CPSC 411: Mobile App Development (iOS) - Spring 2019

Final Project, **Unreal Engine 4 game in 2D/3D,** due Sunday, 19 May 2019 (by 2355)

In this assignment, you will create a **3d/2d game** on your **iOS/Android mobile device,** using the game engine **Unreal Engine 4 (or Unity)**. As an option, you may choose to use Unity, but it is not recommended as (a) it is much less powerful, and (b) it is not recognized as a professional development game engine by companies interviewing for internships and jobs. Ideally, you would choose to use Unreal 4 to develop a 3D game deployed to iOS. Minimally, you would choose to use Unity to develop a 2D game for Android.

There are a variety of games you may choose to implement, but they all share something in common, they use the power of the game engine to move about the world, with one or more cameras following your movement, and multiple light sources. In general, moving obstacles of different types oppose your motion, spawning off-screen, crossing the screen, and being destroyed, only to respawn again.

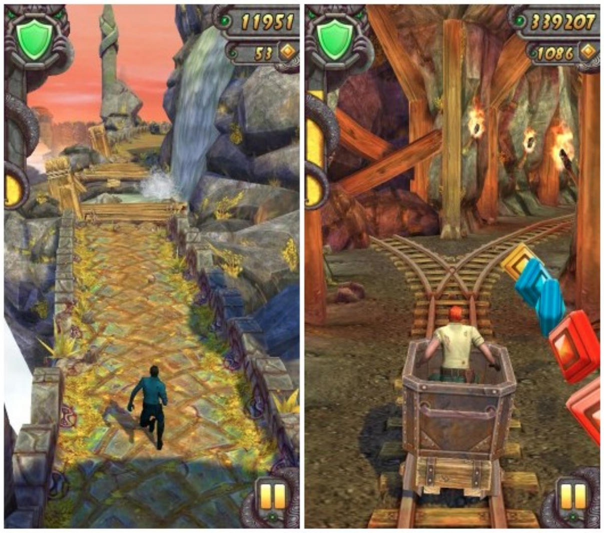
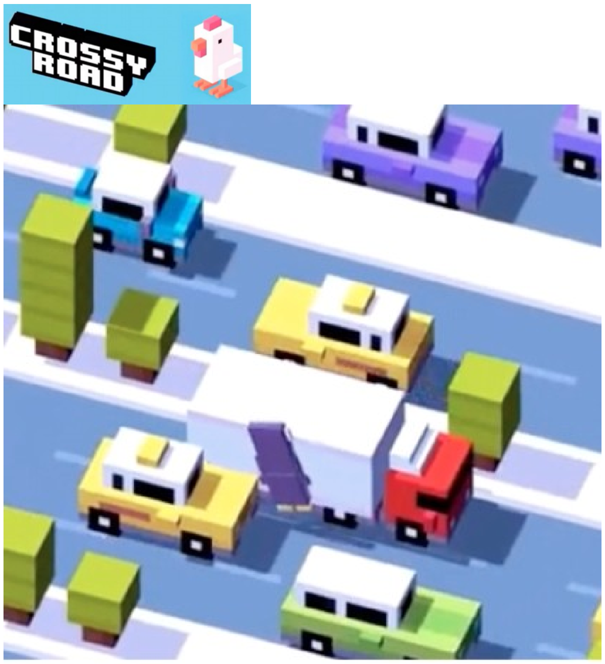
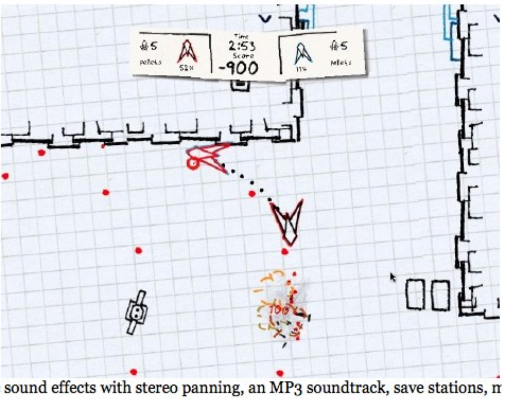
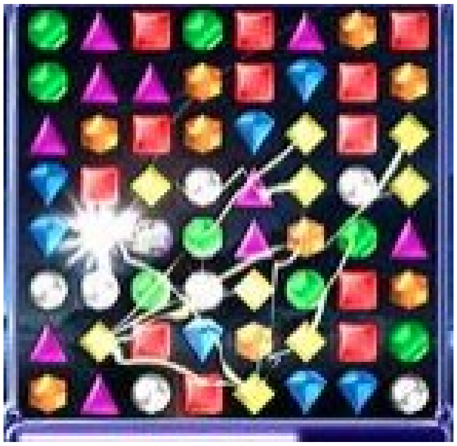
Assets (static meshes or sprites) can be created with pixel editors (inkscape, Piskel) or voxel editors (MagicaVoxel), or static mesh editors (Blender), or can be downloaded from asset libraries for Unreal or Unity. **Games are programmed** using a visual editing tool called Blueprints (for Unreal), or using C++ (Unreal) or C# (Unity). Downloaded assets have more functionality built-in, but you have complete artistic freedom over hand-created assets.

The **audio resources** you will need can be captured using an audio editor such as Audacity running on a laptop, or running as an iOS or Google app on a mobile phone, or can be downloaded from audio sites. Audio (humans shouting, swords clashing, traffic noises, rewards (ka-ching!), or losses in health, game starting, game over) add considerably to the fun of playing your game.

Animation effects can either be downloaded, or are frequently built-in to the game engine. If you wish a custom character to have custom animation, then you will have to use a tool such as Blender to rig and animate the static mesh.

Landscaping your game adds a great deal to its appeal, and there are excellent tutorials on landscaping for Unreal and for Unity. Both of these engines can create realistic looking hills, water, trees, bushes, grass, rocks, sand, …, with branches and grass whipping in the wind.

Carefully consider camera placement (height, angle, number, movement) to add to your game's interest. The cameras are your player's eyes, so the player sees what your camera(s) see. Similarly, consider **light placement**, their brightness, whether they are on only some of the time, and so on.

 An endless running game, using swipes to move forward, sliding under, jumping over, or turning direction. **Objective: survival by running as far as you can**	 Taps on the screen indicate if you are moving forward, backward, left, or right. (in this case, the pony misjudged it jump, ending up on the side of the truck). **Objective: survival by crossing the most obstacles possible**
 le sound effects with stereo panning, an MP3 soundtrack, save stations, n Particles can be emitted as a ship turns, either as a weapon shooting, or the exhaust from the ship **Objective: explore the map and survive**	 Animated effects with sound are more interesting **Objective: match the most color combinations**

Above are some examples of games you might choose to implement (CrossyRoad is not allowed for students co-enrolled in CPSC-386, the Intro to Game Development class, as they are already building it in that class using Unreal).

Your game should have a **clear objective** (clear to the player within a short time). The game's rules should be easy to grasp, the game should have many outcomes due to built-in randomness, and (c) it should be difficult to win (but not impossibly difficult). Example: Go, Chess, Crossy Road, Temple Run, Bejeweled.

Submission

Turn in the code for this project by uploading all of the Unreal (Unity) source files you created, the images directory, and the sounds directory to a single public repository on GitHub. While you may discuss this homework assignment with other students. Work you submit must have been completed on your own.
To complete your submission, print the following sheet, fill out the spaces below, and submit it to the professor in class by the deadline. Failure to follow the instructions exactly will incur a 10% penalty on the grade for this assignment.

CPSC 411 Final Project, due Friday, 19 May 2019 (at 2355)

Verify each of the following items and place a checkmark in the correct column.
Items incorrectly marked items will incur a 5% penalty each on the grade for this assignment.

Your name: ____

Your github repository ____

Completed	Not Completed	Unreal4 Engine 3d/2d Game
□	□	**Game successfully installs** on your mobile device (□ iOS or □Android)
□	□	Game has **startup screen,** which shows each time the game is (re-)started.
□	□	**Implemented the game's HUD (head's up display)** showing the high score, and the current score. If the game has a time limit, the game should end at timer=0.
□	□	**Implemented game assets** using a Pixel or Voxel editor, or static mesh creator
□	□	**Imported all assets into Unreal**, and rotated and scaled them properly.
□	□	**Bound gestures, keys** to their corresponding movements/rotations.
□	□	Actors **jump, run, look in direction they are moving** (WSAD) (no sweeping).
□	□	**Dynamically create actors** when the game starts, when characters are killed and are brought back to life, or when opponents or obstacles are needed.
□	□	**Dynamically created aspects** of the world as the map is explored, and populate them with needed static objects (e.g., roads with cars/trucks, hills with trees,...
□	□	**Created cameras, attached them to actors** or map locations as needed
□	□	**Collisions with obstacles** like trees, rocks, walls, etc. cause the actor to stop.
□	□	**Collisions with high-speed obstacles** causes loss of health or death.
□	□	**Incorporated particle systems in the game (fire, explosion, splash, shots).**
□	□	**Implemented the dynamic generation/destruction code** for allowing the level to be continuously populated as the actor moves forward.
□	□	**Implemented background music and game sounds**
□	□	**At least one other player has played your game** and signed off on it **as fun**.
□	□	Optional (extra credit): **1st AND 3rd person perspective**
□	□	Project directory **pushed to new GitHub repository** listed above

Comments on your submission