مجلة جامعة بني وليد للعلوم الإنسانية والتطبيقية
Bani Waleed University Journal of Humanities and Applied Sciences
تصدر عن جامعة بني وليد - ليبيا
Website: https://jhas-bwu.com/index.php/bwjhas/index
ISSN 3005-3900 الصفحات (145 -155) المجلد الحادي عشر – العدد الخامس – 2026

Evaluating Early Childhood Computer Science Education in Libya: A Gap Analysis of Primary Years 1 & 2 Textbooks Against CSTA K–2 Standards

Nesrin Muftah Elrgibe [1]*, Fawzeyia Elhashmi Mohamad Salah [2], Nesrin Ali Elharbi [3]
[1,2,3] Computer Department, Faculty of Education, Tripoli University, Tripoli, Libya.

n.elrgibe@uot.edu.ly

تقييم تعليم علوم الحاسوب في مرحلة الطفولة المبكرة في ليبيا: تحليل الفجوة بين منهج الحاسوب للصفين الأول والثاني من التعليم الأساسي مع معايير جمعية معلمي علوم الحاسوب *CSTA*

نسرين مفتاح الرقيبي*[1] ، فوزية الهاشمي محمد صلاح [2] ، نسرين علي أبوعجيلة الحربي[3]
[1،2،3] قسم الحاسوب، كلية التربية، جامعة طرابلس، طرابلس، ليبيا.

Received: 07-06-2026	Accepted: 19-07-2026	Published: 28-07-2026

الملخص

شهد تعليم الحاسوب في المرحلة الابتدائية على المستوى الدولي خلال العقد الماضي تحولًا من محو الأمية فيما يتعلق بتقنية المعلومات والاتصالات نحو تنمية التفكير الحاسوبي في الناشئة، حيث يُنظر إلى الطفل بوصفه الطرف الذي يقوم بإنتاج وتكوين المحتوى والمنتجات الحاسوبية لا كمجرد مشغّل لا غير للأجهزة والتقنيات الحاسوبية. ولا يزال مدى مواكبة المناهج الوطنية في الدول النامية والمتأثرة بالنزاعات لهذا التحول غير موثّق بدرجة كافية، ولا سيما في المغرب العربي. تقيّم هذه الدراسة مدى مواءمة كتابي الحاسوب الرسميين للصفين الأول والثاني من مرحلة التعليم الأساسي في ليبيا (2025-2026) مع معايير جمعية معلمي علوم الحاسوب (CSTA) للمستوى A1) الصفوف من الروضة إلى الثاني، الأعمار 5-7). واعتُمد أسلوب التحليل التوثيقي المقارن نفذ على الوثائق المذكورة، حيث صُنّف كل معيار من المعايير الثمانية عشر إلى (مستوفى كليًا) أو (مستوفى جزئيًا) أو (فجوة) وفق قواعد ترميز صريحة تحكم مطابقة المنهج مع المعايير محل المقارنة من حيث السياق. وقد استُوفيَ معيارٌ واحدٌ كليًا (5.6%)، وأربعة معايير جزئيًا (22.2%)، بينما شكّل ثلاثة عشر معيارًا (72.2%) فجوات كاملة، بمؤشر مواءمة مرجّح بلغ 16.7%. وتراوحت المواءمة بين 50.0% في مجال أنظمة الحوسبة و0.0% في مجال الشبكات والإنترنت. ويتميز المنهج بمصطلحات خاصة بالعتاد المادي قوية ومعالجة وافية للسلامة الصحية والبدنية، غير أنه يخلو من أي محتوى يخص البرمجة أو بناء الخوارزميات أو تمثيل البيانات، ومن أي إشارة إلى كلمات المرور أو قواعد السلوك الرقمي السليم أو مبادئ الأمن السيبراني. ويمكن سدّ 69% من الفجوات المحددة دون أي أجهزة حاسوب، مما يشير إلى أن جوهر المشكلة لا تكمن في نقص في البنية التحتية إلى بل إلى قصور واضح في تصميم المناهج وإعداد المعلمين.

الكلمات الدالة: التفكير الحاسوبي، تحليل فجوات المناهج، معايير CSTA، التعليم الأساسي المبكر، تعليم علوم الحاسوب، ليبيا، علوم الحاسوب بدون حاسوب.

Abstract
Primary education of computer science has shifted internationally over the past decade from information and communication technology (ICT) literacy toward computational thinking, in which young children are positioned as producers of computational artefacts rather than operators of devices. Whether national curricula in developing and conflict-affected contexts have participated in this shift remains under-documented, particularly in the Arab Maghreb. This study evaluates the alignment between Libya's official first-cycle computer science textbooks for Grades 1 and 2 (2025-2026) and the Computer Science Teachers Association (CSTA) K-12 Standards at Level 1A (Grades K-2, ages 5-7). A qualitative comparative document analysis was applied to a corpus of 107 pages, coding each of the 18 Level 1A standards as Fully Met, Partially Met or Total Gap under explicit decision rules governing verb correspondence and register substitution. One standard (5.6%) was Fully Met, four (22.2%) Partially Met and thirteen (72.2%) constituted total gaps, for a weighted alignment index of 16.7%. Domain alignment ranged from 50.0% for Computing Systems to 0.0% for Networks and the Internet. The curriculum exhibits strong hardware terminology and an unusually thorough treatment of physical ergonomics, but contains no programming, no algorithmic construction, no data representation and no reference to passwords or online conduct. Sixty-nine per cent of the identified gaps are closeable without computing equipment, which moves the deficit from infrastructure to curriculum design and teacher preparation.

Keywords: Computational Thinking, Curriculum Gap Analysis, CSTA K-12 Standards, Early Primary Education, Computer Science Education, Libya, Unplugged Computer Science.

Introduction

From the mid-1980s to the mid-2000s, computing entered primary schooling on instrumental grounds. Computers were tools, and the curricular job was to produce competent users of them. That job had a settled content: name the components, operate the interface, and use a small set of standard applications productively. Curricula written on this model, usually called ICT literacy, tend to look alike. They are organised around the machine, and they move from hardware to software to the operating-system shell and then to the office suite.

Wing's account of computational thinking [1] began a shift away from that model. Her argument was that the analytic toolkit of computer science, decomposition, abstraction, algorithmic design and evaluation, is a general problem-solving capability worth teaching to every citizen rather than vocational preparation for a few. The idea was not new. Papert had argued two decades earlier that children should program the computer rather than be programmed by it [2]. What had changed was the policy climate: governments were by then reassessing ICT-literacy curricula whose content dated quickly and asked little of pupils.

Policy moved quickly. England replaced its ICT programme of study with a statutory Computing curriculum requiring pupils from age five to understand algorithms, create and debug simple programs, and reason logically about how a program will behave [3]. The Computer Science Teachers Association published revised K-12 standards in 2017, built on five concept domains and seven practices and reaching down to the earliest grade band [4]. Comparable reforms were documented across more than a dozen European systems [5]. By 2020 an explicit algorithmic strand starting in the first year of compulsory schooling had become ordinary in high-income countries.

Extending computing to five- to seven-year-olds was contested at first on developmental grounds, and that objection has mostly faded. Children in this band can build and run ordered instruction sequences, break familiar routines into steps, and find and repair errors in sequences they or others have written. Five- and six-year-olds have solved problems productively inside a programming environment [6]. A long programme of design research argues that programming at this age behaves less like a technical subject than like a literacy, an expressive medium children use to make meaning [7], and work with tangible robotics in kindergarten and early-primary classrooms points the same way [8]. Two details from this literature matter here. The practices that come most easily at these ages, sequencing, decomposition and debugging, are the ones that need no equipment. The practices that come hardest are formal abstraction and anything requiring fluent reading, which is why icon-based environments [9] and unplugged card activities have become the usual delivery vehicles.

Libyan general education has operated for more than a decade under political fragmentation, periodic conflict, displacement and constrained public spending, with the expected consequences for buildings, teacher retention and the spread of resources between and within regions. Curriculum development is centralised in the Centre for Educational Curricula and Educational Research, which wrote and published the two textbooks examined here. Two

points about those books frame what follows. They are current, carrying the academic year 1447 AH / 2025-2026 CE, so they are the operative curriculum rather than legacy material awaiting replacement. And their front matter sets out a thoroughly modern pedagogy: knowledge-economy framing, learner-centred method, cooperative activity, self-assessment, teacher talk held below a quarter of lesson time, and differentiated pacing for pupils with special educational needs. Whatever this study finds about their content cannot be blamed on pedagogical conservatism among their authors.

The study asks how far the Libyan early-primary computing curriculum matches an established international benchmark for the same age band, and what any divergence implies about the work needed to close it. Four questions are addressed. How far do the Grade 1 and Grade 2 textbooks align with the 18 CSTA Level 1A standards, overall and by domain? Which conceptual areas are covered, which are covered thinly, and which are missing? What curricular paradigm explains the pattern? And which gaps follow from infrastructure constraints rather than from curriculum design and teacher capacity?

Conceptual Background

One distinction governs the analysis, and policy language tends to blur it. Digital literacy is about competent, safe and purposeful use of digital tools. Computational thinking is about the practices by which a problem is formulated so that its solution can be expressed as computational steps. Neither contains the other. A pupil can operate a tablet fluently with no computational thinking at all, and a pupil can decompose, sequence and debug competently with cards and chalk without ever touching a computer.

An early review of the field identified unclear definitions as its main weakness and argued that operational clarity meant naming the constructs: abstraction, algorithmic notation, structured decomposition, iteration, conditional logic, debugging and systematic testing [12]. Work seeking an agreed understanding for compulsory education reached much the same conclusion [13]. The main dissent warns that the term dilutes when stretched to cover any structured thinking, and that curricular claims should stay anchored to the practices of computing rather than to problem-solving in general [14]. That warning is useful methodologically, because it justifies a conservative coding stance in which ordered classroom routines earn no credit as algorithmic content unless they are named and used generatively. A K-6 framework has since made explicit what curriculum documents usually leave unsaid, that such a curriculum makes demands on teacher knowledge that an ICT-literacy curriculum does not [15].

