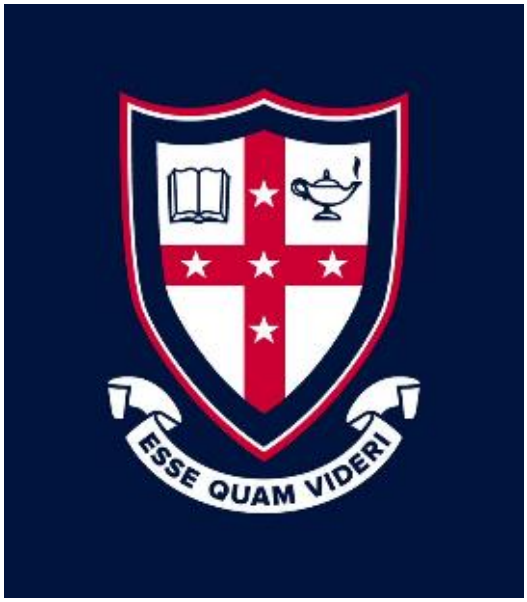

HSC Software Engineering

Software engineering project support

Contents

Task Description....2

Project Description....3

Submission Details....5

Steps To Success....6

What Is The Teacher Looking For?....9

Marking Guidelines....11

Student-Facing Rubric....13

Student Support Material....16

Scenarios For Students Who Do Not Have A Client....16

Scenario 1 – Parking Roster....16

Scenario 2 – Typing Tutor Game....19

Task description

Type of task: develop a software engineering project containing a solution, project documentation and a presentation.

Outcomes being assessed:

A student:

- justifies methods used to plan, develop and engineer software solutions **SE-12-01**
- applies structural elements to develop programming code **SE-12-02**
- analyses how current hardware, software and emerging technologies influence the development of software engineering solutions **SE-12-03**
- evaluates practices to safely and securely collect, use and store data **SE-12-04**
- explains the social, ethical and legal implications of software engineering on the individual, society and the environment **SE-12-05**
- justifies the selection and use of tools and resources to design, develop, manage and evaluate software **SE-12-06**
- designs, develops and implements safe and secure programming solutions **SE-12-07**
- tests and evaluates language structures to refine code **SE-12-08**
- applies methods to manage and document the development of a software project **SE-12-09**

Weighting: 30%

Students identify a real-world problem or opportunity that can be addressed through the development of a software engineering project. Students develop project documentation and engineer a software solution that addresses this real-world problem or opportunity. The software engineering project is presented to the class by simulating a client handover.

Project description

Students find a client for whom they can develop a software solution. Students unable to find a client may use Scenario 1 or Scenario 2 in the Student support material section of this assessment task.

Possible clients include:

- school-based positions: a teacher, librarian or school administrator
- a member of the wider community
  - business owner
  - a non-government organisation that works on various humanitarian and social issues
  - a sporting club, coach, manager or gym
  - a PCYC, YMCA, Scouts, Guides or similar
  - a community group, religious group or hobby group
  - any other organisation or client approved by the teacher.

Students meet and negotiate with their client to:

- define and analyse requirements
- assess scheduling feasibility
- define boundaries
- continually check progress
- document all correspondence.

The software solution:

- demonstrates the design, development and construction of algorithms
- addresses all the clients functional and performance requirements.

The program should also:

- use a language-dependent code optimisation technique
- be configurable in terms of the interface layout, colour scheme, accessibility requirements, legislative and technical requirements as per client needs
- incorporate security elements such as usernames, passwords, basic encryption and decryption as per client requirements.
- include a proposal for an additional innovative solution using a prototype and user interface (UI) design.

Submission details

Students submit:

Component A – System Report documentation

- Complete System Report documentation using the template provided including:
  - process diary
  - client correspondence and feedback
  - Gantt chart
  - use of modelling tools.

Component B – software solution

- Develop, code and upload a software solution for the problem as indicated in the project documentation.
- This will be submitted in the form of a Github link.

Component C – presentation

- Present their software engineering project to peers
- Respond to Q&A.

Steps to success

Table 1 – assessment preparation schedule

Steps	What I need to do
Component A – **Project documentation**	Ongoing completion of the Software engineering project documentation booklet provided that involves 4 key stages. 1. Identifying and defining 2. Research and planning 3. Producing and implementing 4. Testing and evaluating
Identifying and defining	• Begin by defining and analysing the problem requirements including: — demonstrating need(s) or opportunities — assessing scheduling and financial feasibility — generating requirements including functionality and performance — defining data structures and data types — defining boundaries • Provide ongoing screenshots of tools used to develop ideas and generate solutions including: — brainstorming, mind mapping and storyboards — data dictionaries — algorithm design — code generation

Steps	What I need to do
	— testing and debugging — installation — maintenance.
Research and planning	• Apply ongoing project management to plan and conduct the development and implementation of the project including: — scheduling and tracking using a software tool, including Gantt charts — using collaboration tools • Explore communication issues associated with project work including: — involving and empowering the client — enabling feedback — negotiating • Investigate how software engineering solutions are quality assured including: — defining criteria on which quality will be judged — ensuring requirements are met using a continual checking process — addressing compliance and legislative requirements • Demonstrate the use of modelling tools including: — data flow diagrams — structure charts — class diagrams

Steps	What I need to do
	— flowcharts and/or pseudocode
Component B – complete software solution	• Develop and code a software solution for the problem as indicated in the project documentation • Upload the solution.
Producing and implementing	• Design, construct and implement a solution to a software problem using appropriate development approach(es) • Develop, construct and document algorithms • Implement version control when developing a software engineering solution
Testing and evaluating	• Apply methodologies to test and evaluate code • Use a language-dependent code optimisation technique.
Component C – presentation	• Present their software engineering project to peers • Respond to questions and answers.
Testing and evaluating	• Analyse and respond to feedback • Evaluate the effectiveness of a software engineering solution. • Propose an additional innovative solution

What is the teacher looking for?

Students are required to submit 3 components by the end of this project.

Component A – project documentation

The project documentation comprises of the 4 sections, based on the Software Engineering syllabus:

1. Identifying and defining
2. Research and planning
3. Producing and implementing
4. Testing and evaluating

The content of the project documentation should follow the System Report template resource provided.

Component B – software solution

Students, in consultation with their teacher, choose the programming language with which their solution is developed. Examples of languages include the VS suite, Python, Java and C.

Students' programs should (where relevant) incorporate combinations of the following features: control structures, global and local variables, use of simple and structured data types, classes, objects, attributes and methods, functions, modules and libraries and file handling.

Other examples of tools could include web page creation tools, front-end web development frameworks and SQL databases.

The software engineering project should solve a real-world problem or meet a real-world need or opportunity as determined through negotiation with a client. These clients could be associated with health, education, business, entertainment or social. Note that this list is not exhaustive.

Component C – presentation

Students are to develop a 4-minute presentation in the appropriate format (for example, PowerPoint) that demonstrates their software solution to a client (simulated by peers) using a professional language, style and format. The presentation will include screenshots of the

development and documentation of the process. During a 3-minute question and answer session students will answer questions about their project and receive feedback from the audience.

The presentation **must** have:

- **Presentation slides:** A set of slides in the appropriate format (for example PowerPoint) that highlights your solutions features, benefits and challenges using appropriate visual aids such as slides, diagrams and screenshots.
- **Software demonstration:** a live demonstration of your system that allows your teacher and peers to see how you interact with the system and provide feedback.

Marking guidelines

Table 2 – assessment marking guidelines

Grade	Marking guideline descriptors
A	• A student demonstrates extensive knowledge and skill in the application of data, tools and resources to develop and evaluate a fully functional software engineering solution. • A student demonstrates extensive knowledge and skill in the development, testing and implementation of safe and secure code. • A student demonstrates extensive knowledge and understanding of the social, ethical and legal implications of the application of software engineering solutions on the individual, society and the environment. • A student applies comprehensive skills in developing, managing and documenting software engineering projects. • A student communicates logically and effectively using a range of terms, conventions, and methods.
B	• A student demonstrates thorough knowledge and skill in the application of data, tools and resources to develop a functional software engineering solution. • A student demonstrates thorough knowledge and skill in the development, testing and implementation of safe and secure code. • A student demonstrates thorough knowledge and understanding of the social, ethical and legal implications of the application of software engineering solutions on the individual, society and the environment. • A student demonstrates high-level skills in developing, managing and documenting software engineering projects. • A student communicates logically using appropriate terms, conventions and methods.

Grade	Marking guideline descriptors
C	• A student demonstrates sound knowledge and skill in the application of data, tools and resources to develop a software engineering solution. • A student demonstrates sound knowledge and skill in the development, testing and implementation of safe and secure code. • A student demonstrates sound knowledge and understanding of the social/ethical and/or legal implications of the application of software engineering solutions. • A student displays sound skills in developing software engineering projects. • A student communicates using sound terms and conventions and/or methods.
D	• A student demonstrates basic knowledge and skill in the application of data, tools and resources to develop software. • A student demonstrates basic knowledge and skill in the development, testing and implementation of safe and secure code. • A student displays basic skills in developing software. • A student communicates with a basic use of terms.
E	• A student displays limited skills in developing a software engineering project. • A student communicates with limited use of terms.

Common Grade Scale for Preliminary Courses

Student-facing rubric

Table 3 – rubric for Components A and B – project documentation and software solution

Section	Limited	Basic	Sound	High	Outstanding
1. Identifying and defining	Minimal identification of the project's problem and needs, requirements and limitations and applicable tools and processes.	Basic definition and understanding of the project's scope including problem and needs, feasibility, boundaries and applicable tools and processes.	Sound description of the project's requirements and chosen opportunity including problem feasibility, boundaries and applicable tools and processes.	Comprehensive explanation and rationale supporting the project's selection including needs, feasibility, boundaries and applicable tools and processes.	Comprehensive analysis of the problem. Insightful detail explaining the client's functional needs and requirements including feasibility and boundaries.
2. Researching and planning	Minimal understanding of the role of planning and modelling tools in project development.	Basic creation of a project management that uses planning and modelling tools.	A sound project management plan using planning and modelling tools where necessary.	Detailed project management plans including key planning and modelling tools as well as client correspondence.	Comprehensive project management plans. Use of all relevant planning and modelling tools, including responding to client correspondence.

Section	Limited	Basic	Sound	High	Outstanding
3. Producing and implementing	Minimal attempt at a software solution.	An attempt at a software solution that meets basic needs.	A software solution that meets the clients' basic needs. Proposes an addition. Includes an appropriate development approach.	Functional software solution meeting client requirements. Uses code optimisation, demonstrates version control and proposes an additional solution. Includes an appropriate development approach.	Fully functional software solution surpassing client requirements, uses code optimisation, demonstrates version control, proposes an additional innovative solution and an appropriate development approach.
4. Testing and evaluating	Identifies basic testing outcomes without much insight.	Defines effectiveness of the software with some evaluation.	Describes the software's performance with reference to problem definition and need.	Explains the software's effectiveness and necessary improvements based on testing.	Clear evidence of analysis and response to feedback. Evaluates software performance thoroughly, providing detailed feedback for future modifications.

Table 4 – rubric for Component C – presentation

Item	Limited	Basic	Sound	High	Outstanding
Software features, benefits and challenges	Minimal to no description of the software's features, benefits or challenges.	Basic description of the software's features, benefits and challenges. Provides a simple outline of how this was achieved.	Sound overview of the software's features, benefits and challenges with examples. Some explanation of how this was achieved.	Detailed presentation of the software's features, benefits and challenges. Detailed explanation of how this was achieved.	Comprehensive and insightful presentation of the software's features, benefits, and challenges with clear links to improvement and future development. Thorough explanation of how this was achieved.
How the Software meets the project requirements and the user needs	Provides limited to no information to demonstrate how the software meets requirements or client needs.	Provides a basic connection between the software and project/client needs with minimal evidence.	Clearly outlines how the software addresses project requirements and client need, with some examples.	Effectively demonstrates the software's alignment with project requirements and client need, supported by specific examples.	Excellent justification of a comprehensive match between the software and project requirements and client need, with detailed evidence.

Student support material

Scenarios for students who do not have a client

Scenarios where clients could be simulated by the teacher or classmates could include:

- online prefect voting system
- canteen ordering app
- market day business studies website
- school musical booking system
- a biometric roll call attendance system (controversial)
- school jersey and name design app
- sports carnival administration system
- event management system
- an interactive school map and timetable.

These should be expanded into detailed scenarios to provide guidance for students to:

- extract functional requirements
- model with tools
- develop software solutions and innovative prototypes.

The following 2 scenarios demonstrate the depth of detail required to identify, define and analyse to produce and implement a software engineering solution.

Scenario 1 – parking roster

Cranbrook West School has an exceedingly small staff car park. They have a total of 30 spaces allocated for 120 staff members (including Student Administration and Support), split across the 10 faculties of:

- English – 15 teachers
- Mathematics – 13 teachers
- Science – 10 teachers
- Technological and Applied Studies (TAS) – 12 teachers
- Creative and Performing Arts (CAPA) – 7 teachers
- Personal Development, Health and Physical Education (PD/H/PE) – 7 teachers
- History and Languages – 7 teachers
- Human Society and Its Environment (HSIE) – 9 teachers
- Learning Support – 6 teachers
- Student Administration and Support – 34 staff members

The number of teachers for each faculty is given above. There are a total of 10 files provided, one representing each faculty, that need to be read into the program. In each file, there are the names of each of the teachers.

A timetable of how many spaces each faculty gets per day as well as who gets what spot each day for a 2-week cycle needs to be generated.

The program **must**:

- present a form for the user to select the files containing the names for each faculty
- read the names of each of the teachers in each of the faculties from the files
- display which faculty has what spaces on particular days linked to the teachers' names, using a graphical method such as a block of colour or a picture of a car in a car park.
- enable the user to click on 'Monday' and get the allocations for Monday, or 'Tuesday' for Tuesday's allocations and so on.

Note: Teachers do not have to be allocated the same days in a row, nor do they have to have the same parking spots each time.

The program **should** also:

- ensure that 50% of each faculty is allocated a parking spot in the fortnight
- give the user the option to either read the names of each teacher from the files, or to enter each teacher's name manually with the number of days each teacher is in per fortnight
- have a menu system that allows the user to switch between the default mode as specified in the 'must' section, and this optional mode.

The program **could** also:

- ensure that 70% of each faculty is allocated a parking spot in the fortnight
- allow the user to change the number of staff members per faculty so long as the total of 120 staff members is reached. This means that the user should be presented with a form with each faculty name and an associated input box with how many staff members are in each faculty
- increase or decrease the number of total staff members in the school car parking timetable. This should in turn check to see that the total entered does not exceed the number of total staff members.

Scenario 2 – typing tutor game

There is a lot of vocabulary in the Software Engineering course. There are also a lot of large, complex assessment tasks like this one which require a lot of typing. A typing game which builds up the vocabulary of the students studying Software Engineering, as well as keyboard skills, is required to bridge this gap. Technical vocabulary for this typing program is to be derived from the topics of:

- Programming fundamentals – 10 pieces of technical vocabulary
- Object-oriented programming – 10 pieces of technical vocabulary
- Programming mechatronics – 10 pieces of technical vocabulary
- Secure software architecture – 10 pieces of technical vocabulary
- Programming for the web – 10 pieces of technical vocabulary
- Software automation – 10 pieces of technical vocabulary

In addition, definitions and exemplar paragraphs for each of the below NESA key terms need to be included in the typing tutor program.

- Identify
- Define
- Describe
- Explain
- Analyse
- Justify
- Evaluate

These exemplar paragraphs are to be read from data files provided.

Additional exercises that train students on how to place their hands on the home row, feel for the bumps on F and J, punctuation marks and so on must also be included. Some examples of typing programs are given here:

- Speed Typing Online (https://www.speedtypingonline.com/typing-tutor)
- Typing Trainer (https://www.typingtest.com/trainer/)
- Typing Academy (https://www.typing.academy/).

The program **must**:

- read the provided data file for the NESA key term exemplar paragraphs
- progress the students, from training them how to place their hands on the home row to typing in key words, then whole sentences and paragraphs
- calculate their words per minute and percentage (%) of accuracy
- contain at least 5 levels, from absolute beginner at Level 1 to typing in whole sentences and paragraphs at Level 5.

The program **should** also:

- give the administrator of the typing tutor the option of entering the technical vocabulary for each of the given Software Engineering topics into a form, which then gets written into at least one file
- read the technical vocabulary for each topic from the data file generated by the program.

The program **could** also:

- contain up to 10 levels, slowly progressing from absolute beginner at Level 1 to expert at Level 10, where the students are typing in exemplars of code that perform various tasks from Object-oriented programming such as
  - class definition and object instantiation
  - polymorphism
  - inheritance and/or Secure software architecture to create MD5 hashes such as those in Python or in JavaScript
  - AES encryption and decryption ciphers in Python, or PGP encryption and decryption in Python and OpenPGP in JavaScript as typing practice
- read these code exemplars from data files made for the program.

Note: the exemplars given below for each key term are a guide for students to model their responses on.

Table 5 – NESA key terms

NESA key term	Definition	Example
Identify	Recognise and name.	The computer has Microsoft Office 365, Chrome, Internet Explorer and so on.
Define	State meaning and identify essential qualities.	The computer has Microsoft Office 365, which has programmes such as Word, Excel, Access and PowerPoint. Internet Explorer and Chrome are also there, which serve as the main web browsers.
Describe	Provide characteristics and features.	The computer runs Office 365, Chrome, Internet Explorer and so on. They are all applications which are pre-installed on the computer. Chrome and Internet Explorer are web browsers that allow access to Office 365, email, ManageBac and so on. Office 365 is on the computer, though it can also run through a web browser, meaning you can access it at anytime, anywhere.
Explain	Relate cause and effect; make the relationships between things evident; provide why and/or how.	The applications Office 365, Internet Explorer and Chrome are on the computers. Office 365 contains applications such as Word, Excel and Access which are available both at school and online through Internet Explorer and Chrome. Internet Explorer and Chrome are both web browsers, which are not only used to provide access to Office 365, but

NESA key term	Definition	Example
		also to information and YouTube videos to help with assessments from anywhere, at any time.
Analyse	Identify components and the relationship between them; draw out and relate implications.	Office 365, Internet Explorer, and Chrome are on the computers. Internet Explorer and Chrome are both web browsers, which provide a graphical interface to the information and content available online. Some of that content includes, but is not limited to, the Office 365 application suite, allowing students to access Word, Excel, PowerPoint and so on anywhere, at any time. These are accessible inside the Office 365 interface through easily identifiable web-based versions of these applications; the computer-based versions of these applications are more powerful.
Justify	Support an argument or conclusion.	Office 365, Internet Explorer, and Chrome are on the computers. Internet Explorer and Chrome are both web browsers, which provide a graphical interface to the information and content available online. Some of that content includes, but is not limited to, the Office 365 application suite, allowing students to access Word, Excel, PowerPoint and so on anywhere, at any time. This easy access allows students to complete school assessments and tasks without being in any way disadvantaged by needing to have the Office suite installed on their own computers, behaving much like the applications on the desktop machines at school.

NESA key term	Definition	Example
Evaluate	Make a judgement based on criteria; determine the value of.	Office 365, Internet Explorer, and Chrome are on the computers. Internet Explorer and Chrome are both web browsers, which provide a graphical interface to the information and content available online. Some of that content includes, but is not limited to, the Office 365 application suite, allowing students to access Word, Excel, PowerPoint and so on anywhere, at any time. This easy access allows students to complete school assessments and tasks without being in any way disadvantaged by needing to have the Office suite installed on their own computers, behaving much like the applications on the desktop machines at school. The main advantage of this open access is the reduction in gaps between those who have access to the Office suite and those who do not, meaning less excuses for those who may have used this lack of access in the past. However, this also may reduce skill sets or exposure to other application suites such as OpenOffice which some may perceive as a failing of being homogenous.