Computer Science Bridging Menu

You are about to start an exciting journey into the world of Computer Science, good luck!

Remember

- Complete the booklet attached.
-  The red hot chili indicates that the task is more challenging than the others (and must be done by Further Maths students).
- On the next few slides, we have some recommended reading, watching, listening and visiting for learners of mathematics.

Computer Science Bridging Portfolio

Read 	Watch 	List 	Visit (virtually or [illegible]lly at a later date) 	Do 
A free learning platform for A level computer science by browsing different list of topics. **https://isaaccomputerscience.org/**	Craig & Dave – full A level: OCR Specification utube.com/watch?v=dVi2B7fGVm4&list=PLCiOXwirraUBj7HtVHfNZsnwjyZQj97da	BBC – Computing Britain http://www.bbc.co.uk/programmes/b06bq6j1/episodes/downloads	The National Museum of Computing - http://www.tnmoc.org	MOOCs Learn object-oriented programming principles by creating your own text-based adventure game in Python. Supported by Google. https://www.futurelearn.com/courses/object-oriented-principles
MIT News - http://news.mit.edu/topic/computers	. BBC Click - http://www.bbc.co.uk/programmes/n13xtmd5	Wired - http://www.wired.co.uk/series/wired-podcast	Bletchley Park - https://bletchleypark.org.uk/	Learn to code — Build projects. https://www.freecodecamp.org/
			The UK Computer Museum, Cambridge http://www.computinghistory.org.uk/	

A-level Computer Science Transition workbook

The A-level course is assessed by two exams (40% each exam) and a programming project (20%)

- The topic of **Computer Science** is at the heart of the modern world
- Studying it can make you extremely sought after in today's job market
- The transition from GCSE to A level is significant, this includes:
    - An increased emphasis on **technical content**
    - An increased emphasis on developing **practical programming skills**
    - An increased emphasis **independent research**

This workbook is designed to allow you to build on your existing knowledge, develop your skills and get off to a flying start on the A-level course.

Please complete by your first lesson back in September.

1 Independent research task

Emerging computer technology

In this task you get to investigate an area of emerging computer technology that interests you.

You can research any area of emerging computer technology. For example:

- Artificial intelligence
- Robotics
- Autonomous self-driving cards
- Quantum computing

In no more than ONE side of A4, summarise your findings under the following four headings:

1. What is it?
2. What are the possible Social, Moral, Cultural and Ethical **benefits** of this technology on society
3. What are the possible Social, Moral, Cultural and Ethical **risks** of this technology on society
4. My conclusion on this technology and what it will mean for our world 10 years from now

Additional help:
For additional help and support in structuring your answer you might like to watch some of the videos from the following Craig 'n' Dave playlist:

OCR:
SLR 17 – Ethical, morale and cultural issues
https://student.craigndave.org/videos/slr-17-ethical-moral-and-cultural-issues

2 Systems architecture task

Looking under the hood of the processor

The CPU “Central Processing Unit” is the central core of any computer system. You will study in depth what it contains and how it works.

1. Start by watching the following 3 videos from Craig ‘n’ Dave:
    1. **OCR:** https://student.craigndave.org/videos/ocr-alevel-slr01-alu-cu-registers-and-buses
    2. **OCR:** https://student.craigndave.org/videos/ocr-alevel-slr01-fetch-decode-execute-cycle
    3. **OCR:** https://student.craigndave.org/videos/ocr-alevel-slr01-performance-of-the-cpu

2. Draw a diagram of the CPU, showing the following components and describing what each is for:
- Arithmetic Logic Unit (ALU)
- Control Unit (CU)
- Cache

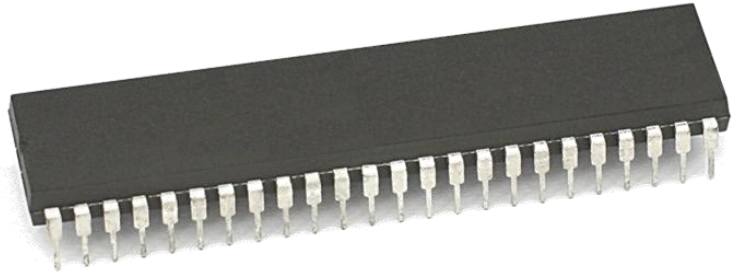
3. Describe the steps of the fetch-decode-execute cycle, with reference to the following registers:
- Program Counter (PC)
- Memory Address Register (MAR)
- Current Instruction Register (CIR)
- Memory Data Register (MDR)

4. Describe how CPU performance can be improved by:
- Increasing the clock speed
- Increasing the cache size
- Increasing the number of cores

3 Memory task

Different types of memory

Computer memory comes in different forms. Carry out some research and answer the following questions:



Random Access Memory (RAM)
- What is RAM?
- Is RAM volatile or non-volatile?
- Is RAM read-only or read-write?
- What sort of information does RAM typically hold?

Read Only Memory (ROM)
- What is ROM?
- Is ROM volatile or non-volatile?
- Is ROM read-only or read-write?
- What sort of information does ROM typically hold?

Virtual Memory
- What is Virtual Memory?
- Why is Virtual Memory needed?

Additional help: For additional help and support in completing this task you might like to watch the following videos from Craig 'n' Dave:

RAM and ROM: https://student.craigndave.org/videos/ocr-gcse-slr1-2-ram-and-rom

The need for Virtual Memory: https://student.craigndave.org/videos/ocr-gcse-slr1-2-the-need-for-virtual-memory

4 Storage task

Types of secondary storage

Virtually all secondary storage devices in use today fit into one of three broad categories:

- Magnetic
- Optical
- Solid state

For each of these three categories of secondary storage, answer the following questions:

- How does it work?
- What are the advantages?
- What are the disadvantages?

Additional help:
For additional help and support in completing this task you might like to watch the following videos from Craig 'n' Dave:

Magnetic, Flash and Optical storage:
https://student.craigndave.org/videos/ocr-alevel-slr03-magnetic-flash-and-optical-storage

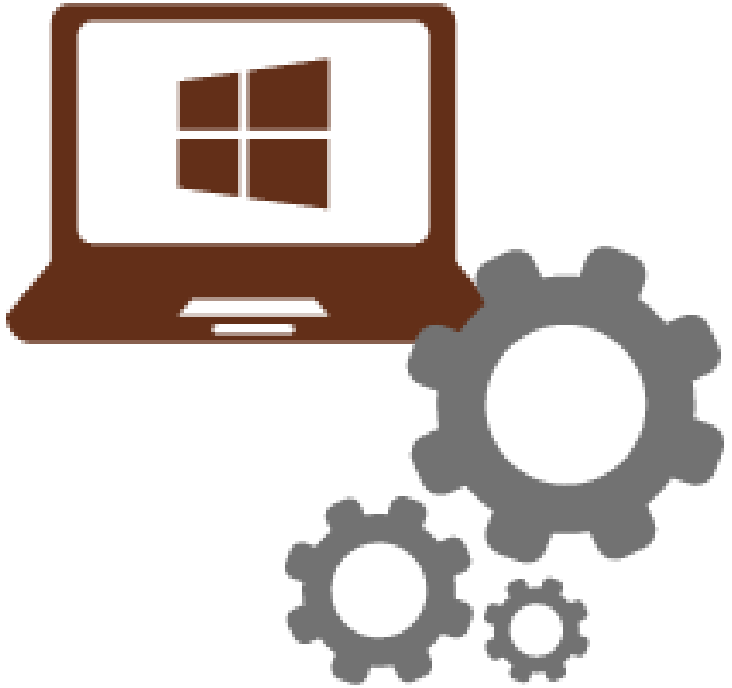
Comparing capacity and speed of storage media:
https://student.craigndave.org/videos/aqa-alevel-slr18-comparing-capacity-and-speed-of-storage-media

5 Systems software task

Operating systems

The operating system is arguably the most important piece of software installed on a computer.