Three benchmarks dominate the field. However, this study uses the CSTA K-12 standards [4], which at Level 1A specify 18 performance expectations across five domains: Computing Systems (3 standards), Networks and the Internet (1), Data and Analysis (3), Algorithms and Programming (8) and Impacts of Computing (3). The framework suits gap analysis for two reasons. Every standard carries a descriptive statement that pins down the intended performance and supplies examples, which limits interpretive drift during coding. And the weighting itself carries information: putting 8 of 18 standards into Algorithms and Programming is a claim about what early computing education is for, so how a curriculum covers that domain says a good deal about its orientation. The English National Curriculum for Computing [3] offers a shorter statutory alternative whose Key Stage 1 requirements map closely onto Level 1A, and it is used here to corroborate. The ISTE Standards for Students [16] organise around learner dispositions rather than disciplinary content, which makes them a poor gap-analysis instrument because their generality admits too many readings.

The CS Unplugged tradition [10] grew out of an interest in conceptual clarity rather than economy, but it has acquired a second use in systems where a working laboratory cannot be assumed. Unplugged activities teach algorithms, data representation and debugging through movement, cards, string, chalk and role-play. A later review cautions that the approach complements programming and does not replace it permanently [11], and that caution is kept in view below. One further finding runs through the reform literature: what limits implementation is usually teacher knowledge rather than curriculum text or equipment, with subject-matter confidence predicting fidelity better than resource availability does [17].

Methodology

This study compares different documents using an existing framework (the Theory-first model). This method was chosen because the goal is to see how well the documents match a specific outside standard, rather than trying to find new ideas within the texts themselves. The study only looked at documents and did not involve any people.

The corpus is the two official pupil textbooks that make up the whole computing curriculum for the first two years of Libyan basic education: كتاب الحاسوب للصف الأول من مرحلة التعليم الأساسي (52 pages, 14 lessons) and كتاب الحاسوب للصف الثاني من مرحلة التعليم الأساسي (55 pages, 14 lessons), both published by the Centre for Educational Curricula and

Educational Research for the academic year 1447 AH / 2025-2026. This is a complete population rather than a sample, since it is everything officially prescribed for pupils in the target age band. Grade 1 runs from the computer and its hardware and software components through peripherals, the computer in daily life, laboratory conduct, mouse use and start-up, to the desktop and shut-down. Grade 2 opens with health rules, then covers main and peripheral equipment and their functions, software types, the Windows environment, desktop components, keyboard and mouse use, desktop properties, launching a program and window controls.

The authors carried out the review, working through both textbooks page by page from cover to cover and examining each lesson, illustration, worked procedure and exercise against the descriptive statement of every Level 1A standard. Working from the printed volumes rather than digital versions was deliberate. Much of the teaching load in early-primary textbooks sits in illustrations, iconography, screen captures and the visual arrangement of exercises rather than in running text, and those elements are judged most reliably in the form pupils and teachers actually meet. Each of the 28 lessons was read twice, once to establish what it teaches on its own terms and once to test it against the standards. Evidence was logged at page level organised by the five domains.

Each of the 18 standards served as a coding unit, taken with its full official descriptive statement rather than its headline sentence. Three mutually exclusive codes were used. Fully Met means the concept is taught explicitly and the pupil performs the action named by the standard's verb, with at least one practical exercise to support it, so that the performance expectation could be demonstrated from the textbook alone. Partially Met covers attenuated evidence: indirect, pitched below the cognitive demand of the verb, delivered in a physical rather than a digital register, or reaching only part of the standard's scope. Total Gap means no lexical, conceptual, illustrative or task-level evidence anywhere in the corpus.

Two rules controlled over-credit, a risk that pulls in one direction only, toward inflated alignment. The verb-correspondence rule holds that where a standard asks the pupil to create, develop, collect or decompose, evidence that the pupil merely follows, identifies or recognises earns a partial code at most. The register rule holds that where a standard specifies a digital register and the corpus treats the analogous concern physically, the code is partial at best, and a total gap where the digital concept is absent altogether. So laboratory conduct is not coded as digital citizenship, ergonomic safety is not coded as cybersecurity, and powering a device down is not coded as logging out of an account.

For aggregate reporting a weighted alignment index was computed as $W = (F + 0.5P) / N$, where F is the count of fully met standards, P the count of partially met standards and N the total in scope. The 0.5 weighting is a convention, and raw category counts appear alongside every index value so that readers can substitute their own. Three procedures support trustworthiness. Every code carries page-level evidence, so the audit trail is open. The decision rules were fixed in advance and applied uniformly, which biases the analysis against inflated alignment rather than against the curriculum. And contestable cases are stated rather than suppressed. A second full pass was made over every standard coded as a total gap, looking specifically for evidence missed the first time.

