**Project Idea: 2D Games using Monogame (XNA)**

Games can give you many opportunities to explore some advanced programming features but it is important that you are comfortable with object orientated programming in C#. You will have downloaded monogame ([www.monogame.net](http://www.monogame.net)) and any other add- ins e.g. Xamarin if needed.

Using **UNITY** instead (https://**unity**3d.com/)

1. – Although Unity is another option for game development its licensing makes it impossible for us to support this in college.

You must understand the game lifecycle:

* Instantiation
* Initialization
* Loading
* Update

Repeat 30 times a second!

* Draw
* Unloading

**Preparation and Research**

Complete these tutorials and create some sort of basic 2D (or if you are very ambitious 3D) game.

**Text tutorials**

<https://roguesharp.wordpress.com/>

<http://rbwhitaker.wikidot.com/monogame-tutorials>

<http://www.informit.com/articles/index.aspx?st=98709> 🡨very good for writing OOP games

**Video tutorials**

There are several video series too you can follow for example

<https://www.youtube.com/watch?v=agt9-J9RPZ0>

<https://www.youtube.com/watch?v=EXzB_MyZRKM>

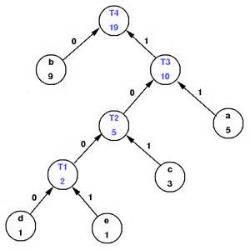
Features you will have to use in a game in order to maximise the complexity of the game.

* Data structures
* Vectors
* Matrix operations for transforming vectors – rotation, movement etc.
* Fully object-orientated.
* Game AI algorithms- e.g. minimax
* Recursion

**HEALTH WARNING**

Games can be hard. The biggest danger is not having a good end user with clear user requirements – i.e. you don’t have a clear idea of what the game is going to do. Without this you will struggle with the analysis and design aspect of the project. Games also tend to have limited data storage requirements which puts more emphasis on the quality of the program itself. Do not attempt a game if you are not prepared to do the spadework and learn how monogame works first.

**Project Idea: Data Compression using Huffman Trees**



Huffman encoding is a lossless compression technique used compression software such as WinZip. It is based upon counting the frequency of letters in a piece of text and then constructing a binary tree where each character is represented by a binary value whose length is proportional to the frequency it appears in the text.

This should be a windows based project but could be done in console mode (wouldn’t look as nice)

The goal of any project would be to develop a text compression app. It gives you the opportunity to load and save data, to use data structures such as trees and dictionaries and there is also the opportunity for recursion when processing the tree structure.

You could carry out some mathematical processing to identify compression ratios too. You could even go on to develop some sort of dictionary encoding technique as well and compare the performance of the two.

**Preparation and Research**

Make sure you understand how Huffman encoding works

<https://en.wikipedia.org/wiki/Huffman_coding>

<https://www.youtube.com/watch?v=dM6us854Jk0>

You will also need to familiarise yourself with the binary tree structure. This is something we shall be doing in upper 6th theory anyway. Here’s an object orientated implementation of a binary tree in C#

<https://msdn.microsoft.com/en-us/library/ms379572(v=vs.80).aspx>

Features you will have to use to maximise the complexity.

* Data structures
* Objects
* Recursion
* File operations
* Sorting

**Project Idea: An encrypted messaging app.**

Although these are the bane of the intelligence services an encrypted app makes for a pretty good project as it combines encryption with networking.

This would be a windows based project.

You would have to develop an app which is more client-server than peer-to-peer. The message server would listen on an unassigned port (this is a bit of a problem in college as we block unassigned ports! – don’t worry we have work-arounds) for clients to connect. They would have to send a password (hashed) which the server would check. Clients connected to the server via the port and would encrypt and send data to the server probably using some sort of Vignere XOR encryption. The server stores the message ready for the message recipient to start a session. When the receiver logs on and connects to the server the server could decrypt the message with the sender’s key, re-encrypt it with the receiver’s key and then send the message to the receiver’s inbox.

You could extend the encryption further using public-key encryption (very handy for initial key exchange) and the mathematics involved here.

This is actually easier than maybe I’ve made it sound above!!

**Preparation and Research**

An introduction to socket programming.

<http://www.codeproject.com/Articles/10649/An-Introduction-to-Socket-Programming-in-NET-using>

<https://msdn.microsoft.com/en-us/library/w89fhyex(v=vs.110).aspx>

<http://csharp.net-informations.com/communications/csharp-socket-programming.htm>

There are some examples of client server chat system using TCP e.g.

<http://www.dreamincode.net/forums/topic/33396-basic-clientserver-chat-application-in-c%23/>

XOR encryption

<https://en.wikipedia.org/wiki/XOR_cipher>

Features you will have to use to maximise the complexity.

