This is a United Kingdom government regulated qualification which is administered and approved by one or more of the following: Ofqual, Qualifications Wales, CCEA Regulation or SQA.
CONTENTS

3. Introduction

4. Qualification Suitability and Overview

4. Trainer Criteria

5. SFIA Levels

6. Learning Outcomes

7-21. Syllabus

22. Examination Format

23-24. Question Weighting

25. Recommended Reading

25. Using BCS Books

26. Document Change History
Introduction

Software touches every organisation, in every sector either directly or indirectly through bespoke development, the use of off-the-shelf business solutions or the realisation of an idea. Software developers turn requirements into reality, through a cycle of analysis, planning, building and testing using a range of tools and concepts.

This Level 4 module covers the key concepts, skills and tools required of anyone working in a Software Developer role, to be able to successfully undertake the tasks required for the development of quality software solutions.
Qualification Suitability and Overview

There are no mandatory requirements for candidates to be able to undertake this certificate qualification, although candidates will need a good standard of written English and Maths. Centres must ensure that learners have the potential and opportunity to gain the qualification successfully.

This qualification is suitable for candidates who are looking to progress their career within a data role. It can be taken as a standalone qualification, or in combination with other units and modules as part of a wider programme, such as an Apprenticeship.

This is an occupationally focused qualification which will:

- Test a learner’s applied knowledge, skills and behaviours to a range of scenarios
- Demonstrate a practical understanding of key concepts across the topic areas
- Enable a learner to progress in their career

Candidates can study for this certificate by attending a training course provided by a BCS accredited Training Provider or through self-study.

### Trainer Criteria

It is recommended that to effectively deliver this certification, trainers should possess:

- 10 days training experience or have a train the trainer qualification
- A minimum of 3 years practical experience in the subject area
## SFIA Levels

This module provides candidates with the level of knowledge highlighted within the table, enabling candidates to develop the skills to operate successfully at the levels of responsibility indicated.

<table>
<thead>
<tr>
<th>Level</th>
<th>Levels of Knowledge</th>
<th>Levels of Skill and Responsibility (SFIA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K7</td>
<td>Evaluate</td>
<td>Set strategy, inspire and mobilise</td>
</tr>
<tr>
<td>K6</td>
<td>Synthesise</td>
<td>Initiate and influence</td>
</tr>
<tr>
<td>K5</td>
<td>Analyse</td>
<td>Ensure and advise</td>
</tr>
<tr>
<td>K4</td>
<td>Apply</td>
<td>Enable</td>
</tr>
<tr>
<td>K3</td>
<td>Understand</td>
<td>Apply</td>
</tr>
<tr>
<td>K2</td>
<td>Remember</td>
<td>Assist</td>
</tr>
<tr>
<td>K1</td>
<td>Remember</td>
<td>Follow</td>
</tr>
</tbody>
</table>

### SFIA Plus

This syllabus has been linked to the SFIA knowledge skills and behaviours required at level 4 for an individual working in Programming and Software Development.

#### KSB15:
Taking innovative approaches to problem solving and/or devising inventive and creative solutions.

#### KSB19:
Understanding the needs, objectives and constraints of those in other disciplines and functions.

#### KSB24:
Working collaboratively with others to achieve a common goal.

#### KSC19:
Applying standards, practices, codes, and assessment and certification programmes relevant to the IT industry and the specific organisation or business domain.

#### KSC22:
Methods and techniques for structured reviews, including reviews of technical work products, test plans, business cases, architectures and any other key deliverables.

#### KSC23:
Testing techniques used to plan and execute software tests of all application components (functional and non-functional) to verify that the software satisfies specified requirements and to detect errors. Examples, but not limited to: dynamic testing techniques and test automation techniques.