- Describe at least eight different roles of an Operating System
- Describe what an “interrupt” is and how it affects the fetch-decode-execute cycle



Additional help:
For additional help and support in structuring your answer you might like to watch some of the videos from the following Craig ‘n’ Dave playlists:

OCR: SLR 4 – Operating systems – Systems software
https://student.craigndave.org/videos/slr-4-operating-systems-systems-software

6 Binary task

Representing negative numbers in binary

In GCSE computer science you will have learnt how to represent positive whole numbers in binary e.g. 47. At A Level you will need to know how to represent negative as well e.g. -47

- Start to recapping (or learning if you didn't do the GCSE) how to represent positive whole numbers between 0-255 in binary
- Now research how to represent negative numbers in binary using the method known as Two's Complement

1. Write out the positive number 107 using 8-bit binary:

128	64	32	16	8	4	2	1

2. Write out the negative number -107 using 8-bit binary:

3. How would you represent the lowest negative number possible using Two's Complement 8-bit binary?

128	64	32	16	8	4	2	1

4. How would you represent the largest positive number possible using Two's Complement 8-bit binary?

Additional help:
For additional help and support in structuring your answer you might like to watch some of the following videos from Craig 'n' Dave:

- GCSE recap: How to represent positive binary values 0-255 https://student.craigndave.org/videos/aqa-gcse-slr13-number-bases
- A Level: Representing negative binary values using Two's Complement https://student.craigndave.org/videos/aqa-alevel-slr11-twos-complement

7 Algorithms task

Algorithms: from theory to practice

Probably the most basic algorithm is that of the “linear search”. If you have done the GCSE course you will have learnt about this searching algorithm already.

1. Start by learning or refreshing your knowledge of the linear search algorithm by using the videos on this page: https://www.craigndave.org/algorithms-linear-search

2. Answer the following questions:
- What does the linear search algorithm do?
- What are some applications of it?
- Write out the steps of the linear search algorithm in simple, structured English.

3. Write out **pseudocode** for the linear search algorithm.

- The algorithm should use an array, `items`, which is pre-populated with the following values: "Florida","Georgia","Delaware","Alabama","California“
- The algorithm should ask the user to “Enter the state to find:”
- If the algorithm locates the state entered by the user in the array it should report back to the screen “Item found at position n”
- If the algorithms can not locate the state entered by the user in the array it should report back to the screen “Item not found”

8 Getting started with Python

Programming basics

Learning to “code” is a fun and essential part of A Level Computer Science.

Complete this task if you haven't done the GCSE in Computer Science or you simply want a nice refresher ahead of starting your A Level course.

1. Head over to the web site: https://www.learnpython.org/
2. Complete the following python tutorials under the heading:
    - Hello, World!
    - Variables and Types
    - Lists
    - Basic Operators
    - String Formatting
    - Basic String Operations
    - Conditions
    - Loops
    - Functions
3. Each section presents you with theory, code to run and exercises to try out.

Additional note: This task is most suited to students who intend to do the A Level and have not previously gained much / or any programming experience from the GCSE Computer Science course. The list of topics above cover the standard set of programming concepts you would be expected to know having completed a GCSE in Computer Science and so will prepare you well for the A level.

Python programming tasks

The rest of the tasks in this Transition Workbook aim to improve your algorithmic thinking and your Python programming skills.

Download Python from here:
https://www.python.org/downloads/
and install it on your PC / laptop.

If, for any reason, you're unable to do this, you can instead use this site to develop and run your Python programs:
https://replit.com/

Save your programs and copy your source code to a document that you can submit when you return to school in September.

Remember to include comments in your programs, including a comment at the start of each program to describe what the program does.

9 Python task: Password reset program

Write a program to allow the user to input a new password. Only accept a new password if it is:

1. At least eight characters long
2. Has lower case and upper case letters.

The password reset program should also make the user input their new password twice so that the computer knows that the user has not made any mistakes when typing their new password.

Extensions:

1. Make some sort of algorithm to suggest how strong the password is (Weak, Medium, Strong) depending on length, whether or not the password has special characters in etc
2. Let the user input their username. The program should go to a text file with a list of usernames and old passwords, and the program should only let you change your password if you input your old password.