* Database tables – messages, users and associated SQL
* Hashing algorithms
* Objects
* Multi-threading for managing concurrent client connections.
* Data structures e.g. hash table/dictionary of connected users.
* Network protocols – ports, TCP listeners, TCP servers, TCP clients, streamreaders and streamwriters.
* Mathematics of PK encryption

**Project Idea: Web-based Apps.**

***About web-based projects***

Web based projects will require a bit of practice first. There is a bit of a learning curve regarding the events that trigger server sided-code (which is just C#) as well as HTML and CSS but it’s pretty straightforward to pick-up. The Visual Studio environment allows us to run these projects locally too so no messing about setting up remote web-server.

There are two approaches to setting up websites: MVC or Web-forms. For the single developer newbie I would probably recommend Web-forms.

With web projects you can get a bit hung up with what they look like and if they look ok on computer, tablet or phone. Don’t worry too much about this, in fact you can download bootstrap - <http://getbootstrap.com/> which provides a nice ‘look’ to your website and addresses many of these concerns (it’s basically a bunch of CSS rules and JavaScript goodies so you don’t have to write them yourself). In fact later versions of Visual Studio (2013 onwards I think) provide it as a built in option.

***Social media App***

You could create some sort of social media app that incorporates posts, likes, dislikes, friends/followers. Users can ask to be friends, post, like and dislike, repost other’s posts. You could, by applying graph theory, recommend people that you should follow/befriend. You could extend it further to incorporate multi-media posts. One option would be to give it an education focus.

***E-Commerce App***

http://tse1.mm.bing.net/th?&id=OIP.Me20a0c53808e78d54e4a426b22e3fa4fo0&w=198&h=198&c=0&pid=1.9&rs=0&p=0&r=0You could create an online shopping or booking site. You could incorporate shopping baskets, stock lists, customer login as well as generating management reports. Again this could be extended to recommend products based on previous purchasing habits.

**Preparation and Research**

**Understanding html and CSS**

[**http://www.w3schools.com/css/**](http://www.w3schools.com/css/)

[**http://www.w3schools.com/html/default.asp**](http://www.w3schools.com/html/default.asp)

**and bootstrap…** [**http://www.w3schools.com/bootstrap/default.asp**](http://www.w3schools.com/bootstrap/default.asp)

**How to develop websites in Visual Studio**

[**http://www.asp.net/web-forms**](http://www.asp.net/web-forms)

**How to create a database driven website in Visual Studio**

[**https://www.youtube.com/watch?v=JOJV-Z7cmkc**](https://www.youtube.com/watch?v=JOJV-Z7cmkc)

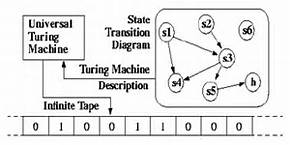
Features you will have to use to maximise the complexity.

* Lots of database tables and associated SQL
* Objects (these make for a very good object orientated systems)
* By representing info in a graph we can use this to recommend new friends or products.
* Analytics for management

**Project Ideas: A Virtual Turing Machine**

Throughout the course we have (and will) use simulators and other software. Developing an equivalent or better version would be a good upper 6th challenge. In Computer Science we use the following (may be you can think of something in another subject you study).

**A Virtual Turing Machine**

In the A Level course we will be studying the concept of Turing machines. A Turing machine is a Finite State Machine with memory in the form of an infinitely long tape accessible at the tape head. We would like a program that allows a student to put information on the tape (within the constraints of the alphabet) and create an FSM in the form of a state transition table or rules. They can then ‘run’ their program and see how this affects the tape. Programs can be saved. There is opportunity to validate data entered (possibly using regex expressions).

**Preparation and Research**

**Background to Turing machines**

[**http://www.alanturing.net/turing\_archive/pages/Reference%20Articles/What%20is%20a%20Turing%20Machine.html**](http://www.alanturing.net/turing_archive/pages/Reference%20Articles/What%20is%20a%20Turing%20Machine.html)

**Example of online simulators (it’s really worth getting your head round these)**

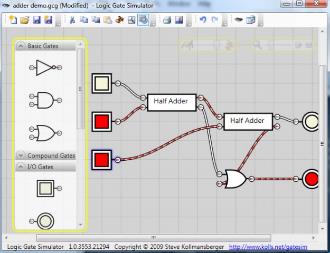
[**https://turingmachinesimulator.com/**](https://turingmachinesimulator.com/)

[**http://www.turing.org.uk/book/update/tmjavar.html**](http://www.turing.org.uk/book/update/tmjavar.html)

Features you will have to use to maximise the complexity.

