Integrating Computer Science and ICT Concepts in a Cohesive Curriculum for Middle School - An Experience Report

Saqui Razak
Carnegie Mellon University Qatar
Doha, Qatar
srazak@andrew.cmu.edu

Salar Mujeeb
Carnegie Mellon University Qatar
Doha, Qatar
salark@andrew.cmu.edu

Nehal Hussein
Carnegie Mellon University Qatar
Doha, Qatar
nmahmoud@andrew.cmu.edu

Hanan Alshikhabobakr
Carnegie Mellon University Qatar
Doha, Qatar
halshikh@andrew.cmu.edu

Huda Gedawy
Carnegie Mellon University Qatar
Doha, Qatar
hgedawy@andrew.cmu.edu

Abdul Wahab Yousef
Carnegie Mellon University Qatar
Doha, Qatar
abdulwaz@andrew.cmu.edu

Abstract

The paper elaborates on the experience gained during the first year of an intervention in the Middle Schools to implement a 3 year curriculum. The growing importance of computing knowledge in most professional fields has led many countries and stakeholders around the world to introduce Computer Science (CS) education in K-12 curriculum. These curricula replace the existing Information Communications Technology (ICT) curriculum by adding a new dedicated CS curriculum, adding topics of CS (predominantly "programming") to the ICT curriculum, or providing CS as an additional elective course. This paper aims to describe a new comprehensive curriculum that integrates topics of ICT with CS in a rich context to create need-based learning for the students (year 1 of the 3-year curriculum). The proposed curriculum has been designed to seamlessly replace existing ICT curriculum giving students added proficiency in CS along with ICT. This experience report provides the pedagogy behind the inclusion of ICT and CS topics, the detail of the curriculum, teacher training, and data collection methodology for impact evaluation. This paper goes into the detail of the 1st year of the curriculum and briefly explains the curriculum for the 2nd and 3rd year of Middle School grades.

CCS Concepts

• **Social and professional topics** → **Computing education; Computer science education; Model curricula; K-12 education.**

Keywords

Curriculum, Computer Science Education, K-12, ICT, Computing, Alice

ACM Reference Format:

Saqui Razak, Salar Mujeeb, Nehal Hussein, Hanan Alshikhabobakr, Huda Gedawy, and Abdul Wahab Yousef. 2021. Integrating Computer Science and ICT Concepts in a Cohesive Curriculum for Middle School - An Experience

This work is licensed under a Creative Commons Attribution International 4.0 License.

SIGCSE '21, March 13–20, 2021, Virtual Event, USA.

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8062-1/21/03.

<https://doi.org/10.1145/3408877.3432528>

Report. In *The 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21), March 13–20, 2021, Virtual Event, USA*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3408877.3432528>

1 Introduction

As the world is inclining more and more towards digital automation, the demand for hiring computer scientists and software developers is increasing significantly. According to a prediction from the White House article, only 28% qualified individuals shall be available for Computer Science (CS) related jobs by 2020 [25]. To meet this increasing demand, most countries have started introducing computing courses as part of their K-12 curriculum. To make room for these new concepts in their existing curriculum, different countries and stakeholders have designed different solutions such as replacing existing Information and Communication Technology (ICT) curriculum with a computing curriculum [24], providing selected Computer Science courses within ICT curriculum or other subjects [12] [17], or providing a CS curriculum as an elective course [2] [6]. Our research [2, 7, 8, 12, 15, 19], shows that no comprehensive curriculum has been devised so far that integrates the topics of ICT with CS in a meaningful context.

Earlier we had developed a CS focused curriculum to fulfill specific needs for local schools in Qatar. This curriculum used Alice software to teach CS. Alice is an interactive visual programming environment that allows users to learn CS concepts in the context of an animation or a game that is coded using drag and drop commands [13]. The students select 3-D objects from the object library in Alice and then insert these objects into the scene that they are creating. To animate their objects or to create interaction between various objects, student drag and drop code to formulate the sequence of commands they want executed to get the desired animations. This allows students to focus on CS concepts instead of worrying about underlying syntax. To accommodate this CS focused curriculum, schools had to move existing ICT topics either to lower or higher grades. This CS focused curriculum is currently being used in 50+ schools by more than 6000 students in Qatar. However, based on the feedback obtained in the form of surveys and focus groups from teachers and students [23], a need for a curriculum that incorporates both ICT topics and CS concepts was identified.

