INFT3960 – Game Production

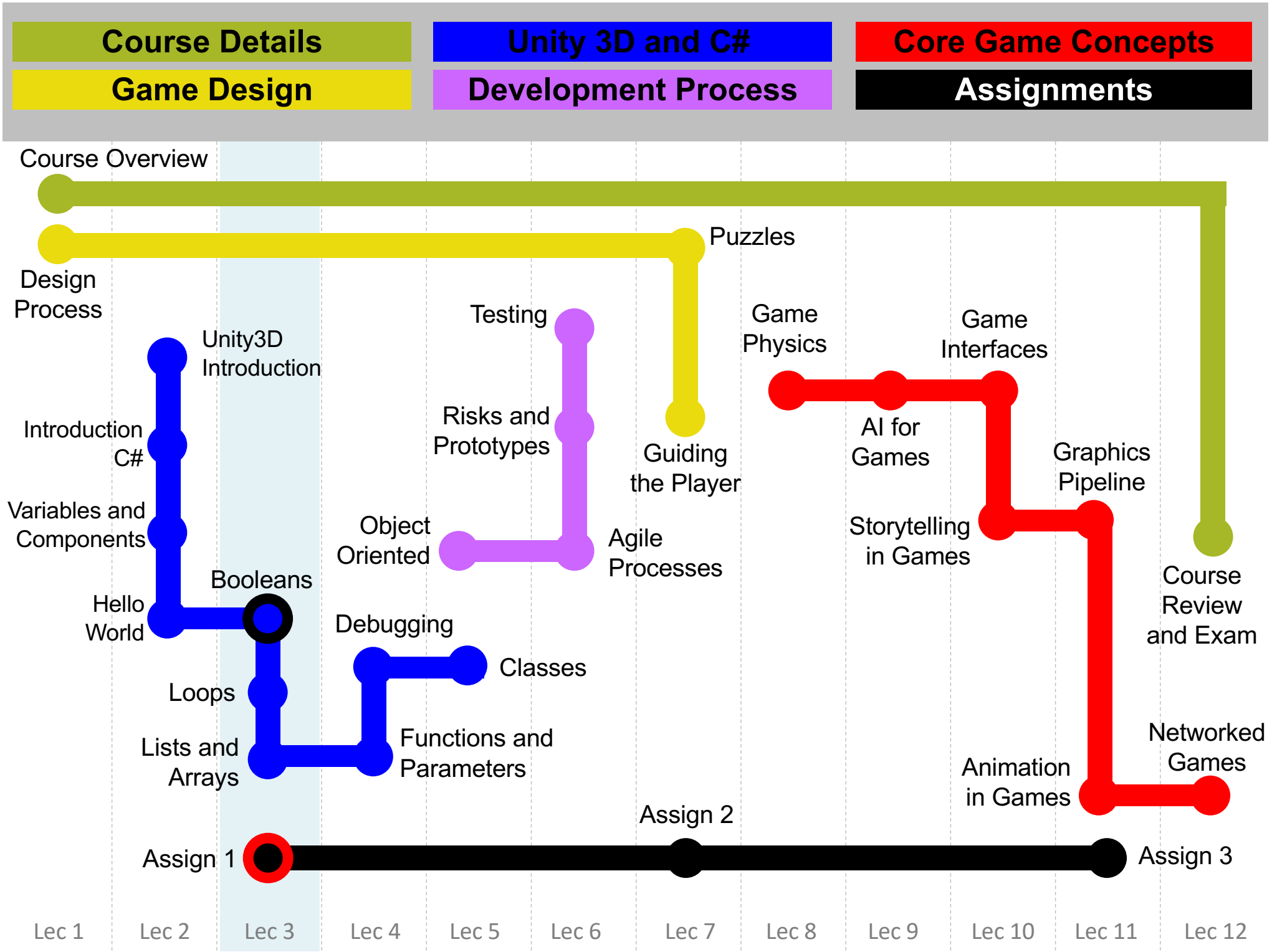
Week 03

Module 3.1

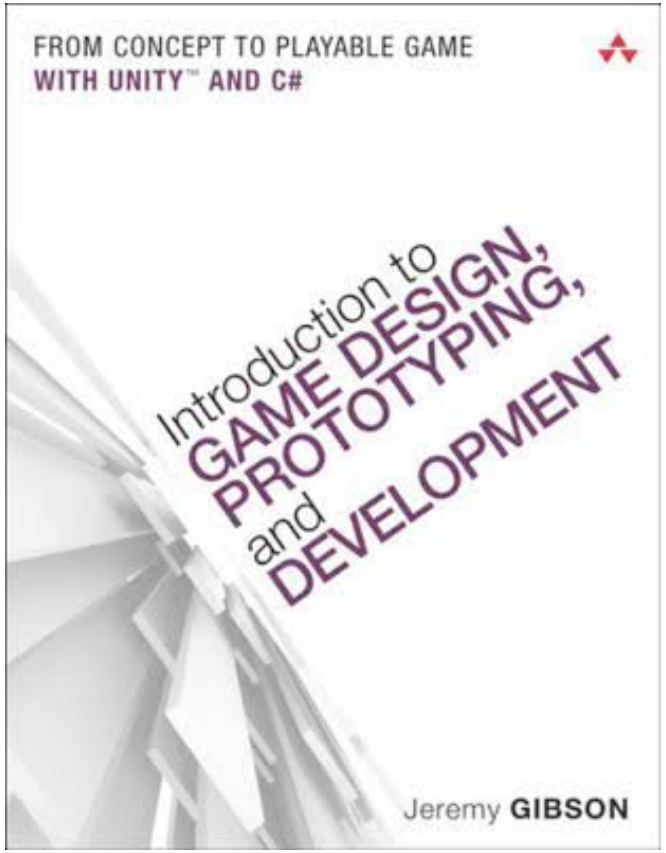
Booleans and Conditionals

Course Overview

Lec	Start Week	Modules	Topics	Assignments
1	3 Aug	Mod 1.1, 1.2	Course Overview, Design Process	
2	10 Aug	Mod 2.1, 2.2, 2.3, 2.4	Unity3D Introduction, Introduction C#, Variables and Components, Hello World	
3	17 Aug	Mod 3.1, 3.2, 3.3	Booleans, Loops, Lists and Arrays	Assign 1 21 Aug, 11:00 pm
4	24 Aug	Mod 4.1, 4.2	Functions and Parameters, Debugging	
5	31 Aug	Mod 5.1, 5.2	Classes, Object Oriented	
6	7 Sep	Mod 6.1, 6.2, 6.3	Agile Processes, Risks and Prototypes, Testing	
7	14 Sep	Mod 7.1, 7.2	Puzzles, Guiding the Player	Assign 2 18 Sep, 11:00 pm
8	21 Sep	Mod 8.1	Game Physics	
9	12 Sep	Mod 9.1	AI for Games	
10	19 Oct	Mod 10.1, 10.2	Game Interface, Storytelling in Games	
11	26 Oct	Mod 11.1, 11.2	Graphics Pipeline, Animation in Games	Assign 3 1 Nov, 11:00pm
12	2 Nov	Mod 12.1, 12.2	Networked Games, Course Review	


Course Details
Unity 3D and C#
Core Game Concepts
Game Design
Development Process
Assignments
Course Overview
Puzzles
Design Process
Testing
Game Physics
Game Interfaces
Unity3D Introduction
Introduction C#
Risks and Prototypes
Guiding the Player
AI for Games
Graphics Pipeline
Variables and Components
Object Oriented
Agile Processes
Storytelling in Games
Course Review and Exam
Booleans
Hello World
Debugging
Classes
Loops
Functions and Parameters
Lists and Arrays
Networked Games
Animation in Games
Assign 2
Assign 1
Assign 3
Lec 1
Lec 2
Lec 3
Lec 4
Lec 5
Lec 6
Lec 7
Lec 8
Lec 9
Lec 10
Lec 11
Lec 12

Booleans etc– (Chapter 20)



BOOLEAN
OPERATIONS
AND
CONDITIONALS

Booleans – Topics

- Boolean Operations
  - Shorting vs. Non-Shorting
  - Combination of Boolean Operations
  - Order of Operations
- Comparison Operators
- Conditional Statements
  - `if`
  - `if/else`
  - `if / else if / else`
  - `switch`