#### KSC84:
Understanding and application of different development approaches e.g. iterative/ incremental methodologies (Agile, XP, TDD, SCRUM) or traditional sequential methodologies (Waterfall or V-Model) and their energy and resource footprints. Irrespective of development methodology a DevOps approach may also be taken where development and operational staff work collaboratively.
Learning Outcomes

Upon completion of the certificate candidates will be able to demonstrate a practical understanding of:

• How to use the software development lifecycle and the roles of individuals within it
• How to use the project lifecycle and the roles of individuals within it
• How to select and apply a suitable software development methodology
• How to use a combination of software development approaches and concepts to develop software in line with organisational and policies and procedures
• How to build and use algorithms and databases
• How and when to apply different types of testing

Further detail regarding the SFIA Levels can be found at www.bcs.org/levels.
1. The Software Development Lifecycle (SDLC) (10%) (K2)

Learners will be able to:

1.1 Explain the role and function of the Software Development Lifecycle.

Indicative content
41. Software Development Lifecycle. c. Facilitate The Creation of Software.(229,518),(909,544)

Guidance
The Software Development Lifecycle (also known as the systems development lifecycle) provides structure to the management of software, from initial feasibility and requirements identification to ongoing maintenance.

1.2 Describe the seven stages of the Software Development Lifecycle.

Indicative content
a. Feasibility Study. g. Maintenance. b. Requirements Analysis. c. Design. d. Code Development. e. Testing. f. Deployment /Implementation Guidance This lifecycle although appearing Linear/waterfall in nature, can also be applied to Agile environments, where stages may be revisited/looped back to in a more iterative manner, all activities are still undertaken, but not necessarily in this order.(495,633),(932,752)

1.3 Describe the main activities in each stage of the Software Development Lifecycle.

Indicative content
a. SDLC Guidance Candidates should be aware of the main activities undertaken at each stage of the SDLC, including – but not limited to- investigation into the brief to find potential solutions, what the problem and system are and what’s needed to make the change/solve the problem, post-analysis creation of new system, code development, maintenance.(494,784),(932,901)
List the high-level deliverables from each stage of the Software Development Lifecycle.

**Indicative content**

a. Return on investment.
b. Outcomes.
c. Costs and benefits.

**Guidance**

Candidates should be able to recall and identify the deliverables from each stage of the SDLC, for example the feasibility report which outlines the project plan and costings created during the feasibility study, or the full documented analysis of the problem and system created during requirements analysis.
Learners will be able to:

2.1 Describe the duties associated with each of the Software Development roles.

**Indicative content**

a. Requirements engineer.
b. Business analyst.
c. Software designer.
d. Programmer/coder.
e. Software tester.
f. Software release engineer.
g. Technical architect.
h. Domain expert.
i. Independent tester.
j. Product owner.

**Guidance**
Candidate should be able to identify and explain the duties commonly associated with these roles, including identifying the most suitable role to undertake a particular task or responsibility and the relationships/dependencies between roles. Note that the individual job titles may vary in workplaces, and some titles used in an Agile development environment may encapsulate other roles.

2.2 Relate software development roles to the expected involvement in each stage of the SDLC.

**Indicative content**

a. Roles from 2.1.

**Guidance**
From the list provided in 2.1, candidates should be able to identify the roles which are required at each stage of SDLC, what their expected contribution is and their reasoning for investment at that stage.

2.3 Describe the skills required to fulfil each role within the SDLC.

**Indicative content**

a. Leadership.
b. Creativity.
c. Communication.
d. Analytical skills.
e. Team-work.

**Guidance**
Each role and stage of the SDLC require different skills from the teams. Candidates should be able to identify the skills required or in use. The list provided is not exhaustive.
3. The project lifecycle (5%) (K2)

Learners will be able to:

3.1 Describe the phases of the project lifecycle.

**Indicative content**

a. Initiation Phase.
b. Planning Phase.
c. Execution Phase.
d. Termination Phase.

**Guidance**

Candidates should be able to describe the purpose and scope of each stage of the project lifecycle, including activities undertaken at each stage.

3.2 Explain the characteristics of the project lifecycle.

**Indicative content**

a. Cost.
b. Staffing.
c. Risk and uncertainty.
d. Ability to accommodate change.

**Guidance**

Candidates should be aware of how the costs, risks and demands of a project can vary throughout the PLC and factors which could influence these.