The proposed curriculum is a 3-year curriculum. In each year, the curriculum is supposed to achieve a certain objective. The objective

of each year of the curriculum is provided as such: 1st year focuses on combining ICT and CS topics in a cohesive manner and teaching CS concepts using a drag-and-drop coding environment, 2nd year focuses on teaching CS concepts using turtle graphics, and 3rd year focuses on teaching CS concepts using Python.

In this paper, we explain the first year curriculum in detail and briefly explain the topics being taught in year 2 and year 3 of the curriculum. The reason year 2 and year 3 are not being explained in detail is because they have been developed using the same philosophy that is used to develop year 1 curriculum as explained in this paper.

Year 1 curriculum is an introductory level curriculum for middle schools and assumes no prior knowledge for the students in either ICT or CS concepts. One of the major goals of our curriculum is to integrate the CS and ICT topics cohesively. The topics in the designed curriculum are strategically linked and streamlined so the flow from one topic to another is in the continuum of the context. In other words, the consecutive topics appear as either a tool to solve the task requirement of previous topics or as a new topic that supplements the need or requirement created by the previous topic. By doing so we can integrate the ICT topics as the tools required by the CS topics to yield certain tasks, thereby, creating a need-satisfying process. The transfer of knowledge can be facilitated if it is conducted under a meaningful context [14], which also helps in knowledge retention [16, 20].

As students study CS topics, the use of ICT tools like word processors, presentation software, sound editing tools, and web development using HTML is presented in an appropriate need-based context. Our previously implemented computing focused curriculum will be referred to as "CS-only curriculum" and the new integrated CS and ICT curriculum will be referred to as "CS/ICT curriculum". In this paper, we present the sequence and depth of topics and the flow that connects them together.

2 Background

Rising generations have access to technology like never before. Hence, many countries are trying to develop a curriculum that covers essential concepts of CS and engages young students. Different paths are being pursued to achieve CS inclusion in K-12. Recently, schools in the UK reintroduced CS, as it was the mainstream until mid 90's where ICT replaced it [10]. Similarly, countries like Germany, Poland, India, New Zealand and Israel are incorporating Computing in their K-12 school systems.

CS K-12 education is still undergoing a lot of discussions on the material that should be included in the curriculum [18]. It should be noted that factors beyond the covered material play a vital role in ensuring the effective delivery of the curriculum. More and more case studies are published in the field to promote best practices and teacher's readiness was seen as a common concern across countries. In the UK, for instance, teachers are threatened by the new curriculum due to their unfamiliarity with the subject [9]. Similarly, New Zealand's recent initiative to teach CS course in high schools struggle to gather sufficient well-trained teachers [11].

Another common theme is the mindset of stakeholders. It is noted that when computing is an optional course, it is referred to as "the difficult option" by teachers and parents [3]. This mindset

spreads to children even before they are exposed to the subject, repelling them from an initial unbiased experience. The unfamiliarity with computing also makes some stakeholders fear the change.

Our societies still do not distinguish between ICT and CS [9]. Therefore, our paper attempts to stitch both words in a cohesive well-merged curriculum, where both subjects go hand-in-hand to introduce various topics in one context. The goal is to help students understand why and how each field is useful in practice.

The team working on this paper had previously run a full-fledged CS curriculum for Grade 7 students (year 6 in the UK curriculum) over a few years. Their experience in developing and evaluating their curriculum gave them insights to create an enhanced curriculum that bridges the gap between ICT and CS, with the goal of combining the best of the two worlds.

3 Curriculum Scope & Outcomes

The curriculum duration is intended for three academic years. The following detail is for year 1 curriculum only. Due to variation of the duration of teaching days in an academic year between different schools ranging between 150-190 days [4], the curriculum breakdown was distributed in weeks analogous to the term weeks in the local schools where our previous CS-only curriculum was implemented. The curriculum can, however, be distributed to cater to different semester requirements of different schools.

