

IGCSE Computer Science — Unit 4.2

Types of programming language, translators and IDEs

Student name: _____

Date: _____

Learning goals (what you should be able to do)

- Explain what **high-level** and **low-level** languages are
- Compare their **advantages** and **disadvantages**
- Explain **assembly language**, **mnemonics**, and the role of an **assembler**
- Describe how a **compiler** and an **interpreter** work, and how they report errors
- Explain the role of an **IDE** and common IDE functions

Language levels

High-level language (HLL) vs low-level language (LLL) comparison table

HLL	LLL
Abstract	
Readable	
Portable	
Translate	

Examples

- HLL examples: _____
(add one more: _____)
- LLL examples: _____
(add one more: _____)

Advantages and disadvantages

HLL	LLL
pros	
cons	

Relative level

- High-level/low-level are _____, but also _____
- For example: _____ is usually considered _____ than _____
- There is no lower level than _____ (binary instructions the CPU executes directly)

Abstraction example: variable names

Complete the idea:

- HLLs let you use _____ (e.g. _____) instead of tracking _____.
- Consequence: programs are usually _____ in an HLL.

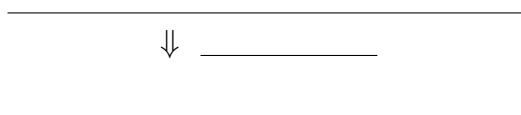
Assembly and the assembler

Key vocabulary

- Assembly language: _____
- Mnemonic: _____
- Assembler: _____

Why an assembler is needed (pipeline)

Fill the missing words:



Assembly depends on the hardware

- Assembly is tied to a CPU's _____ .
 - Different CPUs have different instructions, so they use different _____ .
-
-
-

Translators: compiler and interpreter

Big picture (compare)

Compiler	Interpreter
Translates _____	the Translates and executes _____
Produces an _____	No separate executable _____
Then you _____	Runs _____

Pipelines (fill in)



How errors are reported

- Compiler: can produce an _____ (finds many errors in one go).
- Interpreter: _____ during execution.

Advantages and disadvantages

Compiler	Interpreter
pros	
cons	

Context: “choosing compiler vs interpreter”

- Usually this is really choosing a _____, because many languages are mostly compiled or mostly interpreted.
- Some environments mix approaches (VM/bytecode, packaging, etc.).

Integrated Development Environments (IDEs)

Definition

- **IDE:** _____
- An IDE is not the same as a compiler/interpreter, but it often _____ them.

Common IDE functions (know these)

- | | |
|---------|---------|
| – _____ | – _____ |
| – _____ | – _____ |
| – _____ | – _____ |
| – _____ | |

Write one sentence describing what each does (use your own words):

- Code editor:
- Run-time environment:
- Translator:
- Error diagnostics:
- Auto-completion:
- Auto-correction:
- Prettyprint / formatting:

Exam focus (sentence starters)

Fill the blanks using the words from the unit:

- Low-level languages are _____ to read but allow _____ hardware control.
- High-level languages are _____ to write/debug and are _____.
- Assembly uses _____; an _____ translates it to _____.
- A _____ translates the _____ and can give an _____ for the whole program.
- An _____ translates/executes _____ and _____.

Practice (quick checks)

Definitions

Write short definitions (1 line each):

- High-level language (HLL):

- Low-level language (LLL):

- IDE:

Compare (exam style)

State **two advantages** and **one disadvantage** of HLLs and LLLs:

- HLL:

- LLL:

Your notes

Use this space for anything you struggled with: