

Computational thinking assessment at primary school in the context of learning programming

Manargul Mukasheva^a, National Academy of Education named after Y. Altynsarin, Mangilik El st. 8, Nur-Sultan 010000, Kazakhstan; <https://orcid.org/0000-0002-8611-8303>

Aisara Omirzakova^b, L.N. Gumilyov Eurasian National University, Faculty of Information Technology, Pushkina st. 11, Nur-Sultan 010000, Kazakhstan; <https://orcid.org/0000-0003-1068-0145>

Suggested Citation:

Mukasheva, M. & Omirzakova, A. (2021). Computational thinking assessment at primary school in the context of learning programming. *World Journal on Educational Technology: Current Issues*. 13(3), 336-353. <https://doi.org/10.18844/wjet.v13i3.5918>

Received from; January 03, 2021; revised from March 23, 2021; accepted from October 05, 2021.

Selection and peer review under responsibility of Prof. Dr. Servet Bayram, Yeditepe University, Turkey.

©2021 Birlesik Dunya Yenilik Arastırma ve Yayıncılık Merkezi. All rights reserved.

Abstract

The study was carried out from 2018 to 2020 with the challenge - how to assess the level of computational thinking. The research design is mixed since the disclosure of mutual influence of the components of the chain 'learning programming – computational thinking – evaluating computational thinking' requires the use of both qualitative and quantitative research methods. The conceptualisation of the 'computational thinking' idea is based on the premise of the impact of abstract thinking and computers on human thinking evolution. The structural interpretation of 'computational thinking', consisting of nine components, reflects the presence of a semantic link between teaching programming and the development of abstract computational thinking. Four levels (phenomenological, analytic-synthetic, set-prognostic and axiomatic) of computational thinking have been identified for each of these nine components. The study involved 102 elementary school students who are learning programming in Scratch. The guiding questions and problems we have developed for elementary school students are designed following the characteristics of the four levels of computational thinking. The results of the study showed that the ratio of 'structural components' to different levels of computational thinking, with the corresponding characteristics, allows one to determine the degree of its development or its individual components.

Keywords: Computational thinking, educational taxonomies, evaluation methodologies, levels of development;

* ADDRESS FOR CORRESPONDENCE: Manargul Mukasheva, National Academy of Education named after Y. Altynsarin, Center for Scientific Research, Mangilik El st. 8/1507, 010000 Nur-Sultan, Kazakhstan,
Email Address: mg.mukasheva@gmail.com

1. Introduction

With each passing year, there are more and more scientific studies on the impact of the global information flow and ubiquitous digitalisation on human cognitive abilities. This phenomenon of computational thinking is attracting more and more attention from scientists. Researchers have paid attention to the fact that targeted training in modelling the surrounding reality, algorithmisation and programming for computer systems has a significant impact on the development of cognitive abilities of students.

Assumptions and hypotheses that programming knowledge and skills can influence the development of cognitive abilities, in particular the development of thinking abilities, have initiated the solution of other research tasks in this direction. The most relevant issues for today are 'Are there any tools that can determine the level of development of computational thinking of a student?' and 'In what ways is the level of development of computational thinking assessed?'

This research, like other diagnostic works, assumes that to study the dynamics of computational thinking, it is necessary to develop level characteristics and corresponding qualitative and quantitative indicators that facilitate the assessment of individual structural components of computational thinking as a result of learning. Qualitative indicators determining the dynamics of computational thinking development characterise the dynamics of the development of computational thinking components, and quantitative indicators allow us to get an interpretation of empirical data of the study. This methodology of studying the dynamics of computational thinking development can become the most promising tool that will contribute to systematisation and generalisation of approaches, concepts to computational thinking studies, as well as identification of new phenomena and patterns in the development of computational thinking.

2. Theoretical foundations

In connection with the objective of our research, during the review of most scientific research, the impact of the learning process on the development of cognitive abilities, in particular, on the development of thinking and related intelligence, was studied; methods to diagnose the levels of development of thinking ability and intelligence were also considered. It would be wrong not to take into account the strong influence of the initial foundations on the development of thinking and intellect, which is often mentioned by Piaget (1951) in his research: 'sensorimotor intelligence is a foundation of thinking and it will continue to influence it throughout life through perceptions and practical situations'.

