Achieving Computational Thinking competencies using the BBC micro:bit

Markus Wieser[1], Nina Lobnig[1], Stefan Pasterk[1] and Andreas Bollin[1]

[1]*University of Klagenfurt, Universitätsstraße 65-67, 9020 Klagenfurt am Wörthersee, Austria*

Abstract

Computational thinking becomes more and more important in society and teaching, as it is often part of digital literacy. In Austrian lower secondary education, "basic digital education" is implemented to cover these topics. Its curriculum includes a section for computational thinking, even though the included competencies are vaguely formulated and loosely connected to those promoted by literature. Another issue is that the specific competencies that characterize computational thinking are not clearly defined in literature either. This contribution presents a competency model for computational thinking to determine which topics are part of computational thinking. This model is then used to check to what extent the micro:bit units designed by the Informatics Lab at our university teach computational thinking and which areas are covered. For a meaningful comparison, the competencies of the learning materials are defined and then compared to the developed model of computational thinking. The results answer the question of which areas of computational thinking these micro:bit learning materials are covering and they make visible what is possible with the BBC micro:bit in this context.

Keywords

computational thinking, competencies, competency model, computer science education, micro:bit.

1. Introduction

Although it is often associated with programming, computational thinking is a really broad field and goes back to early work of Seymour Papert [1]. It became more important for society and teaching informatics in the last years, as it is often part of digital literacy. This situation was leading us to the Question: "How could the amount of computational thinking in learning materials be measured and improved?" In order to solve this question we had to start at the beginning, asking the question: "What is computational thinking?" However, there are quite some definitions, best known is the one formulated by Jeanette Wing [3]. Based on it this work aims to systematically collect all the necessary competencies and maps them to a competency model. This can be used on the one hand to get specific competencies, for example for curriculum developers, and on the other hand to find out how much computational thinking can be found in individual materials and how they can be improved in this area. For testing its applicability we created another competency model of the micro:bit materials designed by the Informatics Lab at our University for Students between the ages of 10 and 14. We then compared the

Informatics in Schools. A step beyond digital education. 15th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2022, Vienna, Austria, September 26–28, 2022

markus.wieser@aau.at (M. Wieser); nina.lobnig@aau.at (N. Lobnig); stefan.pasterk@aau.at (S. Pasterk); andreas.bollin@aau.at (A. Bollin)

models asking the Question: "How much computational thinking do the materials contain?" and finally formulated some recommendations to improve the amount of computational thinking of the materials. In the following chapter computational thinking is defined and related work and "basic digital education" are discussed, then, in chapter three, a competency model for computational thinking is created. In chapter four we create another competency model based on the materials for the micro:bit, which we compare to the first model in chapter five, where we also formulate some recommendations to improve the materials. In the final chapter we talk about conclusions and future work.

2. Computational thinking

2.1. Definition and related work

When talking about computational thinking, there is no way around the paper of Jeanette Wing with the title "Computational Thinking" [2]. In her article she writes: "It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use." and "Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction." [2]. For Wing, computational thinking is conceptualizing, not programming, there are used fundamental, not rote skills, and it is a way that humans, not computers think. This means, computational thinking is about representing a real-world problem in an abstract way, not about the ability to program, even though this is of course very important for implementing the idea.

Another important point that is often misunderstood is, that computational thinking means thinking like a human. Computational thinking represents the way people solve problems. It does not aim at making people think or act like computers. The goal is to use computers to solve problems that probably could not be solved without them.

In her paper "Computational thinking. What and why?" published in 2010, Jeanette Wing answers the question what is computational thinking and she defines it as follows [3]:

> "Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent."

Informally, a problem should be formulated in a way it can be processed by humans, computers, or more generally, combinations of humans and computers.

Most definitions of computational thinking found in literature are related to the one of Wing. For example David Barr, a K12-Teacher, says [4] that computational thinking is a problem solving process including:

- Formulating problems in a way that enables us to use a computer and other tools to help solve them.
- Logically organizing and analyzing data,

- Representing data through abstractions, such as models and simulations,
- Automating solutions through algorithmic thinking (a series of ordered steps),
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources,
- Generalizing and transferring this problem-solving process to a wide variety of problems.

