Student Name: contact: wilde.m1@welearn365.com

Southam College Computing & ICT Department

Summer Project

This booklet provides several tasks for you to update your current knowledge and learn new concepts required for A Level

Please complete the research and tasks. These will form the basis of a suitability test when you return in September.

Bring this booklet back in for checking on your first lesson in September

Some excellent sources Of information:
www.teach-ict.com --- YouTube: CraignDave --- www.orc.org.uk --- bbc bitesize
Isaac Computer Science - https://isaaccomputerscience.org/

Contents

Student Name: contact: wilde.m1@welearn365.com

Using this pack

The transition from working at a GCSE standard to an A-Level is significant, including an increasing emphasis on technical content, extended answers and independent research. This pack is designed to allow you to practice some of these skills, building on the work that you may have covered at GCSE. Whether you have studied GCSE Computer Science or not, and whatever your grade, there will be something here to support your preparation for A-Level.

This transition pack is organised into three sections:

- Computer Science Theory
- Algorithmic Thinking and Problem Solving
- Writing Code

This broadly matches the examination and non-examination assessments of the new GCSEs and A Levels. Within each section there will be practice questions to support both the content and style of writing required at A-Level, plus various links to books and other resources that you can use to study any topics that require attention. Each section is based on the GCSE specification, so that the content should be familiar if you have already studied GCSE Computer Science; if you are new to the subject, this should give you an overview of the main topic areas that you will study.

The questions are designed to go beyond GCSE standard and prepare you for A-Level study. Some questions are quite straightforward, and test core knowledge. Others are chosen to give you a chance to extend both your thinking and writing skills and to demonstrate your creativity in solving problems. There are also some genuinely hard extension questions if you want them!

Note that this is not a "self-study" document on its own. This resources contains questions, prompts, starting points and solutions to help you study one or more core topics before starting the A-Level.

Computer Science Theory

Recommended resources

To complete you can use Teach ICT, Isaac Computer Science, GCSEPOD and the A level Craig and Dave resources.

I highly recommend that you purchase the following book as it is really good for your A level and we will be using it in your A level

OCR AS and A Level Computer Science by P.M Heathcote

ISBN: 9781910523056

Computational Thinking – Theory

Computational Logic and Calculations

1. Complete the truth tables for the following expressions
    a. A AND (B OR C)

A	B	C	B OR C	A AND (B OR C)
0	0			
0	0			
0	1			
0	1			
1	0			
1	0			
1	1			
1	1			

    b. (NOT A) OR (NOT B)
        i. What single logic gate produces the same result as this expression?

A	B			

    c. Draw a circuit to represent each expression

2. Calculate each of the following, showing any appropriate working you need
    a. 13 MOD 2
    b. 16 MOD 6
    c. 15 MOD 3
    d. 7 MOD 8
    e. 13 DIV 2
    f. 16 DIV 6
    g. 15 DIV 3
    h. 7 DIV 8
    i. 2^0
    j. 2^7
    k. 2^8
    l. 2^10

3. Covert the following into the units given
    a. 4 bytes = bits
    b. 1 TB = bytes
    c. 80 kB = GB
    d. 40 MB = nibbles

4. Complete the table, converting between binary, hexadecimal and denary as required

Binary	Hex	Denary
0010 1010		
	0B	
		255
0110 0111		
	F5	
		48
	CD	

5. Complete the following calculations
    a. 0110 0011 + 0011 0001
    b. 1010 0110 + 1100 1111
    c. 0110 0011 << 2 (bit shift left two places)

6. Check if these are valid ASCII characters. If they are, give their character equivalent. Note that the first bit is a check digit using even parity, and the remaining 7 bits are the character
    a. 1110 0010
    b. 1100 0111
    c. 0011 0110
    d. 1100 1010

Programming Tools and Standards

1. Compare the use of jpg, png and gif to store images, explaining the benefits, properties and uses of each image format

2. Produce an annotated diagram of the IDE you prefer to use to write code, explaining any features of the IDE that help you to produce your code. You may need to show examples of the IDE in use to highlight the different features

Writing Code

Coding challenges

The coding challenges below will let you check your skills. Part of the transition to A-level is combining skills, and also ensuring that you plan and test your work thoroughly, so think about how you can re-use components and design your code for readability and robustness.

1. Write a program to:
    a. Ask the user to input
        i. The name of a product
        ii. Its cost in pounds
        iii. The program should keep asking for inputs until the user types "None"
    b. The program should then output:
        i. The name and price of the most expensive item
        ii. The name and price of the least expensive item
        iii. The average price of the items iv. The total cost of the items
            1. Items over £50 get a 5% discount
            2. VAT is added at the end at 20%
    c. The program should validate any inputs

Plan your algorithm first, using a flowchart or pseudocode

Code your algorithm, and provide evidence of both your code and the working output

Create a test plan for your algorithm, including testing your validation with normal, boundary and erroneous data.

Student Name: contact: wilde.m1@welearn365.com

Algorithmic Thinking and Problem Solving

The following puzzles will help you to develop you logical thinking skills. There are many good books of puzzles, plus countless online sources to test your skills. Some recommendations are given later.

The following puzzles are representative of classical problems and problem solving strategies. You can solve each one by trial and error, but you encouraged to think about the strategy you employed to solve the problem. Note that there are discussions of each problem available online if you want to investigate them in more detail.

Two good general strategies to try are:

- Can you solve a simpler version of the problem first?
- Can you draw a diagram to help you *visualise* the problem?

After that, you have your standard computer science strategies:

- Decomposition
  - Can you split the problem down into smaller parts to solve?

- Abstraction
  - Can you remove any unnecessary details to focus in on only what you need to solve the problem?
  - Be careful – are you sure that you have kept the right information?

- Generalisation and problem/pattern recognition
  - Is this puzzle a specific example of a problem for which there is a general solution? If so, how does it apply in this case?
    - Do you recognise the problem from somewhere else, or is it similar to something else?
  - You may need to generalise the problem to identify the core features so that you can spot equivalent problems.

Another important strategy is to ensure that the problem is *well-defined*. This means that you know:

- The goal: what you are trying to achieve
- The givens: what you know at the start, or your starting conditions
- The resources: what you have available to solve the problem
- The constraints: any rules that limit your solution
- The ownership: who or what is carrying out each part of the solution

Sometimes just working through the problem definition carefully is enough to give the required insight.

The complete work on problem solving is Polya's "How To Solve It"; there are many sources for this online if you are interested.

The problem

The Princess in the Castle

Originally heard on Puzzle Panel on BBC Radio 4. http://www.bbc.co.uk/programmes/b00xhd45

A princess lives in a long corridor in a castle. The corridor has 17 rooms, numbered 1 to 17 inclusive. Each night the princess sleeps in a different room according to the following rules:

- On the first night of the year she sleeps in a random room
- Each night she moves to an adjacent room; she never sleeps in the same room on two nights in a row and she always moves exactly one room left or right along the corridor
- For example, if she is currently sleeping in room 12, then on the next night she will either be in room 11 or in room 13
- If she is in room 1, then she must be in room 2 on the next night as she cannot move in any other direction (the same is true for room 17 – she must move to room 16 next)

A prince wishes to marry the princess. To do this he must find her room in the castle. However, whenever he sneaks into the castle at night, the guards quickly find him and throw him out! Therefore he only has time to search one room each night.

The princess is unable to give the prince any clues to her location, and the prince has no knowledge of her location, other than whether or not she was in the room he last tried.

What strategy should the prince follow in order to find the princess in a finite time?

What is the maximum number of nights the prince needs to search before he can guarantee finding the princess?