Papert's interpretation of Piaget's theory, influenced by the results of children's learning programming (Logo), showed that Piaget's defined stages of thought process development have a fundamental significance and are specific to particular life evolutions. Papert's (1980) comprehension of these stages in the context of children's early learning in the fundamentals of computer programming allowed us to highlight one important point concerning the possible stimulating impact of computer science on the development of thought processes. As we have established, Papert's assumption about the late transition to the stage of formal operations is confirmed by the conclusions of psychological studies of famous scientists such as Vygotsky and Bruner.

The study of the famous psychologist Vygotsky (1934), devoted to the development of thinking and speech, the problems of interrelation between psychological development and the process of learning, confirms the fundamental function of the learning process in the development of the child's thinking. The most important is that his experiments have shown that in these periods, the development of thinking and other higher psychological functions are influenced by the child's social and cultural development, the sources of which are cooperation and learning. The results of his experiments also showed that not all the results of the learning process contribute to the development of thinking abilities, but only those that

children understand and accept. In learning, it is important that children can only learn what they are already able to learn, and learning is possible where there is a room for imitation. It is also important to identify the lowest and highest learning thresholds because learning can be fruitful only between these thresholds. Learning that does not take into account children's zone of proximal development and focuses on what children already know how to do cannot influence their development; such learning is simply focused on 'the line of least resistance, the weakness of the child, not the strength of the child' (Vygotsky, 1934). As a whole, in the conclusion of Vygotsky's study, it was defined that learning should be focused not on the past but on children's future development.

Bruner's studies also present numerous theoretical and empirical facts confirming the leading role of learning in the development of children's thinking abilities. According to the scientist, anticipatory learning is the most appropriate form of activity to provide children with 'tempting and feasible opportunities to accelerate their development' (Bruner & Anglin, 1974). The conclusion of Bruner's study concerning the support of the process of learning by cooperation and the formation of a scientific concept largely coincides with the ideas of Vygotsky. The majority of his conclusions confirm Papert's hypothesis that with time, when computers and programming become a part of children's everyday life, the age gap between the stages of thinking development (a specific operation and a formal operation) will gradually disappear. The main objective of learning at each of these points is to encourage children to accelerate the transition from one stage of intellectual development to another through cooperation (teachers) and learning tools (instruments). In the process of mastering basic concepts, the most important thing is to help children gradually move from specific thinking to using abstract ways of thinking (Bruner & Anglin, 1974).

Thus, the analysis of the above and other studies on the impact of learning on the development of thinking shows the advanced position of learning in the development of cognitive abilities, including the thinking of the child. In this context, based on these researchers' conclusions, it may as well be assumed that the main objective of learning programming at an early age in school is also to ensure a favourable transition from one stage of intellectual development to another through cooperation (teachers) and learning tools (instruments).

3. Measurement and evaluation of computational thinking

1.1. Computational thinking: Structure and contents

The preconditions of the impact of abstract computation and computers on the evolution of human thinking were first considered in the works of scientists in the field of programming and artificial intelligence such as Fink (1966), Papert (1980), Knuth (1981) and others.

The concept of 'computational thinking' appeared (Wing, 2006) relatively recently, and despite the absence of its generally accepted definition (Artym, Carbonaro & Boechler, 2017; Denning, 2017; NRC, 2010; Selby, 2015), it has begun to attract more and more scientists and researchers from different scientific fields, such as computer science, pedagogy, psychology, philosophy of thinking and others.

There is also an assumption that thought processes of computational thinking should include practical experience in the field where specific computational models are used (Denning, 2017). It does not contradict the concept of computational thinking, but obtaining and assimilation of new knowledge will take place at a high professional level of education, for example, university.

There are also other reasonable conclusions of scientists in which computational thinking is considered as a result of the natural evolution of the human understanding of computer science and which are not connected with the use of specific programmes or programming languages (Koh & Repenning, 2014).