As already mentioned, the goal of the curriculum is to combine ICT with CS so the students are not only capable of using basic ICT tools but also obtain knowledge of important concepts of CS early on. The intended outcome from this curriculum is that the students can comprehend the key concepts of CS such as problem decomposition, solution synthesis, sequential and parallel programming as well as be able to utilize ICT tools to accomplish daily tasks such as using word processors, web editor, presentation software, etc.

One of the biggest strength areas of the curriculum is that it provides a cultural context so that the students can relate to the content being learned [21]. The curriculum is integrated in full academic year and is intended to provide students with the following competencies. Students should be able to

- decompose a problem into smaller problems
- formulate a solution to the problem
- navigate through User Interface (UI) of any program to search for desired option
- create a document using word processor
- develop a basic website using a web authoring tool
- present an idea using a presentation software
- identify different parts of computer hardware
- understand difference between function and methods
- describe the need for sequential vs parallel programming
- apply event-based programming
- demonstrate the need for using loops
- comprehend basic elements required behind making a movie or an animation

Considering the competencies above, the curriculum can be implemented in any educational system as the curriculum is self-contained (does not require prior knowledge) and comprehensive.

4 Curriculum Design & Breakdown

A good curriculum must include a design intent that aims at solving or mitigating problematic areas experienced in targeted

educational system. Our design intent was based on the following questions:

- How can CS be integrated with ICT in a meaningful way to keep the students motivated towards the subject
- What level of focus shall be given to individual topics to ensure concise and timely coverage of the curriculum
- What strategy should be implemented to ensure the curriculum achieves its intended objectives

The curriculum spans for 1 academic year and since it is self-contained, it is well suited for Key Stage 3 (UK) or Grade 7 (US) students and equivalent grades around the world. Considering the tech-savvy nature of modern students, an interface that offers visual response with 3-D characters is used to explain CS topics and to keep the content "enjoyable". Topics in ICT are included as tools that serve specific purposes and needs of students.

The jump from CS curriculum to ICT curriculum occurs not just to suffice the course outcome of the school, but out of a need to accomplish a portion of the main task or a part of a "purpose".

To accomplish our objective of delivering the topics of CS and ICT in a convenient package, the topics of CS and ICT were intertwined in the curriculum. The details of the curriculum are described below.

4.1 Introduction to Computing

To develop contextual base for the students to enable them to easily grasp concepts for the course, two introductory topics; Introduction to Computing, and Introduction to Alice are taught.

Introduction to computing is taught with reference to computer hardware and software to give students a sense of the composition of a computer system. Students get a chance to see the different components of the computer by getting hands-on experience with a Raspberry Pi [26] to identify the processor, input/output ports, power supply unit and memory.

A brief representation of binary number system is also taught so that students can comprehend the computer's operational domain. As students are already familiar with the decimal number system, we relate the binary number system to the decimal number system which allows the students to find new appreciation for binary as well as the decimal number system.

The introduction to Computing module provides students with a general understanding of the field which allows them to understand the integration of CS with their everyday life. In this module, students learn algorithms and the process of writing these algorithms. They also briefly learn the fundamental concepts of CS, such as databases, concept of networking, concept of abstraction, etc. This gives students a better idea of the different areas and applications that constitute the field of CS.

4.2 Learning Alice Development Environment

Alice is a program that uses 3-D animations and storytelling to teach programming concepts. Students use 3-D models provided within the software to create worlds that allow the students to bring their ideas and stories to life [13]. Since 3-D environments provide several degrees of freedom for each component of a model, there is an initial learning curve associated with the tool. We spend the first few weeks getting familiar with the tool and the features that it provides. We chose Alice as our tool as it provides us with ways of discussing

advanced topics like abstraction, lists, and parameters that are harder to explain in other equivalent drag-and-drop environments. We plan to teach these advanced topics in higher grades as discussed in section 5 and section 6 of this paper. Each model (3-D objects in Alice) has methods associated with it that allow certain actions to be performed on these models. Alice has several coding structures that represent basic programming blocks available in most imperative programming languages. Users can drag and drop these blocks to perform specific actions on or by the models.

Since Alice allows the students to play and view the animations they create, students get instant feedback on their code. This develops a habit of testing and debugging among the students. Students are able to use incremental and iterative development that is driven through testing [22].