10 Python task: Check if palindrome

Write a program that will check if the string entered by the user is a palindrome. A palindrome is a word that reads the same forwards as it does backwards like “racecar”.

11 Python task: Change return program

The user enters a cost and then the amount of money given. You should write a program that works out what denominations of change should be given in pounds, 50p, 20p, 10p etc.

Extensions:

1. The program will figure out the change for the American currency and the number of quarters, dimes, nickels, pennies needed for the change
2. Make an automatic testing part of your program where it automatically generates a random price and an amount that you give the cashier. It then works out what change to give, and then tests that your program works by adding the change back onto the price of the item to prove your program works. It should flag an error if there are problems.

12 Python task: Binary / hexadecimal / decimal

Create a program which will present a menu offering the user the following options:

1. Convert an input decimal number (up to 255) to its 8-bit binary equivalent
2. Convert an input binary number (8-bit) to its decimal equivalent
3. Convert an input decimal number (up to 255) to its hexadecimal equivalent
4. Convert an input hexadecimal number (2 digits) to its decimal equivalent
5. Convert an input binary number (8-bit) to its hexadecimal equivalent
6. Convert an input hexadecimal number (2 digits) to its binary equivalent

13 Python task: Fruit machine

Write a program to simulate a Fruit Machine that displays three symbols at random from Cherry, Bell, Lemon, Orange, Star, Skull.

The player starts with £1 credit, with each go costing 20p. If the Fruit Machine “rolls” two of the same symbol, the user wins 50p. The player wins £1 for three of the same and £5 for 3 Bells. The player loses £1 if two skulls are rolled and all of his/her money if three skulls are rolled. The player can choose to quit with the winnings after each roll or keep playing until there is no money left.

14 Python task: Arithmetic test

A primary school teacher wants a computer program to test the basic arithmetic skills of her students. Generate random questions (2 numbers only) consisting of addition, subtraction, multiplication and division.

The system should ask the student's name and then ask ten questions. The program should feed back if the answers are correct or not, and then generate a final score at the end.

Extensions:

1. Extend your program so that it stores the results somewhere. The teacher has three classes, so you need to enable the program to distinguish between them.
2. The teacher wants to be able to log student performance in these tests. The teacher would like the program to store the last three scores for each student and to be able to output the results in alphabetical order with the student's highest score first out of the three.

15 Python task: Fibonacci sequence

The Fibonacci sequence is defined by the recurrence relation:

$F_n = F_{n-1} + F_{n-2}$, where $F_1 = 1$ and $F_2 = 1$.

Hence the first 12 terms will be:

- $F_1 = 1$
- $F_2 = 1$
- $F_3 = 2$
- $F_4 = 3$
- $F_5 = 5$
- $F_6 = 8$
- $F_7 = 13$
- $F_8 = 21$
- $F_9 = 34$
- $F_{10} = 55$
- $F_{11} = 89$
- $F_{12} = 144$

The 12th term, F_{12}, is the first term to contain three digits.

What is the index of the first term in the Fibonacci sequence to contain 1000 digits?

16 Python task: Fizz Buzz

Create a program that replicates the famous game Fizz Buzz. The program will take an input, e.g. 20, and then print out the list of Fizz Buzz up to and including that number, where:

- Any multiple of 3 is replaced by the word ‘Fizz’
- Any multiple of 5 is replaced by the word ‘Buzz’
- Any multiple of both 3 and 5 is replaced by the word ‘FizzBuzz’

Extension:

1. Replace any prime number with the word ‘OOPS!’
2. Allow the user to enter the base numbers that they want to replace words with. E.g. 2 and 3, which would mean:
    - Any multiple of 2 is replaced by the word ‘Fizz’
    - Any multiple of 3 is replaced by the word ‘Buzz’
    - Any multiple of both 2 and 3 is replaced by the word ‘FizzBuzz’

17 Python task: Happy numbers

A happy number is defined by the following process. Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers, while those that do not end in 1 are unhappy numbers. Have the programme find the first 8 happy numbers.

18 Python task: Happy numbers

A happy number is defined by the following process. Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers, while those that do not end in 1 are unhappy numbers. Have the programme find the first 8 happy numbers.