OCR Advanced GCE in Computing

**F454**

**Project Advice**

**2015-2016**

**“aka Mr. Sargent’s Coursework Bible”**

F454 – Project Advice

Contents

[F454 Project Deadlines 3](#_Toc356210600)

[Forward by the chief examiner 4](#_Toc356210601)

[General Comments 5](#_Toc356210602)

[Keeping a diary 5](#_Toc356210603)

[RAD or traditional design? 5](#_Toc356210604)

[About this document 5](#_Toc356210605)

[The user 5](#_Toc356210606)

[Definition, Investigation and Analysis 6](#_Toc356210607)

[Definition 6](#_Toc356210608)

[Investigation and Analysis 8](#_Toc356210609)

[Design 12](#_Toc356210610)

[Nature of the solution 12](#_Toc356210611)

[Algorithms 15](#_Toc356210612)

[Test Strategy 16](#_Toc356210613)

[Software Development and Testing 18](#_Toc356210614)

[Software Development 18](#_Toc356210615)

[Testing 20](#_Toc356210616)

[Documentation 22](#_Toc356210617)

[Evaluation 24](#_Toc356210618)

[Discussion of the degree of success in meeting the original objectives 24](#_Toc356210619)

[Evaluate the user’s response to the system 25](#_Toc356210620)

[Desirable extensions 27](#_Toc356210621)

[A Brief note on the format of your f454 project reports 28](#_Toc356210622)

[F454 project hand in checklist 29](#_Toc356210623)

# F454 Project Deadlines

# Forward by the chief examiner

It is impressive to see so many original ideas and obvious enthusiasm for developing coded solutions. It was also clear that the key to success lies in careful, thorough and detailed investigation of the problem and potential solutions.

Where candidates have gone beyond a simple interview and looked at the problem in greater depth the development became more focused and effective.

There were a number of impressive coded games and activities including retro games like asteroids and teaching aids such as a binary tutor.

It is clear that when a student finds the topic area interesting they focus more on the task in hand and produce good quality solutions.

The best reports provided a commentary on the whole process in chronological order using segments of code, test results and further investigation to illustrate their progress towards a working solution.

It is worth noting that a number of centres have submitted electronic evidence alongside the printed report to good effect.

OCR will now accept the work for this unit in electronic form and no longer require a printed report.

We are also quite happy to accept electronic forms of evidence for testing to supplement these reports, for example, avi files showing testing in progress.

# General Comments

## Keeping a diary

The student should keep a diary from the start of the project work up to completion. The diary should contain:

* Aspect being tackled – the stage of development.
* User contact – who was met, when and why?
* Outcomes from user contact.
* Problems – date of the problem, what the problem was, what action was taken to try to resolve the problem?

The diary is not part of the project write-up, but should be submitted alongside the work as a genuine record of the project development, not written up retrospectively. The diary / development log should form one of the Appendixes of your project write up.

## RAD or traditional design?

Although you have studied both approaches to the SDLC during your course the RAD approach has proven unrealistic due to the time constraints on you.

Therefore it is strongly recommend that you tackle your F454 project using the traditional system lifecycle approach with all the design up-front before development.

If you still wish to have a RAD approach you must have a very strong case for doing so, and must come and see me about it before work on your project begins.

## About this document

What follows is the mark scheme for each section of the report write-up, together with comments about that section from the external examiner in blue. Comments for markers are shown in green which may help in understanding how marks are awarded within the section. Advice from Mr. Sargent is written in red. Not all projects will necessarily easily fit the mark scheme so these comments inform you how to tackle each section.

## The user

Every project needs an identified user or target audience. They play a key role in an OCR Computing project.

For the purposes of the project development, your teacher can be a pseudo-user: someone acting on behalf of a real user. For example, if you are making a game for young children or a pub quiz machine, it may be difficult to have regular contact with a user. Your teacher can be that person in these cases.

# Definition, Investigation and Analysis

## Definition

A candidate should not expect the examiner to be familiar with the theory and practice in the area of the chosen system. There should be a brief description of the end user (for example, firm or business) involved; and the current methods used or details of the area for development that may form the basis of the project. A clear statement of the origins and form of any relevant data should be given. At this stage, the exact scope of the project may not be known and it may lead to the arrangement of an interview with the user.

3 marks - Excellent description with all elements present.

2 marks - Some description of both the stages of study and end user involved.

1 mark - Vague description of the end user or area for development.

External Examiner advice

When defining what was required for the A2 project, the writing team were deliberately vague about exact requirements. This approach allows a wide range of project ideas to be available to students. Anything from fully coded solutions to database solutions with a coded element are possible. You should note that specific mark points are almost impossible to identify. We have opted to use a set of marking criteria into which teachers can map the students’ work on a ‘best fit’ basis.

