3D Games Programming
CI7500
Day 01: Introduction to C# and Unity

Working with C# and Unity

Exercises

Exercise 1: Overview of C#

- Create a solution in C# and add the classes Vehicle, Car, DrawingObject, Line, Circle and Square and their dependences. Create an object called myCar; give it the same initial location as for the myVehicle object; display its location, move it, display again its new location and finally open the third door.

- For the classes related to the DrawingObjects create a method called Draw for all of them and in order to test them create an array of 4 DrawingObjects elements. Define the first as a Line, the second as a Circle, the third as a Square and the forth as DrawingObjects. For all of them call the Draw method

- Open the project called 'ControlFlow'; observe the code and the syntax of the commands; execute it and use some of the internal commands (e.g. hello, exit, time, count, dir, game).

Exercise 2: More on C#

- Implement an algorithm to determine if a string has all unique characters.
  bool isUniqueChars(string str)

- Implement a function which reverses a null terminated string.
  string reverse(string str)

- Given two strings, write a method to decide if one is a permutation of the other (e.g. "asd" "ads")
  bool permutation(string s, string t)

- Write a method to replace all spaces in a string with'%20'. You may assume that the string has sufficient space at the end of the string to hold the additional characters, and that you are given the "true" length of the string. (Note: use a character array so that you can perform this operation in place.)
  <u>EXAMPLE</u>
  Input: "Mr John Smith"
  Output: "Mr%20John%20Smith"
  char[] replaceSpaces(char[] str, int length)

- Implement a method to perform basic string compression using the counts of repeated characters. For example, the string aabcccccaaa would become a2blc5a3. If the "compressed" string would not become smaller than the original string, your method should return the original string.
  string compressBad(string str)

- Given an image represented by an NxN matrix, where each pixel in the image is 4 bytes, write a method to rotate the image by 90 degrees. Can you do this in place?
  int[,] rotate(int[,] matrix, int n)

- Write an algorithm such that if an element in an MxN matrix is 0, its entire row and column are set to 0.
  int[,] setZeros(int[,] matrix, int s1, int s2)

- Assume you have a method Contains which checks if one word is a substring of another. Given two strings, s1 and s2, write code to check if s2 is a rotation of s1 using only one call to Contains (e.g.,"waterbottle"is a rotation of "erbottlewat").
  bool isRotation(string s1, string s2)

Exercise 3: Overview of Unity3D

Create a new project and try to familiarise yourselves with the editor and the tools following the first lecture notes. Create objects and play with the transformation options etc. Try to import models e.g. Torus.fbx, see how the Textures and Materials are working e.g. BrickWall

Exercise 4: Terrain design

Create a new scene and add a terrain, textures for grass, trees, details, a skybox, fog, water, etc. following the instruction in the second lecture. Add also character controller.

Exercise 5: Lights

Create a new scene and following the instructions in the second lecture add lights, halos, and cooklie.

Exercise 6: Cameras

Create a new scene and add multiple cameras, create a split screen and a Picture-in-Picture Effect. Also try to play a bit with layers.

Exercise 7: Game Demo

Create a new scene and build the racing demo presented in the third lecture.