Despite the existence of different approaches to the definition of computational thinking, most researchers note that computation and computational models are getting into all areas of science, including social and human sciences, and the impact of these phenomena on the development of cognitive abilities of man is emphasised (Artym et al., 2017; Denning, 2017).

The three aspects of computational thinking for Scratch suggested by Brennan and Resnick (2012) – computational concepts, computational practices and computational perspectives – include syntactic, semantic, schematic and strategic programming knowledge. This approach to defining computational thinking is supported by researchers as the most appropriate framework for studying various aspects of computational thinking in the context of programming instruction (Kong, 2019; Lye & Koh, 2014).

Different concepts for defining computational thinking and its structural components have been discussed in detail in Artym et al.'s (2017) study. These and other approaches to defining computational thinking, as well as the characteristics often used to measure programming skills, led to defining the following components of computational thinking for its generalised interpretation (Appendix A).

The descriptions of each component of computational thinking presented by us are obtained based on summarising the findings of studies by various authors in different periods (Appendix A). As the table (Appendix A) shows, in most studies, the concept of 'computational thinking' is based on the premise of the impact of abstract thinking and computers on the development of human thinking. Nevertheless, all these components are united by the focus on the development of high-level thinking skills. Considering them together allows computational thinking to be presented as a holistic and multifaceted process. We also believe that it is this holistic interpretation of computational thinking that contributes to the exploration of the semantic relationship between learning programming and computational thinking.

Probably, not all components of computational thinking are presented in this interpretation, since the concept of 'computational thinking' and its properties are dynamically developing and supplemented by new components. Nevertheless, this interpretation of computational thinking was the basis for studying computational thinking in this research.

3.2. Educational taxonomies: evaluation of cognitive skills in the process of learning

The authors consider educational taxonomies of Bloom, Engelhart, Furst, Hill and Krathwohl (1956), Structure of the Observed Learning Outcomes (SOLO) (Biggs & Collis, 1982) and Bepalko (1989) as a methodological tool for revealing the levels of development of computational thinking.

Bloom's taxonomy, which covers the cognitive (field of knowledge and thinking), affective (domains of emotions and feelings) and psychomotor (manipulation and field of action) fields of learners' development, is noted for its broad functionality and flexibility in evaluating learning outcomes, as well as for its ease of perception.

Studies on computational thinking present a model of the relationship between learning programming, the skills (components) of computational thinking and the cognitive domain of Bloom's taxonomy (Selby, 2015). The author of the study notes that Bloom's taxonomy was chosen as a basis for arranging programming skills, as this study does not focus on how well the student has mastered programming skills, but on identifying the highest level of cognitive ability developed through these skills.

Other educational taxonomies define the level nature of cognitive processes and the development of various learning and cognitive skills, including thinking skills.

The quantitative (prestructural, unistructural and multistructural) and qualitative (relationship and extended abstract) SOLO taxonomies show that learning and development go from a monostructured explanation to a multi-structured explanation of a problem, and then to understanding the

interrelationships of elements and finally to developing the skill of abstracting general conclusions and results (Biggs & Collis, 1982).

In his studies, Bepalko (1989) notes that under the conditions of the constantly growing volume of information nobody can study the entire content of any branch of science, but it is quite possible for any person to master the main methods of thinking and activity in the scientific field. To achieve this goal, it is recommended to select the most representative objects of science that provide full and reasonable activity, including successful further self-learning.

According to Bepalko, the important qualitative parameters of knowledge assimilation include the stage of knowledge abstraction (scientific content of learning), the degree of automation of the reproduction of the assimilated knowledge, awareness and strength of assimilation. According to Bepalko's taxonomy, the level of knowledge of students (learning outcomes) is described in two dimensions: on the one hand, taking into account the stage of abstraction β in the presentation of information about the phenomena of reality, and on the other hand, taking into account the level of assimilation α of this information. With the help of these terms, it is possible to show the dynamics of the development of knowledge, experience and thinking skills of students.