As one can see, a lot of skills are necessary to achieve the goal of formulating a problem in a way, humans and machines are able to process it. Thus we are using the definition of Jeanette Wing for our paper and further explanations.

The definition of computational thinking is only the first step. Based on it, we will define a competency model. In literature there are many papers defining competencies of computational thinking like David Barr. These are for example Grover and Pea [5], Weintrop et al. [6], Computational thinking for professionals [7] and Jeanette Wing herself [3]. The competencies are very similar to each other. The main difference is that Jeanette Wing defined two sets of computational thinking skills. One basic skill set for everyone and one set of advanced skills for scientists, engineers, and other professionals.

David Barr [4] defined another set of meta skills, which support the learning and using of computational thinking, but are not necessary for achieving it. These are: *confidence in dealing with complexity*, *persistence in working with difficult problems*, *tolerance for ambiguity*, *the ability to deal with open-ended problems*, and *the ability to communicate and work with others to achieve a common goal or solution.* As these skills can be seen as requirements for computational thinking they are not considered in the process of the development of our competency model.

Apart from papers about computational thinking competencies there are also many tests and surveys for testing computational thinking skills, like CTT [8], BCTT [9], the Callysto CTi [10] or the Bebras Test [11]. We used these tests as base for creating the competency model first, but then learned, these tests mostly just check competencies related to algorithmisation and computational thinking is a lot more than just that.

2.2. Computational thinking in "basic digital education"

Taking a look into education, computational thinking appears in different curricula all around the world. One example for it is the current curriculum for "basic digital education" in Austria's first four years of secondary education [14]. The curriculum is competency-oriented and contains learning outcomes for different areas. It is structured into eight topics including one called "computational thinking". This topic consists of all together ten competencies, which are categorised into the subtopics "Working with algorithms" and "Creative use of programming languages". Sample competencies for the first subtopic are "Students formulate clear instructions for action (algorithms) verbally and in writing." or "Students discover similarities and rules (patterns) in action instructions." The second subtopic focuses with competencies like "Students

know different programming languages and production processes." or "Students master basic programming structures (branching, loops, procedures)." on programming. This shows that in this sample curriculum computational thinking is limited to algorithms and simple programming. The new competency model is intended to change this one-sided view of computational thinking.

3. Competency Model for Computational Thinking

The approach of creating a competency model of computational thinking based on the available tests was not as successful as we thought at the beginning, because it only covered a small part of computational thinking. Thus we decided to create our model based on the definition of computational thinking by Jeanette Wing [3] using a top-down approach. Starting from the definition she defined two sets of computational thinking skills. One for everyone and one for experts. Computational thinking for everyone means being able to:

- (WB1) Understand which aspects of a problem are amenable to computation
- (WB2) Evaluate the match between computational tools and techniques and a problem
- (WB3) Understand the limitations and power of computational tools and techniques
- (WB4) Apply or adapt a computational tool or technique to a new use
- (WB5) Recognize an opportunity to use computation in a new way
- (WB6) Apply computational strategies such divide and conquer in any domain

Computational thinking for scientists, engineers, and other professionals further means being able to:

- Apply new computational methods to their problems
- Reformulate problems to be amenable to computational strategies
- Discover new science through analysis of large data
- Ask new questions that were not thought of or dared to ask because of scale, but which are easily addressed computationally
- Explain problems and solutions in computational terms

Starting with these two sets of high level competencies, following our top-down approach, we had to break them down into lower level ones. In this step we found two types of competencies. Some could be derived semantically, others only syntactically. For example: If we want to split the competency "(WB3) Understand the limitations and power of computational tools and techniques" we could create the competencies "(WB3-1) Understand the limitations of computational tools and techniques" and "(WB3-2) Understand the power of computational tools and techniques" by splitting them analyzing the syntax of (WB3). Both are obviously child competencies of the first one. Deriving semantically on the other hand is not that simple. In order to create child competencies from "Understand which aspects of a problem are amenable to computation" one has to understand the meaning of the competency. First one has to understand a problem. This could be done by using models, diagrams or graphs. Then one has to decompose the now understood problem into smaller problems or aspects in order to find out which parts are amenable to computation. The next step is to understand computation. This is not as easy

