The Integrated Development Environment

2.5.2 · Algorithms and programming (Paper 2) · revision notes

An Integrated Development Environment (IDE) is software that gives programmers the tools to write, translate, run and debug code in one place. You need to know the common tools (editors, error diagnostics, run-time environment and translators) and explain how each helps. Expect 'describe how this tool helps' questions.

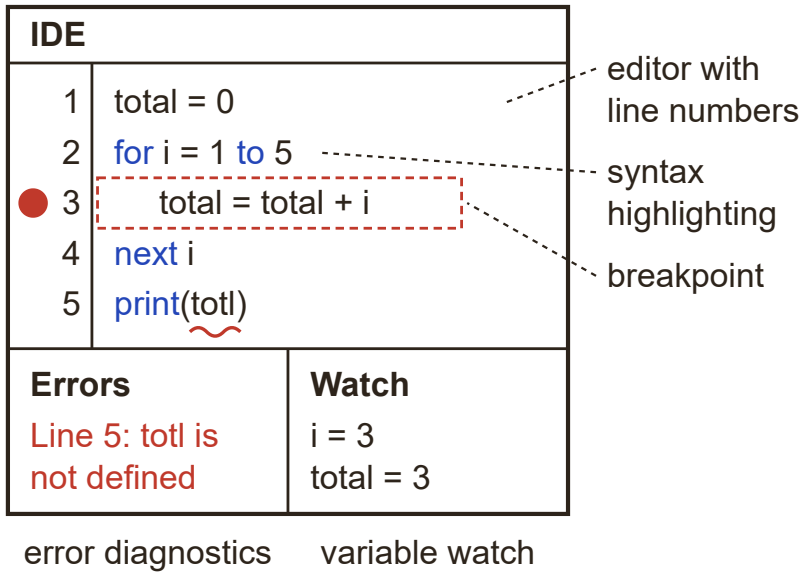
Key facts

- IDE = Integrated Development Environment: tools for writing, running and debugging code
- Error diagnostics: report errors with messages and line numbers
- Translator: built-in compiler or interpreter
- Editor: write and edit code; syntax highlighting, auto-indent, auto-complete, line numbers
- Run-time environment: run and test the program inside the IDE
- Debugging: breakpoints, stepping, variable watch

Notes

The main tools

- **Editor**: where the source code is written and changed. It often has **syntax highlighting** (keywords, strings and comments shown in different colours), **automatic indentation**, **auto-completion** (suggests keywords and variable names as you type), **line numbers** and bracket matching.



The main parts of an IDE

- **Error diagnostics**: detect errors and report them with a message and the line number, and often highlight the line, so errors can be found and fixed quickly.
- **Run-time environment**: lets the programmer run the program inside the IDE to test it, and see its output and any errors that happen while it runs.
- **Translator**: a built-in compiler or interpreter that converts the source code into machine code so it can be run.

Debugging tools

- **Breakpoints**: stop the program at a chosen line, so the programmer can see what has happened so far.
- **Stepping**: run the program one line at a time, to follow exactly which lines run.
- **Variable watch**: shows the values of variables as the program runs, so you can see where a value becomes wrong.
- These tools help find **logic errors**, which the translator cannot detect.

Why use an IDE

- All the tools are in one place, so code is written, run and tested faster.
- Errors are found and fixed more quickly.
- Features such as auto-completion and syntax highlighting reduce typing mistakes.

How to answer each type of question

Identify tools that help find errors

2 to 4 marks | Grade 6

1. Name the tool.
2. Describe how it helps find or fix errors.
3. Repeat for the second tool; make the two tools different.

Example. Identify **two** tools in an IDE that help a programmer find errors, and describe how each one helps. [4]

Model answer.

Error diagnostics (1): highlight the line where there is an error and give a message about it (1).
Breakpoints (1): stop the program at a chosen line so the values of the variables can be checked (1).
Also accept: stepping, runs one line at a time; variable watch, shows how values change.

Explain how an editor feature helps

2 marks | Grade 5

1. Say what the feature does.
2. Say how that helps the programmer: fewer mistakes, faster, easier to read.

Example. Explain how syntax highlighting helps a programmer. [2]

Model answer. Keywords, strings and other parts of the code are shown in different colours (1), so the code is easier to read and mistakes such as a misspelt keyword or a missing quotation mark stand out (1).

Describe how to find a logic error with an IDE

3 marks | Grade 7

1. Pick a place to pause: set a breakpoint near where the error seems to be.
2. Step through the code one line at a time.
3. Watch the variables to see where a value first goes wrong.

Example. A program runs without crashing but calculates the wrong total. Describe how the programmer could use the tools in an IDE to find the error. [3]

Model answer. Set a breakpoint at the start of the code that calculates the total (1). Step through the code one line at a time (1). Watch the value of the total (and other variables) to see the line where it becomes wrong (1).

Don’t lose marks

- Naming a tool without saying how it helps.

- Saying error diagnostics find every logic error: they report errors the translator or run-time environment detects, not wrong results.
- Describing features of the editor when the question asks about finding errors.
- Giving two answers that are really the same tool, such as 'shows errors' and 'highlights errors'.

More tips

Memory tricks

- IDE tools: 'Every Error Rarely Tricks' = Editor, Error diagnostics, Run-time environment, Translator.
- Syntax errors: the error diagnostics find them. Logic errors: you find them with breakpoints, stepping and watches.
- Think of the editor as a word processor made for code.

Exam technique

- Use the OCR names: editor, error diagnostics, run-time environment, translator.
- For each tool, write what it does, then 'so that …' to show how it helps.
- In scenario questions, link the tool to the problem described, such as a wrong total or a crash.

What each grade needs

What you need to be able to do, from the first marks up to the top grade.

Grade 3 — **State what an IDE is**
Software with tools to help programmers write, run and debug programs.

Grade 4 — **Name the main tools in an IDE**
Editor, error diagnostics, run-time environment and translator.

Grade 5 — **Describe features of the editor**
Syntax highlighting, auto-indentation, auto-completion, line numbers.

Grade 6 — **Explain how error diagnostics help**
They highlight errors and give messages with line numbers, so bugs are found quickly.

Grade 7 — **Describe using debugging tools to find errors**
Use breakpoints, stepping and watching variables to find a logic error.

Free notes, Full packs, practice papers and mock exams at chhetriacademy.com. Written for this site; not taken from past papers. Stuck? 1-to-1 tutoring with a free 30-minute trial lesson.