3.3 Describe the duties associated with each of the roles in the project lifecycle.

**Indicative content**

a. Project Manager.
b. Project Team Member.
c. Project Sponsor.
d. Executive Sponsor.
e. Business Analyst.

**Guidance**

Candidates should be able to identify and explain the duties commonly associated with these roles at each stage of the project lifecycle including identifying the most suitable role to undertake a particular task or responsibility and the relationships/dependencies between roles.
Indicative content

a. Project life cycle stages and principles.

Guidance
Candidates should be able to relate the project lifecycle to a given situation or task, detailing the roles and activities at each stage in that project.
4. **Software Development methodologies**  
(10% K4)

4.1 Describe the primary characteristics of Software Development methodologies.

**Indicative content**

- Waterfall Development.
- Agile Development.
- DevOps Deployment.
- Rapid Application.
- Behaviour driven development.
- Test driven development.

**Guidance**
Candidates should be able to list and explain the key characteristics of the given methodologies, including their common uses, strengths and weaknesses.

4.2 Compare and contrast the respective strengths and weaknesses of each of the software development methodologies listed in 5.1.

**Indicative content**

- Methodologies from 5.1.

**Guidance**
Candidates should refer to the methodologies listed in 5.1. Candidates should be able to discuss the strengths, weaknesses, differences and similarities in these methodologies and how they may be used to compliment or support one another.

4.3 Describe the circumstances under which the use of a particular software development methodology would be appropriate.

**Indicative content**

- Methodologies from 5.1.

**Guidance**
Candidates should refer to the methodologies listed in 5.1. Candidates should be able to explain and describe when and why a specific methodology would be applied and the expected outcome and output.
5. Software design approaches and solutions
(10% K6)

5.1 Explain the following software design concepts:

Indicative content

a. Abstraction.
b. Control Hierarchy.
c. Data Structure.
d. Information Hiding.
e. Modularity.
f. Software Architecture.
g. Structural Partitioning.

Guidance

Understanding the meaning and application of these concepts are fundamental to the process of software design. Candidates should be able to explain the purpose and practical application of each concept.

5.2 Compare and contrast the respective strengths and weaknesses of each of the software development methodologies listed in 5.1.

Indicative content

a. Compatibility.
b. Extensibility.
c. Fault Tolerance.
d. Maintainability.
e. Modularity.
f. Performance.
g. Portability.
h. Reliability.
i. Reusability.
j. Robustness.
k. Scalability.
l. Usability.

Guidance

All listed characteristics have a varying level of importance in individual circumstances. The candidate must decide and understand which is the most important – what is the greatest risk.

5.3 Describe examples of common software design patterns and frameworks.

Indicative content

a. Adapter.
b. Decorator.
c. Iterator.
d. Observer.
e. Singleton.

Guidance

Candidates should be able to identify and explain these patterns in use, including providing examples of when their use would be required/appropriate.
6. Organisational policies and procedures relating to the tasks being undertaken (10% K2)

6.1 Describe the relationship between policies and procedures and explain how different procedures can implement the same policy.

**Indicative content**

a. Difference between policy and procedure.

**Guidance**

Candidates should be able to understand how policy drives procedures. For example, a policy may state that any code developed should be “clean and maintainable”, and therefore a procedure should exist to detail the acceptable standard of code, any specific organisational requirements and the steps required to build it. Multiple procedures may exist to ensure compliance with a single policy.

6.2 Explain the importance of well-defined policies and procedure for the effectiveness of an organisation’s operations.

**Indicative content**

a. Governance.
b. Legal requirements.
c. Culture.
d. Customer satisfaction.
e. Working standards.

**Guidance**

The need for policy and procedure in an organisation may be driven by a legal need, an industry standard or an organisational preference or style. Each policy or procedure will impact the way a task or role is undertaken and the expected outcome. Candidates should be able to explain each of these and recognise examples of them in practise.

