SQA PDA (Level 8) in Software Development: Training and Assessment Plan

Purpose

This is your training and assessment plan for the Professional Development Award: Software Development at Level 8 (GL1348) comprised of the following SQA National Units:

- Software Development: Analysis & Design (HA4D 35)
- Software Development: Implementation & Testing (HA4G 35)
- Software Development: Project (HA4K 35)

You need to complete all the above Units to gain the PDA Award. You can use this plan to keep track of your progress. You should become familiar with each Unit's Outcomes, knowledge and skills, what is to be assessed and how it is to be assessed. In order to produce the evidence required for each Unit, you will need to undertake a number of assessment activities including oral or written tests and completing practical tasks.

You will be assigned an Assessor who will review your progress at regular intervals and facilitate assessment activities. They will also be responsible for judging your evidence, performance and understanding of the Unit outcomes, knowledge and skills.

Assessment & Marking

Evidence for each Unit is marked on a pass/fail basis and you must successfully complete all three Units to achieve the PDA.

You may complete some of your assessment activities here at CodeClan under controlled conditions. Other times you may be submitting work as evidence which may have been completed outside the centre. In this case your Assessor will ask you to complete an Authenticity Statement and may follow this up with direct questions about your work.

If you do not achieve a 'pass' for a Unit, or specific Outcome(s), you will be offered a chance for re-assessment, or opportunity to provide alternative evidence of achieving the standard(s). Your Assessor can give you support and guidance to help you consolidate your learning prior to any re-assessment activity.

Equality and Inclusion

The Unit Specifications for the PDA have been designed to ensure that there are no unnecessary barriers to learning or assessment. This plan can be individualised. If you require different assessment arrangements to meet specific additional support needs, please discuss this with your Assessor. You will need to complete the Additional Support Needs Questionnaire prior to undertaking any assessment activity.

Mandatory Policies & Procedures
Learning and assessment of the PDA will take place within the context of the CodeClan course at our training centre. You must adhere to the policies and procedures outlined in the Student Handbook for the CodeClan course and in the Student Code of Conduct. In addition the following policies (Appendix A) are applicable:

- Assessment Statement
- Complaint & Grievance
- Plagiarism & Malpractice
- Assessment Appeals

If you have any questions about the PDA, eligibility, assessment arrangements or any other aspect of this qualification, please speak to a member of staff. If you wish further information about assessment arrangements, you may contact SQA's Assessment Arrangements team on 0345 213 6890 at aarequests@sqa.org.uk

Evidence

NB. You will need to consult with your Assessor as to the minimum requirements to meet the standards.

Analysis & Design Unit, Level 8 (HA4D 35)

Evidence of cognitive competence for Outcomes 1, 2, 3 and 4 will take the form of oral or written tests where definitions, descriptions and explanations are required. Pass rate = 60% (or higher).

Evidence of practical competence for Outcomes 2, 3 and 4 will be demonstrated in the application of object-oriented programming techniques to analyse requirements and design and model software solutions as required. The same program could be used for all three Outcomes (2, 3, 4) or a different program could be used for each Outcome. You may also gather evidence from other programs where required.

Analysis & Design Unit, Level 8: Assessment Plan			
Outcome	Knowledge & skills		Assessment method/ Evidence
1. Describe the use of analysis and design techniques in the software development process.	1.1	Describe conventional and contemporary approaches to software development	Cognitive tests
	1.2	Describe analysis and design tools and models	
	1.3	Describe the Waterfall Development Approach	
	1.4	Describe the Agile Development approach	
2. Define software requirements using object-oriented analysis techniques.	2.1	Define requirements using common models (use cases, object models)	- Use Case diagram(s) - (used to define program requirements) - Object diagram(s) (developed from Use Case diagrams) - Class diagram(s) (developed from Use Case diagrams) - Activity Diagram(s): developed from Class diagram(s) and Object diagram(s)
	2.2	Identify objects and group them into classes	
	2.3	Specify object internals (attributes)	
	2.4	Specify object interaction	
	2.5	Specify object behaviour	