In this section the student should set the scene for the project. The contents of this section are intended to come from a brief introductory meeting. Typically this section will identify:

The organisation

What the organisation does

The area of interest or department within the organisation (if applicable)

The end user

The end user’s role in the organisation

What the intended project is about

The sort of data that is likely to be involved and, if possible, some indication of the source of this data

While this might be considered a check list, simply presenting the facts with no description is not sufficient for 3 marks; some brief discussion will also be required.

Teacher advice

In this section you are giving a brief background to the problem. Answer the questions:

What is the company?

What do the company do?

Who is the end user?

What problem do they have?

How are you going to find out more in order to write the analysis section?

This section should be no more than a couple of paragraphs.

If you are writing a computer game, give a description of the type of game it is with a very brief explanation. Explain who will play the game and on what platform. Identify the key requirements outside the actual game play itself, e.g. in the case of a mobile phone game: easy to pick up and put down, pause at any point and continue later.

If the problem has already been solved, pretend you are solving it for a new platform or user. E.g. with space invaders, perhaps it is a mobile phone version or a new twist on the old concept.

Don’t be overly ambitious. This is only A’level. You will not be writing CoD: Modern Warfare! It is always better to have a small program well done and well written up than a huge problem you can’t solve.

## Investigation and Analysis

This section is the ‘systems analysis’. The question is not how a system performs detailed tasks, but rather how the project progresses from the original data to the results. The candidate should describe how the user requirements were ascertained (including detailed planning of the

investigation). The results of the investigation should be recorded accurately and analysed

carefully to show how the candidate has arrived at the requirements specification. The specification must be detailed and should include the user, hardware and software requirements of the proposed solution.

9–11 marks - Excellent user involvement with detailed recording of the user’s requirements. All

other items must be present, showing a thorough analysis of the system to be

computerised. A detailed requirements specification, including full justification for the

approach and hardware and software requirements, has been produced.

6–8 marks - Good user involvement and recording of the data collection methods. Most of the

necessary items have been covered. However, one or two items have been omitted.

A requirements specification is present with some attempt to justify the approach

based on the results of the investigations but with some omissions, eg hardware and

software requirements.

3–5 marks - Some evidence that an attempt has been made to identify the end-user requirements and some recording of it has been made. Attempts at some of the other items have been made. An attempt has been made to develop a requirement specification but

with little attempt to justify this based on the results of the investigation.

1–2 marks - Some elements have been discussed but with little or no user involvement.

External Examiner advice

This is the most important phase of the project. This needs to cover:

Information collection

Strategies

Requirements specification

This section tells the story of how the student gets from the brief introduction to the detailed requirements specification agreed with the end user. There are items we would expect to see here including detailed interview planning and data collection although the choice of project and end user will dictate the format of this information.

Interviews may be carried out face to face and we will see the notes taken during this. If the end user is remote, we may see emails or transcripts of telephone conversations. There may be lots of forms already in use. There may only be ideas for data collection forms or reports. The important points to note in this section are that careful planning and details are required and we need to see evidence of the process involved in completing this section. Simple typed transcripts are not sufficient.

The information, once gathered, needs to be analysed and subsidiary investigation instigated in order to develop a clear idea of what is required and how the solution might be developed. It is important for the student to justify their approach to the problem. Students should offer realistic justification for their chosen approach rather than simply presenting to the moderator with two totally unsuitable solutions and one suitable one. We want to see a proposed solution justified by reference to the analysis of the information collected and the end user requirements.

The requirements specification should be written as a report to the end user. It should explain what the student has identified from the investigation and analysis as a potential solution. This requirements specification must also include details of hardware and software requirements. The end user must signify that they have seen this specification and that, possibly with some minor modifications, wishes the student to proceed with the development.

Notes from previous years from the Chief Examiner

The poorest projects often failed because of a poor analysis section. Lack of a genuine end user often produced simplistic and superficial analysis. The interviews need to be improved with questions being asked which lead to important facts being discovered which can be used in the design section. There should be much more probing interaction between the student/analyst and the client. Students should look at similar systems to provide further research.

The analysis section should show the interview responses, a discussion of these and then a return with the suggestions, e.g. the player ship designs, the way the lives are represented etc. This process cannot be completed in a single interview. The report should show the full process of interview, deliberation, response and agreement.

Suggestions for alternative solutions need to be credible. Students should be looking at exisiting commercial products and seeing exactly what they can do.

Requirements like “user friendly” are too vague. The objectives need to be specific and measurable.

June 2011 Comments:

Often the investigations were limited to basic, poorly structured transcribed interviews with little evidence of end user participation in the process.

Candidates need to look beyond the simple interview and research the area in which the problem exists, looking at existing solutions to similar problems, and the background to the problem.

Often alternatives failed to provide the rationale for the chosen approach to solving the problem simply identifying alternatives, a database or a spreadsheet or a paper based solution.

Candidates at this level should be able to provide a meaningful rationale for their chosen approach based on the results of their investigation, research and analysis of the problem area.

This section is worth almost 14% of the marks and needs to be treated in far more detail than is evident in some of the projects.

Notes for markers

If the candidate has provided excellent evidence for investigation with no omissions, has analysed the data carefully but has produced a requirements specification lacking detail and has not discussed hardware and software then we have to try and fit this into the most appropriate mark range.

The investigation may be considered worthy of 9-11 marks, but the requirements specification may be worthy of just 6-8 marks. In this case the teacher may decide on 9 because of the quality of the investigation or on 8 because the omissions are too important to overlook.

The marking criteria should be seen as a continuous scale against which the work can be measuered with the criteria describing typical scenarios for that mark range. It is also important that teachers use the full range in each subsection of the marking criteria.

Teacher advice

In this section you are describing the problem. With a game, take this approach to the write up:

1. Initial research: start by identifying a similar game (perhaps from the internet) and describe the mechanics of how it plays.
2. Form a set of questions to ask the user about how your game should look, sound and play. Document the user responses to these questions. (See note 1) E.g.  
   Q: What does the player control in the game?  
   A: The player controls a spaceship that can move left and right at the bottom of the screen.
3. Deliberate on the answers you are given and the initial research. This will inform the proposals.
4. Propose a solution to the problem by describing each element of the game in detail. You can have mock ups of the graphics from a drawing application at this stage.  
   (See notes 2 & 3)
5. Get a response from the user about whether this meets their expectation.
6. Get an agreement from the user.

Note 1: You need to conduct an interview and/or observation of an existing system to know the details of what you need to know to make the program later. Keep records of the questions and observations you make, together with answers to questions.

Note 2: You need to discuss in detail exactly what the system is going to do, but not how it is going to do it. This is not about algorithms, it’s about the requirements. Don’t misinterpret “The question is not how a system performs detailed tasks, but rather how the project progresses from the original data to the results” that simply means you are discussing the what, not the how. Detail is very important in this section in your descriptions of the system.

Note 3: Consider a typical space invaders game. You would need to discuss that the player controls a ship. The ship can move left or right inside a fixed plane at the bottom of the screen. The ship can fire one bullet at a time. There can’t be more than one bullet from the player on the screen at the same time. The objective is for the player to shoot all the invaders. The invaders start towards the top of the screen and move from left to right together in initially 5 rows of 12 invaders. When the right-most invader reaches the right edge of the screen, all the invaders move down a little on the screen and start moving from right to left. When the left-most invader reaches... etc.

‘Leave no stone unturned’. Your analysis should include sufficient detail so if you were to get a programmer to read the analysis, there is nothing more they would need to ask before making the solution. Of course the really fine details may not be entirely known and will be picked up in the development process. For example, the speed at which the ship moves across the screen. You would need to play the game to know what feels right. That is unlikely to be known at the analysis stage and the necessary dialogue between you and a user will gain you marks in the design section later.

If you are producing a game that has no single identifiable user, you need to choose someone to act as your advisor and tester. You should write your analysis as if you were having a discussion with a user. For example, “The intended audience for the game is...” “I am using my teacher as a representative for that audience.” “I discussed the requirements of the game with...” “It was suggested to me that...”

Whatever the problem, it will always have a target audience and therefore an identifiable user which you should be discussing requirements with and keeping records of these discussions.

You should discuss the hardware and software required to run your program. E.g. an IBM compatible PC with x processor , y memory and z hard disk space running the Visual Basic runtime libraries? Find out the necessary spec to run the development environment i.e. VB or Access on a computer. That will help.

As a summary of the analysis, create a numbered point list of the exact and actual requirements. Call this the requirements specification. Avoid requirements that cannot be measured. E.g. “It must be easy to use” is too vague. “The user should be able to find a product within 20 seconds” is better. “A player scores 50 points for each invader.” Remember, specific and measurable. It should contain numbers. Numbers of records, users, invaders, points etc.

# Design

## Nature of the solution

A detailed systems design (including appropriate diagrams) should be produced and agreed with the users. Proposed record, file and data structures should be described and design limitations should be included. Design of data capture forms, input formats (with examples of screen layouts) and output formats should be included. A detailed summary of the aims and objectives should also be included. These items are the design specifications, which should be agreed with the user.

5–6 marks - A clear set of objectives with a detailed and complete design specification, which is logically correct. There is evidence to show that the end user has seen and agreed these designs. There are also detailed written descriptions of any processes/modules and a clear, complete definition of any data structures. The specification is sufficient for someone to pick up and develop an end result using the software and hardware specified in the requirements specification.

3–4 marks - The major objectives of the new system have been adequately summarised, *but omissions have been made*. There is a brief outline of a design specification,including mock-ups of inputs and outputs, and the process model has been described(including a diagram: structure diagram, data flow diagram or system flowchart).There is some evidence that the end user has seen these designs. However, there isa lack of completeness with omissions from the process model, inputs and outputs.Data structures have been identified but there may be inadequate detail.