Results

Of the 18 standards, one was coded **Fully Met (5.6%)**, four **Partially Met (22.2%)** and thirteen **Total Gap (72.2%)**, giving a weighted alignment index of $W = 3.0 / 18 = 16.7\%$. Table 1 shows how unevenly that total is distributed. Almost all of it sits in Computing Systems, which supplies half the weighted score while accounting for one sixth of the standards. Algorithms and Programming, which holds 44% of the Level 1A standards and therefore the largest share of the benchmark's stated priority, contributes 0.5 of the 3.0 weighted points.

Table 1: Alignment by CSTA Level 1A concept domain

Concept domain	Standards	Fully Met	Partially Met	Total Gap	W (%)
Computing Systems	3	1	1	1	50.0
Networks and the Internet	1	0	0	1	0.0
Data and Analysis	3	0	1	2	16.7
Algorithms and Programming	8	0	1	7	6.3
Impacts of Computing	3	0	1	2	16.7
All domains	18	1	4	13	16.7

Computing Systems

This is where the curriculum is strongest, and standard 1A-CS-02, on identifying hardware components and describing what they do, is the one standard met without reservation. The hardware component is built deliberately

across the two years. Grade 1 sets out four principal units, system unit, monitor, keyboard and mouse (pp. 12-13), the anatomy of the mouse itself with left button, right button and wheel (pp. 41-43), and a peripheral set of scanner, printer, speakers and camera (pp. 18-22). Grade 2 returns to the four units (pp. 17-18), adds microphone, headphones and digital camera (pp. 19-22), and then supplies the functional layer Grade 1 left out. Page 20 pairs each device with what it does, conveying input, output, storage and processing roles in language a seven-year-old can follow and without abstract vocabulary. Terminology is bilingual and used consistently, which will help pupils later when computing content switches to English.

Standard 1A-CS-01, on selecting and operating appropriate software, was coded Partially Met. Pupils do operate software: Grade 2 (pp. 50-53) teaches program launch by two routes, double-clicking a desktop icon and choosing from the Start menu, and the catalogue runs to a word processor, a drawing program, a calculator, a browser, a spreadsheet and a presentation program (pp. 25-26). What never happens is selection. No exercise sets a purpose and asks the pupil to pick a program that fits it, and the standard's second clause, on differing user needs and preferences, goes untouched. Software arrives as a closed catalogue to memorise rather than a toolkit to choose from.

Standard 1A-CS-03, on describing hardware and software problems in accurate terms, is a total gap. There is no vocabulary for a malfunction and no troubleshooting strategy, not even the two the descriptive statement names as ideal, restarting the device and closing and reopening an application. What the curriculum offers instead is prohibition. Grade 1 (pp. 31-34) tells pupils not to tamper with laboratory contents, not to alter settings and not to switch a machine on without the teacher's permission. These rules invert the standard, casting the pupil as custodian of the machine rather than competent user of it. A pupil finishing this part can name and describe the components of a computer as well as international peers can, and still cannot say what is wrong when something fails.

Networks and the Internet

The one standard here, 1A-NI-04 on passwords and protection from unauthorised access, is a total gap, and the absence is complete. A systematic page-by-page search of both volumes found no occurrence of كلمة المرور، تسجيل الدخول، حساب مستخدم or any equivalent. The Internet appears twice in the whole curriculum, both times as a product label rather than a concept: once as a desktop browser icon glossed as browsing the Internet (Grade 1, p. 46) and once as a named browser in the application-software list (Grade 2, pp. 25-26). The screen captures in Grade 2 (pp. 30, 42, 50) show an authenticated session with nothing said about how the authentication happened.

Level 1A gives this domain a single standard, and that standard is about passwords, the framework having judged that what five- to seven-year-olds most need from networking is not topology or packets but protection of their own credentials. Two features of the Libyan context sharpen the finding rather than excusing it. The usual objection about unreliable connectivity does not apply, since the standard concerns the idea and practice of a password, which can be taught on an offline machine, on paper, or through a role-play in which pupils invent and guard a secret word. And Libyan children meet authentication early and outside school, on a parent's phone, a shared family device or a games console. The curriculum falls silent exactly where the child's real exposure starts, which leaves credential habits to be picked up by imitating adults.

Data and Analysis

Standard 1A-DA-05 was coded Partially Met, and the file-handling behind that code is more substantial than the corpus as a whole would suggest. Grade 1 (pp. 46-47) introduces document storage and the recycle bin. Grade 2 (pp. 42-45) teaches double-click navigation into the file system, shows local drives with free and total capacity in gigabytes, teaches drag-and-drop relocation and right-click access to context menus and folder properties, and at pp. 46-49 has pupils browse the file system through a picker dialogue to choose an image. Save, delete, move, browse and inspect are all genuinely taught. Copy, search and modification of file content are not. What settles the coding is the standard's defining clause, to define the information stored as data: the term بيانات never appears in either volume, so pupils pick up file operations without the abstraction that would gather images, text, audio and programs into one category.

Standards 1A-DA-06 and 1A-DA-07, on collecting and presenting data visually and on reading patterns to predict, are empty. No survey, tally, table, chart, pictograph or graph appears anywhere, so no pattern identification and no prediction follow. The corpus holds exactly one numerical prompt, a question asking how many computers the school laboratory contains (Grade 1, p. 8), and it works as an isolated recall item with no collection, organisation or representation attached. It sits one instructional step away from a compliant activity. This is the domain where the distance between deficit and remedy is widest, since both standards need only tally marks, squared paper and coloured pencils, and both fit naturally beside the Grade 1-2 mathematics curriculum, where counting, comparison and simple graphing already appear.

Algorithms and Programming

Seven of the eight standards here are total gaps and one is partial. No programming environment of any kind appears in either volume: no block environment, no floor robot, no command cards, no Logo derivative. The vocabulary is missing too. Neither خوارزمية nor any child-level paraphrase working as a named concept occurs in either book, and repetition, loops, conditionals, decomposition, planning artefacts, attribution and debugging go unrepresented.

The single partial code, at 1A-AP-08 on modelling daily processes by creating and following algorithms, rests on a real habit of the books: operations are presented as explicitly numbered ordered sequences. The two-step start-up (Grade 1, pp. 44-45), the three- and four-step shut-down (Grade 1, pp. 48-51; Grade 2, pp. 31-34) and the five-step wallpaper change (Grade 2, pp. 46-49) all take this form, and Grade 2 page 50 goes further by giving two routes to launching the same program, which is an implicit brush with algorithmic equivalence. Under the verb-correspondence rule, though, the standard pairs creating with following, and only the second appears. No exercise asks a pupil to build a sequence, and the sequences on the page are framed as instructions for operating a machine rather than lifted into a transferable idea. This is the most contestable code in the study. Applying the warning against crediting ordinary structured routines as computational content [14] more strictly would make it a total gap and bring the overall index down to 13.9%.

Two further points are worth drawing out. Decomposition (1A-AP-11) is done for the pupil by the authors and never by the pupil, since no exercise presents an unbroken task and asks the child to break it into steps. And error goes untreated altogether, whether the error belongs to the machine (1A-CS-03) or to the pupil (1A-AP-14). The implicit epistemology is that the correct procedure is given and is to be reproduced faithfully, which forecloses the productive-failure stance debugging depends on. Because this domain holds 44% of Level 1A by standard count, it accounts for most of the shortfall. The consequence is definitional: without it, the subject being taught is computer operation rather than computer science.

Impacts of Computing

Standard 1A-IC-16, on comparing life before and after a computing technology arrives, was coded Partially Met on the strength of Grade 1's societal lesson (pp. 24-27), which places computing in schools, banks, hospitals, airports, shops and games and supports it with a matching exercise. The lesson is locally grounded and well illustrated. It is also synchronic and descriptive where the standard is diachronic and comparative. Nothing frames a before and after, no paired image shows the same activity done without computing, and negative change is not considered alongside positive.

Standards 1A-IC-17 and 1A-IC-18 are total gaps under the register rule. There is a substantial behavioural strand in the books: Grade 1's laboratory-order and computer-handling lessons (pp. 28-40) and, above all, Grade 2's opening lesson on health rules (pp. 6-16), eleven pages and a fifth of the volume, covering posture, screen distance, lighting, a 30-40 minute working limit with 4-5 minute breaks and periodic medical examination. As a treatment of physical safety it is well done, more thorough than in many better-resourced systems. But 1A-IC-17 is about conduct in online collaborative spaces, withholding personally identifying information, giving kind feedback and reporting harmful behaviour to an adult, and no online space exists anywhere in the corpus. Standard 1A-IC-18 is about credential privacy and proper log-off; both volumes teach shut-down, which changes the power state, and neither teaches sign-out, which changes the identity state, even though the menus reproduced in Grade 2 (pp. 33, 50) show the very control from which sign-out is chosen. With 1A-NI-04 also a total gap, the cybersecurity and safety-law-ethics coverage of Level 1A stands at zero. Table 2 gives the full coding.

Table 2: Coding of all 18 CSTA Level 1A standards against the Libyan corpus

Code	Standard (abridged)	Status	Basis for the coding decision
1A-CS-01	Select and operate appropriate software	Partial	Pupils start apps by two ways; no task requires to select a fitting program, and user preference is untouched.
1A-CS-02	Identify and describe hardware components	Full	Bilingual terminology across both volumes with an explicit device-function map.
1A-CS-03	Describe hardware and software problems	Gap	No fault vocabulary and no troubleshooting; laboratory rules train the pupil not to intervene.
1A-NI-04	Explain and use passwords	Gap	No occurrence of password, login or user account in 107 pages; the Internet appears twice, only as an icon.
1A-DA-05	Store, retrieve and modify information as data	Partial	Save, delete, move, browse and inspect are taught; copy and search are not, and the term data is never defined.
1A-DA-06	Collect and present data in visual formats	Gap	No survey, tally, table, chart or pictograph anywhere in the corpus.

1A-DA-07	Identify patterns in data to make predictions	Gap	No data visualisation, and therefore no pattern identification and no prediction.
1A-AP-08	Create and follow algorithms for daily processes	Partial	Numbered operating procedures are followed but never constructed; the concept is never named or abstracted.
1A-AP-09	Represent information using numbers or symbols	Gap	Symbols appear only as answer-marking notation; no encoding or decoding activity.
1A-AP-10	Develop programs with sequences and loops	Gap	No programming environment of any kind; loops never mentioned in any register.
1A-AP-11	Decompose steps into a precise sequence	Gap	Decomposition is performed for the pupil by the authors and never by the pupil.
1A-AP-12	Develop plans describing a program's sequence	Gap	No storyboard, story map or planning artefact; no program development to plan.
1A-AP-13	Give attribution when using others' creations	Gap	Ownership of work and crediting are never raised, including where an image is appropriated as wallpaper.
1A-AP-14	Debug errors in an algorithm or program	Gap	No error-finding or repair activity; the corpus contains no treatment of error at all.
1A-AP-15	Describe steps and choices during development	Gap	No coding journal, presentation or discussion of choices, despite front matter urging such practice.
1A-IC-16	Compare life before and after new technology	Partial	Computing's ubiquity is well illustrated across sectors, but the comparison is synchronic, not diachronic.
1A-IC-17	Work respectfully and responsibly online	Gap	Offline laboratory conduct is taught; no online space exists anywhere in the corpus.
1A-IC-18	Keep login information private and log off	Gap	Shut-down is taught as a power-state transition; sign-out and credential privacy are absent.

Resource Attribution of the Identified Gaps

Each of the thirteen total-gap standards was classified by whether closing it to full specification would need working computing equipment (Table 3). Nine of the thirteen would not, and neither would raising 1A-AP-08 and 1A-IC-16 from partial to full; paper, cards, chalk, floor space and crayons would do. Implementing that subset alone would take weighted alignment from 16.7% to roughly 58%.

Table 3: Resource attribution of the thirteen total-gap standards

Closeable without computing equipment	Requires equipment
1A-CS-03 fault-description vocabulary	1A-AP-10 develop programs with sequences and loops
1A-NI-04 passwords, concept and practice	1A-AP-15 describe the development process
1A-DA-06 collect and represent data	1A-IC-17 online conduct, requiring an online space
1A-DA-07 patterns and prediction	1A-IC-18 log-off, requiring accounts on a device
1A-AP-09 symbolic representation	
1A-AP-11 decomposition	
1A-AP-12 planning artefacts	
1A-AP-13 attribution	
1A-AP-14 debugging	
9 of 13 (69%)	4 of 13 (31%)

Discussion

The pattern in these results is too consistent to be incidental. The Libyan curriculum is organised around the machine: its parts, its peripherals, their functions, its operating-system shell, its start-up and shut-down, and the rules for correct bodily and behavioural conduct in its presence. Judged inside that frame the books are competently made. The CSTA framework is organised around a different object, the child as producer of computational artefacts, and its allocation of eight of eighteen standards to Algorithms and Programming asserts that early computing education exists to build constructive and analytic practice, with device knowledge as supporting infrastructure rather than as the subject. Read this way, 16.7% is less a measure of missing content than a sign that the two documents are about different things. The distinction matters for reform, because it means the usual remedy, naming more devices and cataloguing more applications, would leave the index roughly where it is.

The most telling structural feature of the corpus is an asymmetry in what it protects. Hazards to the child's body and to the equipment are recognised and treated at length: poor posture, eye strain, weak lighting, long sessions, dust, food and drink near the machine, unauthorised changes to settings. Hazards to information, to identity and to the child's social experience online are not recognised at all. Every safety concept in the two books is a bodily one. That asymmetry is what produces three of the most consequential gaps, which together make up the whole of the cybersecurity and safety-law-ethics coverage at Level 1A and stand at zero. The near-miss at 1A-IC-18 shows how narrow the distance can be: the curriculum teaches the pupil to switch the computer off but not to sign out of it, having printed on its own pages the menu that signing out comes from.

There is an encouraging reading of the same finding. The authors of these textbooks clearly care about protecting children; eleven pages on ergonomics is the evidence. The task is not to create that commitment but to widen a frame that already works, carrying it from the body to the account. Presenting a password lesson as the second half of the existing health-and-safety lesson, سلامة الجسم and سلامة معلوماتي, works with the logic of the book rather than against it, and stands a better chance of surviving revision. Other near-misses sit close by. Because the books habitually present numbered ordered sequences, the structural form of an algorithm is already familiar to pupils and teachers, so what is missing is the generative act and the name rather than an unfamiliar structure. And what blocks 1A-DA-05 is one absent term, بيانات, which would turn an operational skill into a conceptual one at almost no cost.

A predictable response to findings like these is that a system under conflict-related strain cannot fairly be held to a benchmark written for well-resourced schools. The objection deserves an answer rather than a dismissal, and the classification in Table 3 is the attempt at one. In its general form the objection does not hold: 69% of the absent standards need no computing equipment at all, and the unplugged tradition [10], [11] exists precisely to make these concepts available without hardware. The objection does hold for the remaining 31%. Developing programs needs a programming environment. Online conduct needs an online space. Log-off needs a device with user accounts. Describing the development process presupposes artefacts to describe. A Libyan curriculum reaching 58% alignment through unplugged delivery while treating those four as medium-term goals would be making a defensible and honest choice, provided it kept in view the caution that unplugged work complements programming rather than replacing it for good [11]. The deficit, then, is mostly one of curriculum design and teacher capacity rather than infrastructure, which is the more tractable diagnosis, since design and capacity cost far less to fix than national laboratory provision.

That conclusion puts the weight of reform on teacher preparation. A teacher who has never personally decomposed a problem, traced an algorithm by hand, encoded information symbolically or debugged a faulty sequence cannot teach those practices from a textbook page, however well the page is written, and the international evidence consistently shows subject-matter confidence rather than equipment predicting fidelity of implementation [15], [17]. Three implications follow for Libyan faculties of education. Primary-teacher preparation needs a dedicated computational-thinking course, separate from existing computer-skills or educational-technology provision, since a course teaching student-teachers to use office applications equips them for the current curriculum and not the reformed one. That course should be taught mostly unplugged, both because it models the pedagogy graduates will need in under-resourced schools and because it stops the laboratory being a precondition for teacher preparation itself. And teaching-practice placements should require at least one assessed computational-thinking lesson, without which the content stays inert. An in-service route is needed alongside this for teachers already in post, since revision cycles run in years while the current cadre is in classrooms now.

Recommendations for Curriculum Revision

What follows is addressed to curriculum developers at the Centre for Educational Curricula and Educational Research, and is sequenced so that each phase pays off whether or not the later ones happen. The first phase needs no additional hardware and would raise weighted alignment from 16.7% to roughly 58%.

1. Insert an unplugged algorithms component into Grade 1. Two lessons, on ordered steps and on instruction and execution, in which pupils order, execute and correct cut-out picture cards for daily routines such as washing hands, packing the school bag or the classroom tidy-up. Existing textbook production already includes colouring and matching pages, so nothing about the production model needs to change.
2. Add a floor-grid command game in which one pupil directs another along a chalk-marked route using literal step commands, with the class finding and repairing faulty commands together. Executing commands literally is what teaches precision, and the failure-and-repair cycle is what teaches debugging.

3. Add a class-survey data lesson to Grade 2, cross-referenced to the mathematics scheme of work, in which pupils tally a class survey, represent the same data twice as a pictograph and as a bar chart, and then answer a prediction question.
4. Extend the Grade 2 health-rules lesson into a two-part safety lesson covering what a password is, why a secret protects what belongs to the child, strong secrets against weak ones, and never telling anyone a password except a parent or teacher, delivered through a secret-word role-play. This is the single highest-value addition available, both because the networks domain stands at zero and because it connects primary education to national cybersecurity capability-building.
5. Convert the societal-relevance lesson into a comparative one using paired before-and-after images, with discussion of what became easier and what was lost.
6. Add a symbol-encoding activity in which a picture key assigns a symbol to each letter, so pupils encode their own name and decode a classmate's, and a fault-description panel supplying the language of malfunction together with two safe first actions.

The second phase addresses teacher enablement, and the first phase depends on it. A teacher's guide with fully scripted unplugged activities for each addition, including likely pupil responses and common misconceptions, does more work than anything else in the programme, because without it the textbook additions will be delivered as reading passages. Alongside it should come computational thinking in pre-service preparation, an assessed requirement inside teaching practice, and a short in-service cascade built on four scripted activities: sequencing, debugging, class-survey data and the password role-play.

A third phase, wherever equipment allows, would bring in a block-based environment such as ScratchJr in the final term of Grade 2, across four lessons: moving a character, sequencing three commands into a story, adding a repeat block, and finding and fixing a broken sequence. Environments of this kind are free, run offline once installed, use icons rather than text and so suit pre-readers, and work on low-specification tablets [9], [18]. Where tablets are unavailable, one teacher machine can drive whole-class demonstration while pupils hold physical block cards. This phase would also add task-driven software selection, name data explicitly and add copy and search to the existing file lesson, teach sign-out as distinct from shut-down using menus the textbook already reproduces, and attach an attribution rule to every creative activity [19].

Structurally, the subject title deserves reconsideration, since the present one describes the current content accurately and thereby entrenches it, whereas a title in the register of التفكير الحاسوبي would signal the intended shift and give teachers licence to teach past the device. The two-year time allocation should also be rebalanced from its present concentration of roughly 70% on hardware, software cataloguing and operating-system operation, toward something closer to 40% computing systems, 25% algorithms and programming, 15% data, 10% impacts and digital citizenship, and 10% networks and cybersecurity. A periodic benchmarking cycle against a published international framework, with findings made public, would help prevent the drift recurring.

Limitations

The study covers pupil textbooks only. The teacher's guide, ministry schemes of work, assessment instruments and classroom practice as enacted were not available, so what is described is the intended curriculum as the pupil text represents it rather than the implemented or attained curriculum. Teachers may well deliver algorithmic or data content the books omit, and data representation may be covered in the parallel mathematics curriculum, in which case part of that gap is a coordination failure between subject curricula rather than an absence from the system. The benchmark encodes one defensible weighting of early computing priorities among several, the 0.5 weight given to partial codes is conventional, and one code is openly contestable, with the stricter reading bringing the overall index down to 13.9%.