3. Design software solutions using object-oriented techniques.	3.1	Produce a functional design solution using object-oriented modelling techniques.	- Object diagram(s) (developed from Use Case diagrams) - Class diagram(s) (developed from Use Case diagrams) - Activity Diagram(s): developed from Class diagram(s)
	3.2	Map technology-independent concepts onto implementing classes and interfaces to produce a model of the solution domain.	- Object diagram(s) (developed from Use Case diagrams) - Inheritance diagram(s) developed from Class Diagram and Object Diagram)
	3.3	Take account of implementation constraints (hardware and software platforms, performance requirements, persistent storage and transactions, usability, budgets and time limitations).	- Implementation Constraints Plan(s): (developed from Class Diagrams, Object Diagrams and Activity Diagrams)
4. Model software solutions using object-oriented techniques.	4.1	Use an object-oriented modelling language to model solutions.	Use Case diagram(s) - (used to define program requirements) - Object diagram(s) (developed from Use Case diagrams) - Class diagram(s) (developed from Use Case diagrams) - Activity Diagram(s): developed from Class diagram(s) and Object diagram(s)
	4.2	Model dynamic behaviours (business processes, use cases, sequence, activity, statechart diagrams)	- Use Case diagram(s) - (used to define program requirements) - Activity Diagram(s): developed from Class diagram(s) and Object diagram(s)

	4.3	Model static structures (classes, class diagrams, attributes, operations, visibility, association, aggregation, inheritance, relationships between classes components)	- Object diagram(s) (developed from Use Case diagrams) - Class diagram(s) (developed from Use Case diagrams) - Inheritance diagram(s) developed from Class Diagram and Object Diagram)

Implementation & Testing Unit, Level 8 (HA4G 35)
Evidence of cognitive competence for Outcomes 1, 2, 3 and 4 will take the form of oral or written tests where definitions, descriptions and explanations are required. Pass rate = 60% (or higher).

Evidence of practical competence for Outcomes 2, 3 and 4 will be demonstrated in the application of object-oriented programming techniques, algorithms, data structures and testing approaches to specific problems. The same program could be used for all three Outcomes (2, 3, 4) or a different program could be used for each Outcome. You may also gather evidence from other programs where required.

Implementation & Testing Unit, Level 8: Assessment Plan			
Outcome	Knowledge & skills		Assessment method/ Evidence
1. Describe structured programming constructs	1.1	Describe structured programming constructs	Cognitive tests
	1.2	Describe simple data types, data structures and algorithms	
	1.3	Describe basic software testing methods	
	1.4	Describe contemporary programming paradigms	
2. Apply object-oriented programming concepts.	2.1	Write programs constructed from objects and classes	Examples of: - one or more classes in one program. - one or more objects in one program -the object(s) calling a method *The evidence for 2.1 can be covered by the evidence in sub-section 2.2*

	2.2	Create new classes by inheriting properties and methods from existing classes.	Examples of: - a second class that inherits both properties and methods from a first class (this can be used to cover 2.1 or can inherit from Class evidenced in 2.1) - one or more objects in one class that inherits properties and methods from another - the object(s) calling an instance of a method that was inherited from another class
	2.3	Hide internal workings of objects by encapsulation.	Example of: - encapsulation in a program
	2.4	Create a single interface to entities of different types by means of polymorphism.	Example of: - polymorphism
3. Construct programs that make use of algorithms and data structures.	3.1	Create data structures.	Examples of: - one or more arrays used within a program - one or more hashes used within a program
	3.2	Select or construct algorithms to traverse, sort and search data structures	Example of: - one or more function/algorithm designed to sort data - one or more function/algorithm designed to search data.
	3.3	Carry out operations on data structures	Example of: - the result of one or more function/ algorithm designed to sort data - the result of one or more function/algorithm designed to search data.
4.1 Test programs using a range of approaches	4.1	Carry out static testing (verification).	Practical Tests - practical competency assessed See Testing Activity A
	4.2	Carry out dynamic testing (validation).	
	4.3	Carry out unit testing	Practical Tests - practical

	4.4	Carry out integration testing	competency assessed See Testing Activity B
	4.5	Check that software meets specified requirements prior to User Acceptance Testing	

Project Unit, Level 8 (HA4K 35)
Evidence of cognitive competence for Outcomes 1, 2, 3 and 4 will take the form of oral or written tests where definitions, descriptions and explanations are required. Pass rate = 60% (or higher).

