Accredited

# AS and A LEVEL

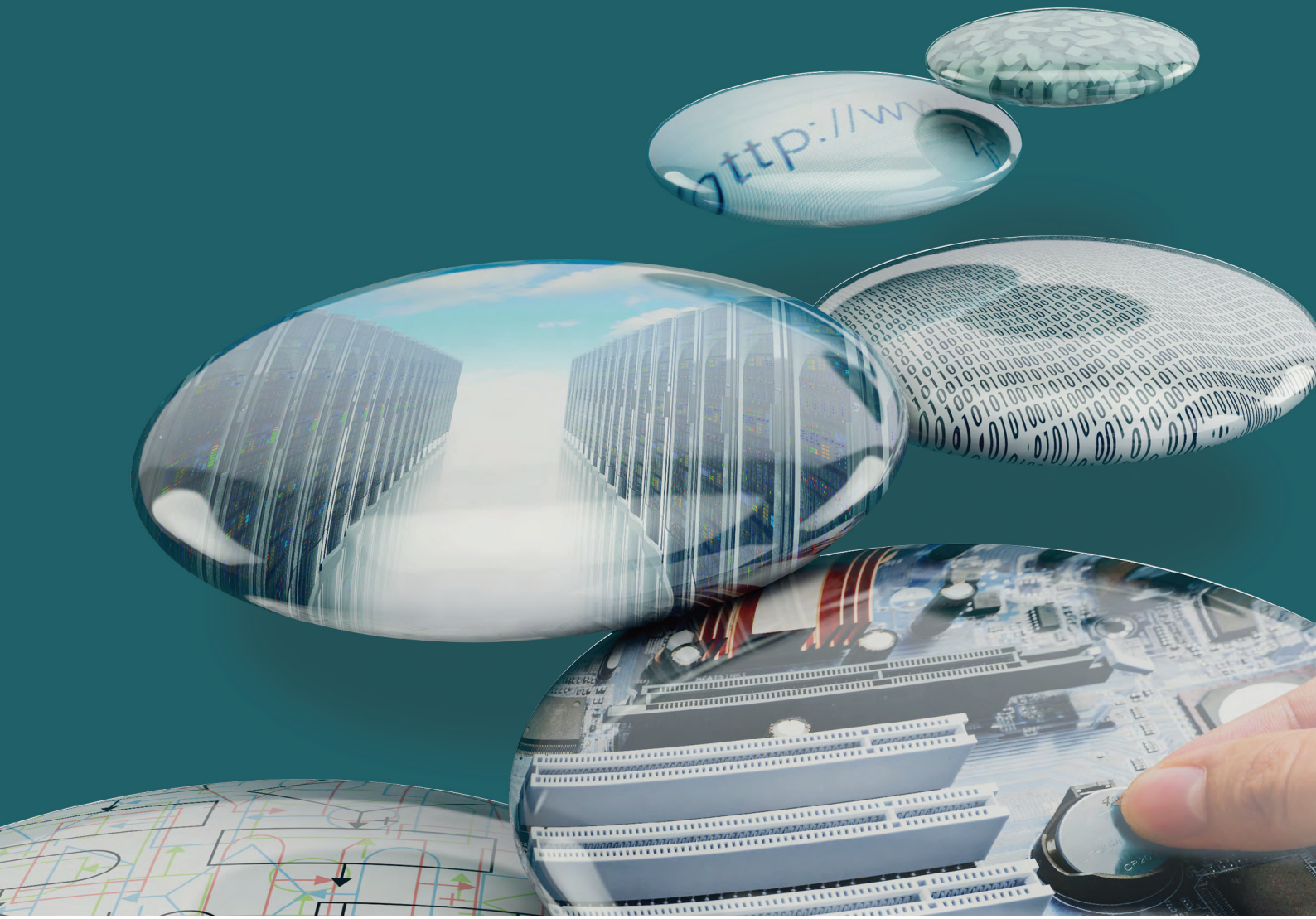
*Topic Exploration Pack*

H046/H446

# *COMPUTER SCIENCE*

Theme: Thinking ahead

September 2015



CAMBRIDGE UNIVERSITY PRESS
www.cambridge.org

OCR
Oxford Cambridge and RSA

We will inform centres about any changes to the specification. We will also publish changes on our website. The latest version of our specification will always be the one on our website (www.ocr.org.uk) and this may differ from printed versions.



Oxford Cambridge and RSA Examinations is a Company Limited by Guarantee.
Registered in England. Registered company number 3484466.

Registered office: 1 Hills Road
Cambridge
CB1 2EU

OCR is an exempt charity.

# Contents



This Topic Exploration Pack should accompany the OCR resource ‘Thinking Ahead’ learner activities, which you can download from the OCR website.