* Database tables and associated SQL
* Objects (these make for a very good object orientated systems - A Turing machine is an object which contains a program object which contains a state transition rule object etc.)
* Searching and processing instructions
* Using regex to validate instructions
* Data structures

**Project Idea: Circuit Builder & Boolean expression generator**

It would be useful if we could have a windows-based program to draw logic circuits. The app would have a menu of gates and other components which could be dragged and dropped onto the drawing service. These could then be connected to form circuits. These circuits could be ‘run’ to show whether a circuit is generating an off or on result.

As an added extra the circuit could then be inorder ‘parsed’ to generate the appropriate Boolean expression for an output (If you consider a circuit as a Boolean tree). Circuits could also be saved and retrieved.

**Preparation and Research**

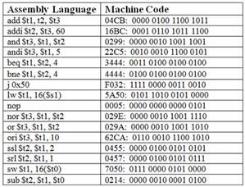
**Example of online circuit drawing tools**

[**http://www.neuroproductions.be/logic-lab/**](http://www.neuroproductions.be/logic-lab/)

[**https://scratch.mit.edu/projects/785418/**](https://scratch.mit.edu/projects/785418/) **(a scratch version)**

Features you will have to use to maximise the complexity.

* Database tables or files and associated SQL/file handling
* Objects
* Parsing (with recursion) to form Boolean expression.
* Data structures – linked objects, graphs, trees.

[](http://www.bing.com/images/search?q=assembly+language&view=detailv2&adlt=strict&id=63B71074D35C9EBEA078BFDD0234151DB6F32A18&selectedIndex=0&ccid=56RqKLAR&simid=608011853651575158&thid=OIP.Me7a46a28b011c480892cd58049950b56H0)**Project Idea: A Virtual Machine for writing Assembly Language**

During the course we have used a virtual machine to practise assembly language programming. It allows the computing student to see the contents of memory and registers. It animates the fetch-execute cycle by showing the movement of data, address and control signals along the system busses. The program would allow you to pause and restart execution and would validate instructions entered.

Your challenge would be to create something similar.

However it would also require the ability to save and retrieve assembly language programs and monitor student progress on challenges set by the teacher.

**Preparation and Research**

**Example of an online virtual machine**

[**http://www.peterhigginson.co.uk/RISC/**](http://www.peterhigginson.co.uk/RISC/)

Features you will have to use to maximise the complexity.

* Database tables and associated SQL
* Objects
* Interpreting instructions
* Using regex to validate instructions
* Data structures

**Project Idea: Processing Bitmaps**

C# allows you access to the individual pixels in a bitmap file loaded. This allows you the option of adjusting their values to achieve various affects.

For example…

* You could choose to write a graphics program to apply various artistic effects on an image (or images).
* You could investigate compression algorithms to reduce the image size.
* You could encrypt images.
* You could develop a drawing program to create animated gifs.

**Preparation and Research**

Here are some examples of code that processes bitmap images.

Image Processing

[**http://www.codeproject.com/Articles/33838/Image-Processing-using-C**](http://www.codeproject.com/Articles/33838/Image-Processing-using-C)

Converting a bitmap image into ASCII Art.

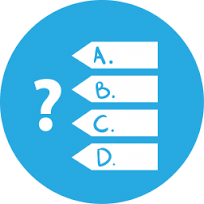
[**http://www.c-sharpcorner.com/article/generating-ascii-art-from-an-image-using-C-Sharp/**](http://www.c-sharpcorner.com/article/generating-ascii-art-from-an-image-using-C-Sharp/)

Image Blending

[**https://softwarebydefault.com/2013/04/22/bitwise-bitmap-blending/**](https://softwarebydefault.com/2013/04/22/bitwise-bitmap-blending/)

Features you will have to use to maximise the complexity.

* Mathematics - Using matrices to process images
* Saving and retrieving image files
* Objects

**Project Idea: A Quiz App**

This application would allow the users to set questions and quizzes – these could be, say, multiple choice or short answer type questions. Users could then run the quizzes and record the results which could then be analysed. You could have different levels of access –teacher and student with option of setting students into groups and having the teacher see how their students are performing. This project could developed on any platform – windows, phone or web.

It could be developed in a game format such as ‘who wants to be a millionaire’ or ‘pointless’.

The project could be extended further in the following ways

1. Allow multi-media content (images, video, sound) to be added to questions
2. Introduce time limits on questions.
3. Search feature for relevant questions
4. Create an intelligent system that has questions set at different levels of complexity and students would be asked questions based on their past performance. i.e. if a student starts getting a lot of simple questions right then start asking harder questions.
5. Approach the problems as an object orientated system
6. Develop multiplayer options – such as kahoot or socrative.

