Quick demo of each game, plus login/scoreboard

- Each person talks about one game
- There are 4 games so the odd one out talks about the general GameCentre features
- Roles will be assigned over the weekend

Design Patterns

- Model-View-Controller
- 3 / 4 games use this design pattern - it is very useful for this situation
- Solve problem of translating an abstract version of a game (Model) to a display (View) that can be interacted with (Controller)
- Also implemented iterator in SlidingTiles during phase 1

Important Classes

- The most important classes of each game correspond to the model, view and controller classes
- Also of note are important classes outside the games: Score and User

Test Cases:

- Strive for 100% coverage where possible
- The Model class for 2048 unfortunately needs to be passed a view object to run animations
- 100%: 2048's Grid, Tile and Cell classes - TileAndCellTest
- 0%: ActualStartingActivity - composed entirely of loading/saving data and button listeners

Score object

- Use inheritance to account for scores with extra features, like difficulty in CheckersScore, or board size for SlidingTiles score. 2048 doesn't use extra features, so it uses the parent class Score

High score storing file

- File name format: (username)_(the game)_score_save_file.ser

Viewing scores on scoreboard

- ScoreActivity class
- If the saved score file does not exist, the scoreboard will show "No record found for (game)".
- If the file exists, then the scoreboard will display the score accordingly.
- For the games with different level of difficulties, top 5 scores will be displayed for each difficulty level