Clitheroe Royal Grammar School Sixth Form: Transition Pack

Welcome to the Computer Science Department

We look forward to meeting you and welcoming you to the Sixth Form.

This Transition Pack contains information to support your transition from GCSE to A Level study.

Please read all the pack ready to begin Year 12:

1. What Independent Learning looks like in Computer Science
2. Frequently Asked Questions (FAQs)
3. Computer Science Introductory Task

✓ **Read the Subject Information Sheet** which is available here:
CRGS Sixth Form Subject Sheet - Computer Science

✓ **Download the exam board specification** which is available here:
AQA A Level Computer Science

✓ **Read the section called 'Specification at a Glance'**, focusing on the A Level content.

✓ Some of these resources will become more useful when you have moved further through the course, such as the A Level specification, so store them where you can revisit them over the next 2 years.

✓ Don't worry if some of the work sounds challenging. A Level work is more difficult than GCSE work after all. Your teachers will be supporting you through this transition. Please talk to us if you are unsure about any aspect of the course.

We look forward to seeing you soon.

Mr Latty - Head of Learning, Computer Science

A Level Computer Science FAQs

How much of the course is theory? How much is practical, writing programs?
Over the whole course, approximately half and half, but there is a lot of practical work to do when learning how to write programs/software. You can expect to spend almost all of the first term of Year 12 doing practical work.

What is the weighting of the above?
Approximately 50% each. In Year 13 the practical coursework makes up 20% of the total A Level. You will also sit a practical on-screen examination in which your programming skills will be assessed.

Is there a lot of maths involved? Do I need to take Maths A Level as well?
Taking Maths at A Level will certainly help with Computer Science, but it is not compulsory. The Computer Science syllabus contains some maths when covering number bases (e.g. binary, hexadecimal) and also in Boolean algebra.

What's the difference between ICT and Computer Science?
Put in simple terms, ICT is concerned with learning to use software written by others (e.g. databases, spreadsheets, etc) whereas Computer Science is concerned with problem solving and writing algorithms to solve these problems which can then be converted into programs in a programming language.

What other subjects fit well with Computer Science?
Maths and physics in particular.

Do I need any prior knowledge? Will I have needed to program before?
No, you just need a good grade in GCSE maths. We don't assume any previous experience of programming and will start from scratch.

Can I practise programming at home?
Yes, we will be learning to program with C# using Microsoft Visual Studio. This is available as a free download from https://visualstudio.microsoft.com/vs/community/. An excellent resource for learning C# is 'The Yellow Book', which can be downloaded for free from www.robmiles.com/c-yellow-book.

What will I expect to be able to program by the end of Year 12? Year 13?
By the end of Year 12 you will have learned all the basic building blocks of coding structured software such as assignment, iteration, selection, subroutines, file handling, etc. By the end of Year 13 you will be a very competent programmer and have experience of developing object-oriented programs and graphical user interfaces. In year 13 you will also cover functional programming and SQL, whilst many of our students also learn another language such as Java or Python.

Do we write games software?
Yes, and depending on your choice of coursework in Year 13, you may write quite a lot of this!

Do I get much homework?
In line with all other A Level subjects at CRGS, we would expect you to do at least 4 hours a week.

Clitheroe Royal Grammar School Sixth Form: Transition Pack

Is there any coursework?
There is a practical examination set by the exam board each year which involves modifying and adding extra functionality to the skeleton program code provided by the exam board. These are usually very interesting problems. In year 13, the non-exam assessment is worth 20% of the A Level, and involves students writing a substantial piece of software.

What jobs would I be able to apply for with Computer Science A Level? What courses could I take at university which link with Computer Science?
You may end up becoming a programmer, systems analyst, IT technician, etc. Courses at university would include Computer Science (some universities require A Level Maths for this degree), Games Programming, Systems Design, etc. Even if you don't pursue this subject any further after your A Levels, the problem-solving skills you will acquire will be extremely useful in other subject areas.

Mr M Latty, Head of Learning – Computer Science

Clitheroe Royal Grammar School Sixth Form: Transition Pack

A-Level Computer Science
Introductory Task

A large part of A-Level Computer Science is learning to program. In Year 12 you will be taught to program using C#. You will also learn a great deal of theory about exactly how computers work.

The following information should give you an indication of the type of topics that you will encounter in Year 12. Please read it carefully and check your understanding of the worked examples.

Data Representation
You can think of computers as just a very clever set of on and off switches that use the flow of electrons to control and represent information. Computers do not understand anything other than machine code, so data and instructions need to be converted and stored in binary. Binary is a number system that uses just two symbols, 0 and 1.

The Binary Number System
We (usually) count using the denary (or decimal system), also known as base 10. In this system, there are 10 digits, 0 to 9. When we write out a number we know that the columns represent, from right to left, units, tens, hundred, thousands, etc. So the number 1010 is:

Thousands ($1x10^3$)	Hundreds ($1x10^2$)	Tens ($1x10^1$)	Units ($1x10^0$)
1	0	1	0

One thousand
Zero hundreds
One ten
Zero units
..or the number **one thousand and ten**

The binary number system (base 2) uses only two digits: 0 and 1. This means that the place value is different – instead of going up by 10 to the power of *n*, we go up by 2 to the power of *n*. Therefore our number 1010 in binary is:

Eights ($1x2^3$)	Fours ($1x2^2$)	Twos ($1x2^1$)	Ones ($1x2^0$)
1	0	1	0

One eight
No fours
One two
No ones
..or the number **ten** in denary

Converting Binary to Denary
Converting binary numbers to denary is easy! Simply place the binary number under the correct place values. For an 8-bit binary number, these are the place values:

128	64	32	16	8	4	2	1

We will convert the number 10111001 to denary:

128	64	32	16	8	4	2	1
1	**0**	**1**	**1**	**1**	**0**	**0**	**1**

To convert the number we use simple addition: 128+32+16+8+1 = **185**
Converting Denary to Binary

Clitheroe Royal Grammar School Sixth Form: Transition Pack

There are several techniques that can be used to convert numbers from denary to binary. One common method is to write out a table with the binary place values. Anything below 255 will be an 8-bit number:

128	64	32	16	8	4	2	1

In this example we will convert the number 215 to binary. Starting from the left hand column, we can put a one in the 128 and take this away from 215:

128	64	32	16	8	4	2	1
1							

215-128 = 87...87 remaining, so put a one in the 64 column and subtract 64 from 87:

128	64	32	16	8	4	2	1
1	1						

87-64 = 23...23 remaining, so we cannot put a one in the 32 column; we put a zero instead:

128	64	32	16	8	4	2	1
1	1	0					

Still 23 remaining, so we can put a one in the 16 column, then subtract 16 from 23:

128	64	32	16	8	4	2	1
1	1	0	1				

23-16=7...7 remaining, put a zero in the 8 column:

128	64	32	16	8	4	2	1
1	1	0	1	0			

Still 7 remaining, put a one in the four column and subtract four from 7:

128	64	32	16	8	4	2	1
1	1	0	1	0	1		

7-4=3...3 remaining, put a one in the twos column and subtract 2 from 3:

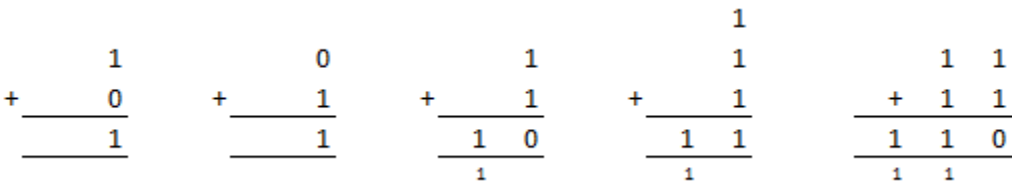
128	64	32	16	8	4	2	1
1	1	0	1	0	1	1	

3-2=1...1 remaining, so put a one in the ones column:

128	64	32	16	8	4	2	1
1	1	0	1	0	1	1	1

Clitheroe Royal Grammar School Sixth Form: Transition Pack

Binary Addition
To add binary numbers, work from right to left (as usual) and carry into next column as necessary.
Rules for adding binary numbers



Example

	0	1	0	1	0	1	0	1
+	1	0	0	0	1	1	1	0
	1	1	1	0	0	0	1	1
			1	1	1			

Sometimes, we find that we need a ninth column. If our computer only had 8 bits to store numbers this would mean the carry from the eighth column would be lost. This is called overflow; the calculation has overflowed the available space. In 8 bits the largest number we can store is 255.

Negative Numbers
All of the examples that we have looked at so far are **unsigned** binary **integers** (i.e. positive whole numbers). What if we need to store a negative number? For this we can use **two's complement binary**. To convert from denary to two's complement binary:

- Convert the absolute (i.e. positive) value to binary
- Flip all of the bits – ones become zeroes, zeroes become ones
- Add one

For example, write -99 in binary:
Convert the absolute value (i.e. positive 99) to binary:

-128	64	32	16	8	4	2	1
0	1	1	0	0	0	1	1

Note that the leftmost column is **minus** 128 when using two's complement. This means that the highest positive number we can store in 8 bits is 127
Flip the bits

-128	64	32	16	8	4	2	1
1	0	0	1	1	1	0	0

Add one

-128	64	32	16	8	4	2	1
1	0	0	1	1	1	0	0
							1
1	0	0	1	1	1	0	1

+

When stored in two's complement binary format, a negative number always starts with a one and a positive number will start with zero.

To convert from two's complement binary to denary, add up each place value in turn. Don't forget that the leftmost digit is negative. To convert the number above back to denary:
-128 + 16 + 8 + 4 + 1 = -99

Clitheroe Royal Grammar School Sixth Form: Transition Pack

Past Paper Questions

Q1.

(a) Represent the denary number 123 in binary using 8 bits.

Answer .....

(1)

(b) How many different denary numbers can be represented using 8-bit binary?

Answer .....

(1)

(Total 2 marks)

Q2.

The table below shows the contents of three memory locations.

Address	Memory contents
56	0011 0111
57	1000 1001
58	1100 0000

If the binary codes each represent a pure binary integer, what are the denary numbers stored at locations 56 and 57?

Address	Memory contents	Denary
56	0011 0111	
57	1000 1001	

(Total 2 marks)

Represent the denary value -18 as an **8-bit two's complement binary integer**.

Use the space below for rough working.

(2)

(d) What is the **largest positive denary value** that can be represented using **8-bit two's complement binary**?

Use the space below for rough working.

(1)