According to Bepalko's taxonomy, in the process of learning, in the development of levels (stages) of abstraction (β), there are the following levels: phenomenological level, analytic and synthetic level, attitudinal and prognostic level and axiomatic level.

i. Comparison of Bepalko's taxonomy with other educational dimensions

Bloom's taxonomy and Structure of the Observed Learning Outcomes (SOLO) taxonomy have shown that the main difference of Bepalko's method is the clearest representation of the characteristic of the abstraction levels β or scientific content of learning (Appendix B).

Moreover, in educational taxonomy (diagnostics), psychological features of age categories of students, as well as the content of the subject, forms and methods of education following levels of education (elementary level, basic senior level or students of higher education institutions) are of special importance. For example, there is an assumption that visualisation, pattern recognition and generalisation can be studied in K-2, and abstraction and critical thinking in grades 6–8 (Juskeviciene and Dagiene, 2018). The age and gender characteristics of learners in creating a tool for assessing the levels of development in computational thinking are also equally important for researchers.

We also noted that continuity in the development of thinking processes, conditional continuity in the process of learning, including learning the basics of programming, involves continuously learning the dynamics of computational thinking.

Thus, the methodological review of known educational taxonomies and the results of studies in the field of computational thinking allowed us to define the following important points that should be taken into account when developing taxonomic tools for its measurement and evaluation:

- considering the process of development of computational thinking as a complex set for the development of cognitive skills (cognitive components) with level characteristics;
- presenting the description of levels of development of computational thinking (structural components) in a broad context with the most detailed and flexible indicators and characteristics;
- comparison of quantitative indicators (e.g., scores) with qualitative characteristics of computational thinking components;
- correlating (or matching) the traditional system of evaluating learning outcomes of individual subjects/disciplines, such as programming;

- integrating into other taxonomies to study the development of different types of thinking;
- applying a wide range of diagnostic methods and tools for the study of thinking ability.

4. Methods and tools

The levels of development of computational thinking are defined by taking into account the fundamental function of the learning process in the development of thinking. This approach allowed us to use the level elements of Bloom’s, SOLO, and Bepalko’s taxonomy to describe qualitative characteristics of structural components of computational thinking for each level of its development in students.

The generalised structural interpretation presented (Appendix A), which consists of the nine best-known components of computational thinking, served as the basis for the development of a diagnostic method of evaluating the levels of development of computational thinking in the process of learning programming.

The methodology of evaluating the development of computational thinking developed by us, following Bepalko’s taxonomy, consists of four levels with corresponding qualitative and quantitative indicators. These indicators characterise the development of structural components of computational thinking in the learning process: phenomenological, analytic and synthetic, attitudinal and predictive and axiomatic level (Appendix C).

During the study, students were offered five tasks which aimed at determining the level of development of different components of computational thinking. The purpose of the first task (Task No. 1, Table 1) was to find out the level of abstract and conceptual perception of information about an object or a process, in particular, how they represent a situation, whether they understand the meaning of the task correctly and whether students can abstract text information. The tasks are presented in the form of text and verbal descriptions of the plot of the task and the characters involved in it (Table 1).

Table 1. Task No.1. Selection of scenario – plot for the Scratch project

Plot No.	Scenario – plot for the project
Plot 1	A big shark wanted to catch a fish. A boat is sailing. The shark saw the boat and sailed away. The fish is saved.
Plot 2	One day Red Kitten, Jolly Colt and Little Dinosaur met in the yard. They were playing hide and seek. Red Kitten hid in the woods. Jolly Colt and Little Dinosaur hid in the car.
Plot 3	It is Sailor Goose’s birthday. His friends, Golden Fish and Big Shark came to see him. They gave Goose a real car.
Plot 4	Jolly Colt is solving the problem, ‘Guess whose voice it is?’ ‘Meow’ is the voice of Red Kitten, ‘bow-wow’ is the voice of puppy Laika, ‘quack-quack’ is the voice of Sailor Goose. If Jolly Colt guesses right, he will get three apples.
Plot 5	Little Dinosaur is lost in the woods. He saw a boat in the woods.

The following four tasks were a continuation of the first task and were based on Raven’s progressive matrices made for children. The plots with Scratch characters were supposed to help identify the skill levels such as information analysis, information decomposition, action (process) forecasting and others. For example, the results of the second task (Task No. 2, Table 2) and the fifth task (Task No. 5, Table 3) show how the student understands the characters’ actions and operates the objects of his future project.

Table 2. Task No.2. Selection of objects for the Scratch project

















Jolly Colt is solving the problem, 'Guess whose voice it is?' 'Meow' is the voice of Red Kitten, 'bow-wow' is the voice of puppy Laika, 'quack-quack' is the voice of Sailor Goose. If Jolly Colt guesses right, he will get three apples. Indicate the characters and objects which take part in the story.							
1	2	3	4	5	6	7	8
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
							

Table 3. Task No.5. Creating the project plan. Selection of characters and objects for the Scratch project

Select the characters and objects for the story 'In the woods'. Prepare a scenario for the project.							
1	2	3	4	5	6	7	8
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
							

Each task was attributed scores on a 4-point scale, for example, Plot 5 was attributed 1 score (Task No. 1, Table No. 4) because it has no clear meaning and does not carry the essential information and details that are to be analysed to get the completed result, when task Plot 4 is chosen, the student gets the maximum score of '4'. For other tasks, such as task No. 2 and No. 5, students receive 1 score for one correct answer, 2 scores for two correct answers, 3 scores for three and a maximum of 4 scores for four or more correct answers.

Task No. 3 and task No. 4 demanded students to write the names of hidden characters in empty cells. Task No. 4 was intended to determine detailisation skills and understanding of classification, students distributed characters and objects to two scenarios suggested by a teacher.

The characters that were included in the test tasks were mainly taken from the Scratch characters' library. The colour version of the matrix, natural landscapes (sea, forest, winter, summer, etc.) and characters suitable for them, allowed us to make test tasks for different subjects which are attractive to the children of primary school age (9–10 years).

The calculation of Cronbach's alpha coefficient by SPSS Statistics, which is often used in psychology to check the reliability of the test, showed a sufficient level of reliability for the developed tasks ($\alpha = 0.75$). The average values of inter-point correlation and covariance were 0.374 and 0.412, respectively, which shows the acceptability of the use of such story problems to assess the development of computational thinking of primary school students.

5. Collection of data and results

The developed methodology was tested for the first time in October–December 2019. Students of 4th and 5th grade (9–10 years old) of general education schools who study the fundamentals of programming in the Scratch environment as part of their school programme took part in the research. The content of school programmes of the 4th and 5th grades includes a section on ‘Computational Thinking’. The main concepts, with which students become familiar within this section, are ‘object’, ‘command’, ‘action algorithm’, ‘executor’, ‘data types’, ‘variables’, ‘variable values’, ‘assignment operators’, ‘logical operators’, ‘choice of conditions’ and ‘simple and nested loops’.

Moreover, when choosing a group, the number and age of students in the group are taken into account, as well as the duration of students learning the fundamentals of programming in the computer science course and the experience of teachers who teach computer science. Measurements were taken within 1 academic hour (not exceeding 40 minutes) in writing.

Respondents were divided into three groups because the duration of students learning the fundamentals of programming was different. The number of respondents was 102; 33 of them studied programming in the Scratch environment during 2 academic semesters (1st group, 0.5 years), 35 of them studied programming during 6 semesters (2nd group, 1.5 years) and 34 of them studied programming during 10 semesters (3rd group, 2.5 years). In general education school, the school year is divided into four semesters.

The obtained results have shown that in the 1st group indicators in the form of the total sum of scores on tasks 1–5 are subject to the universal law of normal distribution with a standard deviation $\sigma_1 = 2.7$ (Figure 1).