1–2 marks Some vague discussion of what the system will do, with a brief diagrammatic representation of the new system.

External Examiner advice

What appears in the design section depends very much upon the chosen problem and method of solution but some things will always need to be included. These include:

System objectives developed from the requirements specification

Input, output and general user interface

Process models in a suitable format

There will in most cases, be record and file structures.

System objectives should be detailed performance objectives for the system derived from the user requirements. Rather than stating in general terms what the system must achieve they specify what the system must do in measurable terms. These will form the basis for detailed system testing later on in the process.

The level of detail is a critical factor here. For the top range of marks, it should be possible for a suitably skilled person to pick up the design and build the product without further investigation of the problem.

Once again, make sure there is evidence the end user remains involved with the system development. Without end user agreement to the work, the marks available will be limited.

Notes for markers

If the system objectives are clear but not quite complete, there is evidence of input and output designs and proposed processes but little end user involvement and insufficient detail to send this to another developer, then we are looking at 2 or 3 marks for this section.

One frequently asked question about the mock up designs is, “Do these have to be hand drawn?” The answer is no. Most systems will be developed using RAD and providing there is evidence of this process, that is sufficient.

Teacher advice

In this section you are presenting your proposals for how you will solve the problem. You should include the following:

1. A systems diagram

It should identify the key stages of the solution. See the exemplar your teacher has given you.

1. Sub headings for each box from the systems diagram
   1. Proposed screen designs

Include sketches of what the screens or form designs will look like. These can be ‘mock-ups’ done in a Word document, Publisher, Excel etc. or hand drawn sketches scanned in. These should include the real graphics and dimensions of the graphics at this stage.

If there are output reports these should be planned too. Be careful to include all major screen designs. If you are making a game, each level should be sketched, together with the main menu, help screen, high score screen etc. You may conclude that from the External Examiner advice, proposed screen shots are not necessary. This is not true as initial designs are part of “Nature of the Solution” and we need to see them early on.

If your project is a database all the forms and reports should be sketched.

* 1. Detailed summary of the process

Under the sub headings a detailed summary should be written of what happens at each stage. This would include some description of what coding/algorithms would be necessary. This is not algorithms themselves, but descriptions. Enough for an experienced programmer to derive pseudo code from. E.g. move the invaders 10 pixels according to the invader direction variable. +10 moves right, -10 moves left. If the invader has reached the edge of the screen, reverse the values to cause the invader to move in the opposite direction next time the procedure is called.

Under the relevant sub-heading identify all the ‘objects’ required in the program if it is a game, such as the ship, invaders etc. You should include class diagrams for these.

Identify classes, arrays and files that would be needed. For example, a high score table may be stored in memory as an array and on a disk in a sequential file. For games with levels, include one example of a text file that will be read in for the level data.

In a database, the tables, fields, data types and relationships should be identified.

In a databases solution under the relevant sub-heading identify all the queries, macros or code elements needed.

1. Explain the traditional SDLC process

You should explain that you are taking a traditional approach to the stages of the Software Development Life Cycle and justify why. However, you should make it clear that you will have serious amounts of user involvement and testing all the way through the development of your solution and not simply bolted onto the end of your project.

1. Sign off the proposal

You should show your proposed screen designs to your user as well as talking them through your solution. Record some feedback from them that they are agreed in principle, subject to any changes that need to be made before development commences. Get a dated signature at the bottom of this section to prove you did it.

## Algorithms

Detailed language-independent algorithms should be developed together with evidence that the algorithms have been tested to ensure they meet the design objectives.

5 marks - A complete set of algorithms with evidence to show that they have been assessed by

the candidate to show that they will meet the design specification. (Evidence should show how these algorithms form a complete solution and that they have been tested for functionality using appropriate techniques.)

3–4 marks - A complete set of detailed algorithms covering the system as specified.

1–2 marks - Some vague algorithms detailing how the system will be developed.

External Examiner advice

There must be a coded element in any solution, unless the student is limited to half marks in the software development section. A prerequisite of any coded element will be algorithms. There are marks allocated to the quality of these algorithms along with the testing that has been applied to them. There must be evidence that the algorithms have been properly tested to ensure they do the job for which they are intended.

Notes from previous years from the Chief Examiner

June 2011 Comments:

In a number of cases the algorithms provided were superficial top level algorithms that failed to describe the data processing within the solution.

Moderators were often left in doubt about the validity of the set of algorithms provided and whether they actually described a solution to the problem.

Notes for markers

The marking criteria here is quite clear but for those who have chosen to provide a completely coded solution, these algorithms may in fact also form part of their design, for example as part of the top down design. In this case, these two sections will probably be assessed from the same section of the report. This is acceptable.

Teacher advice

This is where the real thought about the internal workings of your solution begins. This section is often over looked and performed poorly by candidates.

You may come back to this section later to explain anything complex about your solution. For example, your collision detection may use a pruning tree technique to speed up the checking process. That sort of detail ought to be explained here but is likely to be done once you have written the code.

## Test Strategy

A detailed test strategy and plan, together with appropriate test data, should be developed and

documented. It is vital to produce test cases and to show that they work. To do this, it is necessary not only to have test data, but to know what the expected results are with that data.

5 marks - A detailed test strategy and plan covering all aspects of the system with data to test

under normal, extreme and abnormal circumstances.

3–4 marks - A detailed test strategy and a plan covering several aspects of the system but with

inadequate data to effectively test the system, eg data covers only normal

circumstances or covers only a limited part of the design specification.

1–2 marks - A vague discussion of how the system might be tested.

External Examiner advice

The final element of any design is planning for the testing of the system. Much of the system testing will take place during the development stage and the candidate needs to ensure this is carefully planned to cover all of the system design objectives. Evidence should include an overview of the test strategy including how this relates to the system objectives and the test data to be used.

Notes from previous years from the Chief Examiner

Test plans should be designed showing when things will happen such as modular testing, integration testing, beta testing and acceptance testing.

June 2011 Comments:

The test strategy should also set the tone for the development and testing sections describing the data to be used at each stage. A number of candidates merely described alpha, beta and acceptance testing in generic terms plus a post development test table.

Once again, at this level, candidates should provide the rationale behind their choice of test data for each stage of the development and testing process.

Teacher advice

Explain you are going to test the program as it is developed, but will have a black box approach to the final beta testing.

Create a table showing everything that needs to be tested by an independent test team or beta-testers to be assured the system or game works as intended. This is not the actual beta-testing, but the plan for it.

The tests should include: valid inputs, invalid inputs and extreme cases. It also needs to check all the ‘events’. E.g. “When a bullet touches an invader”.

|  |  |  |  |
| --- | --- | --- | --- |
| Test No. | What is being tested | Input data or actions | Expected outcome |
| 1 | The player’s ship cannot leave the screen to the left or right. | Move ship to far left and hold left key down.  Move ship to the far right and hold right key down | Ship stops and no longer responds to key press |
| 2 |  |  |  |

# Software Development and Testing

## Software Development

A technical description of how the solution relates to the design specification produced and agreed with the user should be included. It is the responsibility of the candidate to produce evidence of their development work. This section must show how the candidate tested each section during development and the responses to this alpha testing. The code must be documented adequately to explain its function and there must be clear evidence of how modular code has been used to develop the final solution.

13–16 marks - There is complete evidence showing how the solution was developed using suitable alpha testing at each stage to inform the process. The modular code is fully annotated indicating clearly the purpose of each section and the interrelationship between the sections. The developed solution fulfils all of the design specification.

9–12 marks – Program listings are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate, detailing their purpose. There is sufficient annotation evident to illustrate how the solution was developed for a particular purpose indicating the purpose of sections of code. The code will be modular and there will be good evidence to show how testing was used during the development process to inform each stage. The developed solution fulfils the design specification but there are some minor flaws in the solution.

5–8 marks – Program listings are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate. There is some annotation evident to illustrate how the solution was developed and some limited evidence that some testing took place during development. The developed solution has significant flaws and only partially fulfils the design specification. The code may be linear but with some of the annotation indicating how the code relates to the problem and some limited evidence of alpha testing.

1–4 marks – Program listings are provided in the form of printouts but with no annotation or evidence of alpha testing. The developed solution does not fulfil the design specification. There is some evidence of system development.

External Examiner advice

The system should be developed and the candidate provide good evidence to show the development stages and the testing processes that took place during this development. A project scoring in the top band must include detailed evidence of the tests and the outcome of each test.

The section should also show the coded element of the solution as a fully annotated print out. Fully annotated means that if the code is removed leaving just the annotation, then the code could be rewritten exactly. This annotation is also vital during the moderation process; it cannot be assumed that the moderator is familiar with the programming language used, but will be able to follow the logical processes involved from the commentary.

Notes for markers

One frequently asked question is, ‘can a student who produces a detailed solution but without any coded element score any marks in this section?’ The answer is yes and it illustrates how the marking criteria can be applied to work not quite matching the criteria to credit candidates for what they have achieved. Such a solution would suggest a mark in the 5-8 category, probably a maximum of 6 depending upon the quality of the other elements.

Teacher advice

Although you are most likely doing a Traditional SDLC model the development section musn’t be done totally in isolation. During the production of any major solution you must make sure to **“Tell the development story”**

During your development section of your report you must make sure to include the following details:

* Name of the stage e.g. player movement.
* Date development started.
* Commented, indented and annotated code listings.
* Explanation of testing being carried out and modifications to make.
* Screen shot of output.
* Feedback from user.

A solution that includes no code written by you can still gain 6 marks in this section.

You will be able to gain marks for many other sections of your report such as “Testing” and “Evaluation” under this section if you do it well. You should not simply leave any mention of testing & evaluation until after you have finished coding, this is an unrealistic way to develop a solution and the examiner will be aware of this!