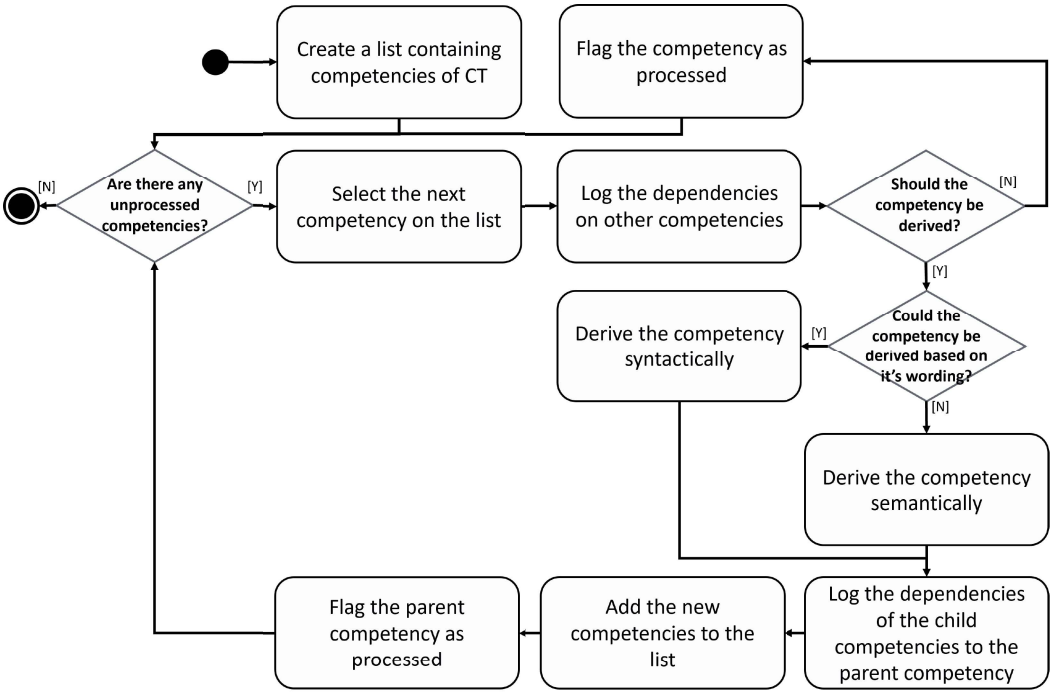

Figure 1: Workflow for deriving competencies

as it seems, but anyway it will be another competency. Based on this, one has to evaluate, if a problem is amenable to computation and how much power or time is required for the algorithm.

Now we created four new child competencies based on the competency "(WB1) Understand which aspects of a problem are amenable to computation". These are: "(WB1-1) Understand a problem using modelling, diagrams and graphs", "(WB1-2) Decompose a problem into smaller problems", "(WB1-3) Understand amenable to computation" and "(WB1-4) Evaluate, if a problem is amenable to computation and how much power or time is required for the algorithm". In Fig. 1 one can find the workflow for the derivation of the competencies.

If we look at the skill set, we learn, that the second competency (WB2), additionally depends on the first competency (WB1). "Understand which aspects of a problem are amenable to computation" is needed to achieve "Evaluate the match between computational tools and techniques and a problem". Based on the first competency (WB1) we get the structure seen in Fig. 2. The parent competencies are located in "Layer 0", the child competencies in "Layer 1". Each of these competencies could be derived and new child competencies could be created and the child competencies also could be derived again. To keep it as small as possible, we decided to limit the model to three layers, so each main competency is derived two times. The first layer competencies (Layer 0) are those defined by Jeanette Wing [3]. The main competencies also depend on each other and the basic ones are necessary to achieve the expert competencies.

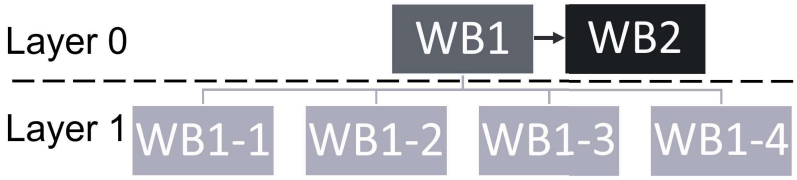

Figure 2: The competencies directly depending on WB1

In total, the competency model created consists of 130 competencies derived from the sets above. In this paper we will only show selected parts, the whole competency model could be found here [12].