4.3 Sequential and Parallel Execution

The first thing that students start learning in the CS part of the course is the concept of algorithms. At the heart of this concept is the sequence of step-by-step instructions that help us achieve an end goal. We start with the concept of sequence of instructions and the importance of their order. As mentioned earlier, students learn to manipulate objects in Alice by making the objects move and turn as a whole, or manipulate the different parts of the object individually. The correct execution of any animation will depend on the sequence in which different objects move. Alice provides a block called "do in order", where each instruction in the block is executed in the same sequence as specified in the block. Students also learn that once an instruction is executed, no memory of it is retained except any effects that the instruction produced. Students practise the concept of algorithmic thinking where, for an algorithm to work correctly, all steps of an algorithm must exist. These steps must be in the right sequence, and there should not be any extra steps that are not part of the algorithm. Since Alice provides visual feedback of user code, students immediately see the problems with their algorithm.

The second concept we focus on is parallel execution. In traditional programming, the concept of parallel processing in the shape of threads is considered an advanced topic and definitely not suitable for middle schools students in most cases. Animations, on the other hand, require many steps to be performed together. A character walking in an animation will look completely unnatural if the arms and legs move in sequence as the character moves forward. Alice provides a "do together" block that allows for parallel execution of each instruction. Furthermore, Alice also allows for a sequence of "do together" blocks where each parallel block executes in sequence. Students learn the difference between sequential and parallel execution at the very beginning of their CS education.

These two blocks in conjunction with each other allow students to produce substantially complex animations. At this point, they are ready to start looking at design strategies for their animations before writing any code.

4.4 Design with Storyboarding

Upon completion of sequential and parallel execution modules, students obtain skills required to create a moderately sophisticated animation. To capture students' imagination, storyboarding is taught

so that students can plan the story for their animation. "A storyboard is a graphic representation of how your video will unfold, shot by shot" [5]. Storyboarding provides students a convenient ground, where students can focus on their story rather than directly coding in Alice.

To achieve this, a paper template is provided where students draw a scene or several scenes for their story. Storyboarding also helps students in planning which is relevant to Alice such as selection of the environment they shall be using, objects they shall be adding and the actions or events that shall take place in the story. In reference to CS, storyboarding allows students to visualize the problem, the story, as a whole and then breakdown the problem into manageable steps that can be executed in Alice.

In the storyboard phase of the course, students have to create several scenes that require pictures, story lines or dialogues as text, and objects they would need for each scene. This is an ideal situation to introduce our first ICT topic of using a word processor. Due to the widespread use of Microsoft Word as a word processor, we focus our curriculum on this tool.

Students learn to conveniently add text to a document, insert images, make bullet lists, highlight important parts of the text, add headings for better presentation, and perform spell checks. To learn the formatting of documents we also cover topics of adjusting page layout, adding page numbers, and they also learn to save the file to be opened later. This provides a motivated approach by which students learn MS Word to solve a real problem i.e. composing their story in words. We make sure that the topics we covered in MS Word are sufficient for the students to develop storyboard documents. This teaching in context allows students to understand the use of each feature and its application.

4.5 Camera and Sound

To create animations that are realistic and professional, most animation tools provide means of manipulating camera position and sources of light. Camera motion also allows for quick changes in scenes in different parts of an animation. Alice allows for Camera manipulation using code. Students learn ICT aspects in camera motion in animations along with CS concepts of sequential and parallel camera motion to obtain desired effects.

The next module in the curriculum deals with sounds. Alice allows for custom sounds to be uploaded in an animation and played in sequence or in parallel with code. Based on our experience in using Alice to teach CS in local schools, as described in section 7, students at this point are getting saturated. Hence, processing their own sounds in their animations piques their interest again.

Adding sounds to animations motivates the need to understand how sound works and studying tools that allow us to manipulate sound files. This naturally leads us to covering a sound processing tool like Audacity®.

4.6 Audacity®

Audacity is a free tool for recording, modifying and composing sound tracks [1]. The motivation behind teaching students Audacity is to help them learn an ICT tool to control another dimension of their animation i.e. sound. Students in this module learn to record their sound in Audacity and add effects to their recorded track. The features taught in this module can help students tackle real

problems such as removing background noise, creating fade in/fade out effects, adding echoes, deleting the portion of the track, increasing/decreasing sound frequencies to create drama, composing two or more sound tracks into a single file etc.

4.7 Communications and Presentation

Communications and presentation form a basic necessity for 21st century competency and is irrespective of any field. A major part of our curriculum is to train students in tools and techniques that allow them to present their ideas to a wider audience. At this point in the curriculum, students are able to make animations that are substantially sophisticated and include the use of cameras and sounds where appropriate. At this point, we ask them to work on a project that will include all the skills they have learned so far, but there is a caveat. Before they can start working on implementing their animations, they need to present their ideas plus their storyboard designs to their classmates. This is a natural cue for us to introduce a presentation software as the next ICT topic. Again due to its ease of access and wide spread usage, we pick Microsoft PowerPoint as the presentation software.

It is worthy of noting that PowerPoint is not taught when students are introduced to MS Word during storyboarding, as there was no "need" in the context and flow of the curriculum. Also, since students are already familiar with the MS Word interface and capable of adding and formatting text and pictures, features such as adding slides and choosing themes is taught in this module. This module also encourages students to explore various entertaining features of the PowerPoint such as slide transitions, slide animations, animation controls, etc. The PowerPoint slides are utilized by students during oral presentation of their projects in class. Students present their ideas to their peers, which helps their presentation and communication skills as well as increases their self-confidence.

4.8 HTML

Another part of ICT that is commonly taught is web design using HTML. At this point in the curriculum, we want students to take their ideas, designs, and presentations to the wider audience of the World Wide Web. The students learn HTML tags and their usage to render a basic website and to create an index page with a header to link other pages. The students also learn to change the background color of the website and to design the body by using page break tags as well as embedding the video into the body. At the end of this module, students are required to have a complete web page with an introduction about themselves, story line of their animation and an embedded video of their animation.

4.9 Using Functions and Math Expression

To be able to create more sophisticated animations, students need familiarity with more Alice features. Students are taught to use functions to add more control into their animation but most importantly the context is used to teach students "abstractions" which is one of the fundamental concepts of CS. Alice provides built-in functions that allow users to inquire about different properties of an objects. Functions return values based on the property that is being evaluated. For example, we can use the "distanceto" function to check the distance between two objects. Functions provide a means of abstraction where the details of how a task is working

is abstracted out in order to utilize "what" a task does in a useful manner. Although, at this point, students do not write their own functions, using built-in functions gives them a better understanding of the concept. The use of math goes hand-in-hand with these functions. For example, the notion of the two objects "close enough" is carried out by making sure the distance between two objects is less than some specific value. This forces students to make decisions about ranges of appropriate values according to the needs of their animations.

4.10 Methods, Events, and Loops

At this point in the curriculum students already have an animated story that is substantially complicated. Due to that, there is a very clear need for students to be able to organize their code better and to be able to reuse some of the code they have already written. This "need" allows us to introduce the concept of task decomposition in CS. Students learn to decompose their "useful chunk" of the code in Alice into individual modules or methods. The useful chunk here pertains to the part of the code that can be used and tested separately. Alice supports two methods of two scope, **World Level Methods** and **Object level Methods**. Students learn to utilize object level methods to code in isolation to the rest of its environment, and world level methods to segregate useful code of an object's interaction with other objects.

Alice also allows creation of event driven animations or games, which are one of the most engaging topics among youths. This module teaches the students how to add keyboard events to the animation to make the animation responsive. Students learn the basics of interactive programming by small exercises. For example, one exercise features a helicopter in an airport and requires the use of keyboard cursors to move the helicopter around the airport before landing it on the ground. This exercise helps students learn how to link the events to specific methods and how to take advantage of this feature in designing their own game or interactive scene in Alice.

The last topic in this curriculum is added to ease repetition of tasks in Alice by introducing loops. Students learn For-loop in this module to automate the part of the code that is repeated several times in a row. Alice provides a "Loop block" that allows users to specify the number of times instructions in the block should be repeated. Assuming a scenario where student's animation features helicopter rotors turning a complete circle, to create 100 complete rotations, students will have to copy-paste the code 100 times to achieve the desired effect. Learning loops allows the students to realize how effortlessly they can bring the same functionality by using the simple loops command.

At this point, we only introduce repetition in the form of "for loops". We coverage "while" loops in later grades since that requires them to understand Boolean logic - topics not covered in this grade.

Table 1 shows the breakdown of the curriculum.

5 Advance Topics (Year 2 and Year 3)

The topics of this curriculum were selectively chosen to ensure students are not overloaded with more content than they can handle. Certain topics were removed from the curriculum based on the teachers' response in the pilot curriculum and were introduced in the next grade. The topics in this curriculum are self-contained to

Table 1: Breakdown of the Topics in Weeks

Topics	Weeks
Introduction to Computing	3.5
Introduction to Computer Science	1
Introduction to Alice	2
Sequence and Parallel Execution	3
Storyboarding	4
Camera and Sound	4
Audacity	2
Communications and Presentation	2
HTML	4
Function and Math Expressions	2
Methods, Events, Loops	4

create a continuum of context. The topics which were removed shall be introduced in the later grade and include Boolean logic, conditional execution, while loops, random numbers, parameters, variables, functions, and lists, and recursion.

6 Year 2 and Year 3 curriculum pedagogy

In the first year of the curriculum, we designed a curriculum that uses a drag-and-drop and visual interface to learn programming concepts. Once the students are comfortable with these concepts, they take Python in year 2 to learn computing concepts in the context of a text-based programming language. However, in year 2, our goal is to take away the drag-and-drop interface only and focus on teaching the syntax of text-based programming while keeping the visual aspect to assist the students with the transition. We introduce Python through the use of Turtle Graphics library. Turtle Graphics is a library in Python that introduces programming through the creation of pictures and shapes. Upon the use of the Turtle Graphics library, students are provided with a virtual (x-y) plane with a turtle starting at (0,0). Students get the feeling of text-based programming in a fun and interactive way. In the third year of the curriculum, now that students are comfortable and familiar with text-based programming in a visual environment, we take away the visual environment. Text-based programming is all about the steps the computer must take to create what we want to give to the user. The reason behind taking away the visual environment is to assist the students to think like a computer to be able to translate human needs to processes the computer can process.

7 Curriculum Implementation

One important aspects of the curriculum design is implementation methodology. CS-only curriculum was implemented in more than 50 schools encompassing more than 6000 students [21, 22]. Schools introduced this CS-only curriculum by shifting part of their existing ICT curriculum in earlier grades to make room to allocate 2 days a week, 60 minutes class per day dedicated to teaching the CS-only curriculum. In the process, a great deal of experience and feedback from teachers was generated, which formed the basis of our implementation strategy for the CS/ICT curriculum.

The CS/ICT curriculum is implemented by providing five support media namely, continuous teacher training, course books, lesson plans, student assessment, and surveys. In academic year 2019-2020, 4 schools that were part of the CS-only curriculum switched

to using the CS/ICT curriculum. In the Academic year 2020-2021, 8 new schools and 4 old schools are adopting this CS/ICT curriculum. In the year 2021-2022, a total of 37 schools have committed to use our CS/ICT curriculum. These schools teach UK based curriculum, ICTA based curriculum, CBISE, FBlSE, and other regional curricula.

The steps taken to implement a curriculum and then the follow up with the teachers and students are provided below

- (1) Conduct information sessions where schools are invited to understand the curriculum
- (2) Teachers are trained for the Schools interested in using the curriculum
- (3) Course books, lesson plans, assessment and surveys are provided to the teachers
- (4) During the year, focus groups and surveys are conducted
- (5) Assessment and survey results are shared with the schools and teachers
- (6) The curriculum is upgraded based on the response of the stakeholders

As teacher's readiness was identified as a major concern to the CS education community, [9, 11], dedicated workshops were required so teachers are able to implement the curriculum in class and develop their own lesson plans for the course.

8 Teacher Training

The CS-only curriculum started off with training teachers before implementing the curriculum for Grade 7 [21]. Local and public schools that use UK, US or national curriculum were teaching only ICT until very recently. Even schools that teach programming were not teaching CS concepts instead, they were teaching systematic ordered steps to get a program to run in a regimented fashion without any focus on concepts. When our CS-only curriculum was implemented in 2013, almost none of the teachers were familiar with CS.

Hence our team designed a teacher training workshop that provided multiple training sessions for teachers on the CS concepts. These workshops focused on teaching the tool (Alice) as well as methodologies for teaching CS concepts. Learning from the successful experience of CS-only teacher training [23], we continued the same training approach in the CS/ICT curriculum training. Teachers are trained on the new curriculum in a 5 day workshops with 2 hour of training per day.