**Preparation and Research**

Here are some examples of quiz programs

[www.socrative.com](http://www.socrative.com) or [www.kahoot.com](http://www.kahoot.com)

Online quizzes e.g. <http://www.triviaplaza.com/>

Setting up a basic quiz in windows forms

<https://msdn.microsoft.com/en-us/library/dd492172.aspx>

Features you will have to use to maximise the complexity.

* Database tables and associated SQL
* Objects
* Data structures
* Possible use of graphs if creating ‘adaptable’ quizzes

**Project Idea: Using APIs**

APIs (application programming interface) are provided by many websites e.g. YouTube, Instagram, Twitter, facebook, GoogleMaps. These allow you to interrogate their database and will return data to you in the form of JSON or XML which you can then process for your own application. You have to be slightly careful about what APIs you use. You will require an API key to access the data and there will be restrictions on the number of requests you are able to make. In some cases the website may charge you but often relatively minimal use is free.

We could process information in all sorts of interesting ways.

However another source of APIs is the operating system. For example you can make an API call to find your geolocation (on mobile devices)

**Preparation and Research**

Here are some bizarre APIs you could incorporate.

<http://blog.mashape.com/list-of-fun-and-interesting-apis-to-try-out/>

The programmable web lets you see what APIs are out there…

<http://www.programmableweb.com/>

Article on using social media APIs

<http://www.poynter.org/2011/how-to-use-apis-from-google-facebook-twitter-to-find-data-ideas/141786/>

Try twitter console for an idea of the sort of stuff you can get

<https://dev.twitter.com/rest/tools/console>

Features you will have to use to maximise the complexity.

* Database tables and associated SQL
* Objects
* Parsing xml/json data
* Data structures

**Project Idea: Optimisation algorithms**

Optimisation algorithms are ways of identifying a best (or at least a good) solution to certain types of problems. Some typical applications are…

**Route finding**

When trying to find the best route from one location to the other (e.g satnav type problems) on a graph then and this would be an application of Dijkstra’s shortest path algorithm. E.g. an app to show the quickest route for new students to find their way between classrooms in college.

**Timetabling and Rotas**

Optimisation algorithms could be used to find ways of organising staff rotas or simple timetabling problems (e.g. parents evening appointments – with a solution that minimises waiting time between appointments). These often use hill-climbing type algorithms where a random solution is continually refined until a best solution is found based on a scoring system.

**Puzzle Solving**

Solving Sudoku puzzles, Rubik cubes and maze escaping are other applications of optimisation algorithms. These types of problems also give an opportunity to develop sophisticated user interfaces too.

**Preparation and Research**

<https://www.youtube.com/watch?v=oSdPmxRCWws>

A video explain the principle of hill-climbing algorithms

<https://www.youtube.com/watch?v=0nVYi3o161A>

Dijkstra shortest path algorithm explained.

Features you will have to use to maximise the complexity.

* Recursion
* Objects
* Data structures – trees and graphs

**Some other ideas..**

**Simulations**

We can write programs to model various situations. Anything from the basic ‘Conway’s Game of Life’ to more sophisticated modules of predator-prey relationships, economic models or Physics Simulations e.g. Planet orbits (some sort of virtual orrery), projectiles etc.

**Booking Systems**

Booking systems allow us to develop an underlying database as well as developing quite sophisticated interfaces – if you look on line at booking systems for, say, football matches you know the sort of thing I mean. There are also various reporting opportunities which would allow you to run sophisticated queries.

For example, a booking system which would allow you to book a computer in the drop in centre using a visual representation of computers in the room whilst allowing the administrator to design/ alter the layout. It could then monitor usage patterns and even send reminder emails to students to tell them they have a booking soon.

**Control Systems**

You don’t have to limit yourself to a .NET project on a PC. You can use devices such as an Arduino or raspberry pi along with sensors to develop a control or monitoring systems. We have to be careful that such systems whilst having the hardware setup recognised have a sophisticated program to monitor and analyse data. It’s also incumbent on you to buy the kit – sensors, breadboards etc. – we’re skint!

**Turn Based Games**

You can write games that don’t have to be a real-time graphics fest but rather focus on the machine AI. Good examples can range from connect4 to games such as a risk or GO. The AI can be as sophisticated as you want to make it but some sort of game strategy would need to be programmed in - again based on some sort of minimax tree. So opportunities for tree data structures and recursion are present as well as the opportunity for developing more sophisticated interfaces.

**Finally..**

**None of the above.**

At the end of the day it is **your** project and you don’t have to follow any of the suggestions above. You are free to choose any sensible project. You will need to discuss your proposal with your tutor to ensure…

1. There’s an end-user
2. It’s practicable
3. It’s complicated enough
4. It’s not too complicated!