Qualification Accredited

OCR
Oxford Cambridge and RSA

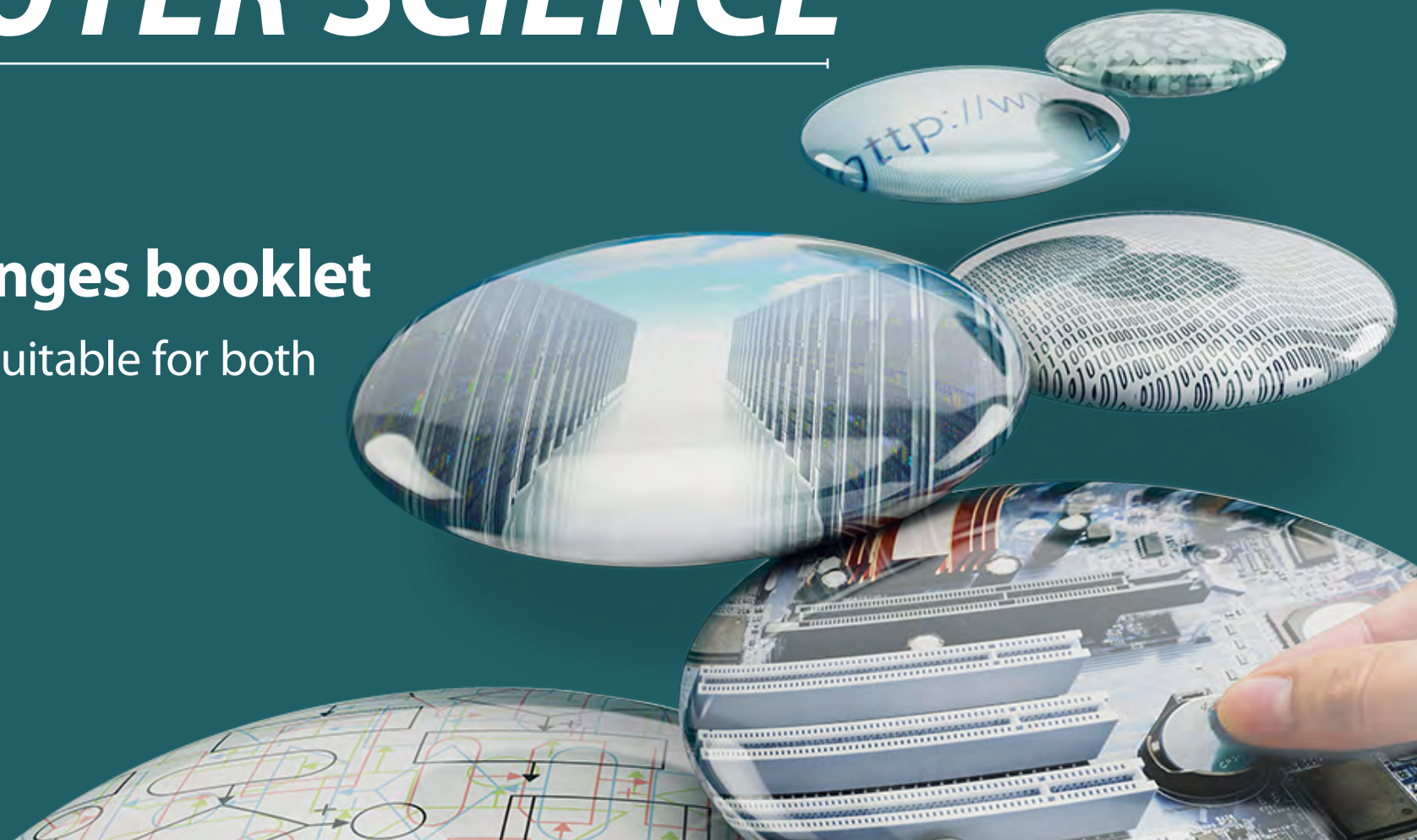
GCSE (9-1) and A LEVEL

COMPUTER SCIENCE

Coding challenges booklet

Coding challenges suitable for both GCSE and A Level

Version 1



www.ocr.org.uk/computerscience

Contents

CONTENTS

INTRODUCTION

These coding challenges provide real world problems for teachers and students to help develop their coding skills.

For GCSE: responses only need command line interfaces, and focus on using the programming techniques contained within the current specification.

For A Level: any solutions should have a graphical use interface created for it. Use of OOP methodologies is to be encouraged, as many problems lend themselves to a class system.

This is an active document and likely to receive regular updates with challenges throughout the lifetime of the specification.

We do not publish solutions, as there are many ways in which these problems could be solved. Discussions regarding approaches are beneficial at a cohort/class level to encourage candidate's realisation that each problem has many unique solutions that will fulfill the success criteria that have been identified. Where we do provide a solution - it should be used for discussion and comment, rather than being taken as 'the only and/or best way to solve the challenge'.

1	**Factorial Finder** The Factorial of a positive integer, n, is defined as the product of the sequence n, n-1, n-2, ...1 and the factorial of zero, 0, is defined as being 1. Solve this using both loops and recursion.
2	**Speed Tracker** Create a program that takes a time for a car going past a speed camera, the time going past the next one and the distance between them to calculate the average speed for the car in mph. The cameras are one mile apart. **Extensions:** 1. Speed cameras know the timings of each car going past, through number plate recognition. Valid number plates are two letters, two numbers and three letters afterwards, for example XX77 787. Produce a part of the program that checks whether a number plate matches the given pattern. Tell the user either way. 2. Create a program for creating a file of details for vehicles exceeding the speed limit set for a section of road. You will need to create a suitable file with test data, including randomised number plates and times. You will then use the code you've already written to process this list to determine who is breaking the speed limit (70mph) and who has invalid number plates.
3	**Thief!** A thief has managed to find out the four digits for an online PIN code, but doesn't know the correct sequence needed to hack into the account. Design and write a program that displays all the possible combinations for any four numerical digits entered by the user. The program should avoid displaying the same combination more than once. Submit a fully detailed Showcase for your program.
4	**Classification** A simple classification system asks a series of Yes/No questions in order to work out what type of animal is being looked at. Eg Does it have 4 legs? Does it eat meat? Does it have stripes? These systems can often be drawn using a "tree" structure. Carry out some simple research on classification trees, then write a program to help the user decide between the following: horse, cow, sheep, pig, dog, cat, lion, tiger, whale, dolphin, seal, penguin, ostrich, sparrow, spider, ant, bee, wasp, termite, octopus, squid Is there a better way to do this than using 101 IF...ELSE...END IFs? Develop your classification system for your own area of interest: pop bands; pokemon; cars; footballers; teachers; diseases etc.

5	**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.
6	**Unit Converter (temp, currency, volume)** Converts various units between one another. The user enters the type of unit being entered, the type of unit they want to convert to and then the value. The program will then make the conversion.
7	**Credit Card Validator** Takes in a credit card number from a common credit card vendor (Visa, MasterCard, American Express, Discoverer) and validates it to make sure that it is a valid number (look into how credit cards use a checksum).
8	**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.
9	**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. Display an example of your output here. Find the first eight happy numbers.

10	**Number Names** Show how to spell out a number in English. You can use a pre-existing implementation or make your own, but you should support inputs up to at least one million (or the maximum value of your language's default bounded integer type, if that's less). **Extensions:** 1. Create support for inputs other than positive integers (like zero, negative integers, and floating-point numbers).
11	**Regex Query Tool** This is a tool that allows the user to enter a text string and then in a separate text box enter a regex pattern. It will run the regular expression against the string and return any matches or flag errors in the regular expression.
12	**Quiz Maker** Make an application which takes various questions from a file, picked randomly, and puts together a quiz for students. Each quiz can be different and then reads a key to grade the quizzes.
13	**Caesar Cipher** Implement a Caesar cipher, both encoding and decoding. The key is an integer from 1 to 25. This cipher rotates the letters of the alphabet (A to Z). The encoding replaces each letter with the 1st to 25th next letter in the alphabet (wrapping Z to A). So key 2 encrypts "HI" to "JK", but key 20 encrypts "HI" to "BC".
14	**Events calendar** Create a menu driven program that allows the user to add or delete events from a list of dates and timings, just like a calendar. The program should warn you if any of the events overlap when entering them. **Extensions:** 1. Make it so that none of the events are hard-coded into the program
15	**Pangrams** **"The quick brown fox jumps over the lazy dog";** note how all 26 English-language letters are used in the sentence. Your goal is to implement a program that takes a series of strings (one per line) and prints either True (the given string is a pangram), or False if it is not.
16	**Kaprekar** Determine whether a number is a Kaprekar number or not. See http://mathworld.wolfram.com/KaprekarNumber.html for more information.

17	**Number Table** Write a program that takes a symbol (+,-,* or /) and a natural number (>0) and makes a table like below for the operation from 0 to n For this example the user has entered "+ 4": + | 0 1 2 3 4 ---------- 0 | 0 1 2 3 4 1 | 1 2 3 4 5 2 | 2 3 4 5 6 3 | 3 4 5 6 7 4 | 4 5 6 7 8
18	**Years in a Range** Write a program to count the number years in a range that has a repeated digit. For example, 2012 has a repeated digit, but 2013 does not.
19	**Logic Gate** Write a program that will give the students the answer to logic gate questions For example: Enter logic gate : OR Enter first input : 1 Enter second input : 0 Result = 1 It should work for the logic gates OR, AND, XOR, NAND and NOR
20	**Palindromes** Write a program that checks if a string entered by the user is a palindrome. A palindrome is a word that reads the same forwards as backwards like "racecar"

21	**Data Entry** Create a program that retrieves the membership details for a Rock Climbing Club. The program should take a range of details and then repeat them back, with headings, for confirmation. Once confirmed, the program stores these details; else it clears them and allows a new input. **Extensions:** 1. Allow entry of more than one membership 2. Store membership details to a file 3. Retrieve details from a file 4. Allow searching for stored users
22	**Simple Life Calculator** Create a program that has 3 simple calculators within it, e.g. VAT, Tax and Times table. Allow users to choose which calculator they want to use and then carry out that calculation. **Extensions:** 1. Use a an option menu so that the user can use more than one calculation before the program closes
23	**Fibbing** Create a program that will calculate the Fibonacci Sequence to 10 places. **Extensions:** 1. Allow the user to specify the number of places generated 2. Print this in reverse order 3. Display the total of all the numbers shown

24	**Hack-proof** Create a program that will only open a text document if the correct password is entered. The user should choose the username and password first and it should also verify the password before allowing it. **Extensions:** 1. Create a random password first of at least 8 characters first as a suggested password 2. Create a random password that contains at least a lowercase, uppercase and special character of at least 8 characters in length 3. Verify that the password given by the user matches: a. The limits in Extension 1 above b. The limits in Extension 2 above
25	**Ordering** Create a program that allows entry of 10 numbers and then sorts them into ascending or descending order, based on user input. **Extension:** 1. The user can input a word or string, and it arranges the string into alphabetical order. E.g. My Rabbit would be shown as "abbimty". (Punctuation placement is not essential) 2. Repeat Extension 1, but include the sentence structure
26	**Truth or not** Create a program that would take the number of inputs in a logic circuit and works out the number of output lines are needed for the truth table. Have it draw the truth table on screen, using Columns for Inputs (A, B, C etc) and rows for the 1's and 0's. **Extension:** 1. Fill in the rest of the truth table if you can!
27	**Word Subtraction** Create a program that takes two strings/words. Then then converts this to an ASCII value and subtracts the values from each other. **Extension:** 1. Also add a function that removes any characters in the second word that occur in the first word. E.g. Fish and Tin, would return "Fsh" and "Tn"

28	**Name that Number** Telephone Keypads often have letters associated with each number. This means that 0141 117 2556 could be stored as 0141-CAT-DOOR. Create a program that can convert a phone number with "letters" into one that only contains digits. **Extensions:** 1. Can you develop your program so that only words in the dictionary are allowed?
29	**Item Merge** Create a program that will compare two shopping lists from "Week A" and "Week B". It will return any unique items contained on the list. **Extension:** 1. Append the two lists, with no repetition 2. Develop this to 4 Weeks of shopping an highlight the top 3 most popular items
30	**Year Addition** Create a program that accepts a year in the format ####, e.g. 2015. The program then adds each digit of the year together and outputs the answer. E.g. 2015 becomes the output 8. **Extension:** 1. Develop this so that the user can guess an integer value. If the MOD division is "0" they score a point, if it isn't they can guess again, up to 3 attempts in total
31	**Forwards and Backwards** Create a program that is able to detect if an input is the same as the reverse of the same input – i.e. a Palindrome
32	**Code it up** Create a program that adds 25 to the value of each character of a string that a user enters. This new string should be saved and output. **Extension:** 1. Develop your program to include a conversion from a 'coded' string back to a normal string 2. Develop your program to allow the user to enter the number they want the string coded by (e.g. 12) 3. Develop your program to then decode a string, based on the coded value that the end user enters

33	**Mor-se Coding** Create a program that allows you to enter a string and encode it into Morse code, using '.' and '-' notation. Spaces between words should be replaced with the "|" (pipe) character. Use a normal space for gaps between each character. **Extension:** 1. Develop your program to translate from Morse to alphanumeric, using the standards above
34	**What's the day?** Design a program to take 3 inputs, one for day, one for month and one for year. Get your program to validate if this is an actual day, and, if it is, output the day of the week it is! *Hint: How do leap years affect this program?*
35	**Game of Chance** A user can bet on any number from 0 to 30. If it's an even number they 2x their money back. If it's a multiple of 10 they get 3x their money back. If it's a prime number they get 5x their money back. If the number is below 5 they get a 2x bonus. Create a program that allows the user to guess a number. A random number is generated. If the guess == the random number then the user wins and gets a pay-out. Combinations of the win scenarios should be catered for.. e.g. 20 wins 2 x 3 bonus = 6x their money. **Extension:** 1. Develop your program to allow a user to enter the amount they want to place for that bet, and work out the resulting pay-out 2. Develop your program to store the user's current balance and stop them from betting if they have no money left 3. Develop your program to finally incorporate validation so that they cannot enter into a negative about of cash ever, and that a bet should be between 1 and 10 units of currency 4. Develop your program to allow multiple bets on different numbers
36	**Triangulate** Create a program that accepts 3 sides of a triangle. It then works out if these sides form a triangle, and if so, what type of triangle (e.g. Scalene, Isosceles, Right-Angle….) **Extension:** 1. Develop your program to allow 2 sides of a triangle and an angle, to work out the length of the missing side

37	**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'
38	**Sing Along** Create a program that prints the lyrics to the song '10 green bottles' in as few lines of code as possible. **Extension:** 1. Develop this program so that you can enter any starting number and it will count down from there
39	**Even more Odd** Create a program that accepts and random integer array (at least 10 integers) and orders them firstly by size, (small to large), and then puts all the even numbers AFTER the odd numbers within the array. It then echos the original array and the modified array to screen. E.g. an array 1,2,3,4,5,6,7,8,9,10 would be output 1,3,5,7,9,2,4,6,8,10. **Extension:** 1. Develop your program to allow Character input as well, and these come before the integers, and are listed in reverse alphabetical order
40	**Base of Numbers** Create a program that converts a denary number into its hexadecimal equivalent. **Extension:** 1. Allow the user to specify the base that they want to convert the number into, using an integer, e.g. 16 for Hexadecimal

The small print

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.

Whether you already offer OCR qualifications, are new to OCR, or are considering switching from your current provider/awarding organisation, you can request more information by completing the Expression of Interest form which can be found here: www.ocr.org.uk/expression-of-interest

Please get in touch if you want to discuss the accessibility of resources we offer to support delivery of our qualifications: resources.feedback@ocr.org.uk

Looking for a resource?

There is now a quick and easy search tool to help find **free** resources for your qualification:

www.ocr.org.uk/i-want-to/find-resources/

www.ocr.org.uk/alevelreform

OCR Customer Contact Centre

General qualifications

Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk

OCR is part of Cambridge Assessment, a department of the University of Cambridge. *For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored.*

A DIVISION OF CAMBRIDGE ASSESSMENT

LLOYD'S REGISTER · LRQA ISO 9001 001

UKAS MANAGEMENT SYSTEMS