Although you will have snippets of code throughout your project report to help you tell the develop story your entire code listing should be printed and fully annotated and should be included as an Appendix called “Code Listings” at the back of your project report.

## Testing

An attempt should be made to show that all parts of the system have been tested, including those sections dealing with unexpected or invalid data as well as extreme cases. Showing that many other cases of test data are likely to work – by including the outputs they produce – is another important feature. Evidence of testing is essential. The beta testing should cover all aspects of the test plan produced in the design section, which should cover all aspects of the design specification. The examiner must be left in no doubt that the system actually works in the target environment.

This evidence may be in the form of hardcopy output (possibly including screen dumps), photographs or any format that does not require any specific hardware or software. The end user(s) must be involved in this process and evidence of end-user testing is required.

11–14 marks – The testing covers as many different paths through the system as is feasible, including valid, invalid and extreme cases. The testing covers all aspects of the design specification and the test plan from the design section. There is clear evidence of end-user testing.

8-10 marks – There is evidence of testing covering most aspects of the design specification but with omissions, e.g. test data does not include erroneous data for all tests or there is limited evidence of end-user testing.

5-7 marks – There is limited evidence of testing based on a badly developed test plan with clear omissions. There is no description of the relationship between the test plan and the testing in evidence.

1-4 marks – A collection of hardcopy test run outputs with no clear link to the test plan and covering few aspects of the system. No evidence of end-user testing.

External Examiner advice

Once the system is developed, it needs to be subjected to final black box testing using suitable test data matching the system objectives. The end user must be involved in this process testing with typical data. Evidence needs to show how the system was tested using the test plan from the design section covering all the system objectives. The moderator must be left in no doubt that the system works.

Notes from previous years from the Chief Examiner

Testing of the nature, “does this button work” is not sufficient evidence for this section. It should cover input to output. Testing the algorithms.

June 2011 Comments:

The specification requires candidates to show the development process with testing at each stage. Many candidates simply provided a block of code followed by a test table with limited evidence of testing at any stage.

The purpose of testing, especially during development is to try and break the code before adding the next coded segment. In many cases the test tables were completed with purely functional elements with little or no evidence beyond ‘it worked’, this is not sufficient.

Notes for markers

A system which has been thoroughly tested by the developer, but for which there is no evidence that the end user has been involved, cannot score more than 9 or 10 marks here. A system that is tested fairly extensively, but with only valid data and with little or no evidence of end user involvement, may be worth just 7 or 8 marks.

The project report should show evidence of testing during the system development and post development testing. The testing shown as part of the development will be largely credited in the Software Development section. Testing post-development is credited in this section. Candidates may well find it difficult to distinguish clearly between these items and credit should be given for evidence of testing regardless of whether it is in the ‘correct’ section or not.

Teacher advice

Copy your test plan from the Testing Strategy section and add the columns ‘actual outcome’ and ‘end user verification’. Record in the actual outcome what actually happened when the test was performed. Ask the user to sign to confirm the program works as intended.

Check your test plan contains valid data, invalid data and extreme valid data.

See this section as the beta-testing phase, carried out by a test team or the user rather than the programmer. The program should be complete when you tackle this section.

Evidence that the program works to the test table is very important. This evidence can be before and after screen shots, but could also be a video of the game being played (covering all the tests), captured using software like BBFlashback Pro. If you use a video, submit it together with your write-up.

# Documentation

Quality of written communication is assessed in this section. Much of the technical documentation will have been produced as a by-product of design and development work and also as part of writing the report to date. The software solution should also include sufficient on-screen help to enable the user to make use of the system. Some additional supporting documents will be necessary including initial set-up, getting started and troubleshooting guides, to ensure the end user can implement the solution.

8-10 marks – Candidates will provide detailed and accurate documentation. The documentation will be well presented, in a structured and coherent format. The documentation will cover all aspects of the system, with no omissions, including installation, typical use, troubleshooting, and backup. The on-screen help and supplementary documentation makes a complete guide to the solution and is well presented and easy to follow. Subject specific terminology will be used accurately and appropriately. There will be few, if any, errors of spelling, grammar and punctuation.

4-7 marks – Candidates will provide clear documentation. The documentation will be well presented. There is clear on-screen support to enable the end user to use the system. The supporting documentation and on-screen help is well presented and covers most aspects of the system operation with only one or two omissions, e.g. troubleshooting or backup. Some subject specific terminology will be used. There may be occasional errors of spelling, grammar and punctuation.

1-3 marks – Candidates will provide superficial documentation, with weak supplementary user documentation covering few aspects of the system. The information will be poorly expressed and limited technical terms will be used. Errors of grammar, punctuation and spelling may be intrusive.

External Examiner advice

There is a requirement that the system includes good on-screen support for the user. This implies clear screen layouts with informative data entry screen, informative error messages, possibly context sensitive help or online help files. This will need to be supplemented with those essential ‘getting started’ guides we all get with new software. The marks are awarded for the quality of the documentation and the ease with which the end user can get to grips with the system.

