MODULE P LANGUAGES AND IDES · P4

IDE tools and how they help a programmer: mark scheme

21 marks · Cambridge 0478 syllabus (exams 2026-2028) 4.2.5 (explain the role of an IDE in writing program code and the common functions IDEs provide: code editors, run-time environment, translators, error diagnostics, auto-completion, auto-correction, prettyprint)

How to use. One mark per creditworthy point up to the maximum shown. **/** separates alternative words; **//** separates alternative answers. Words in (brackets) are not needed for the mark. **...** at the start of a bullet means it builds on the bullet before. Credit any other valid answer that shows the same understanding. Do not credit brand names of software. Do not credit functions of an operating system (e.g. memory management) as IDE functions.

QUESTION	ANSWER	MARKS	GUIDANCE
1(a) 0478 4.2.5	Row 2 ticked: Software that provides the functions needed to write, translate, run and test program code in one program	**1**	More than one box ticked = 0. Row 1 is a compiler, row 3 an operating system, row 4 assembly language.
2(a) 0478 4.2.5	One mark for each correct function: Types `REP`, IDE offers `REPEAT`: auto-completion Types `OUPTUT`, IDE changes it to `OUTPUT`: auto-correction Keywords, identifiers and strings in different colours, consistent indentation: prettyprint High-level code converted into machine code: translator Program run inside the IDE, outputs shown for testing: run-time environment	**5**	Accept "compiler" // "interpreter" for translator. Accept "autocomplete", "autocorrect", "pretty print(ing)". Code editor and error diagnostics are not needed. If the same function is written in more than one row, credit only the row where it is correct.
3(a) 0478 4.2.5	Error diagnostics	**1**	Accept "error reporting" // "debugger". Do not credit "translator" / "compiler" / "code editor".
3(b) 0478 4.2.5	Any two from: It identifies / shows the line number (line 03) where the error is // highlights the line with the error ... so the programmer can find the error quickly // knows where to look It describes / explains the error // says a quotation mark is missing ... ... so the programmer knows what to change // to add the missing " (after `gallery` / before the comma) It may suggest a correction It reports the error before the program is run // it can report all the errors found	**2**	max 2. Do not credit "it finds errors" / "tells you there is an error" alone.

QUESTION	ANSWER	MARKS	GUIDANCE
3(c) 0478 4.2.5	Any two from: Prettyprint shows strings / keywords / identifiers in different colours ... the rest of line 03 (`, Gallery`) is shown in the string colour // `Gallery` is not shown in the identifier colour ... so the programmer can see where the string should have ended // that the string has not been closed ... (so) the error can be seen as it is typed // before the program is translated / run Consistent indentation / layout makes the code easier to read // errors easier to see	**2**	max 2. "Makes code look neat" alone is not enough. Credit a general point about colour-coding plus one point applied to line 03.
4(a) 0478 4.2.5 (4.2.3)	Any two from: The program is written in a high-level language // source code The processor / CPU can only execute / understand machine code // binary The translator converts the high-level code into machine code / binary ... ... so the program can be run / tested inside the IDE // without needing a separate translator program	**2**	max 2. Do not credit "to translate the code" alone (repeats the question).
4(b) 0478 4.2.5	Any two from: Lets the program be run / executed inside the IDE // without leaving the IDE Shows the outputs (e.g. the total number of visitors) // the programmer can enter test inputs (e.g. visitor numbers) ... so the actual output can be compared with the expected output Shows errors that happen while the program is running // where the program stops	**2**	max 2. "Tests the program" alone is not enough.

QUESTION	ANSWER	MARKS	GUIDANCE
5(a) 0478 4.2.5	One mark for each correct function (max 3). One mark for each matching explanation of how it helps (max 3). For example: Auto-completion // ... suggests / finishes keywords / identifiers as he types, so code is written faster // with fewer spelling mistakes Auto-correction // ... corrects misspelt keywords / common mistakes automatically, so fewer (syntax) errors reach the translator Prettyprint // ... colours / indents the code, so it is easier to read // errors are easier to spot Error diagnostics // ... show the line / location and a description of each error, so errors are found and corrected quickly Run-time environment // ... runs the program and shows its output inside the IDE, so he can test it without changing to another program (Built-in) translator // ... converts the code to machine code inside the IDE, so he can translate and run with one action // without a separate program Code editor (with line numbers / auto-indent) // ... he can write and change code and find a line given in an error message	**6**	max 6. An explanation scores only if it matches the function given. A function with no explanation scores 1. Listing more than three functions without explanations scores max 3. The role of an IDE (all the functions are in one program, so he does not need to switch between programs) may be credited in place of one explanation mark. Debugging tools (breakpoints, stepping, watching variables) may be credited under error diagnostics.

Total: 21 marks