

**A-Level Computer Science Guide**

**How Computer Science will be taught:**

You will have 5 lessons a week split between two teachers.

Lessons will consist of theory and practical elements. Lessons will have some time available for exercise completion from the course textbook; however, the majority of this will be done outside of lessons as homework. There will also be lesson time available to do the practical programming.

At the end of each chapter an assessment will happen in class, you will be expected to revise for this and there will be time to go through any issues after if required.

**Working expectations:**

In class you will be expected to complete the course notes. You will also be expected to do practical programming and learn how to make advanced programs independently outside of class.

**What 100% effort in this subject looks like:**

Completing all work and notes and submitting on time.

Continually revising the content gone through and where possible to do some advanced reading of the other topics.

Independent developing your programming skills in Python.

**Folder Policy:**

You will be provided with a pocket folder to keep your course notes and examples in. This should be with you every lesson as you may need to refer to previous topic notes as you progress through the course.

You may then keep your completed exercises in a separate folder that you provide yourself. This should be clearly labelled into the sections; it does not need to be with you every lesson.

**What marking looks like:**

The class teacher will look through your work but there will also be some marking that will need to be done by you prior to handing in.

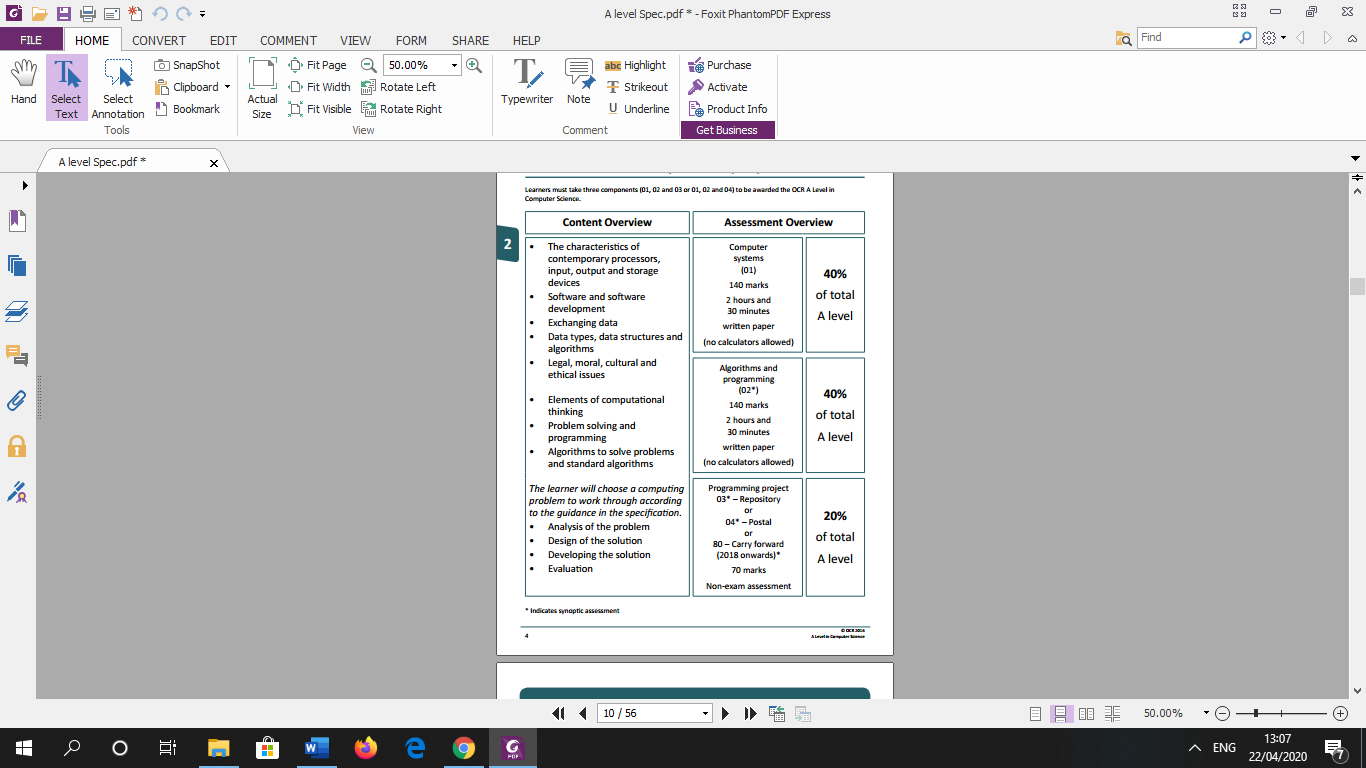
Assessments, either chapter assessments or unit assessments, will be marked by the teacher and allocated a grade. You may also do some “quick recap” assessments through the use of Socrative in class.

**What homework looks like:**

You will be set exercises of work to complete which consolidate the chapters taught in class. These will range in length; however, they are designed to emphasis understanding and interpretation and to check your knowledge. These may be questions from the book or what we have covered in class.

**Specification at a glance:**

Learners must take three components (01, 02 and 03 or 01, 02 and 04) to be awarded the OCR A Level in  
Computer Science.



