Course: Mobile Application Development **Assignment #: 1**

Course #: IFT 390

Instructor: Karam Keyrouz **Due Date: *Thursday, October 18th, 2018***

Guess the Number: Create an application consisting of a game where a player has to guess the value of a randomly selected number. Based on the selected difficulty, the player gains a number of points for correct guesses, and loses a number of points for incorrect guesses. The application should provide a good user experience.

Your Android application should do the following:

1. **Main Menu**
    a. Your application should open to a menu screen with 4 buttons.
    b. The first button will start the game.
    c. The second button will teach the user how to play.
    d. The third button sets the difficulty to easy.
    e. The fourth button sets the difficulty to hard.
    f. Your name must be displayed at the top of the screen.
    g. The selected difficulty must be displayed and updated at the bottom of the screen.
2. **How to play**
    a. The how to play screen should show instructions that enable the user to understand the game mechanics, rules and win/lose conditions.
    b. A button should enable users to return to the main menu.
3. **Easy Mode**
    a. If the easy difficulty is selected, the user should be able to see it on the main menu.
    b. On easy, correct guesses grant 3 points whereas incorrect guesses cost 1 point.
    c. The number to guess is between 1 and 20.
4. **Hard Mode**
    a. If the hard difficulty is selected, the user should be able to see it on the main menu.
    b. On hard, correct guesses grant 6 points whereas incorrect guesses cost 1 point.
    c. The number to guess is between 4 and 59.
5. **The Game**
    a. Once the game starts, the range of the number to be guessed should be displayed at the top of the screen.
    b. Players start with 8 points.
    c. While the game is running, players should be able to see their score and it should be updated whenever a guess is made.
    d. The player enters a number as input and once a button is tapped, that number is compared with the answer.
    e. If the number is correct, the user should be notified and a new number is selected for the user to guess. The score is updated accordingly.
    f. If the number is greater than the answer, the user should be notified to pick a smaller number. The score is updated accordingly.
    g. If the number is lower than the answer, the user should be notified to pick a larger number. The score is updated accordingly.
    h. Once the score reaches 0, the game ends and the player is taken to a screen informing him/her they have lost. From this screen, the only option is to return to the main menu.
    i. Once the score reaches 20, the game ends and the player is taken to a screen informing him/her they have won. From this screen, the only option is to return to the main menu.

j. Players are not punished for empty or out of bounds inputs, but should be informed of their mistake.
k. Your name should be hidden behind the score.
l. The user also has the option to prematurely end the game and return to the main menu. All values will be reset in this case. (score, answer, etc…)
m. Your name should be logged whenever a correct guess is made.

6. **Code**
   a. Your code should be optimized.
   b. Use proper logging mechanics for debugging and testing.
   c. Use comments to explain your work especially for added features or altered mechanics.
   d. Provide a decent user experience.

Note: the requirements above are guidelines to make a working game. You have the option to change certain mechanics, if you think it would make the game more interesting. An example would be the number of points won/lost or the starting number of points. Another example would be the winning/losing conditions. However, note that any changes you make must not alter the core game and that you are responsible for these changes, should they affect performance or user experience. This means that you will be penalized if the changes lead to a negative outcome, but also rewarded if they lead to a positive one. If the changes don't affect the game positively or negatively, no reward or penalty will occur.

Bonus: if you want there are many ways to further enhance the game and test your knowledge of android development even more. Bonus features are also taken into consideration but **only if the main requirements are fulfilled and if they are built using concepts seen in class.** Some ideas for bonus features might be:

- Keeping track of the number of guesses it took for the user to get a correct answer and awarding/removing more points accordingly.
- Keeping track of the guesses made, and preventing users from repeating the same wrong guess.
- Informing the user if they are closer/farther from the answer after the first guess. A harder challenge would be to do it with colors.
- Instead of winning when the player reaches a certain score, they go to another level. Levels gradually get harder, and once they reach the final level, they win.
- Multiplayer:
  - Players take turns trying to guess the same number in a competition to reach victory first.
  - Players take turns trying to guess a different number each in a competition to reach victory first
  - Since this is an application on a mobile device, note that having more than two players is not practical.

The following should be returned to the instructor:

1- A **cover page** with your Name, Major, Project Number.

2- A **listing of your program** well **documented** (use comments) with the following guidelines:

> **White space**
> The format of the source code is important in the readability of a program. The term "format" refers to both horizontal and vertical white space. In higher level languages, blank lines can be used for vertical spacing and, for horizontal spacing, blocks of code can be indented.
> **Identifiers**
> The identifiers (variable names) which are used also affect the readability of a program. Make identifiers more self-documenting by carefully and creatively choosing names which are close to English but long enough to be meaningful in terms of the task being performed.

A **soft copy** of your ***java and XML files*** must be submitted via email. Let the file names be "IFT 390 Assignment I – *yourname* - filetype" for example:

"IFT 390 Assignment I – Karam Keyrouz – Guess the Number JAVA"

Or

"IFT 390 Assignment I – Karam Keyrouz – Guess the Number XML"

To submit the files and make them of reasonable size, copy each file in a text editor such as Notepad++ or sublime and name them as instructed. In this assignment **at least TWO files** should be submitted.

NOTES

- No correction or grading will be performed if I don't receive a hard copy in class even if a soft copy was submitted.
- No late submissions.
- All submitted documents must be typewritten.
- You may only use the Android programming concepts covered in in the course so far. Do not use any more advanced concepts we will cover later on or any other programming concepts that you have had experience with. If you do I'll remove points.