The average value is 11.9

25th percentile – 10.0

50th percentile – 11.1

75th percentile – 14.0

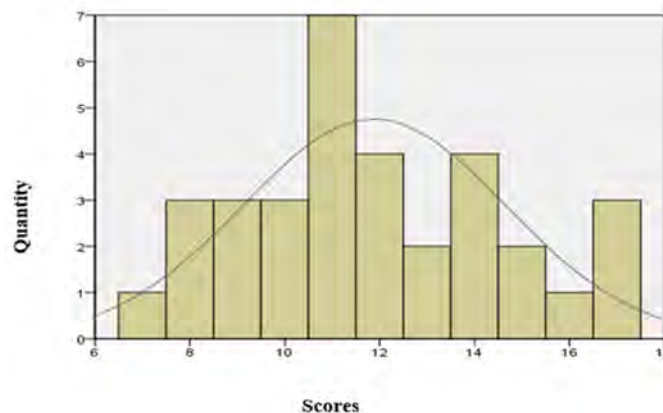


Figure 1. Histogram with a normal curve for the 1st group

The indicators of 34 students in the 2nd group, who studied programming for 1.5 years, deviate significantly from the normal distribution. For these indicators, the standard deviation is $\sigma_2 = 3.7$ (Figure 2).

The average value is 14.7.

25th percentile – 13.0

50th percentile – 15.0

75th percentile – 18.0

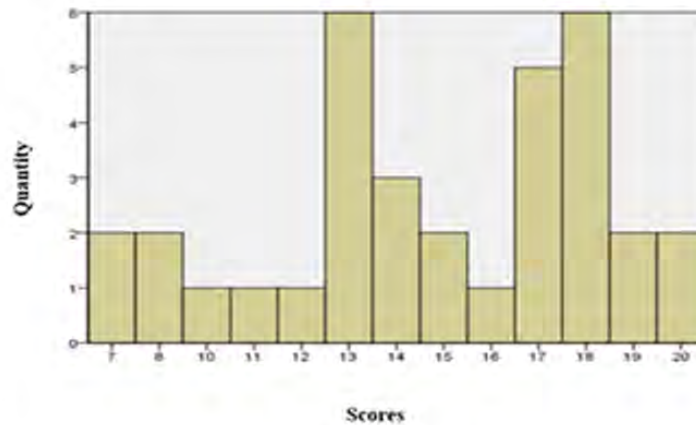


Figure 2. Histogram for the 2nd group

The data of the 3rd group correspond to the normal distribution law with the standard deviation $\sigma_3 = 2.4$ (Figure 3). The average value is 17.2.

25th percentile – 17.0

50th percentile – 18.0

75th percentile – 19.0

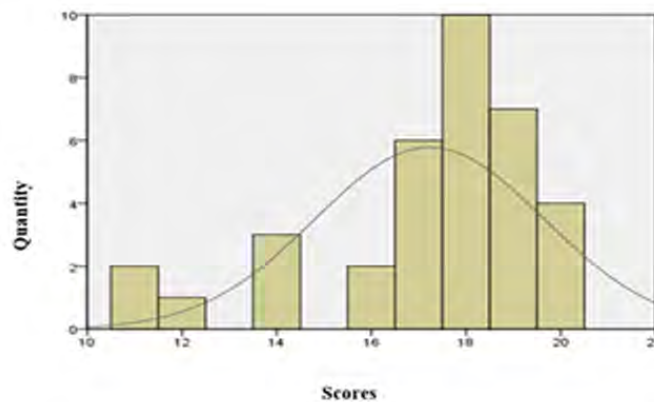


Figure 3. Histogram with a normal curve for the 3rd group

The indicators of all 102 participants in the above five tasks are within the interval [7; 20]. Taking into account the average value of 14.66 and the standard deviation $\sigma = 3.7$, the density of the distribution of the scores over the four levels of computational thinking development is shown in Table 4.

Table 4. The density of distribution of student scores by level computational thinking development

Levels of computational thinking development	Interval of scores density	Respondents
Phenomenological	[7,10]	15%
Analytical–synthetic	[11,15]	35%
Attitudinal–prognostic	[16,18]	35%
Axiomatic	[19,20]	15%

Furthermore, the χ^2 -Pearson test was used to determine the correlation between the score values of tasks 1–5 and the final semester grades in computer science. The analysis showed significant correlation links at the level of $\rho \leq 0.01$ between scores and grades ($\chi^2 = 0.658$), as well as between scores and duration of learning programming ($\chi^2 = 0.306$) (Table 5).