Notes from previous years from the Chief Examiner

June 2011 Comments:

Candidates are generally providing on screen help but not so commonly providing evidence of this for the moderator. On screen help is the major element for the documentation mark and must be evidenced.

Notes for markers

If the student produces a system with poor on-screen support and a detailed printed user guide then very few marks are available. With no on-screen help and a detailed printed documentation then we are limited to 2 or 3 marks. If there is some useful on-screen help and detailed printed documentation then 5 marks become available.

Teacher advice

This section is the user manual provided with your solution to the user.

On-screen help is also essential for all projects.

In a game scenario, maybe a help screen shows the points that can be gained, the key controls, the background story etc. Game screens may also show the number of lives a player has, their score, time available etc.

In other types of projects, this may include tool-tips, or labels explaining what the user has to do and pop-up message boxes.

Games also include small getting started guides. Study those from Wii, DSi, PS3, PSP or Xbox games inside the case next to the disk. You should include one of these and follow their format almost exactly.

The quality of the writing is very important in user guides. Make sure you check the spelling of all your work, including your report write-up.

# Evaluation

## Discussion of the degree of success in meeting the original objectives

This discussion should demonstrate the candidate’s ability to evaluate the effectiveness of the completed system. The original objectives stated in the requirements specification should be matched to the achievements, taking into account the limitations. User evaluation is also essential and should arise from direct user evaluation.

3-4 marks – A full discussion, taking each objective in the Nature of Solution section and explaining the degree of success in meeting them (including where in the project evidence can be found to support this), or reasons why they were not met.

1-2 marks – Some discussion about a number of objectives, but some omissions or inadequate explanation of success or failure.

0 marks – no discussion present.

External Examiner advice

Candidates need to provide an evaluation of their system, identifying any issues and how these were dealt with or why they were not met. It is important that the discussion covers the requirements specification. The student should also ensure that they have identified the evidence in the report to support any assertions made.

The evaluation is also an opportunity to tell the moderator how clever they have been by highlighting the best bits of their system or showing how they dealt with particularly difficult aspects of the development.

Notes from previous years from the Chief Examiner

June 2011 Comments:

The evaluation should cover each of the objectives and provide references to the evidence from testing and user testing and feedback to show how well these have been achieved.

The response to the end user feedback follows on from this evaluation and is not based on the end user saying, ‘it works’ but on the detail in their feedback and how the candidate views this and would deal with it if continuing with the project.

This once again leads nicely into the final section of the evaluation.

Teacher advice

Copy the objectives from the Nature of the Solution section and write about each one. Indicate at which stage during development the requirement was met and give a page number to that development in the Algorithms section.

## Evaluate the user’s response to the system

It is important that the user is not assumed to be an expert in computer jargon, so some effort must be made to ensure that the system is user-friendly. It will be assumed that the user will have considerable knowledge of the underlying theory of the business or area being computerised. Clarity of menus, clear on-screen help and easy methods of inputting data are all examples of how the system can be made user-friendly. Here marks are awarded for the degree of satisfaction that the user indicates in the acceptance procedure. Could the system or its results be used? Was the system specification achieved? Do any faults still exist? The candidate should evaluate the user’s response to the final version of the system.

3 marks – The user indicates that the system could be used but there are some faults which need to be rectified. The candidate provides a detailed discussion of how these inadequacies may be dealt with.

OR

A fully functional user-friendly system has been produced. The user indicates that the system fully meets the specification in the Nature of the Solution section, and there are no known faults in the system.

2 marks – The system is, in the main, user-friendly, but there is room for improvement (e.g. no on-screen help has been provided). The user indicates that the system could be used but there are some faults which need to be rectified. The candidate has made some limited attempt to discuss how these inadequacies may be dealt with.

1 mark – The system does not meet the design specification and the end user is not able to make use of the system. The candidate briefly discusses these issues in terms of their project management.

External Examiner advice

The end user needs to provide feedback on the system. Simply saying in a letter that the system does everything is not sufficient. The end user should make meaningful comments on:

* The clarity of menus.
* The quality of the on-screen help.
* The ease of inputting data.
* Could the system or its results be used?
* Was the system specification achieved?
* Do any system faults still exist?

Refer to your project diary in this section, as outcomes of meetings will be recorded there.

Teacher advice

A game also requires user feedback. Typically, they may discuss:

* The clarity of menus – navigating to on-screen help and starting the game.
* The quality of on-screen help. Does it include the controls, how to score, how to win, how to lose etc.
* How responsive and easy the controls are. How they may be ported to other platforms, e.g. touch screen, gyro etc.
* The speed of the game, is the pace right? Is it too fast or slow?
* The difficulty of the game, is it too hard, too easy, about right? Does it get harder? Is the challenge increase about right?
* Quality of the graphics and sound.
* Is any artificial intelligence appropriate? Does it play a good game?
* Is it fun? How could it be more enjoyable? What would the sequel include?
* Whether they like the prototype, was the specification achieved?
* Any glitches that still exist that may be fixed in a patch later.

## Desirable extensions

As a result of completing the system, the candidate should identify the good and bad points of the final system, highlighting any limitations and necessary extensions to the system, and indicating how the extensions could be carried out.

3 marks – The candidate clearly portrays the good and bad points of the system indicating the limitations, possible extensions and how to carry out the extensions.

2 marks – The candidate clearly identifies good and bad points and any limitations.

1 mark – The candidate identifies the obvious good points of the system and possibly some bad points or limitations.

External Examiner advice

Any system developed for A level is going to have significant limitations and parts of the original design may have been problematic and not completed or even attempted. The student should take this opportunity to explain these limitations and describe how the system might be modified to include these elements.

Teacher advice

It is very unlikely the system you developed could/would be actually used. Why is that? This is a chance to discuss what limitations your system has. Also discuss what achievements you are proud of.

In the context of a game, you may consider:

* All the aspects that work well: menus, on-screen help, collision detection, retro graphics etc.
* The number of levels you made, could this be extended?
* The quality of the graphics, could it be improved with the help of a digital artist?
* The portability: if you made the game for the i-phone or another platform it is unlikely to work with VB code. Perhaps it would have been better to make it in the target platform development kit? This may have been prohibitively expensive? No development tools in school? That said, if it’s a prototype, why not RAD in VB first for “proof of concept”?
* Perhaps there were better languages to use for game development? C++, C# with XNA or Lua scripting?
* Perhaps there are better techniques within VB extensions? Using DirectX for graphics and DirectSound for sound?

# A Brief note on the format of your f454 project reports

Every single time you hand in your F454 project you must make sure to do the following:

1. Hand in the entire project “so far” in a folder. Not acceptable in loose sheets, plastic wallets or via email / pen drive.
2. At the very front of the project before everything else always include a copy of the “F454 Project Mark Cover Sheet.pdf”. Spare copies found under Teaching Resources/Computing/A2.
3. Your project must have a cover page, which includes:
   1. Title of project
   2. Subject
   3. Your full name
   4. Your candidate number
   5. Centre number and name
   6. See examples in box or in Teaching Resources/Computing/A2
4. Your project must have an automatically generated contents page that works off Heading 1 / Heading 2 styles etc
5. Every page of your project, with the exception of the front cover must have a header and footer which details the following:
   1. Section of project (Definition, I&A, Testing etc)
   2. Your name
   3. Your candidate number
   4. Centre number
   5. Page number
6. **EVERY** heading in your project **MUST** be labelled up to reference the exam mark scheme section it relates to, you **MUST** use labelling which matches the “F454 Project Mark Cover Sheet.pdf” (e.g. Ai, Aii, Bi etc).
7. **You MUST** follow this project advice booklet. How many of you have read it? How many of you have started your project diary and are keeping it going? I  **WILL** want to see this, and it will form an APPENDIX to your final project.

# F454 project hand in checklist

1. Make sure you project as a title page with:
   1. “Title of Project”
   2. The wording F454
   3. Your full name
   4. Your candidate number
   5. Your centre name & number
2. Make sure every section of your project is included and clearly labelled:
   1. ai – definition
   2. aii – investigation & analysis
   3. bi – nature of the solution
   4. bii – algorithms
   5. biii – test strategy
   6. ci – software development
   7. cii – testing
   8. d – documentation
   9. ei – discussion of the degree of success in meeting original objectives
   10. eii – evaluate the users’ responses to system
   11. eiii – desirable extensions
3. Make sure every single page is clearly numbered.
4. Make sure one of your Appendix’s is clearly labelled as “Code Listings”
   1. This should include a full print out of **ALL** code in colour.
5. Make sure you have included a copy of your entire solution on a CD, DVD or Pen Drive and that your documentation section clearly indicates how your program should be run and installed.
6. Make sure **EVERYTHING** is printed and in order, in colour ready for handing in and binding on **Friday 29th April 2016.** Do **NOT** present work to me in folders or plastic wallets.

# Initial Project Ideas

Use this area to record your initial ideas / thoughts for your F454 Coursework project.

Bring this with you to your first lesson in September. Your ideas will be used as the basis for writing your “Project Definition”.

# F454 Computing Coursework Tracking

Bring this along to each check point meeting, it will be filled out with your teacher.

|  |  |
| --- | --- |
| **Check Point Date** | **Notes on progress required for next checkpoint** |
| **04 Jan** |  |
| **18 Jan** |  |
| **08 Feb** |  |
| **22 Feb** |  |
| **07 Mar** |  |
| **21 Mar** |  |



**Year 13 OCR Computing F454 Coursework Guide**

**If found please return to a member of the Computing department.**