VCE Computing: Software Development: Administrative information for School-based Assessment in 2016

School-assessed Task

The School-assessed Task (SAT) for contributes 30 per cent to the study score.

Teachers will provide to the Victorian Curriculum and Assessment Authority (VCAA) a score against each criterion that represents an assessment of the student's level of performance for Unit 3 Outcome 2 and Unit 4 Outcome 1. For Unit 4 Outcome 1 the recorded scores must be based on the teacher's assessment of the student's performance according to the criteria on pages 6-9. This assessment is subject to the VCAA's statistical moderation process.

The 2016 VCE Computing: Software Development assessment sheet on page 11 is to be used by teachers to record the SAT score. The completed assessment sheet for each student's SAT must be available on request by the VCAA. Student scores for each outcome will be submitted separately via VASS. The score for Unit 3 Outcome 2 must be submitted no later than 15 June (SIAR 1). The score for Unit 4 Outcome 1 must be submitted no later than 10 October (SIAR 3).

The performance descriptors for the assessment criteria are published annually on the Computing: Software Development study page of the VCAA website. The performance descriptors for the assessment criteria 5 to 8 (Unit 4 Outcome 1) are included in this document. Details of authentication requirements and administrative arrangements for School-assessed Tasks are published in the *VCE and VCAL Administrative Handbook 2016*.

The School-assessed Task has five components. They relate to:

- Unit 3 Outcome 2 (three components)
- Unit 4 Outcome 1 (two components).

Unit 3

Analysis and design

Outcome 2

Analyse and document a need or opportunity, generate alternative design ideas, represent the preferred solution design and formulate a project plan for creating the solution.

Nature of task

- An analysis that defines the requirements, constraints and scope of a solution in the form of a software requirements specification
- A folio of two to three alternative design ideas and the detailed design specifications of the preferred design
- A project plan (Gantt chart) indicating times, resources and tasks.

Scope of task

Analysis: software requirements specification

Students must identify a need or opportunity for a software solution created through the use of a programming language. It needs to be a real-world problem or opportunity, not a problem determined by the teacher. Students should engage in the activities of the Analysis stage of the problem-solving methodology and document their findings in the form of a software requirements specification (SRS). The SRS should be assembled as a formal document that addresses the purpose and environment (audience and technical characteristics) of the software solution and details of the solution requirements, including constraints and scope. Typically the analysis tools (use case diagram, context diagram, data flow diagram) would form an appendix to the SRS. There is no prescribed template for the SRS, hence flexibility exists in its presentation.

Teachers should approve each student’s need or opportunity before they conduct an in-depth analysis. The performance descriptors at the highest level for criteria 1 and 2 should be used to help make this decision.

Folio

The folio will comprise design ideas, student-generated evaluation criteria and the detailed specifications of the preferred design.

Students should generate two or three alternative design ideas for their solution. These ideas do not have to be detailed – they represent broad ideas of key features and functions of possible solutions. Students apply their criteria to select the design idea that will be developed into a detailed design for the solution, using a set of tools, methods and techniques. The design should be able to meet the functionality and user requirements stated in the SRS. The evidence from this task is assessed through criterion 3.

Project plan

Students prepare a project plan, in the form of a Gantt chart, for the full implementation of the problem-solving methodology (analysis, design, development and evaluation), hence the plan incorporates the requirements of Unit 4 Outcome 1. Students should only be assessed on their ability to document the key concepts and processes relevant to their project; they must not be assessed on their software skills. Students should take account of any assessment dates and other deadlines provided by the teacher; however all other scheduling must be determined by the students. As the project progresses in Unit 3, students should be recording any adjustments to their original plan.

The evidence from this task is assessed mainly through criterion 4.

Unit 4

Software solutions

Outcome 1

Apply stages of the problem-solving methodology to create a solution using a programming language that fulfils identified requirements and assess the effectiveness of the project plan in monitoring progress.

Nature of task

- A software solution that meets the software requirements specification and the results of the useability test.
- An assessment of the extent to which the project plan (Gantt chart) assisted in monitoring project progress in one of the following:
    - A written report
    - An annotated visual plan.

Scope of task

Software solution

Students use the detailed design produced in Unit 3 as the basis for developing a software solution. Students should apply the necessary features of the programming language published in the *VCAA Bulletin*. If the detailed design generated and assessed in Unit 3 is incomplete or contains significant errors and would prevent demonstration of the highest level of achievement on the relevant criteria, students have the opportunity to make adjustments to their design. Teachers can provide feedback on the quality of the designs but the adjustments must be student initiated, not teacher directed. The modified design is not reassessed but this opportunity prevents negative consequential effects in the development stage.

Students must present evidence of the useability test that they designed, conducted and documented. Two or more ‘users’ other than the student must undertake the test. The useability test should address the core features of the solution and these would be reflected in the SRS. Students make any necessary adjustments to their solution, and evaluate its efficiency and effectiveness based on the evaluation criteria generated in Unit 3. The evidence from this task is assessed through criteria 5, 6 and 7.

Assessment of project plan

Students are required to explain, either through annotations or a report, how their plan assisted in monitoring the progress of the project. Students can use a variety of ways of showing adjustments on their plan. In addition to acknowledging adjustments, students must explain the usefulness of the plan in monitoring their progress through the stages of the problem-solving methodology. The evidence from this task is assessed through criterion 8.

This following rubric is used to assess student achievement on Unit 4 Outcome 1. Teachers assess evidence produced from two tasks against the criteria and performance descriptors to grade achievements on criteria 5 to 8. All criteria are published on the assessment sheet on page 11.

The criteria identify specific characteristics that are used to judge levels of performance against the outcomes. Performance descriptors describe typical evidence associated with five different levels of performance for a criterion (five levels; 10 marks).

Criteria 1 to 4 relate to Unit 3 Outcome 2.

Criteria 5 to 8 relate to Unit 4 Outcome 1.

VCE Software Development: School-assessed Task 2016

Assessment Criteria	Levels of Performance					
	Not shown	1–2 (low)	3–4	5–6 (medium)	7–8	9–10 (high)
5. Skills in organising and managing data and files.		Organises and manipulates limited data through the use of few data structures. Organises limited files in a way to allow quick access to data. Applies few procedures or techniques to secure files or enhance access to required files.	Organises and manipulates some data through the use of some data structures. Inaccuracies exist. Stores and organises some files in a way that allow quick access to some data. Applies some general procedures and techniques to handle and manage some files. Errors expose potential security vulnerabilities and problems accessing required files.	Organises and manipulates some data through the use of some appropriate data structures. Devises a file management plan for storing and organising files that considers some relevant file features that affect the speed of accessing data. Applies some appropriate procedures and techniques to handle and manage most files. Some errors expose potential security vulnerabilities or problems accessing some files.	Organises and manipulates most data through the efficient and effective use of mostly appropriate data structures. Devises an appropriate file management system for storing and organising most files to increase the speed of accessing data. Applies an appropriate set of procedures and techniques to handle and manage the security and accessibility of all files. Minor efficiency or effectiveness errors exist.	Organises and manipulates all data through the efficient and effective use of appropriate data structures. Devises a coherent and appropriate file management system for storing and organising files that maximise ready access to data. Applies systematically a comprehensive set of procedures and techniques to efficiently and effectively handle and manage the security and accessibility of all files.

VCE Software Development: School-assessed Task 2016

Assessment Criteria	Levels of Performance					
	Not shown	1–2 (low)	3–4	5–6 (medium)	7–8	9–10 (high)
6. Skills in using a programming language to develop a software solution that meets specific needs or opportunities		Applies limited processing features of the language to develop a partial solution. Writes limited internal documentation with minimal formatting. Applies limited data validation techniques to check the reasonableness of some input data. Limited evidence of a sorting algorithm in the solution.	Applies some processing features of the language to develop a solution. Inaccuracies and omissions affect the operation of the solution. Applies inconsistently some coding conventions. Writes some formatted internal documentation. Applies some relevant data validation techniques to check the reasonableness of most input data. Writes a solution with a sorting algorithm with some level of complexity.	Selects and applies a range of processing features of the language to develop a solution. Some errors of correctness or completeness exist. Coding conforms to some accepted conventions. Writes some formatted internal documentation with relevant program comments, however, inconsistencies exist. Applies efficiently and effectively some relevant data validation techniques to check the reasonableness of input data. Documents a solution with a mostly complex sorting algorithm.	Correctly selects and applies a wide range of relevant processing features of the language to develop a solution. Minor errors exist. Coding conforms to most accepted conventions. Writes internal documentation that contains relevant program comments and is formatted. Applies efficiently and effectively most relevant data validation techniques to check the reasonableness of input data. Documents a solution with a complex sorting algorithm.	Correctly selects and skillfully applies an extensive range of relevant processing features of the language to develop a correct solution. Coding conforms to all accepted conventions. Writes clearly internal documentation that is comprehensive, contains relevant program comments and is well formatted. Applies efficiently and effectively all relevant data validation techniques to check the reasonableness of input data. Documents the efficient use of a complex sorting algorithm in the solution.

VCE Software Development: School-assessed Task 2016

Assessment Criteria	Levels of Performance					
	Not shown	1–2 (low)	3–4	5–6 (medium)	7–8	9–10 (high)
7. Skills and strategies for testing and evaluating a software solution to meet specific needs or opportunities		Documents limited test data and performs limited tests. Many logic errors are undetected.	Documents generally a small range of test data and applies some suitable testing techniques. Some key logic errors are undetected.	Documents formally a range of test data and applies suitable techniques for detecting most logic errors.	Documents formally a wide range of relevant test data and applies suitable testing techniques. Minor logic errors exist but to not reduce the capacity of the solution to meet is requirements.	Documents formally and clearly a comprehensive range of relevant test data. Applies suitable techniques to detect all logic errors.
		Prepares and conducts, for one user, a brief and general useability test. Uses limited testing techniques.	Prepares a general useability test. Conducts the test using some suitable techniques. Meets user requirements.	Prepares a useability test that is specific to some requirements of the solution. Conducts the test for the required number of users and uses suitable testing techniques.	Prepares a useability test that is appropriate to targeting specific key requirements of the solution. Conducts the test using a range of suitable techniques and meets user requirements.	Prepares a useability test that comprehensively covers all targeted requirements of the solution. Conducts test using a set of suitable techniques and meets user requirements.
		Documents informally limited evidence of the useability test.	Documents general results of the useability test.	Documents accurately most results of the useability test in an organised manner.	Classifies and documents accurately all the results of the useability test.	Documents in detail a logically classified set of accurate results of the useability test.
		Implements few modifications to the solution as a result of useability testing.	Implements some modifications to the solution as a result of useability testing.	Implements most modifications to the solution as a result of useability testing.	Implements key modifications to the solution as result of useability testing.	Implements all necessary modifications to the solution as a result of useability testing.
		Lists few approaches to evaluating the solution, making limited reference to the criteria.	Identifies some feasible strategies for evaluating the solution, making some reference to criteria.	Proposes some feasible strategies for evaluating the efficiency and effectiveness of the solution. Makes reference to most criteria.	Proposes feasible strategies for evaluating the efficiency and effectiveness of the solution. Considers all criteria.	Proposes a coherent set of feasible strategies for evaluating the efficiency and effectiveness of the solution. Strategies align with all criteria.
		Outlines generally how some requirements of the solution are met.	Describes generally how some functional and non functional requirements of the solution are met.	Explains in terms of efficiency and effectiveness how some functional and non functional requirements of the solution are met.	Explains in terms of efficiency and effectiveness how some specific features of the solution meet most functional and non functional requirements.	Explains in terms of efficiency and effectiveness how specific features of the solution meet all functional and non functional requirements.

VCE Software Development: School-assessed Task 2016

Assessment Criteria	Levels of Performance					
	Not shown	1–2 (low)	3–4	5–6 (medium)	7–8	9–10 (high)
8. Understanding of strategies for assessing the effectiveness of the project plan in monitoring progress		Provides limited evidence of adjustments to the initial plan of the project.	Records some adjustments to the initial plan during the progress of most stages of the project. Some errors exist.	Records most adjustments to the initial plan during the progress of the entire project using some appropriate techniques. Minor errors exist but do not reduce the usefulness of the plan.	Records correctly adjustments to the initial plan during the progress of the entire project using appropriate techniques.	Records correctly and clearly all adjustments to the initial plan during the progress of the entire project. Applies a range of appropriate recording techniques.
		Refers to limited approaches for assessing the usefulness of the project plan.	Uses some general strategies for assessing the usefulness of a project plan.	Applies some specific strategies for assessing the usefulness of project plans.	Applies an appropriate set of strategies for assessing the usefulness of the project plan.	Applies systematically a coherent set of strategies for assessing the usefulness of the project plan.
		Refers to limited measures of effectiveness when outlining some general improvements to some parts of the project as a result of using the plan.	Describes generally some factors that influence the effectiveness of the project plan.	Explains the importance of some factors that influence the effectiveness of the project plan.	Explains clearly the importance of relevant factors that influence the effectiveness of the project plan.	Explains clearly and coherently the importance of relevant factors that influence the effectiveness of the project plan.
			Describes how the plan and some of its adjustments assisted in monitoring and improving most stages of the project.	Explains generally how the plan and its adjustments assisted in monitoring the progress of the entire project.	Compares and explains how the specific adjustments recorded as part of the project plan assisted in monitoring the progress of the entire project.	Reports clearly and comprehensively the usefulness of the initial plan and its adjustments in monitoring the progress of the entire project.

Authentication of VCE Computing: Software Development School-assessed Task (SAT)

Teachers are reminded of the need to comply with the authentication requirements specified in the Assessment: School-based Assessment section of the *VCE and VCAL Administrative Handbook 2016* This is important to ensure that 'undue assistance [is] not … provided to students while undertaking assessment tasks'.

Teachers must be aware of the following requirements for the authentication of VCE Computing: Software Development School-assessed Task.

1. The body of work created for the School-assessed Task (SAT) is based on work developed and completed in Unit 3 Outcome 1 and Unit 4 Outcome 1.

2. Teachers are required to fill out the Authentication Record Form and provide the student with feedback on their progress at each observation.

3. Undue assistance should not occur at any time during the development of the body of work, and teachers need to be vigilant. Students are required to demonstrate development of their thinking and working practices. Teachers are reminded that it is not appropriate to provide 'detailed advice on, corrections to, or actual reworking of students' work'.

4. Teachers must sight and monitor the development and documentation of the student's thinking and working practices throughout the unit to authenticate the work as the student's own. Students must acknowledge the source of materials and information used to support the development of their work.

5. Students should be encouraged to complete their work at school. Where students use external service providers, their documentation should demonstrate ongoing progress throughout the SAT.

6. During the generation of the software solution teachers must plan and use observations of student work in order to monitor and record each student's progress as part of the authentication process. Teachers must ensure that all source and reference material, all use of non-school (home, outsourced) resources and any external assistance (for example, tutors) are acknowledged on the authentication form. If a student acknowledges using external resources or receiving external assistance, the teacher should record complete details as an attachment to the Authentication Form.

7. Teachers are reminded that the authentication procedures must be followed for all student work in relation to this SAT. School-based audits include the inspection of authentication records. If authentication records are not provided, the school is automatically audited the following year.

2016

Victorian Certificate of Education
Computing: Software Development Assessment Sheet
School–assessed Task

STUDENT NAME

STUDENT NUMBER □□□□□□□□□

ASSESSING SCHOOL NUMBER □□□□□

Teachers need to make judgments on the student's performance for each assessment criterion. Teachers will be required to choose one number from 0–10 to indicate how the student performed on each criterion with comments, as appropriate.

Assessment criteria **The extent to which the student demonstrates:**	Not Shown (0)	Low (1–2)	(3–4)	Med (5–6)	(7–8)	High (9–10)
1 skills in using analytical tools to depict relationships between data, users and networks	□	□	□	□	□	□
2 interpretation of data to identify and document the software requirements specification	□	□	□	□	□	□
3 skills in generating design ideas and designing preferred solutions	□	□	□	□	□	□
4 understanding of project management concepts and processes	□	□	□	□	□	□
5 skills in organising and managing data and files	□	□	□	□	□	□
6 skills in using a programming language to develop a software solution that meets specific needs or opportunities	□	□	□	□	□	□
7 skills for testing and strategies for evaluating a software solution	□	□	□	□	□	□
8 understanding of strategies for assessing the effectiveness of the project plan in monitoring progress	□	□	□	□	□	□

LEVELS OF PERFORMANCE: TEACHER'S COMMENTS
You may wish to comment on aspects of the student's work that led to your assessment of High, Medium, Low or Not Shown for specific criteria.

If a student does not submit the School–assessed Task at all, N/A should be entered here. □