The workflow described above has been applied to all the competencies of the basics set leading to the following competencies on "Layer 1":
Child Competencies of WB1:

- (WB1-1) Understand a problem using modelling, diagrams and graphs
- (WB1-2) Decompose a problem into smaller problems
- (WB1-3) Understand amenable to computation
- (WB1-4) Evaluate, if a problem is amenable to computation and how much power or time is required for the algorithm

Child competencies of WB2:

- (WB2-1) Understand and select computational tools and techniques
- (WB2-2) Validate whether a computational tool or technique is suitable for solving a problem.
- (WB2-3) Apply given computational tools and techniques to solve a problem.
- (WB2-4) Evaluate which computational tools and techniques are most appropriate for a problem.
- (WB2-5) Apply computer science tools and techniques to a problem

Child competencies of WB3:

- (WB3-1) Understand the limitations of computational tools and techniques
- (WB3-2) Understand the power of computational tools and techniques
- (WB3-3) Understanding computer architecture
- (WB3-4) Understand the concept and operation of register machines
- (WB3-5) Understand and apply basic algorithms and data structures

Child competencies of WB4:

- (WB4-1) Applying a computational tool for a new purpose

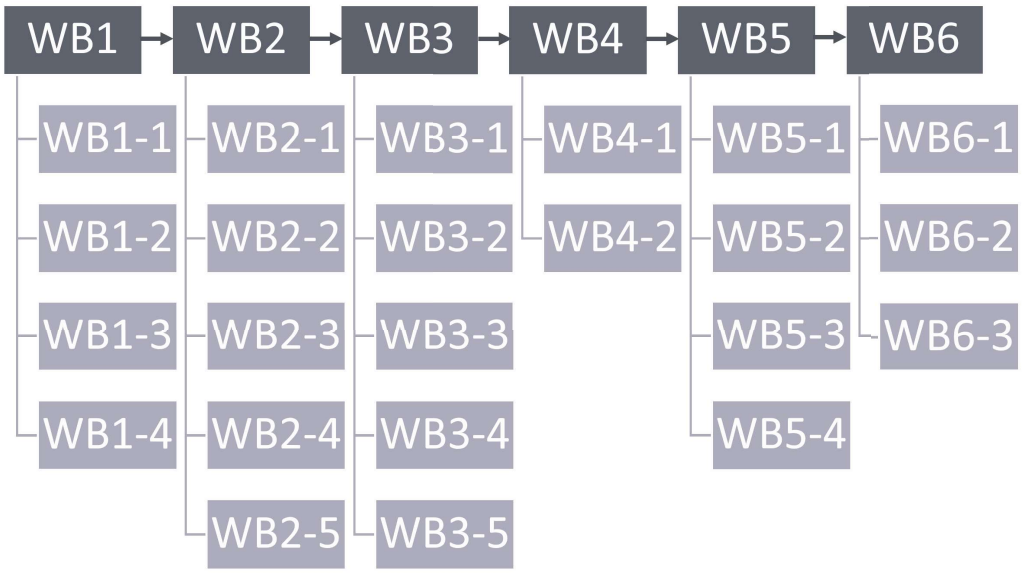

Figure 3: Overview of the structure of the basic skill set competency model in two layers

- (WB4-2) Adapting a computational tool to other requirements

Child competencies of WB5:

- (WB5-1) Identify commonalities between problems
- (WB5-2) Transferring a problem solving process to a different problem.
- (WB5-3) Generalization of a problem solving process to a class of problems.
- (WB5-4) Find similarities and differences between computational tools and techniques

Child competencies of WB6:

- (WB6-1) Divide problems into categories and assign a problem to a category
- (WB6-2) Apply design paradigms, such as Divide & Conquer or the Greedy Method, in any domain
- (WB6-3) Apply programming concepts, such as alternative or recursion in any domain

This leads to the competency model shown in Fig. 3.