Boolean Operations

Operations that combine and compare bools

`!`	The NOT Operator
`&&`	The AND Operator
`||`	The OR Operator

! the NOT Operator

! is pronounced either "not" or "bang”

Reverses value of the bool

```
print( !true );      // Outputs: false
print( !false );     // Outputs: true
print( !(!true) ); // Outputs: true (the double negative of true)
```

! also called the "logical negation operator"

This differentiates it from ~, the bitwise not operator

&& the AND Operator

Returns true only if both operands are true

```
print( false && false );     // false
print( false && true  );     // false
print( true  && false );     // false
print( true  && true  );     // true
```

|| the OR Operator

Returns true if either operand is true

```
print( false || false );    // false
print( false || true  );    // true
print( true  || false );    // true
print( true  || true  );    // true
```

| (the pipe) is Shift-Backslash (Just above the return or enter key on a US keyboard

Shorting and Non-Shorting

&& and || are shorting operators

- If the first operand of && is false, the second is not evaluated
- If the first operand of || is true, the second is not evaluated
- Only important if you have a boolean function as the second operand and it does something important

& and | are non-shorting operators

- Both operands are evaluated regardless of value

& and | are also bitwise operators

- & and | compare each bit of the values passed into them
- Bitwise operators will be used much later when dealing with Unity layers and collisions

Comparison Operators

Allow the comparison of two values

Return a bool (either true or false)

```
==    Is Equal To
!=    Not Equal To
>     Greater Than
<     Less Than
>=    Greater Than or Equal To
<=    Less Than or Equal To
```

COMPARISON BY VALUE OR REFERENCE

Simple variables are compared by value

```
bool, int, float, char, string,
Vector3, Color, Quaternion
```

More complex variables are compared by reference

When variables are compared by reference, the comparison is not of their internal values but of whether they point to the same location in memory

GameObject, Material, Renderer, HelloWorld (and other C# classes you write)

```
1 GameObject go0 = Instantiate( boxPrefab ) as GameObject;
2 GameObject go1 = Instantiate( boxPrefab ) as GameObject;
3 GameObject go2 = go0;
4 print( go0 == go1 ); // Output: false (different location)
5 print( go0 == go2 ); // Output: true (same location)
```

Comparison – Is Equal To

Returns true if the values or references compared are equivalent

```
print( 10 == 10 );           // Outputs: True
print( 20 == 10 );           // Outputs: False
print( 1.23f == 3.14f );     // Outputs: False
print( 1.23f == 1.23f );     // Outputs: True
print( 3.14f == Mathf.PI ); // Outputs: False
// Mathf.PI has more decimal places than 3.14f
```

Do NOT confuse == and =

== The comparison operator

= The assignment operator

Comparison – Not Equal To

Returns true if the values or references compared are NOT equivalent

```
print( 10 != 10 );          // Outputs: False
print( 20 != 10 );          // Outputs: True
print( 1.23f != 3.14f );    // Outputs: True
print( 1.23f != 1.23f );    // Outputs: False
print( 3.14f != Mathf.PI ); // Outputs: True
```

Comparison – Greater Than

Returns true if the 1st operand is greater than the 2nd

```
print( 10 > 10 );           // Outputs: False
print( 20 > 10 );           // Outputs: True
print( 1.23f > 3.14f );     // Outputs: False
print( 1.23f > 1.23f );     // Outputs: False
print( 3.14f > 1.23f );     // Outputs: True
```

Comparison – Less Than

Returns true if the 1st operand is less than the 2nd

```
print( 10 < 10 );           // Outputs: True
print( 20 < 10 );           // Outputs: False
print( 1.23f < 3.14f );     // Outputs: True
print( 1.23f < 1.23f );     // Outputs: True
print( 3.14f < 1.23f );     // Outputs: False
```

Greater Than or Equal To

True if the 1st operand is greater than or equal to the 2nd

```
print( 10 >= 10 );          // Outputs: True
print( 20 >= 10 );          // Outputs: True
print( 1.23f >= 3.14f );    // Outputs: False
print( 1.23f >= 1.23f );    // Outputs: True
print( 3.14f >= 1.23f );    // Outputs: True
```

Less Than or Equal To

True if the 1st operand is less than or equal to the 2nd

```
print( 10 <= 10 );          // Outputs: True
print( 20 <= 10 );          // Outputs: False
print( 1.23f <= 3.14f );    // Outputs: True
print( 1.23f <= 1.23f );    // Outputs: True
print( 3.14f <= 1.23f );    // Outputs: False
```

Conditional Statements

Control Flow Within Your Programs

```
if
if / else
if / else if / else
switch
```

Can be combined with Boolean operations

Make use of braces { } (to specify conditional statements)

Conditional Statements

`if` Performs code within braces if the argument within parentheses is true

```
if (true) {
    print( "This line will print." );
}

if (false) {
    print( "This line will NOT print." );
}

// The output of this code will be:
//    This line will print.
```

All the code within the braces of the if statement executes

Conditional Statements

Combining if statements with boolean operations

```
bool night = true;
bool fullMoon = false;

if (night) {
    print( "It's night." );
}
if (!fullMoon) {
    print( "The moon is not full." );
}
if (night && fullMoon) {
    print( "Beware werewolves!!!" );
}
if (night && !fullMoon) {
    print( "No werewolves tonight. (Whew!)" );
}

// The output of this code will be:
//     It's night.
//     The moon is not full.
//     No werewolves tonight. (Whew!)
```

Conditional Statements

Combining if statements with comparison operators

```
if (10 == 10 ) {
    print( "10 is equal to 10." );
}

if ( 10 > 20 ) {
    print( "10 is greater than 20." );
}

if ( 1.23f <= 3.14f ) {
    print( "1.23 is less than or equal to 3.14." );
}

if ( 1.23f >= 1.23f ) {
    print( "1.23 is greater than or equal to 1.23." );
}

if ( 3.14f != Mathf.PI ) {
    print( "3.14 is not equal to ” + Mathf.PI + "." );
    // + can be used to concatenate strings with other data types.
    // When this happens, the other data is converted to a string
```

if else

Performs one action if true, and another if false

```
bool night = false;

if (night) {
    print( "It's night." );
} else {
    print( "What are you worried about?" );
}

// The output of this code will be:
//     What are you worried about?
```

if / else if / else

Possible to chain several else if clauses

```
bool night = true;
bool fullMoon = true;

if (!night) {                    // Condition 1 (false)
    print("It's daytime. What are you worried about?");
} else if (fullMoon) {           // Condition 2 (true)
    print( "Beware werewolves!!!" );
} else {                         // Condition 3 (not checked)
    print( "It's night, but the moon is not full." );
}

// The output of this code will be:
//     Beware werewolves!!!
```

Nested `if` statements

Possible to chain several else if clauses

```
bool night = true;
bool fullMoon = false;

if (!night) {
    print( "It's daytime. Why are you worried about?" );
} else {
    if (fullMoon) {
        print( "Beware werewolves!!!" );
    } else {
        print( "It's night, but the moon isn't full." );
    }
}

// The output of this code will be:
//     It's night, but the moon isn't full.
```

`switch`

Alternative to several if statements

Can only compare for equality

Can only compare against a single variable against literals

switch

```
int num = 3;

switch (num) {  // variable in parentheses is being compared

    case (0):  // Each case is a literal - compared against num
        print( "The number is zero." );
        break;  // Each case must end with a break statement.

    case (1):
        print( "The number is one." );
        break;

    case (2):
        print( "The number is two." );
        break;

    default:  // none of the other cases true, default happens
        print( "The number is more than a couple." );
        break;
    }  // The switch statement ends with a closing brace.

// The output of this code is: The number is more than a couple.
```

switch

```
int num = 3;
switch (num) {
    case (0):
    case (1):
    case (2):
        print( "The number is less than 3" );
        break;
    case (3):  // case (3) falls through to case (4)
    case (4):  // case (4) falls through to case (5)
    case (5):
        print( "The number is a few." );
        break;
    default:
        print( "The number is more than a few." );
        break;
}
// The output of this code is: The number is a few.
```

Summary

Boolean Operations: `!` `&&` `||` `&` `|`

Learned about "shorting operations"

Boolean operations can be combined

Comparison Operators: `==` `!=` `>` `<` `>=` `<=`

Conditional Statements: `if` `if…else` `switch`

if and switch statements can be combined in complex ways