a player (human or computer) reaches 101 or more points a pop up window with the message "You win!" is displayed in green colour (if the human wins) or "You lose" in red colour (if the computer wins). You should implement the exact rules of the game described in the Section "The rules of the game" above.

   (5 marks)

- 2 marks: Red/green colours
- 3 marks: Termination of the game and determination of the winner does not follow the "rules of the game"

8. As soon as a winner is determined, the game is not playable any more. In order to start a new game, the user needs to press the Android "Back" button to move to the initial screen of the application from where a new game can be started by pressing the *New Game* button.

   (3 marks)

9. Implement what happens in the case of a tie according to the exact rules of the game described in the Section "The rules of the game" above (the 2 players keep rolling until the tie is broken - no rerolls in this case).

   (10 marks)

   - 1-5 marks for partial implementation of tie rules (e.g. rethrowing allows rerolls)
   - 6-9 marks for minor omissions

10. Extend your implementation by allowing the human player to set the number of points which is the target to reach for the winning player (default value is 101).

    (2 mark).

11. Extend your implementation so that the total number of wins for the human and computer players is displayed on the top left of the playable screen in the form `H:10/C:5` where 10 is the total number of human wins and 5 is the total number of computer wins.

    The score does not need to be persisted using a file or other way, as soon as the application is exited the score should be lost and the next time that the application is restarted the score starts with 0/0.

    (3 marks)

12. Design and implement an efficient (as optimum as possible) strategy for the computer player strategy. The strategy you design should determine whether the computer player should reroll some of its dice and which ones (up to a maximum of 2 rerolls - total maximum of rolls is 3).

    You should assume that the computer player cannot see the current dice of the human player displayed in the application, but it is aware of (i.e. it can see) the current total of the current game for both itself and the human player, i.e. the computer player knows that the score of the current game is, e.g. $79 - 93$.

    If you implement this subquestion, you should make sure that you describe in detail the strategy that you come up with, its justification and why do you think that it will perform well, advantages and disadvantages. The documentation of this should be done inside comments in the very beginning, or the very end of your implementation of this part.

    (16 marks)

    - 9 marks for devising an efficient algorithm
      - 8–9 marks for excellent derivation of a strategy
      - 6–7 marks for minor omissions of a good approach

  - 4–5 marks for some kind of strategy which partially works
  - 1–3 marks for a poor choice of a strategy
- 7 marks for proper documentation and justification
  - 4 marks for documentation
  - 3 marks for justification

13. For all the tasks, the application should behave in a user friendly manner when the device is rotated from portrait to landscape and back to portrait mode. I.e. the application should resume from exactly the same point (same screen and data) when the orientation changes. The rotation of the device should not change what was the user was seeing before the rotation.

    You should NOT disable `Activity` recreation (e.g. when screen orientation changes) for this or any parts of your work. You should also not disable re-orientation of the device.

    (9 marks)

    - 1-5 marks: partial implementation
    - 6-9 marks: minor omissions

Marking Scheme: The marks achieved for each part of the program are indicated in the description of the task above. In addition to these the following will be taken into account:

- *Code readability* (structure, comments, variable naming, etc,): 4%
- *Implementation* (e.g. quality, efficiency, look and feel of the application, based on fonts, colours, etc.): 4%

In addition to the above marks indicated in each of the sub-questions in the specification, additional marks will be deducted if an application behaves in an unexpected way. For example, an application should not crash, the application should work properly even if a user enters invalid data or rotate the device.

The maximum for work which does not compile (or XML or other resources files with syntax errors causes the project not to build) is 30%.

Failure to submit a video or a working link to a video will result to a capping of your mark to 30% (i.e. if your submitted work is given initially a mark above 30% it will be capped to 30%).

Submission of assignments using a different method other than Blackboard will not be accepted and zero (0) marks will be awarded in such cases. The exception is a working link to a video outside Blackboard.

Deadline: Monday 24th of March 2025, 13:00.

Submission Instructions

Files to submit:

1. All of the files of the Android Studio project of your application in a zip file.

2. A video demonstrating all the functionality that you implemented for your application. If the video size exceeds the size allowed by Blackboard you can submit a link to a video to another site (e.g. Google Drive, Youtube, etc.) where you have uploaded the video. Make sure that if your video is located to another web site and not Blackboard that you have given appropriate permissions to external people to access your video.

Referencing code: Any code taken from other resources (i.e. a textbook or internet) should be referenced in comments within your code (full textbook details or full web URL), identifying the exact code that you used it as part of your application and the exact portions of the original source code that you reused.

You should submit via BlackBoard's Assignment functionality (do NOT use email, as email submissions will be ignored.), all the files described above. A single zip file with the name `wNNNNNNNN` (where `wNNNNNNN` is your university ID login name) containing all the above files could be submitted alternatively. You can create such a file by using the main menu in Android Studio and choose `File->Manage IDE Settings->Export to Zip File...`

Note that Blackboard will allow to make a submission multiple times. Make sure before submitting (i.e. before pressing the `Submit` button), that all the files you want to submit are contained there (or in the zip file you submit).

In the case of more than one submissions, only your last submission before the deadline given to you will be marked, so make sure that all the files are included in the last submission attempt and the last attempt is before the coursework deadline.

Request to mark submissions which are earlier than the last submission before the given deadline will be ignored as it is your responsibility to make sure everything is included in your last submission.

The following describes how to submit your work via BlackBoard:

1. Access `https://learning.westminster.ac.uk` and login using your username and password (if either of those is not known to you, contact the Service Desk, tel: +44 (0) 207 915 5488 or log a call via `https://servicedesk.westminster.ac.uk`.).

2. Click on the module's name, `MODULE: 5COSC023W.2023 MOBILE APPLICATION DEVELOPMENT` found under `My Modules & Courses`.

3. Click on the `Assessment->Submit Coursework->Coursework`.

4. Click on `View Assignment`.

5. Attach your zip file containing all the Kotlin source code files and resources of the Android Studio Project, by using the `Browse` button.

6. Attach your video or include a link of your video as the first comment line) in the MainActivity of your code.

7. Create a Word or PDF file with the following information:

- *Comments*: Type your full name and your registration number, followed by:
  "I confirm that I understand what plagiarism is and have read and understood the section on Assessment Offences in the Essential Information for Students. The work that I have submitted is entirely my own. Any work from other authors is duly referenced and acknowledged."

8. Attach the file with the statement above.
9. Check that you have attached both the zip, the video (or the video link) and the statement file.
10. Click the `Submit` button.

If Blackboard is unavailable before the deadline you must email the Registry at `studentcentre@westminster.ac.uk` with **cc:** to myself and your personal tutor before the deadline with a copy of the assignment, following the naming, title and comments conventions as given above and stating the time that you tried to access Blackboard. You are still expected to submit your assignment via Blackboard. Please keep checking Blackboard's availability at regular intervals up to and after the deadline for submission. You must submit your coursework through Blackboard as soon as you can after Blackboard becomes available again even if you have also emailed the coursework to the above recipients.