OCR
Oxford Cambridge and RSA

GCSE (9–1)

COMPUTER SCIENCE

J276
For first assessment in 2018

Version 2: Valid for 2019 and 2020 examination series

Programming Project Guidance

ocr.org.uk/gcsecomputerscience

Contents

1. Introduction

1.1. What is the Programming Project?

The Programming Project is an opportunity for candidates to engage in an authentic programming experience as part of the GCSE (9–1) Computer Science course.

The Programming Project does not count towards the final grade, but does consolidate the learning across the specification through practical activity. The Programming Project also helps to build core programming skills which will benefit the candidate in further study or employment.

Centres must:

- Provide learners with the opportunity to undertake the Programming Project during 20 timetabled hours
- Ensure the work created is authentic and individual to that learner
- Ensure that learners appropriately reference any material used from a source
- Ensure that learners cover each part of the project: Analysis, Design, Development, Testing, Evaluation

1.2. What is the purpose of this document?

This document should be read in conjunction with the GCSE (9–1) Computer Science specification (http://www.ocr.org.uk/Images/225975-specification-accredited-gcse-computer-science-j276.pdf) and provides further guidance on how the Programming Project is to be administered within your centre.

Teachers must ensure that they are familiar with all Programming Project requirements, as a failure to meet the requirements will be regarded as malpractice/maladministration.

1.3 How do I find out more information?

Please contact our Computer Science Subject Advisors for advice and support.

01223 553998

ComputerScience@ocr.org.uk

@ocr_ict

2. Tasks

2.1 Choosing a task

OCR provides three Programming Project tasks each year from which a candidate must complete **one**. This allows centres to either:

a) Choose one task for their entire cohort

b) Choose a separate task for each class group/set

c) Allow candidates to pick the task they prefer

Programming Project tasks are released by OCR on 1 September. Centres must use the correct tasks for submission.

2.2 Where to find the tasks

These will be made available on the OCR Computer Science subject webpage http://www.ocr.org.uk/qualifications/gcse-computer-science-j276-from-2016/

3. Preparing candidates

3.1 Skills required

Candidates should have been taught all of the programming techniques listed within the specification for the Programming Project. Please see Section 2d and 2e of the specification for further details.

The programming techniques for the Programming Project are a subset of those required for the Component 02 written examination.

It is important for candidates to understand *why* and *where* programming techniques may be used. This should support them to be confident in 'thinking outside the box' when solving problems. This aids candidates in generating individual solutions.

3.2 Solutions

Candidates must work individually to produce a solution. Group work is **not** permitted.

Programming Project tasks allow a wide range of solutions and pathways to a solution, there is no single 'perfect solution'.

4. Resources and support

4.1 Use of resources

Candidates have unlimited access to resources and the internet during completion of the Programming Project.

We would recommend that candidates are given advice on where best to seek support as this will help candidates locate sources of support quickly and effectively. Teachers have a responsibility to ensure that any teacher-provided materials do not mimic the Programming Project task.

Candidates must be made aware that their work should be authentic, and all resources used are appropriately referenced.

Submission of plagiarised work may result in a malpractice investigation.

4.2 OCR resources

OCR resources that can support the Programming Project may be downloaded from our Computer Science subject webpage https://www.ocr.org.uk/qualifications/gcse/computer-science-j276-from-2016/.

4.3 Support

Teachers may assist and give feedback to candidates during the Programming Project, but candidates must be the driving force behind their solution. Teachers must not restrict candidates to specific solutions or provide entire solutions.

Where a teacher gives support to a candidate, it must be clear through the use of referencing where that support starts/ends, and where individual candidate work commences.

4.4 Writing frames, worked solutions and templates

OCR has produced a Programming Project Report template which candidates can use to guide them through their write up, providing a structure to ensure that they cover all sections of the Programming Project. This can be found on our website at: https://www.ocr.org.uk/qualifications/gcse/computer-science-j276-from-2016/assessment/

Teachers may customise or create their own document as long as it does not go beyond the OCR provided template in terms of support and structure.

The use of writing frames, third party worked solutions and 'similar but different' solutions are not permitted during completion of the Programming Project.

4.5 Referencing

Candidates must reference all resources and support used during the Programming Project task appropriately. We would encourage candidates to reference at point of use within their Programming Project report, rather than at the end.

Not referencing resources or support may be investigated as malpractice and/or maladministration.

5. Writing the Programming Project report

5.1 The Programming Project report

The report candidates write should document their work on the Programming Project task from start to finish. For any code/solution they develop, they are expected to produce evidence to show that they have completed the following:

- Analysis
- Development
- Evaluation
- Design
- Testing

For an authentic experience, candidates must engage with all areas of the Programming Project. There must be clear evidence that each section listed above has been attempted. Any refinements candidates make as they progress through the Programming Project should be captured within their report.

If any of the sections listed above have not been included or given only token consideration, it may result in a malpractice/maladministration investigation.

5.2 How to present the Programming Project report

Candidates should be encouraged to submit a single narrative for the Programming Project report. To help ensure that the report is clearly laid out, candidates should:

- Use page numbering
- Use headers/footers (Centre Number, Candidate Number, Candidate Name, Title of work)
- Use appropriate headings and subheadings
- Include screen shots with word wrapped text/annotation (this will help identify key steps candidates have taken)
- Complete a spelling and grammar check

OCR has produced a Programming Project Report template which candidates may find helpful to use. This can be found on our website at: https://www.ocr.org.uk/qualifications/gcse/computer-science-j276-from-2016/assessment/.

5.3 School accounts

You may use normal school accounts to complete the Programming Project. As a centre you have an obligation to ensure that your candidates' work is authentic. Some teachers may feel more comfortable in creating separate secured accounts.

5.4 Completion date

The Programming Project must be completed by 15 May for the year of entry.

Centres may complete the Programming Project in either year of the GCSE programme of study.

For centres undertaking a 3 year programme of study, it must be done in the second or third year of study (i.e. Year 10 or Year 11).

5.5 Feedback to candidates

There is no requirement to mark the Programming Project as it does not count towards a candidate's final grade.

We have provided a progress tracker should a centre wish to provide formative feedback to their candidates during the completion of the Programming Project (please see appendix 1).

To support teaching and learning, teachers may provide feedback which enables candidates to make progress through the Programming Project. This feedback should be referenced appropriately. The program and the report must still be candidate driven.

6. Monitoring of the Programming Project

6.1 Monitoring of work by OCR

OCR will contact all centres to request a sample of work for each examination series in which entries are made. OCR will be checking that the following Programming Project requirements have been met. Centres **must** have:

- Provided learners with the opportunity to undertake the Programming Project during 20 timetabled hours
- Ensured the work created is authentic and individual to that learner
- Ensured that learners appropriately reference any material used from a source
- Ensured that learners cover each part of the project: Analysis, Design, Development, Testing, Evaluation

6.2 Submitting your sample of work to OCR for monitoring

All candidate reports being sampled must be submitted electronically via post on CD/DVD/USB-drive. The sample must be posted to the address provided.

Candidates should create a single report file. The following file naming convention must be used:

CentreNumber_CandidateNumber_Surname_Forename_Filename

e.g. 99999_2505_Hopper_Grace_ProgrammingProject.pdf

We advise centres to keep a backup of all candidate work. Please note, work will not be returned to your centre after the monitoring has been completed.

Failure to submit the requested sample will be investigated as maladministration.

6.3 Additional documents

Candidates should include evidence of annotated code within the Development section(s) of their report. There is no requirement to submit raw code files, or compiled code. Where suitable, the raw code may be copied and pasted into the report for reference. Raw code and executable files will not be opened for security reasons and therefore cannot be relied upon as evidence.

Where use of multiple files is unavoidable (e.g. a word document with video evidence for some testing), each file must be suitably named to aid identification of its contents.

e.g. 99999_2505_Hopper_Grace_Videotesting.wmv

6.4 Candidate Authentication Statement

Each candidate must complete a Candidate Authentication statement which confirms the work they have produced is their own. A statement for you to use is located on the OCR webpage http://www.ocr.org.uk/administration/stage-3-assessment/general-qualifications/internal-assessment/.

Completed statements are held within your centre and are not required for submission with the sample.

6.5 Programming Project Authentication Form (CCS161)

There is a Programming Project Authentication Form (CCS161) for J276 (9–1) Computer Science. This must be completed and signed by all teachers who have been involved in delivering the Programming Project task. A senior member of staff will also be required to sign the form to confirm that all Programming Project requirements have been adhered to.

The form can be downloaded from the OCR webpage at http://www.ocr.org.uk/qualifications/gcse-computer-science-j276-from-2016/assessment/.

Once completed the form must be submitted electronically to OCR on the CD/DVD/ USB-drive which contains your centre's sample of work. Failure to submit the signed form will be investigated as maladministration.

6.6 Evidence of 20 timetabled hours

The Programming Project is to be completed within the classroom and evidence of delivery of 20 timetabled hours must be submitted with your signed CCS161 form and sample of work.

OCR provides a tracking spreadsheet that you may wish to use to record this time. This can be downloaded on our website at https://www.ocr.org.uk/qualifications/gcse/computer-science-j276-from-2016/assessment/.

6.7 Checklist: Sample submission to OCR

I have:

- ☐ all Candidate Authentication statements signed and stored in the centre
- ☐ a CD/DVD/USB-drive with a folder for each candidate whose work has been requested for sample monitoring
- ☐ within each candidate folder, a single Programming Project report file (ideally PDF format)
- ☐ used the correct naming convention for the Programming Project report file
- ☐ suitably labelled any supporting evidence within the correct candidate folder
- ☐ an electronic copy of the evidence of delivery of 20 timetabled hours included with the sample on the CD/DVD/USB-drive
- ☐ checked all documents open correctly with no shortcuts copied in error
- ☐ the correct address for posting that was sent to the Exams Officer
- ☐ a backup copy of candidates work kept within the centre.

Appendix 1: The Programming Project progress tracker

See Section 5.5

Programming Techniques

☐ Few techniques are used	☐ Several techniques are used	☐ Most techniques are used
☐ There is no/limited attempt for all parts of the task	☐ There is a reasonable attempt for all parts of the task	☐ There is a complete attempt for all parts of the task
☐ Code is disorganised and lacks commenting	☐ Code is generally well structured and commented	☐ Code is consistently well structured and well commented
☐ The task is approached in a linear fashion	☐ The task has been broken down into component parts, but not always effectively	☐ The task has been broken down into component parts effectively and efficiently
☐ Designs do not reflect the code produced	☐ Designs are comparable to the code produced	☐ Designs accurately reflect the code produced
☐ There is no real efficiency built into the code	☐ There is some efficiency within the code	☐ The code is fully efficient

Analysis

☐ There is little analysis for each part of the task	☐ There is some analysis for each part of the task	☐ There is full analysis for each part of the task
☐ There is a limited attempt to decompose the task	☐ There is some attempt to decompose the task	☐ There is a full attempt to decompose the task
☐ Decomposition is not effective	☐ Decomposition is generally effective	☐ Decomposition is fully effective
☐ Success Criteria identified are generic and lack detail	☐ Success Criteria are defined and generally detailed	☐ Success Criteria are clearly defined and detailed
☐ Success Criteria do not cover task functionality	☐ Success Criteria cover some task functionality	☐ Success Criteria cover all task functionality
☐ Discussion of approaches is limited	☐ Discussion of approaches is present but not justified	☐ Discussion of approaches is present and justified
☐ Testing plans are limited, or are present but have no explanation	☐ Testing plans present and discussed. Some are linked to the requirements.	☐ Testing plans are present and justified in relation to the requirements
☐ Validation for robustness is not discussed or limited	☐ Validation for robustness is discussed but not justified	☐ Validation for robustness is justified against the requirements
☐ There is no, or limited mention of real-world utility	☐ There is some discussion of real-world utility	☐ There is full discussion of real-world utility

Design

☐ The algorithms are either incomplete or missing	☐ The algorithms are complete, but not fully functional	☐ The algorithms are fully complete and clearly functional
☐ The intended approach is not discussed or justified	☐ The intended approach is discussed but not justified	☐ The intended approach is both discussed and fully justified
☐ The user interface is not planned, or planned but not functional	☐ The user interface is planned but may not be fully functional	☐ The user interface is fully planned and functional
☐ The design could not be used to construct the intended solution	☐ The design could be used to construct the intended solution with some guidance	☐ The design could be used to construct the intended solution without guidance
☐ The test plan covers some basic functionality but is limited	☐ The test plan covers most functionality with some robustness testing	☐ The test plan covers full functionality and covers robustness testing
☐ There is limited discussion of variables and data structures	☐ There is discussion of most variables and data structures	☐ There is justification of all data structures and variables
☐ There is little evidence of modular designs	☐ Most of the program is designed in a modular way – but not effectively	☐ An effective modular design is produced

Development

☐ There is little or no evidence of how the solution was built	☐ There is some evidence of key development points as the solution was built	☐ There is comprehensive evidence of the solution as it was built
☐ There is little or no evidence of systematic testing during development	☐ There is some evidence of systematic testing during development	☐ There is full evidence of systematic testing during development
☐ There is little or no evidence that systematic testing is used to refine the solution	☐ There is some evidence that systematic testing is used to refine the solution	☐ There is significant evidence that systematic testing is used to refine the solution

Testing

☐ The test plan has been partially completed, missing key requirements testing	☐ The test plan has been mostly completed, with key requirements tested	☐ The test plan has been fully completed, with all requirements tested
☐ The solution meets a few of the success criteria	☐ The solution meets most of the success criteria	☐ The solution meets all of the success criteria
☐ Testing for robustness has not been completed	☐ Some testing for robustness is evidenced	☐ Testing for robustness is fully evidenced

Evaluation and Conclusions

☐ Unresolved issues are ignored or not addressed	☐ Unresolved issues are commented on, but resolution to these are not discussed	☐ Unresolved issues and their resolutions are fully justified
☐ There is little or no evidence that systematic testing is used to refine the solution	☐ There is some evidence that systematic testing is used to refine the solution	☐ There is significant evidence that systematic testing is used to refine the solution
☐ The report is lacking coherence	☐ The report is generally coherent	☐ The report is coherent

www.ocr.org.uk

OCR Customer Contact Centre

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