*This activity offers an opportunity for English skills development.*



# Thinking Ahead

For the topic ‘Thinking ahead’ students will need to be familiar with the following points:

a) Identify the inputs and outputs for a given situation.

b) Determine the preconditions for devising a solution to a problem.

c) The nature, benefits and drawbacks of caching.

d) The need for reusable program components.

Thinking ahead in terms of computational thinking is the process of thorough planning of your code to ensure an efficient outcome and where possible using caching/pre-fetching.

To start with we will consider the above points using making a meal as an analogy.

Inputs and outputs for a meal would obviously be the ingredients. We could go a bit further and say ‘how much’, ‘are they organic’, ‘does it suit vegans?’. These are all potential abstractions. We only abstract our problem as far as we need to go to solve the problem that we want to deal with. Outputs will be the meal, but then also may extend to how this will affect the persons mood, how tasty it is, what they would change if they would make it again.

For example, a pre-condition to eating a meal would be that you have all the ingredients and equipment available before starting to cook. Without any of these things, this could prolong the time it takes to make the meal, or could even ruin the meal completely. Preconditions can also be found in the recipe. 30 degrees for 1 hour in the recipe? Ok. Are the carrots still hard? Cook them for a bit longer. A post condition for the meal may be that you have a place to eat the meal.

Caching is the process where we build a ‘library’ of pre-formed elements for quick future use. In the cooking example, caching would be explained by if some of the spices you are using are in a cupboard. You know, those cupboards where all the spices are in front of the ones you actually want? An inefficient thing would be to get out a spice you want, use it and then put it back right away into the midst of all the other spices. If you want to use the spice later on? You will have to find it again amongst all the other spices. This could be termed as having no cache at all.



A much more efficient way of dealing with this problem is taking all the spices that you have from the cupboard and putting them on display in a line in front of you. This will dramatically reduce the time it takes to find the spice you want. Even better if you put the spices back in the cupboard that you know you won't be using. Even better if you put the spices in alphabetical order. This would be an efficient cache.

Having reusable program components is really useful as once we have coded something then we know it works. In addition, we can make it so that we can pass different arguments into these components to change how they behave. For example, when we are cooking something we may use a specific mould for a jelly, although the jelly we might use may be different flavours. Algorithms make decisions based upon some pre-conditions. These may be given as initial conditions or given as parameters.

**Prior knowledge**

To complete the first activity students will require some knowledge of Python but apart from that no additional knowledge is needed.

**Common student misconceptions**

Students might become confused that 'thinking ahead' is all about being able to predict what is going to come ahead in the future but actually it's more concerned with good planning. Planning ahead of time what data structures you are going to use, how you are going to break up your program into functions, that sort of thing.

You may even find that you go in later to 're-factor' (basically a term for making your code better, more extensible, faster, etc.) your code, although sometimes the code you start off with may be harder to fix if you do not lay down good practice to start with, especially if you work for a company and someone else has to come along and fix your 'bad code'.



## Activity 1 – Noughts and Crosses

Tic Tac Toe/Artificial Intelligence

Gets students thinking of conditions of win/loss conditions and inputs/outputs/reusable functions.
http://inventwithpython.com/chapter10.html

http://codeboom.wordpress.com/sweet-resources/noughts-and-crosses/
Other ideas relating to game theory:
Nim-like games - http://nrich.maths.org/1209

Chess/Checkers

## Activity 2 – Gantt Charting (See Worksheet)

Students can use the worksheet for Gantt charting to 'plan ahead' for a music concert.

A list of things that need to happen for the concert to be successful are given in the worksheet. Students should put these steps in order, consider who of your team should do that role and also consider if there are any other steps that need to be done.

After students have constructed the Gantt chart there are some further questions to help students consider the choices they have made and make for an interesting discussion point at the end of the lesson.

The activity is not supposed to be serious, and there are no right or wrong answers, but serves as a good introduction to thinking ahead to solve a problem.

This activity should take under an hour.



## Activity 3 – Array Activities

A useful activity on arrays can be found here:
http://moodlenz.net/pluginfile.php/71/mod_data/content/100/Array%20Activities.pdf

To complete the activity you will need a pack of playing cards.

After completing these activities and the accompanying class discussions, students should be able to discuss why a specific algorithm is appropriate to solve a particular problem, and discuss the trade-offs between different approaches.


We'd like to know your view on the resources we produce. By clicking on the 'Like' or 'Dislike' button you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click 'Send'. Thank you.

If you do not currently offer this OCR qualification but would like to do so, please complete the Expression of Interest Form which can be found here: www.ocr.org.uk/expression-of-interest

OCR customer contact centre
General qualifications
Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk