Your notes

GCSE Computer Science: revision notes

Programming languages and translators

Contents

1. High-level and low-level languages
2. Translators
3. The IDE
4. Key terms
5. Quick questions
6. What to do next

High-level and low-level languages

- **High-level languages**, such as Python, Java and C#, use English-like keywords. They are easier to write, read and debug, and they are portable: the same code can run on different types of processor. They must be translated into **machine code**.
- **Low-level languages** are close to the hardware. **Machine code** is the binary instructions the CPU runs directly. **Assembly language** uses short mnemonics, such as ADD or LDA, one for each machine code instruction.
- Low-level code gives direct control of the hardware and can be very fast and small, which suits embedded systems. But it is hard to write, and it only works on one type of processor.

Translators

- A **compiler** translates the whole program into **machine code** in one go, making an executable file. It reports all the errors at the end. The executable runs quickly without the **compiler**, and the source code doesn't need to be shared.
- An **interpreter** translates and runs the program one line at a time. It stops at the first error, which helps debugging, but the program runs more slowly and the **interpreter** is needed every time it runs.
- An **assembler** translates **assembly language** into **machine code**.

The IDE

- An integrated development environment (IDE) is software for writing programs. It usually has an editor (with syntax highlighting and automatic indentation), error diagnostics, a translator and a run-time environment.
- Debugging tools include **breakpoints** (pausing the program at a chosen line), stepping through the code line by line, and watching the values of variables.

Key terms

High-level language	A programming language with English-like keywords, such as Python.
Low-level language	A language close to the hardware: machine code or assembly language.
Machine code	Binary instructions that the CPU can run directly.
Assembly language	A low-level language that uses short mnemonics, such as ADD.
Compiler	A translator that converts a whole program into machine code in one go.
Interpreter	A translator that translates and runs a program one line at a time.
Assembler	A translator that converts assembly language into machine code.
IDE	Integrated development environment: software with tools for writing and testing programs.
Breakpoint	A point where a program pauses so the programmer can inspect it.

Quick questions

Cover the answers and test yourself: remembering is what makes it stick.

1. Which type of language does the CPU run directly?
2. What does a compiler do?
3. What does an interpreter do?
4. What does an assembler do?
5. Which of these is an advantage of a high-level language?
6. Why might a programmer use assembly language?
7. Why is an interpreter useful while a program is being developed?
8. Which translator produces an executable file?

Answers

1. Machine code
2. Translates the whole program into machine code in one go
3. Translates and runs a program one line at a time
4. Translates assembly language into machine code
5. It is easier to read, write and debug
6. For direct control of the hardware and very efficient code
7. It stops at the first error, which makes debugging easier
8. A compiler

Your notes

What to do next

Practise 12 questions on this topic

Try 5 exam questions, with mark schemes

Read the lesson, one step at a time

Free revision for GCSE and KS3 at lycolearn.co.uk. Some notes are for one exam board only, and say so in brackets.

Your notes