Table 5. χ^2 – Pearson test for the correlation of scores in tasks 1–5 and semester grades in Computer Science

		Semester grade	Scores in tasks 1–5	Duration of learning
Semester grade	Pearson’s correlation	1	0.658**	-0.037
	Value (two-sided)		0.000	0.712
	N	102	102	102
Scores in tasks 1–5	Pearson’s correlation	0.658**	1	0.306**
	Value (two-sided)	0.000		0.002
	N	102	102	102
Duration of learning	Pearson’s correlation	-0.037	0.306**	1
	Value (two-sided)	0.712	0.002	
	N	102	102	102

**Correlation is significant at level 0.01 (two-sided).

To calculate the average value of the final semester grades of the respondents of the 1st group, the authors took the grades for the last two semesters, of the 2nd group for 6 semesters and of the 3rd group for 10 semesters in Computer Science.

The empirical data also showed that the average values of scores on tasks in three groups differed significantly from each other, thus showing positive changes in the dynamics of computational thinking among students who studied programming for 1.5 and 2.5 years (Table 6).

Table 6. The average value of development evaluation indicators of components of computational thinking

Group No.	Average scores in tasks 1–5					Average scores
	Q1	Q2	Q3	Q4	Q5	
1st group	2.2	3.1	2.4	2.1	2.2	11.9
2nd group	2.4	3.0	2.6	3.4	3.4	14.7
3rd group	3.2	3.4	3.7	3.5	3.4	17.2

To verify the existence of a positive trend of quantitative indicator change when moving from one group to another, Jonkir’s criterion *S* was used. To answer the question ‘Is it possible to say that there is a certain tendency in the development of computational thinking when moving from one group to another on learning programming?’, the following hypotheses were tested:

H_0 null hypothesis: The trend of change in the quantitative indicator when moving from one group to another is random.

Alternative hypothesis H_1 : The trend of change in the quantitative indicator when moving from one group to another is not random.

The average of 10 randomly selected respondents from each group was used to calculate the empirical value of Jonkir's criterion (Table 7).

Table 7. Quantitative indicators to use the S -criterion of Jonkir's trends

Respondent No.	1st group		2nd group		3rd group
	Average scores	Quantity of high scores on the right	Average scores	Quantity of high scores on the right	Average scores
1	8	19	8	10	12
2	10	19	11	10	14
3	11	18	13	9	16
4	11	18	13	9	17
5	11	18	14	8	18
6	12	17	15	8	18
7	14	13	17	6	18
8	14	13	18	2	18
9	16	11	18	2	20
10	17	9	19	2	20
Sum	124	155	146	66	171
Average value	12,4	15,5	14,6	6,6	17,1

For this research, the empirical value of Jonkir's S -criterion $S_{emp} = 142$ falls into the zone of significance and confirms the validity of the alternative hypothesis (Figure 4).

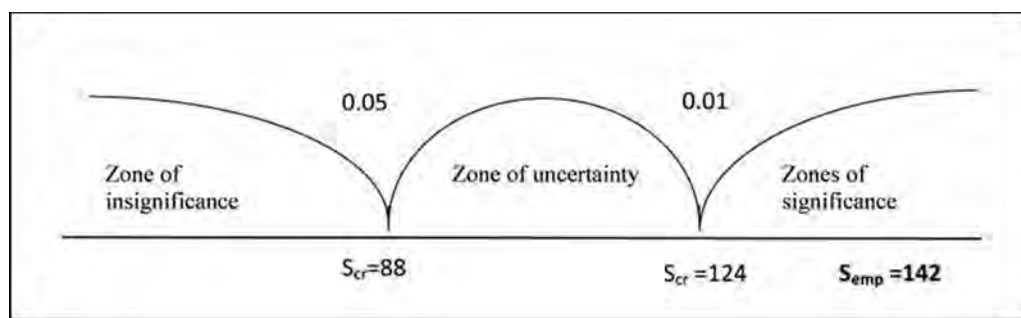


Figure 4. Empirical value of Jonkir's S -criterion

It is assumed that such a methodology provides an opportunity to show consistency, systematisation and integrity of the process of development of computational thinking in the process of learning not only programming but also other subjects. The flexibility of the methodology involves that it can be used to evaluate the levels of computational thinking of students at different levels of learning.