4. Competencies achievable using the BBC micro:bit

After the creation of the competency model of computational thinking we created another competency model from the materials of the Informatics Lab to test what amount of computational thinking is covered by them. The materials were designed to teach programming to children between the ages of 10 and 14, using the BBC micro:bit. We decided to use the materials,

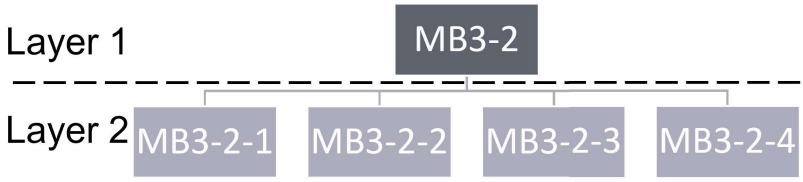

Figure 4: The competencies related to MB3-2

because they are designed for teaching programming, which requires algorithmization. This is part of computational thinking and therefore seems to be a good base for comparison. The materials are split into six parts, covering six different topics. They are: "Basics", "Inputs and random numbers", "Branches", "Variables", "Advanced Branching" and "Loops". Each topic contains tasks for practicing and builds on the previous ones. The materials can be found on the website of the Informatics Lab [13].

However, this time we have different requirements for creating the competency model. In the model of computational thinking we started from the definition and some high level competencies, now our source are the practicing tasks and the information given in the materials. So we had to use a different approach. Instead of top down, which was used to create the competency model of computational thinking, we now use a bottom up approach, starting with the practicing tasks and given information. We formulated competencies for every task and information found in the materials and listed them. For example, for the task "If the micro:bit is shaken, it should display a symbol of your choice, otherwise a question mark. If both keys (A + B) are pressed at the same time, it should randomly decide whether a large or a small heart is displayed." We have two possibilities in the second part, either a small heart or a large heart. This only can be resolved using branching and random numbers,l so we created the competency "Applying branching and random numbers". After the formulating process, we grouped the list into three categories: micro:bit specific competencies like "Use the programming environment", computational thinking competencies and math competencies like "Understand random numbers". This is due to the fact we had one main category about using the micro:bit environment and another one based on computational thinking. In the third category we put the competencies which could not be assigned to any of the previous ones.

After that, we categorized the competencies of computational thinking and formulated parent competencies for them. For example, the competencies "(MB-3-2-1) Be able to write programs sequentially", "(MB3-2-2) Understanding and applying branching", "(MB3-2-3) Understanding and applying variables" and "(MB3-2-4) Understanding and applying loops" are being merged to the parent competency "(MB3-2) Understand and master programming concepts such as branching, variables, and loops". This leads to the structure seen in Fig. 4.

Following the path to the top we get the following main competencies:

- Understanding and using the micro:bit user environment

- Apply (new) computational methods to known problems
- Reformulate problems to make them amenable to computational strategies
- Decompose a problem into individual sub problems, solve them, and combine the partial solutions
- Understanding and applying basic mathematics

The competencies MB2, MB3 and MB4 are connected to computational thinking, the other two are not. Sure one could argue without basic mathematics it is not possible to achieve computational thinking, but again basic mathematics are seen as requirement for computational thinking and for this reason not taken into account.

The whole competency model of micro:bit contains 81 competencies on 4 Layers. 39 of these are child competencies of "Understand and master programming concepts such as branching, variables, and loops". This shows that the main purpose of the materials, teaching basic programming concepts to children, is achieved in those areas. Another property of the micro:bit competency model is the fact that opposite to the computational thinking model, the main competencies are mostly independent.

5. Comparison and Recommendations

After creating both models, we compared them to each other by comparing the competencies. Competencies which are occurring in both competency models are:

- Reformulating problems to make them amenable to computational strategies
- Apply new calculation methods to known problems
- Adapting a computational tool to other requirements
- Recognize a way to use computational methods in a new way
- Finding a specific solution to a problem using appropriate computational tools and techniques
- Understand and master programming concepts such as branching, variables, and loops

As mentioned before, the materials for the micro:bit are designed for students between 10 and 14, so the competencies have to be adapted to the age level. Thus we decided to distinguish between fully achieved competencies and partly achieved ones. In a next step we matched the micro:bit competencies to their suitable counterpart in the computational thinking model. The result on the first two layers can be found in Fig. 5. The framed are competencies of computational thinking which are partly achieved by the micro:bit materials.

After the comparison we analyzed the merged model and the micro:bit materials again, focusing on optimization. With the help of the competency models we found some areas of computational thinking which could easily be covered by the micro:bit materials just with a few modifications. These areas could be:

- A higher focus on understanding and explaining in the tasks

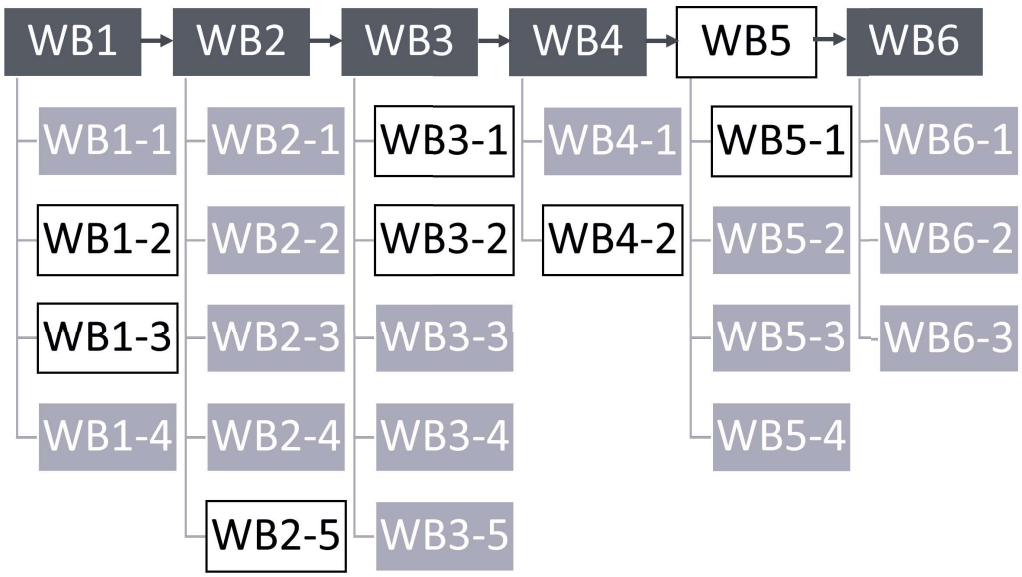

Figure 5: Partly achieved competencies of computational thinking by the micro:bit materials

- Selection of methods and techniques
- Consider debugging
- Discuss classifying problems
- Optimize solutions
- Add basic algorithms and data structures

All these recommendations could easily be applied to the materials with just a few modifications except for the basic algorithms and data structures, which should be added as an extension. These recommendations have a great impact on the coverage of the computational thinking competency model as can be seen in Fig. 6. The black competencies are the ones, which would be additionally covered by applying the recommendations.

6. Conclusion and Future Work

Out of the 49 Competencies of the first two layers of the computational thinking model, there are 15 at least partly achieved by the micro:bit materials of the Informatics Lab. If we apply the recommendations, we talk about 12 added and 27 in total. So we have an initial coverage of 30.6% and after applying the recommendations, which are covering 24.5%, we have a total coverage of 55.1% of at least partly achieved competencies of computational thinking in the micro:bit materials.

Although it is only weakly represented in the some approaches like for instance the

