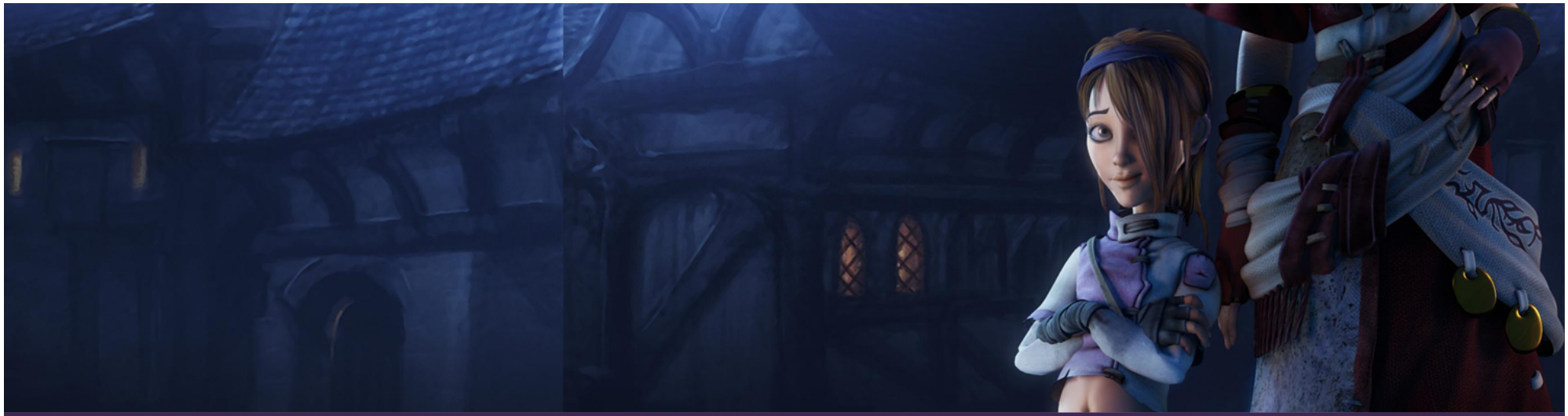

FALMOUTH
UNIVERSITY

Programming Workshop 4: Objected Oriented Programming in C++

GAM340: Professional Practice
BA(Hons) Game Development

FALMOUTH UNIVERSITY

Digital Attendance

- The system replaces paper-based registration with a network of card readers in Learning & Teaching spaces around the college.
- We hope it will save lots of time!
- You 'TAP' your ID card to register your attendance in a session.
- The window for registration is from 15 minutes before a session is timetabled to start through to 15 minutes after the start time.

- Learning Objectives
  - **Understand** the C++ object model
  - **Explain** issues with inheritance in C++
  - **Use** const correctness
  - **Demonstrate** an awareness of design patterns for C++
  - **Solve** basic C++ Architectural class programming interview problems

- structs in C -> classes in C++
  - C doesn't support classes, but it does support structures
    - https://onlinegdb.com/rkJmxZLOB
    - Code and data are split
    - Can use this approach to make fundamental C containers
      - Linked List
      - Binary tree
      - Directed Acyclic Graph
      - 'Safer' string classes
      - 'Maths' vector, matrix etc
      - etc

- structs in C -> classes in C++
  - Object orientation provides 4 features that make programming much easier
    - (Common interview question)
      - Encapsulation
      - Data Hiding
      - Inheritance
      - Polymorphism

      - https://onlinegdb.com/SyAJmWLOB

- structs in C -> classes in C++
  - https://onlinegdb.com/SyAJmWLOB
  - Encapsulation
    - 'Put into a capsule'
    - Code and data are stored together
  - Data Hiding
    - Limit visibility of data and functions
    - Public – is visible to users of a class
    - Private – can only be used in the class
      - » In the Vector class, X & Y can only be accessed by functions in the class
    - Other types are available
      - » Protected & friend
  - Inheritance
    - Base class of 'GameObject'
      - » Note static initialisation
      - » Has private & protected members
    - Vector class derives from GameObject
      - » Where it uses GameObject ctor(), they are explicit
    - Overrides GameObject virtual function
      - » And calls base functionality
  - Polymorphism
    - The 'ability to take on many forms'
    - Static – functions
      - » Functions with the same name take different arguments
        - ctor(), Set() & Add()
    - Dynamic - object casting
      - » Code creates a array of GameObject ptrs
      - » Stores vector ptrs in them
      - » DebugPrint will call derived function, even though it's an array of base object