Table 2: Teacher Training content by days

Training content	days
Introduction to Alice as a tool	Day 1
Review of 3-year curriculum and pedagogy	Day 2
Camera, Loops	Day 3
Methods, Function and Math Expressions	Day 4
Events	Day 5

Furthermore, our workshops include computing concepts, such as: sequential execution, algorithms, conditionals, loops, etc. To execute the workshops we either invited various schools to one venue, or visited new schools adopting the CS/ICT curriculum to provide the training.

Some schools expect their students to cover some specific ICT related topics as part of their school system standards. This also influenced our decision to create the new curriculum presented in this paper. Therefore, we included the same teachers who interacted with our previous CS-only curriculum program in the creation of the new curriculum. Teachers naturally know their students' capabilities and the time needed to cover ICT topic. The CS-only curriculum teachers inclusion in developing the CS/ICT curriculum was extremely vital. The topics of the curriculum were distributed in full consultation from the teachers involved in the CS-only phase.

The key factor is that previously trained teachers at our CS-only phase will be ready to teach the new course. They already had a longer time with ICT topics and they experienced teaching CS during pilot phase. Our focus will be on the newly joined teachers from either new joining schools or first-year teachers.

9 Results and Lessons Learned

Our results and observations complement the concerns of the CS academic community regarding teaching of CS concepts. Based on the response of school management, the ICT coordinators and the teachers to whom this curriculum was presented to, it was evident to see that there exists a gap in the current curriculum where ICT and CS topics are integrated in a cohesive need-based manner and this gap is being filled using our CS/ICT curriculum.

4 teachers participated in the teacher's satisfaction survey. The survey asked questions about teacher's overall experience, student learning, class experience, books, lesson plans and supporting material. 75% of the teachers showed positive response to the questions related to better student learning, enjoying while learning, application of curriculum learning to real life. 2 focus groups were conducted in the year 2019-2020. During the focus group a need for regular training sessions and interaction among other schools was identified.

For this reason, along with having an initial 5 day teacher training, in the Academic year 2020-2021, we will hold teacher's training/ meetings once every 3 weeks. During these training sessions, teachers from all our partner schools will share their teaching experience and will be trained on a new topic every meeting. The proposed topics that are will be taught include pedagogy, educational learning, CS concepts, and teaching methodologies.

Schools were invited to 2 information sessions held in the academic year 2019-2020. Approximately 50% of the school that attended the information sessions agreed to implement the curriculum in the year 2020-2021 and year 2021-2022. Majority of the remaining schools showed keen interest in using the curriculum however, their approval dependent on the board members of the schools therefore we are still waiting for their response. This implies that bureaucracy is an obstacle for the schools in adopting this curriculum.

10 Acknowledgements

This paper was made possible by NPRP grant # 10-0205-170345 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the author[s].

References

- [1] 2012. Audacity Team (2012) Audacity®. Version 2.0.0. Audio editor and recorder. <http://audacityteam.org/>. Accessed: 2019-01-27.
- [2] 2016. ELHS Course Descriptions 2016-18. <http://www.elhs.org/home/140001978/140006806/140142830/2016-18Curriculum.pdf>
- [3] 2017. *After the reboot: computing education in UK schools*. <https://royalsociety.org/topics-policy/projects/computing-education/>
- [4] 2018. The Organisation of School Time in Europe. https://eacea.ec.europa.eu/national-policies/eurydice/sites/eurydice/files/school_calendar_2018_19_final_report_0.pdf
- [5] 2018. What Is A Storyboard And Why Do You Need One? <https://www.vyond.com/resources/what-is-a-storyboard-and-why-do-you-need-one/>. Accessed: 2019-01-27.
- [6] Petia Assenova, Rumen Nikolov, Ivan Stanchev, and Jorjeta Koleva. [n. d.]. Teaching Informatics in the Bulgarian Schools. *Technology-Based Education Series Cross National Policies and Practices on Computers in Education* ([n. d.]), 139–155. https://doi.org/10.1007/978-0-585-32767-9_7
- [7] Kathy Athayde. 2004. *Senior 2 (20S), Senior 3 (30S), and Senior 4 (40S) Computer Science*. Manitoba Education, Citizenship and Youth, Winnipeg.
- [8] Anja Balanskat and Katja Engelhardt. 2015. *Computer programming and coding Priorities, school curricula and initiatives across Europe*. European Schoolnet.
- [9] Neil Christopher Charles Brown, Michael Kölling, Tom Crick, Simon Peyton Jones, Simon Humphreys, and Sue Sentance. 2013. Bringing Computer Science Back into Schools: Lessons from the UK. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. ACM, New York, NY, USA, 269–274. <https://doi.org/10.1145/2445196.2445277>
- [10] Tom Crick and Sue Sentance. 2011. Computing at School: Stimulating Computing Education in the UK. In *Proceedings of the 11th Koli Calling International Conference on Computing Education Research (Koli Calling '11)*. ACM, New York, NY, USA, 122–123. <https://doi.org/10.1145/2094131.2094158>
- [11] Tyne Crow, Andrew Luxton-Reilly, Burkhard C. Wünsche, and Paul Denny. 2019. Resources and Support for the Implementation of Digital Technologies in New Zealand Schools. In *Proceedings of the Twenty-First Australasian Computing Education Conference (ACE '19)*. ACM, New York, NY, USA, 69–78. <https://doi.org/10.1145/3286960.3286969>
- [12] Valentina Dagieni and Gabriele Stupuriene. 2016. Informatics Concepts and Computational Thinking in K-12 Education: A Lithuanian Perspective. *Journal of Information Processing* 24, 4 (2016), 732–739. <https://doi.org/10.2197/ipsjip.24.732>
- [13] Wanda P Dann, Stephen Cooper, and Randy Pausch. 2008. *Learning to program with Alice*. Prentice Hall Press.
- [14] Diana H J M Dolmans, Willem De Grave, Wolfhagen, and Cees P M Van Der Vleuten. 2005. Problem-based learning: future challenges for educational practice and research. <https://onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2929.2005.02205.x>
- [15] G. Gueudeth, L. Bueno-Ravel, S. Modeste, and L. Trouche. 2017. Curriculum in France: A national frame in transition. *International perspectives on mathematics curriculum* (2017), 41–70.
- [16] Susan Imel. 2000. Contextual Learning in Adult Education. Practice Application Brief No. 12. (2000).
- [17] Peter Micheuz. 2008. Some Findings on Informatics Education in Austrian Academic Secondary Schools. *Informatics in Education* (2008), 221–236.
- [18] Miranda C Parker and Leigh Ann Delyser. 2017. Concepts and Practices: Designing and Developing A Modern K-12 CS Framework. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. ACM, 453–458.
- [19] Don Passey. 2017. Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies* 22, 2 (2017), 421–443.
- [20] Keith R Potter. 1999. Learning by doing: A case for interactive contextual learning environments. *Journal of Instruction Delivery Systems* 13 (1999), 29–33.
- [21] Saquib Razak, Huda Gedawy, Wanda P Dann, and Donald J Slater. 2016. Alice in the Middle East: An Experience Report from the Formative Phase. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. ACM, 425–430.
- [22] Saquib Razak, Huda Gedawy, Nour Tabet, Wanda Dann, and Don Slater. 2015. Alice in UK An Alice Based Implementation of UK National Computing Curriculum. In *International Conference on Computer Science Education Innovation and Technology (CSEIT)*. *Proceedings*. Global Science and Technology Forum, 29.
- [23] Samah Al Sabbagh, Huda Gedawy, Hanan Alshikhabobakr, and Saquib Razak. 2017. Computing curriculum in middle schools: an experience report. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*. ACM, 230–235.
- [24] Royal Society. 2017. After the reboot: computing education in UK schools. Retrieved January 20, 2019 from <https://royalsociety.org/~media/events/2018/11/computing-education-1-year-on/after-the-reboot-report.pdf>
- [25] Farnam Jahanian Tom Khalil. 2013. Computer Science is for Everyone! Retrieved January 20, 2019 from <https://obamawhitehouse.archives.gov/blog/2013/12/11/computer-science-everyone>
- [26] Eben Upton and Gareth Halfacree. 2014. *Raspberry Pi user guide*. John Wiley & Sons.