Conclusion

This study offers the first item-level benchmark analysis of Libya's early-primary computing curriculum against an international standard. Set against the 18 CSTA Level 1A standards, the official Grade 1 and Grade 2 textbooks for 2025-2026 meet one standard fully, four partially, and leave thirteen as total gaps, for a weighted alignment of 16.7%, with domain alignment running from 50% in Computing Systems down to zero in Networks and the Internet.

Three of the findings travel beyond the Libyan case. The coverage pattern reflects a coherent device-operation paradigm rather than piecemeal omission, so adding content inside the existing frame would not move the alignment much. The corpus shows a protective asymmetry, thorough and commendable on bodily safety and silent on the

informational equivalent, which explains why the cybersecurity strand is absent entirely and, because the underlying commitment to protecting children is plainly there, points to the most workable route for fixing it. And 69% of the gaps can be closed without computing equipment, which moves the problem from infrastructure to curriculum design and teacher capacity. There is a methodological point of wider use as well: applying explicit verb-correspondence and register rules changes gap-analysis outcomes materially, because a curriculum teaching laboratory conduct, physical ergonomics and power-down procedures looks well covered on the surface until the coding asks what the standards actually require.

Five extensions would strengthen the evidence. Extending the analysis to Grades 3-6 against Level 1B matters most, since it would show whether the computational-thinking gap narrows, holds or widens across the primary phase, and therefore whether what is documented here is a delayed start or a settled orientation. Analysis of the teacher's guide and of lessons as taught would test whether teachers supply what the textbook omits. A baseline assessment of pupils' computational thinking at the end of Grade 2, using an age-appropriate unplugged instrument, would give an outcome measure against which any revision could be judged. A survey of primary teachers' computational-thinking self-efficacy would size the in-service task. And a design-based implementation study trialling the first-phase unplugged additions in a small number of Libyan schools would test the feasibility claim the recommendations rest on.

Judged by the tradition they belong to, these two textbooks are competently made. The hardware component is coherent and spirally sequenced, the ergonomics lesson is more careful than many produced in far better-resourced systems, and the front matter sets out a learner-centred pedagogy with clarity and conviction. The problem is not that they are poor examples of their kind. It is that their kind, an ICT-literacy curriculum built around the machine, is one international practice has moved past, and that the practices they omit are the ones six- and seven-year-olds pick up most readily and that cost least to teach. A child who can name every peripheral on a desktop computer but has never put a sequence of steps in order, never found an error and fixed it, and never been told what a password is has been prepared for the computing of a generation ago. Closing that distance needs no national laboratory programme and no wholesale replacement of the curriculum. It needs cards, floor space, a teacher who has been shown how, and a decision to start.

References (APA 7th edition style)

[1] Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.

[2] Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books.

[3] Department for Education. (2013). National curriculum in England: Computing programmes of study. Department for Education.

[4] Computer Science Teachers Association. (2017). CSTA K-12 computer science standards, revised 2017. CSTA.

[5] Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). Developing computational thinking in compulsory education: Implications for policy and practice. European Commission, Joint Research Centre.

[6] Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5-6 years old kindergarten children in a computer programming environment: A case study. Computers & Education, 63, 87-97.

[7] Bers, M. U. (2018). Coding as a playground: Programming and computational thinking in the early childhood classroom. Routledge.

[8] Sullivan, A., & Bers, M. U. (2016). Robotics in the early childhood classroom: Learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. International Journal of Technology and Design Education, 26(1), 3-20.

[9] Flannery, L. P., Silverman, B., Kazakoff, E. R., Bers, M. U., Bontá, P., & Resnick, M. (2013). Designing ScratchJr: Support for early childhood learning through computer programming. Proceedings of the 12th International Conference on Interaction Design and Children, 1-10.

[10] Bell, T., Witten, I. H., & Fellows, M. (2015). CS Unplugged: An enrichment and extension programme for primary-aged students. Computer Science Unplugged.

[11] Bell, T., & Vahrenhold, J. (2018). CS Unplugged: How is it used, and does it work? In H.-J. Böckenhauer, D. Komm, & W. Unger (Eds.), Adventures between lower bounds and higher altitudes (pp. 497-521). Springer.

[12] Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. Educational Researcher, 42(1), 38-43.

[13] Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agreed understanding. Education and Information Technologies, 20(4), 715-728.

[14] Denning, P. J. (2017). Remaining trouble spots with computational thinking. Communications of the ACM, 60(6), 33-39.

[15] Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. Educational Technology & Society, 19(3), 47-57.
[16] International Society for Technology in Education. (2016). ISTE standards for students. ISTE.
[17] Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. Computer Science Education, 26(4), 235-254.
[18] Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. Computers & Education, 141, 103607.
[19] Kafai, Y. B., & Burke, Q. (2014). Connected code: Why children need to learn programming. MIT Press.