- structs in C -> classes in C++
  - Compare and contrast C# OO with C++
    - C# classes you know and love:
      - » https://onlinegdb.com/S1Jo7ycdB
    - C++ equivalents:
      - » https://onlinegdb.com/r1e_IJc_H

- structs in C -> classes in C++
  - Abstract base classes
    - Can't be instantiated, only used to create derived classes
    - In C#, abstract is used as a modifier for the class definition
    - In C# at least one function is defined as pure virtual
      - `virtual void fn() = 0;`

- structs in C -> classes in C++
  - Operators & operator overloading
    - Operators were added to languages to give a more natural feel to writing algorithms
      - A = B+ C;
      - vector_Add(&A,&B,&C);
    - Operators can be chained together
      - A = ((B + C) * (D*F)) / G;
    - Anything can have operators written for them
      - Doesn't always make sense
        - » Linked List += makes sense
        - » Linked List *= doesn't

- structs in C -> classes in C++
  - Operators & operator overloading
    - In C#
      - https://onlinegdb.com/BJ0Flg9dr
    - In C++
      - https://onlinegdb.com/SJyKxx5OS
    - C# addresses a lot of the ‘issues’ with C++ operators
      - Removes need for == and !=, and += and +
      - Remember deep and shallow copy assignments!
    - Both languages create lots of temp objects
      - Common to see ‘c style’ math functions
        - » Vector_op(&result, &argA, &argB)

- structs in C -> classes in C++
  - const correctness
    - A very common interview question
    - Not a feature in C#, but used extensively in C++ to create read_only object pointers as return types, functions and arguments.
    - Const correctness is a compile time feature
      - It will stop code from compiling, rather than leaving you to find bugs/issues at runtime
    - Const correctness will become pervasive in projects
      - Const return types will lead to const arguments in functions across the code base

- structs in C -> classes in C++
  - const correctness
    - A normal project
      - https://onlinegdb.com/rkwsL3iOH

```
Data* GetData()
{
    return pData;
}
```

```
DataHolder DH;

DH.GetData()->a = 4;
```

- structs in C -> classes in C++
  - const correctness
    - A normal project
      - https://onlinegdb.com/ByEC83suH

```
const Data* GetData()
{
    return pData;
}
```

```
DH.GetData()->a = 4;
```

- structs in C -> classes in C++
  - const correctness
    - A normal project
      - https://onlinegdb.com/ByEC83suH
      - Add const correctness to functions that are to take Data*
      - And functions that are members of the Data class

```
main.cpp: In function 'int main()':
main.cpp:42:23: error: assignment of member 'Data::a' in read-only object
     DH.GetData()->a = 4;
                   ^
main.cpp:44:29: error: invalid conversion from 'const Data*' to 'Data*' [-fpermissive]
     DH.UpdateData(DH.GetData());
                   ~~~~~~~~~~^~
main.cpp:32:10: note:   initializing argument 1 of 'void DataHolder::UpdateData(Data*)'
     void UpdateData(Data* d)
          ^~~~~~~~~~
main.cpp:46:24: error: passing 'const Data' as 'this' argument discards qualifiers [-fpermissive]
     DH.GetData()->GetA();
                   ^
main.cpp:10:13: note:   in call to 'int Data::GetA()'
         int GetA()
             ^~~
```

- structs in C -> classes in C++
  - Programming tests with C/C++
    - Generally looking at
      - Understanding of C++ OO features and issues
      - Ability to use
        - » ABCs
        - » Create and use standard containers
        - » Create sensible class hierarchies
        - » Use of const correctness

- OO Programming tests C++
  - Linked List class (of type x)
    - https://en.wikipedia.org/wiki/Linked_list#Singly_linked_list
  - Sorted linked List (of type x)
    - https://en.wikipedia.org/wiki/Doubly_linked_list
  - Dynamic array (of type x)
    - https://gameprogrammingpatterns.com/object-pool.html
  - Binary tree
    - https://en.wikipedia.org/wiki/Binary_tree
  - Balancing a binary tree
    - https://en.wikipedia.org/wiki/Self-balancing_binary_search_tree
  - Linked List with ABC & derived classes
    - Create ABC linkList node, derive your LL-able classes from it
  - String class implementation with operators & const correctness
    - Use operators to replace strcat, strcpy and anything else that makes sense