Evidence of practical competence for Outcomes 2, 3 and 4 will be demonstrated in the application of object-oriented programming techniques, algorithms, data structures and testing approaches to specific problems. The same program could be used for all three Outcomes (2, 3, 4) or a different program could be used for each Outcome. You may also gather evidence from other programs where required.

Project Unit, Level 8: Assessment Plan			
Outcome	Knowledge & skills		Assessment method/ Evidence
1. Plan the development of a moderately complex software product.	1.1	Apply contemporary development approach	Evidence for this outcome can be gathered over the planning and design stages of the project. - Project Brief + breakdown - Acceptance Test Plan - Wireframe - Object Diagrams (These can be created in conjunction with Class Diagrams and Use Case Models, however is not required to meet Outcome (Outcome 2.3).
	1.2	Gather requirements information	
	1.3	Prioritise requirements	
	1.4	Validate acceptance criteria for product	
	1.5	Outline test plan	- Acceptance Test Plan - Sitemap
2. Design the structure of a moderately complex software product using object-oriented programming techniques.	2.1	Produce object diagrams	- Object Diagram(s):
	2.2	Produce system interaction diagrams	- Sequence Diagram - Collaboration Diagram
	2.3	Produce wireframe designs	- Wireframe designs
	2.4	Write pseudocode	
	2.5	Select algorithm	Examples of: - algorithms
3. Develop a	3.1	Build working software	Examples of:
	3.2	Apply object-oriented programming to meet design	

moderately complex software product using an object-oriented programming language.	3.3	Structure code idiomatically	- Pseudocode - Algorithms - User Input with Results - Interaction with Data Persistence - Results of Interaction with Data Persistence
	3.4	Accept user input	Example(s) of User Input:
	3.5	Process input according to design requirements	
	3.6	Interact with data persistence	Example(s) of Interaction with Data Persistence:
	3.7	Output results and feedback to user	Example(s) of Results of Interaction with Data Persistence:
4. Test the operation and acceptance of a moderately complex software product.	4.1	Check operation of code using a range of techniques	- Acceptance Criteria and Test Plan: - Bug Tracking Report: developed over the course of the project - -Example(s) of Testing:
	4.2	Diagnose causes of errors	
	4.3	Correct identified errors	
	4.4	Ensure acceptance criteria are met	
	4.5	Measure coverage of tests	

Assessment Plan Agreement

Candidate Name:			
Cohort:		**SCN ref:**	
Assessor:			

Review 1	**Date:**
Topics: Review ASN form and agree/personalise Assessment Plan	
Introduced to Professional Development Award - structure, processes and contents of course and award discussed. Additional Comments:	

Review 2	**Date:**
Topics: Assess progress and evidence gathering	
Evidence gathered reviewed and discussed. Advice given on how to proceed. Discussed structure of PDA going forward. Additional Comments:	

Review 3	**Date:**
Topics: Assess progress and evidence gathering	
Evidence gathered reviewed and discussed. Advice given on how to proceed. Discussed structure of PDA going forward. Additional Comments:	

Review 4	**Date:**
Topics: Assess progress and evidence gathering	
Final review - work reviewed and guidance given on individual unit evidence. Timetable for final week explained, as well as submission, marking and verification processes. Additional Comments:	

▢ I have completed an Additional Support Needs questionnaire.
▢ I have read and agree with the mandatory policy and procedures documents
▢ I agree with the Assessment timetable as outlined without any amendments:

Assessor signature:	Candidate signature:
Date:	Date:

OR

▢ I have discussed with my Assessor the following amendments to my Assessment Plan:

Personalised Assessment Plan	
Agreed alterations to Assessment timetable/activities:	
Candidate signature:	Assessor signature:
Date:	Date: