MODULE 1 OF 7

Why Is My Child Learning This?

The honest case for why coding, data, and digital skills matter more than almost anything else your child will study.

A NOTE BEFORE YOU BEGIN

Hello, and thank you for reading this. If you've picked up this guide, chances are you've sat in a parents' evening and heard words like "Python", "algorithms", "data literacy", or "computational thinking" — and nodded along while quietly having no idea what any of it meant. That's completely normal. The school curriculum has changed faster than anyone thought to explain it to parents. This guide is my attempt to fix that.

I'm a quantitative analyst at an investment bank — I work with data and code every day. I also know what it's like to try to explain what I do to people who don't have a technical background, because I do it all the time. Every module in this guide is written for you, not for engineers. There's no jargon. There's no assumption that you know anything about technology. And there's no judgment — these are genuinely confusing topics, and nobody was supposed to already know them.

Each module ends with one practical activity — something you can do or say that week. You don't need to read everything at once. Dip in and out as topics come up with your child.

Why Is My Child Learning This?

Let's start with the most honest version of the question you might actually be asking: *is this really necessary?* After all, you got through school and built a career without learning to code. Your parents managed perfectly well without understanding data. Why is this suddenly so urgent?

It's a fair question, and it deserves a straight answer rather than vague claims about "the future of work". This module gives you that answer — and by the end of it, you'll have a clear picture of why schools are teaching this, what's genuinely at stake, and how to think about it without either panicking or dismissing it.

The world your child is growing up into

Think about the jobs that didn't exist ten years ago. Social media manager. App developer. Data analyst. UX designer. Podcast producer. Drone operator. None of these were careers when most of us were at school, and yet millions of people do them now. The pace of that change is accelerating, not slowing down.

Economists at the World Economic Forum estimate that around 65% of children starting primary school today will end up in jobs that don't currently exist. That's not a scare statistic — it's simply the pattern of the past fifty years, extended forward.

What does this have to do with coding and data?

Quite a lot, actually. Almost every new job created in the last decade has involved working with digital tools, understanding data in some form, or interacting with technology in ways that go beyond just using apps. The shift isn't that everyone needs to become a programmer — it's that the ability to think clearly about information, to understand how digital systems work, and to use technology purposefully has become a baseline professional skill, much like literacy or numeracy.

> *The ability to work with data and digital tools has become a baseline skill — not just for tech jobs, but for almost every job.*

Why it's different from when we were at school

When most of us were growing up, computers were something you used for specific tasks — typing a document, playing a game, looking something up. Learning to use them was about learning particular software: Microsoft Word, Excel, Internet Explorer. That's not what schools are teaching now, and it's worth understanding why.

The curriculum has shifted from teaching children to *use* software, to teaching them to *understand* how it works — and eventually to build their own. The reasoning is simple: if you only know how to use the tools someone else built, you're always dependent on whoever built them. If you understand how those tools work, you have genuine power over your own digital life.

Consider the difference between someone who can use a spreadsheet and someone who understands how to structure data and write the logic behind it. Both can do useful things. But the second person can do things the first person can't even imagine — and in most professional contexts, that difference is increasingly visible in salaries.

> **A USEFUL ANALOGY**
> We teach children to write, not just to read. We could get by teaching only reading — but writing gives you a different kind of power over language. Learning to code is the same shift: from consuming digital things to being able to create them.

What the curriculum actually covers — and why

The UK National Curriculum introduced computing as a core subject in 2014. That might feel recent, but children who started school that year are now in their teens — which means this is no longer a new experiment. It's an established part of education.

Here's a plain-English summary of what children are expected to learn at each stage, and the thinking behind it:

Age	What they're learning	Why it matters
5–11 (KS1 & KS2)	Logical thinking. Simple algorithms. Scratch programming. Understanding that computers follow instructions.	Builds the systematic, step-by-step thinking that underpins all of computing — and a lot of mathematics.
11–14 (KS3)	Python or a similar language. How the internet works. Data representation. Cybersecurity basics.	Moves from visual tools to real programming. Builds genuine digital literacy.

Age	What they're learning	Why it matters
14–16 (KS4 / GCSE)	GCSE Computer Science: algorithms, data structures, programming projects, AI and ethics.	Formal qualifications that open doors to A-levels, apprenticeships, and university courses.

Scotland, Wales, and Northern Ireland have their own curricula, but all have made similar shifts towards computing and digital literacy as core subjects in recent years.

The skills nobody tells you about

Here's something that often surprises parents: the most valuable things children learn from coding and data aren't technical at all.

When a child learns to write a programme, they're learning to break a big problem into small steps. They're learning that when something doesn't work, the answer is to look carefully at what went wrong and try a different approach — not to give up. They're learning that precise language matters, because a computer does exactly what you tell it, not what you meant.

These are habits of mind. They're useful in law, in medicine, in business, in writing, in almost every field you can think of. A child who learns to debug their code is developing exactly the same mental muscle as a scientist who learns to trace a failed experiment back to its source.

Data literacy — the ability to understand, question, and work with information — is similarly transferable. In a world flooded with statistics, graphs, opinion polls, and "research shows that..." headlines, knowing how to read a chart critically and ask where the numbers came from is an incredibly powerful skill. It's what separates people who can be misled by data from people who can't.

> *Learning to code teaches children to think clearly under constraints. That's useful in every field, not just technology.*

What about jobs? The honest picture

Let's be concrete about this, because vague claims about "jobs of the future" can feel unconvincing. Here's what the data actually shows.

The UK government's own figures show that digital skills are now required in over 80% of advertised jobs — and that's not just tech jobs. Marketing, finance, healthcare, teaching, logistics, retail — virtually every sector now expects employees to work with data and digital tools as a matter of course.

The salary gap is real and growing. According to research by Nesta, workers with strong digital skills earn around 30% more on average than those without. That gap has been widening for a decade and shows no sign of narrowing.

Perhaps most tellingly: the fastest-growing occupational category in the UK is not engineering or software development. It's roles that combine domain expertise with digital fluency — the nurse who

can work with patient data systems, the teacher who can create digital learning materials, the accountant who can automate their reporting. These are not "tech jobs". They're ordinary jobs, done better by people who are digitally literate.

> THE SKILL YOUR CHILD ACTUALLY NEEDS
> It's not "how to code". It's the ability to work confidently with digital tools, understand data, and adapt as the tools change. Coding is one route to that — but the destination is broader than any particular language or platform.

What about AI? Doesn't that change everything?

You've probably heard the argument: AI will do all the coding, so why bother learning it? It's a reasonable question, and we'll cover AI in depth in Module 4. But here's the short version.

AI tools can write code, yes. They can also write essays, do maths, translate languages, and summarise documents. But we haven't stopped teaching children to write, do maths, or learn languages. We've adjusted how we teach them and what we emphasise — critical thinking, checking outputs, knowing when to trust a result and when to question it.

The same is true for coding and data. A child who understands how programmes work and how data is structured will use AI tools far more effectively than one who doesn't. They'll know when the AI is right, when it's plausible but wrong, and when it's confidently making things up. That judgement doesn't come from using AI — it comes from understanding the underlying subject.

What if my child doesn't want to work in tech?

Good news: this doesn't matter as much as you might think.

A child who wants to be a doctor will work with electronic health records, diagnostic AI, and data systems. A child who wants to be a journalist will need to understand data visualisation, verify online sources, and navigate AI-generated content. A child who wants to run a small business will use digital marketing, financial software, and automated tools from day one.

The question isn't whether your child will use these skills — they will, in almost any career. The question is whether they'll use them confidently or reluctantly, whether they'll be the person in the room who understands what's happening or the one who nods along hoping nobody asks them a direct question.

More practically: even if a child never writes a line of code in their working life, learning to code has taught them something valuable. The logical thinking, the systematic problem-solving, the resilience when things don't work first time — these are skills that transfer everywhere.

> *You don't have to want a tech career to benefit from learning how technology works.*

What you can do as a parent

You don't need to understand Python to support a child who's learning it. What matters far more is attitude — yours as much as theirs.

Research on how children engage with difficult subjects shows that parental attitude is one of the strongest predictors of persistence. Children who hear "I was never good at maths either" tend to disengage from maths. Children who hear "that sounds tricky — what have you tried so far?" tend to stick at it. The same applies here.

If you express anxiety or dismissiveness about coding and data — even casually — your child is likely to absorb that attitude. If you express genuine curiosity, even from a position of knowing nothing, they're more likely to stay engaged.

That's partly why this guide exists. You don't need to become an expert. You just need enough of a picture to ask good questions, to notice when your child is engaged versus struggling, and to treat the subject as something worth taking seriously.

Module 1 Activity: The 10-year job exercise

Sit down with your child — or just do this yourself first — and write down three jobs that didn't exist when you started secondary school. (Social media manager, data analyst, and podcast editor are good examples, but find your own.) Then ask your child: what do you think those jobs involve day to day? What skills do they need? Do any of them involve coding, data, or AI? There's no right answer. The point is simply to make the abstract concrete — and to open a conversation about change that doesn't start with "you need to learn Python".

IN SUMMARY

The world of work is changing
Digital skills are now expected across almost every sector — not just tech jobs. The salary gap between those who have them and those who don't is real and growing.

This isn't about everyone becoming a programmer
The goal is digital literacy — the ability to work confidently with data and technology, understand how it works, and adapt as it changes.

The hidden skills are the most valuable
Learning to code builds logical thinking, systematic problem-solving, and resilience. These transfer to every field.

AI doesn't make this irrelevant
Understanding the underlying subject makes AI tools more useful, not less necessary. Judgement comes from knowledge.

Your attitude matters as much as your knowledge
You don't need to understand Python. You need to approach it with curiosity rather than anxiety — your child will notice the difference.

UP NEXT

Module 2 — What Is Coding, Really?

Now that you know why this matters, Module 2 demystifies programming itself. What is Scratch? What is Python? What is your child actually doing when they sit down to code — and why does it look nothing like what you'd expect?

THE COMPLETE GUIDE

Want all seven modules?

This is Module 1 of *What Is My Child Actually Learning?* — a complete parent's guide to coding, data, and AI in schools.

Seven plain-English modules · Read on any device · No technical knowledge needed

Register your interest at codifyit.co.uk

ALL MODULES IN THIS GUIDE