Assessment of software design and development with a General Purpose Programming Language

This example rubric defines key criteria for:

- user interface design, and the use of relevant tools such as mockups,
- algorithm design, and the use of relevant tools such as flowcharts and pseudocode,
- the development (coding) of the algorithms in a General Purpose Language, such as Python or JavaScript, including testing.

The criteria in this rubric are ***not*** suitable for assessing a complete digital solution.
They do ***not*** address:

- the overall design thinking process and the value of innovation,
- problem definition and identification of solution requirements,
- evaluation of completed solution,
- collaborative work and project management.

ASSESSMENT STANDARD (extracts - Digital Technologies)

Years 5 and 6: **Design** solutions by **developing** algorithms to address the problems. **Incorporate** decision-making, repetition and user interface design into their designs and **implement** their digital solutions, including a visual program.

Years 7 and 8: **Design** user experiences and algorithms **incorporating** branching and iterations, and **test**, **modify** and **implement** digital solutions**.**

Rubrics templates

DIGITAL TECHNOLOGIES HUB

Achievement standard criteria	***E***	***D***	***C***	***B***	***A***
Design user interface *(text only)*	*No demonstrable attention to interface efficiency or effectiveness for the purpose of solution.* *User is presented with no instructions to interact with the solution.* *No response to invalid user input.*	*Interface is inefficient and/or ineffective for the purpose of the solution.* *User is presented with inadequate instructions to interact with the solution.* *Little response to invalid user input.*	*Interface is somewhat efficient and effective for the purpose of the solution.* *User is presented with adequate instructions to interact with the solution.* *Some responses to invalid user input.*	*Interface is efficient and effective for the purpose of the solution.* *User is presented with clear instructions to interact with the solution.* *Some useful responses to invalid user input.*	*Interface is highly efficient and effective for the purpose of the solution.* *User is presented with clear, complete instructions to interact with the solution.* *Appropriate, helpful responses to invalid user input.*
Design user interface *(graphical)*	*No demonstrable attention to interface efficiency or effectiveness for the purpose of solution.* *Mockup missing or incoherent.* *Interface lacks key elements and is inadequate for the intended user.* *When present, colour, placement and font choices do not demonstrate understanding of design conventions.*	*Interface is inefficient and/or ineffective for the purpose of the solution.* *Mockup inadequate for presenting design.* *Elements of the interface are unclear, incomplete and/or unsuitable to the intended user.* *Colour, placement and font choices do not demonstrate understanding of design conventions.*	*Interface is somewhat efficient and effective for the purpose of the solution.* *Mockup used to present design adequately.* *Elements of the interface are adequate in terms of clarity and suitability to the intended user.* *Colour, placement and font choices make the solution adequately useable, and demonstrate basic understanding of design conventions.*	*Interface is efficient and effective for the purpose of the solution.* *Mockup used correctly to present design.* *Most elements of the interface are clear and suitable to the intended user.* *Colour, placement and font choices make for a mostly consistent experience, and demonstrate moderate understanding of design conventions.*	*Interface is highly efficient and effective for the purpose of the solution.* *Mockup used correctly to present design thoroughly.* *All elements of interface are clear, complete and suitable to the intended user.* *Colour, placement and font choices make for a consistent experience, and demonstrate strong understanding of design conventions.*
Design algorithm	*Little or no apparent use of flowchart / pseudocode.* *Algorithm is incomplete or incoherent, demonstrating little or no understanding of sequence.* *Algorithm does not correctly incorporate more than one or two of the following:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Flowchart / pseudocode used sparingly or incorrectly.* *Algorithm is inefficient and/or ineffective, demonstrating limited understanding of sequence.* *Where appropriate, algorithm correctly incorporates some of the following:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Flowchart / pseudocode used adequately, with some conventions followed.* *Algorithm is somewhat efficient and effective, demonstrating adequate understanding of sequence.* *Where appropriate, algorithm correctly incorporates most of the following:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Flowchart / pseudocode used mostly correctly, with most conventions followed.* *Algorithm is efficient and effective, demonstrating moderate understanding of sequence.* *Where appropriate, algorithm correctly incorporates:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Flowchart / pseudocode used correctly, with all necessary conventions followed.* *Algorithm is highly / maximally efficient and effective, demonstrating strong understanding of sequence.* *Where appropriate, algorithm correctly incorporates:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*

Rubrics templates

DIGITAL TECHNOLOGIES HUB

Develop code – overall functionality	*Basic functional code is missing.* *Program cannot run.* *No demonstrable attention given to functional requirements or design.*	*Code is inadequate or contains syntax errors.* *Program is not functional or contains bugs that affect basic functionality.* *Program meets little or no functional requirements.* *Little or no validation code.*	*Code is adequate but not complete, or contains syntax errors.* *Program contains bugs that affect some functionality. It can be run successfully with minor modifications.* *Program meets some functional requirements, reflects design somewhat.* *Validation code sometimes in place.*	*Code is largely complete and free of syntax errors.* *Program is mostly free of bugs, where reasonable. It can be run successfully with specific input.* *Program meets most functional requirements and mostly fulfils design.* *Validation used when necessary, giving some useful feedback to user.*	*Code is fully complete and free of syntax errors.* *Program is free of bugs, where reasonable. It always runs successfully with expected input.* *Program meets all functional requirements and fulfils design.* *Validation correctly used when necessary, giving useful feedback to user.*
Develop code – programming skills	*Few or none of the following skills are utilized correctly:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Some of the following skills are utilized correctly:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*An adequate range of the following skills are utilized correctly:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Most of the following skills are utilised efficiently:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*	*Where appropriate, all the following skills are utilised thoroughly and efficiently:* • *iteration (loops)* • *branching (decisions)* • *variables* • *user input* • *output*
Develop code – readability & internal documentation	*Code is largely incoherent.* *No attention to rules and conventions to maximise code readability.* *Comments are not present or randomly inserted.*	*Code shows no attention to organisation.* *Little or no appropriate rules and conventions followed to maximise code readability, including:* • *tabbing and whitespace* • *naming of variables / functions* *Comments are rare.*	*Code shows limited attention to organisation.* *Some appropriate rules and conventions followed to maximise code readability, including:* • *tabbing and whitespace* • *naming of variables / functions* *Some comments are present.*	*Most code organised clearly and logically.* *Most appropriate rules and conventions followed to maximise code readability, including:* • *tabbing and whitespace* • *naming of variables / functions* *Comments are mostly present where appropriate and are mostly clear.*	*All code organised clearly and logically.* *All appropriate rules and conventions followed to maximise code readability, including:* • *tabbing and whitespace* • *naming of variables / functions* *All comments present where appropriate, thorough and clear.*
Develop code - testing	*No formal testing apparent.* *No evidence of testing.*	*Formal testing sporadic and rare.* *Testing tool (eg. testing table) used incorrectly or sparingly.*	*Formal testing includes some or one of the following (where appropriate):* • *unexpected user input / data* • *out of range data (boundary checking)* • *wrong type data* *Testing tool (eg. testing table) partly complete and used adequately.*	*Formal testing includes most of the following (where appropriate):* • *unexpected user input / data* • *out of range data (boundary checking)* • *wrong type data* *Testing tool (eg. testing table) mostly complete and used correctly.*	*Formal testing includes all of the following (where appropriate):* • *unexpected user input / data* • *out of range data (boundary checking)* • *wrong type data* *Testing tool (eg. testing table) complete and used effectively.*