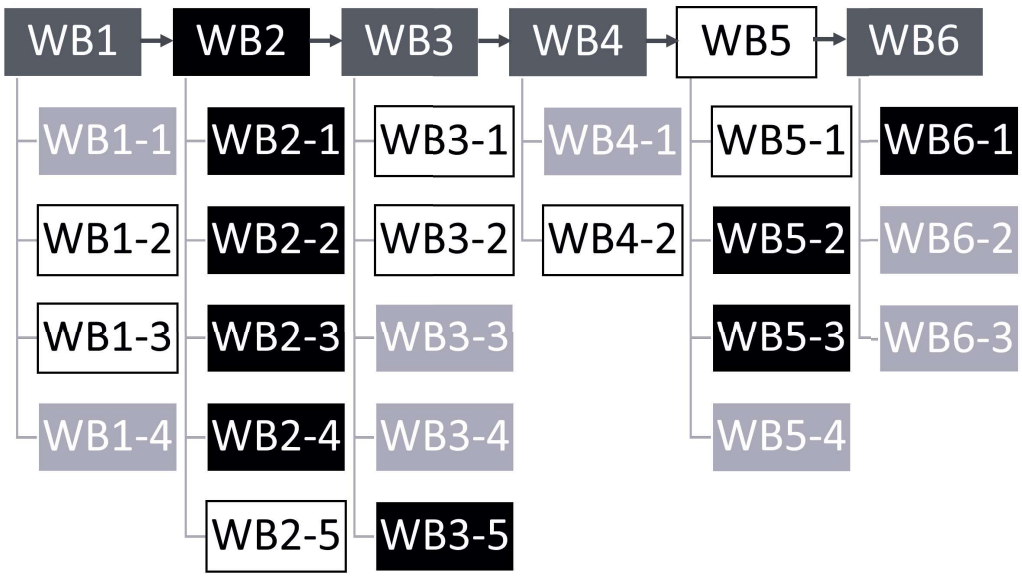

Figure 6: Additionally covered competencies by applying the recommendations

curriculum of "basic digital education" in Austria, computational thinking is a really broad field. Our competency model breaks it down into smaller parts that are easier for curriculum writers, for example, to work with. The model can also be used to check how much computational thinking is found in certain documents. However, this assumes that a competency model of the materials exists. Based on the comparison, recommendations can be made as to what extent the amount of computational thinking could be increased. Our computational thinking model is therefore a good way to analyze, which competencies are covered by some learning materials and to what extent this coverage still could be optimized.

To make it even more usable, an extension by using Bloom's taxonomy would probably be desirable. This would make the future model even more extensive, but the improved structure would make it easier to use, which in turn could encourage more curriculum authors to use the model. This could also lead to an even stronger anchoring of computational thinking in the curricula, which would be a very welcome development.

The competency model of computational thinking could also be relevant for the Informatics Lab at our university. This makes it possible to check all materials of the lab for their content of computational thinking and to determine which competencies are covered and which are missing. For the missing competences, new materials could be developed or existing ones could be extended in order to cover as much of computational thinking as possible.

References

[1] Papert, S.: Mindstorms, Children Computers and Powerful Ideas. Basic Books (1980)

[2] Wing, J.: Computational thinking. Communications of the ACM, **49**, 33–35 (2006)

[3] Wing, J.: Computational thinking. What and why? (2010)

[4] Barr D., Harrison J. and Conery L.: Computational thinking: A digital age skill for everyone. Learning and leading with technology, **38**, 20–23 (2011)

[5] Grover S., Pea R.: Computational thinking in k–12 a review of the state of the field. Educational Researcher, **42**, 38–43 (2013)

[6] Weintrop D. et al.: Defining computational thinking for mathematics and science classrooms. Journal of Science Education and Technology, **25**, 127–147 (2016)

[7] Denning P., Tedre M.: Computational thinking for professionals. Communications of the ACM, **64**(12), 30–33 (2021)

[8] Bati, K.: Computational thinking test (ctt) for middle school students. (2018)

[9] Zapata M. et al.: Computational thinking test for beginners: Design and content validation. (2020)

[10] PIMS. Callysto, https://www.callysto.ca/computational-thinking-tests/. Last accessed 18 May 2022

[11] Bebras, https://www.bebras.org/. Last accessed 18 May 2022

[12] Wieser M., Entwicklung eines Kompetenzmodells des Informatischen Denkens im Kontext blockbasierter Programmierung am Beispiel des micro:bit, https://resolver.obvsg.at/urn:nbn:at:at-ubk:1-43345. Last accessed 18 May 2022 (German)

[13] Regionales Fachdidaktikzentrum Informatik: micro:bit Grundlagen. https://www.rfdz-informatik.at/grundlagenmicrobit/. Last accessed 26 May 2022

[14] Bundesministerium für Bildung, Wissenschaft und Forschung Österreich: Lehrplan Digitale Grundbildung. https://www.ris.bka.gv.at/Dokumente/BgblAuth/BGBLA_2018_II_71/BGBLA_2018_II_71.html. Last accessed 20 May 2022

[15] Bundesministerium für Bildung, Wissenschaft und Forschung Österreich: Digitale Grundbildung - Begutachtungsentwurf. https://www.ris.bka.gv.at/Dokumente/Begut/BEGUT_25796D77_3C78_4325_A420_58ADC71458CC/BEGUT_25796D77_3C78_4325_A420_58ADC71458CC.pdf. Last accessed 20 May 2022