6.3 Discuss the range of policies and procedures that might be implemented in a software development environment.

**Indicative content**

a. Naming conventions.
b. Commenting of code.
c. Source control (committing, pulling, pushing, merging etc).
d. Secure development.
e. Data protection.

**Guidance**

Candidates should be able to provide examples of the types of policies and procedures that are likely to exist in a software development environment and their purpose. This includes, but is not limited to, the examples listed.
7. The principles of algorithms, logic and data structures relevant to software development (15% K4)

7.1 Explain the nature and purpose of an algorithm.

**Indicative content**

- Problem solving.
- Automation.
- Speed of processing.

**Guidance**

An algorithm is a set of instructions provided in order to complete a given task. It is used to assist with processing large volumes of data, solve complex problems and/or automate tasks.

---

7.2 Prepare examples of the use of Sequence, Selection, Iteration and Recursion in an algorithm.

**Indicative content**

- Sequence.
- Selection.
- Iteration.
- Recursion.

**Guidance**

As the building blocks for algorithms, candidates should be able to describe and create examples of these constructs.

---

7.3 Describe the use of abstract data types in the design and analysis of algorithms.

**Indicative content**

- Queue.
- Stack.
- List.

**Guidance**

Candidates should understand and be able to explain that using abstract data means that data can be manipulated using different operations - for example, finding or adding to existing lists.
7.4 Calculate the space and time complexity of an algorithm. Development project.

Indicative content
a. Big O and Little O notation.

Guidance
Candidates should be able to calculate the space and time complexity (memory required to complete, time taken to run) of a given algorithm using data provided.

7.5 Calculate the space and time complexity of an algorithm. Development project.

Indicative content
a. Access individual elements.
b. Subscripts.
c. Searchable - find specific data.

Guidance
Candidates should understand and be able to explain the difference between single and multidimensional arrays including their use and suitability in a specific circumstance.

7.6 Calculate the space and time complexity of an algorithm. Development project.

Indicative content
a. Linked lists.
b. Making changes and additions.
c. Storage requirements.
d. Searching versus subscripts.

Guidance
Candidates should be able to explain the difference between a list and an array, including how and when one may be more suitable than the other, considering their advantages and disadvantages.

7.7 Describe the implementation of a stack and a queue using linked lists and/or arrays. Which a list may be implemented as a linked structure.

Indicative content
a. Stack - push, pop, empty.
b. Queue - add, remove, empty.

Guidance
Candidates should be able to explain the role stacks and queues, including how they are built using a range of lists/linked lists, arrays and pointers.
Describe the implementation of a tree structure and discuss its use in software development.

**Indicative content**

a. Abstract data type.
b. Single element - roots, branches, leaves.
c. Binary trees or N-ARY.

**Guidance**

Candidates should be able to explain the creation and role of a tree structure for storing and finding data, including the role of the roots, branches, and leaves.

Explain how a graph structure can be used to represent directed and undirected graphs and describe the basic operations provided by a graph structure.

**Indicative content**

a. Permits loops.
b. Nodes and relationships.

c. Directed and undirected.

d. Nodes and relationships.

**Guidance**

Candidates should be able to explain the differences between directed and undirected graphs. Candidates should understand the role of the nodes and relationships in the graph structure and how they are represented.

Explain the operation and implementation of common sorting algorithms.

**Indicative content**

a. Bubble Sort.
b. Insertion Sort.
c. Quicksort.

d. List size - volume of data.

**Guidance**

With a number of sorting algorithms available, candidates should be able to explain how each works and its suitability in a given circumstance, considering factors such as time, complexity, and storage requirements.

Explain the operation and implementation of a number of common searching algorithms.

**Indicative content**

a. Linear Search.
b. Binary Search.
c. Tree Search.
d. List size - volume of data.

**Guidance**

Different types of search algorithms exist, and candidates should be able to explain how each works and its suitability in a given circumstance.
Compare and contrast the use of Hash Tables with the search algorithms listed in 9.11.

**Indicative content**

- Speed.
- Collisions.

**Guidance**

Candidates should understand the role of hash tables as a storage space and be able to assess the suitability of each type of algorithm to perform a search. Factors to be considered may include unique or duplicate values, collisions and time.
Describe the use of database software for storing data.

**Indicative content**

- Store and retrieve.
- Run queries.

**Guidance**

Candidates should understand and be able to explain the role of a database as a storage location for data, to which queries may be applied to find, link or manipulate data.

Explain the characteristics of a relational database management system and the nature of Structured Query Language (SQL).

**Indicative content**

- Select/where from statements.
- Based on tables.
- Relational algebra.

**Guidance**

Candidates should be able to describe the key characteristics of a relational database and how it works, as listed. Candidates will be expected to understand the role of SQL in managing data in a relational database.

Compare and contrast the use of relational databases with the use of Not Only SQL (NOSQL) systems.

**Indicative content**

- NOSQL - graphs, documents, key-value pairs.
- SQL - relational database.
  - Scalability, ease of use, performance.
- Singleton.

**Guidance**

Candidates should understand and describe the differences in strengths, weaknesses and suitability of SQL relational databases and NOSQL systems, considering the factors listed.
Compare and contrast differing implementations of NOSQL databases.

**Indicative content**

b. Key value.
c. Graph based.

**Guidance**

Candidates should draw comparisons between the suitability, use, strengths and weaknesses of NOSQL databases, from the list provided. Candidates should be able to explain the differences and similarities in the structure and purpose of these database types.
9.1 Produce flowcharts and pseudocode to represent a software design.

**Indicative content**

- Flowchart - flowline, terminal, process, decision, input/output, pre-defined process.
- Pseudocode - sequence, selection and iteration.

**Guidance**

Candidates should be able to build, complete and error check flowcharts and pseudocode. Candidates should expect to be tested on items such as completing a missing element or highlighting errors in existing items.

9.2 Produce a functional specification for a given requirements document.

**Indicative content**

- Functions that a component must perform.
- Descriptions of system requirements, input/output.

**Guidance**

Candidates should be able to build, complete and error check a functional specification. Candidates can expect to be tested on completing partly built specifications and highlighting errors in existing items.

9.3 Produce a technical specification for a given requirements document.

**Indicative content**

- Specific requirements of a project.
- Language, hardware, software, libraries.

**Guidance**

Candidates should be able to build, complete and error check a functional specification. Candidates can expect to be tested on completing partly built specifications and highlighting errors in existing items.
10. The nature of software testing frameworks and methodologies. (10% K4)

10.1 Compare and contrast functional testing methods.

**Indicative content**

- a. Unit Testing including white box, black box, grey box and dry runs.
- b. Integration Testing.
- d. Acceptance Testing.

**Guidance**

Candidates are required to understand the role and purpose of each type of testing, including the stage at which they are implemented.

10.2 Compare and contrast non-functional testing methods.

**Indicative content**

- c. Usability Testing.
- d. Compatibility Testing.

**Guidance**

Candidates are required to understand the role and purpose of each type of testing, including the stage at which they are implemented.

10.3 Compare and contrast commonly used software testing frameworks.

**Indicative content**

- a. Linear Automation Framework.
- b. Modular Based Testing Framework.
- d. Data-driven Framework.
- e. Keyword-driven Framework.

**Guidance**

Candidates should be able to explain the use, suitability, advantages and disadvantages of each of the frameworks listed.
Examination Format

This certificate is assessed through completion of an invigilated online exam which candidates will only be able to access at the date and time they are registered to attend.

<table>
<thead>
<tr>
<th>Type</th>
<th>Digital exam which includes:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• a knowledge-based assessment consisting of 20 questions</td>
</tr>
<tr>
<td></td>
<td>• a set of scenario-driven situational judgement assessments consisting of 4 scenarios each with a set of 5 questions, 20 in total</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Duration</th>
<th>90 minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td>Yes</td>
</tr>
<tr>
<td>Open Book</td>
<td>No (no materials can be taken into the examination room)</td>
</tr>
<tr>
<td>Passmark</td>
<td>26/40 (65%)</td>
</tr>
<tr>
<td>Delivery</td>
<td>Digital format only</td>
</tr>
</tbody>
</table>

Adjustments and/or additional time can be requested in line with the BCS reasonable adjustments policy for candidates with a disability, or other special considerations including English as a second language.
Question Weighting

Each major subject heading in this syllabus is assigned a percentage weighting. The purpose of this is:

1. Guidance on the proportion of content allocated to each topic area of an accredited course.
2. Guidance on the proportion of questions in the exam.

<table>
<thead>
<tr>
<th>Syllabus Area</th>
<th>Question type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Understand all stages of the software development life cycle (SDLC).</td>
<td>Multiple choice/Scenario based/Written answer 10%</td>
</tr>
<tr>
<td>2. Understand the range of roles and responsibilities within the software development lifecycle (SDLC).</td>
<td>Multiple choice/Scenario based/Written answer 10%</td>
</tr>
<tr>
<td>3. Understand the roles and responsibilities of the project life cycle within your organisation.</td>
<td>Multiple choice/Scenario based/Written answer 5%</td>
</tr>
<tr>
<td>4. Understand the range of software development methodologies.</td>
<td>Multiple choice/Scenario based/Written answer 10%</td>
</tr>
<tr>
<td>5. Understand software design approaches and patterns and be able to identify reusable solutions to commonly occurring problems.</td>
<td>Multiple choice/Scenario based/Written answer 10%</td>
</tr>
<tr>
<td>6. Understand organisational policies and procedures relating to the tasks being undertaken, and when to follow them.</td>
<td>Multiple choice/Scenario based/Written answer 10%</td>
</tr>
</tbody>
</table>
7. Understand the principles of algorithms, logic and data structures relevant to software development. Multiple choice/Scenario based/Written answer 15%

8. Understand the principles and uses of relational and non-relational databases. Multiple choice/Scenario based/Written answer 10%

9. Understand the nature of software designs and functional/technical specifications. Multiple choice/Scenario based/Written answer 10%

10. Understand the nature software testing frameworks and methodologies. Multiple choice/Scenario based/Written answer 10%

Total 100%
Recommended Reading

The following titles are suggested reading for anyone undertaking this award. Candidates should be encouraged to explore other available sources.

<table>
<thead>
<tr>
<th>Title</th>
<th>Software Development in Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>Bernie Fishpool, Mark Fishpool</td>
</tr>
<tr>
<td>Publisher</td>
<td>BCS</td>
</tr>
<tr>
<td>Publisher Date</td>
<td>August 2020</td>
</tr>
<tr>
<td>ISBN</td>
<td>1780174977</td>
</tr>
</tbody>
</table>

Using BCS Books

Accredited Training Organisations may include excerpts from BCS books in the course materials. If you wish to use excerpts from the books you will need a license from BCS. To request a license, please contact the Head of Publishing at BCS outlining the material you wish to copy and the use to which it will be put.
Document Change History

Any changes made to the syllabus shall be clearly documented with a change history log. This shall include the latest version number, date of the amendment and changes made. The purpose is to identify quickly what changes have been made.

<table>
<thead>
<tr>
<th>Version Number</th>
<th>Changes Made</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td></td>
</tr>
</tbody>
</table>
CONTACT

For further information please contact:

BCS
The Chartered Institute for IT
3 Newbridge Square
Swindon
SN1 1BY

T +44 (0)1793 417 445

www.bcs.org

© 2021 Reserved. BCS, The Chartered Institute for IT

All rights reserved. No part of this material protected by this copyright may be reproduced or utilised in any form, or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system without prior authorisation and credit to BCS, The Chartered Institute for IT.

Although BCS, The Chartered Institute for IT has used reasonable endeavours in compiling the document it does not guarantee nor shall it be responsible for reliance upon the contents of the document and shall not be liable for any false, inaccurate or incomplete information. Any reliance placed upon the contents by the reader is at the reader’s sole risk and BCS, The Chartered Institute for IT shall not be liable for any consequences of such reliance.