**Summer preparation**

The purpose of giving you a summer bridging task is:

1. To provide a bridge from level 2 to level 3 study, and lead into the early stages of the course.
2. To engage you in independent learning which is required at level 3.
3. To encourage you to develop your work ethic and commitment to study.
4. To measure your suitability for the course and assess your initial levels of achievement.

**Tasks:**

**Task 1 - Vocabulary**

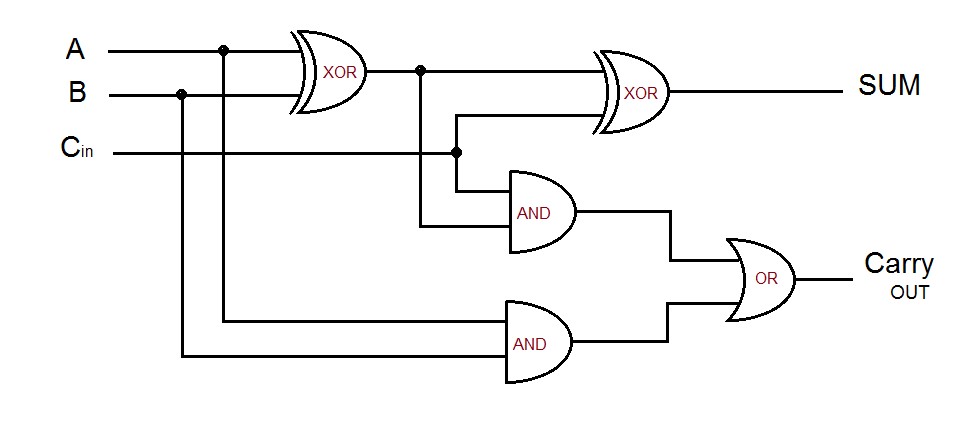
There are a number of key terms that crop up throughout the two-year course. It is important that you are aware what the meaning is of these terms. You will have learned some of this last year but in addition to those you need to research and find out about some the terms below

Define the following key terms:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Pipelining* | Harvard Architecture | CISC | RISC | *GPU* | Multicore System | Parallel Processor System |
| Paging | Segmentation | Interrupt | Distributed OS | Embedded OS | Intermediate Code | *Lexical Analysis* |
| *Syntax Analysis* | Waterfall Model | Agile Methodologies | *Programming Paradigm* | *Procedural Language* | Relational Database | *Normalisation* |
| Two’s Complement | *Floating Point Arithmetic* | *Bitwise Manipulation* | *Binary Search Tree* | *Backtracking* | *Data Mining* | *Big O Notation* |
| *Dijkstra’s Shortest Path* | *A\* Algorithm* | *De Morgan’s Law* | *Karnaugh Maps* | *Linkedlist* | Sign and Magnitude | Floating Point |

**Task 2 – Truth tables**

Given the circuit below, create and complete the truth table:



**Task 3 – Decomposition**

NASA are developing a new Mars Rover robot. The development of such machines costs millions of dollars and they simply cannot afford to lose or destroy the rover once it is launched. To reduce development costs and train astronauts and scientists in its use, they decide to develop a computer based simulation of the rover. The system will enable users to get used to the controls, how it responds to input and the behaviour of the craft in a “real world” situation.

1. Abstract the problem – make two lists titled “Essential components” and “unnecessary components.” Under each, make a decision about what things you would consider *must* be developed in this simulation and what would be irrelevant – for example the number of stars visible in the sky would not be relevant to this simulation and therefore we would not add this to our simulation.
2. Decompose the problem – make a list of what you would consider are the main tasks that would need to be carried out in order to create this simulation. For example, one task may be “implement appropriate gravity”

**Task 4 – Programming**

Using any language of your choice, write a simple program which attempts to predict the outcome of a Formula One race. This can be as simplistic or complicated as you like, but as a guide it should at least consider some prior performance, qualifying positions and perhaps time differences. There should be some “reasonable” maths used to form the prediction. In other words it is not sufficient to say “first place in qualifying will win!”

**Please bring your work with you to your first lesson.**

**Potentially useful websites:**

* The specification for the course is genuinely important reading – especially section 2 which tells you exactly what you will learn and Appendix 5 which gives you examples of code we will be using and you *need* to know.

<https://www.ocr.org.uk/qualifications/as-and-a-level/computer-science-h046-h446-from-2015/>

* We have a revision website that is in continuous development and currently contains all of our lesson resources. Click on A level CS in the top menu and then go through the different sections, there is some good information for the course

[www.learnitwithdavo.co.uk](http://www.learnitwithdavo.co.uk)

* <https://student.craigndave.org/a-level-videos> - These are the the summary videos that we watched for Craig and Dave for each chapter. These will give some flipped learning for the units.

**Further reading:**

* The Design of Every Day Things by Don Norman is a seminal book in computing and design – he worked for many years as a design lead at Apple and is responsible for some of the most intuitive designs/systems we use today. His book can be found online at: [http://www.nixdell.com/classes/HCI-and-Design-Spring-2017/The-Design-of-EverydayThings-Revised-and-Expanded-Edition.pdf](http://www.nixdell.com/classes/HCI-and-Design-Spring-2017/The-Design-of-Everyday-Things-Revised-and-Expanded-Edition.pdf)