6. Discussion

Studies often note that it is difficult to find any tools or methods that would allow evaluating the level of development of computational thinking, including on the semantic level. Many attempts made by researchers have been limited to the study of skills, mainly at the syntactic or functional level (Koh & Repenning, 2014). The results of the content analysis of studies on computational thinking also showed that only 6 out of 65 studies focused on the evaluation of computational thinking ability (Kalelioglu, 2018). After examining the question, 'What are the methods and tools for testing or improving students' CT skills?', and different approaches of assessment of the computational thinking skills, Lockwood and Mooney (2018) concluded that work related to testing and evaluating computational thinking is at its initial stage. Researchers point out that it is necessary to continue research in this area to make computational thinking a common skill taught in schools.

The results of the study of Kong (2019) show that the characteristics of key programming skills prevail when evaluating computational thinking: defining data structure; organising cycles and checking conditions, procedures and parallel processing of data and processes; and using additional modules, libraries as well as interface design. From our perspective, estimating programming skills is more suitable for defining the levels of practical mastering of a programming subject than for developing computational thinking. That is why it will be reasonable to measure and estimate the levels of development of computational thinking as a cognitive skill, as well as to consider it in a wider context with the most detailed and flexible indicators and characteristics.

It was initially assumed that measuring the level of computational thinking will be considered in the context of learning programming and programming tasks will be used to evaluate its levels. However, during the research (especially during the theoretical and methodological analysis of the leading role of learning in the development of thinking), an idea to consider the assessment of computational thinking in a wider range appeared. It was also promoted by diversity and multidimensionality of the structure of the concept of computational thinking. In addition, different assumptions and hypotheses about early learning of the fundamentals of computer science and programming to develop computational thinking had a significant influence on the research. The inclusion of programming (computational thinking) in primary school curricula is being actively discussed in many countries around the world (Artym et. al., 2017; Kong, 2019; Mukasheva & Zhilbayev, 2016; Yadav, Mayfield, Zhou, Hambrusch & Korb, 2014). Some countries have included programming (or computation) in primary school curricula (The National Curriculum in England, 2014; National curriculum for basic schools, Annex 10, 2014).

Some results of our research, in particular, the significant deviation in the responses of students of the 2nd group (Figure 2) assume that continuity and consistency in targeted learning play an important role in the development of computational thinking. The study of the results showed that the 2nd group had respondents who studied computer science and programming with interruptions or repetitions due to different circumstances. It is assumed that these learners got low or high scores in tasks 1–5. A comprehensive study of this issue requires new methods and tools for measuring and evaluating computational thinking. Nevertheless, a sample of three groups with different duration of learning allowed us to test the methodology of evaluating the development of computational thinking in a short period.

In studies measuring and assessing computational thinking, the reliability of data collection tools and the number of scales confirmed by statistical analysis are important aspects. We support the view of Haseski and Ilic (2019) on the reliability and variety of measurement tools for evaluating computational thinking. Tests with open and closed questions can be effective in measuring and evaluating levels of development of computational thinking in primary school students. The development of tests or tasks to measure and assess computational thinking based on known techniques to determine levels of

development of thinking, intelligence and other cognitive abilities (e.g., Raven's Progressive Matrices, Cattell's Test, Eisenk's Questionnaires, etc.) helps to improve their effectiveness and reliability.

7. Conclusion

At this stage, the results of the research serve as a methodological tool for studying the dynamics of computational thinking development in the process of learning programming. It is assumed that the generalised interpretation of computational thinking, the developed method of evaluating computational thinking and story problems for primary school can contribute to the systematisation and generalisation of approaches and concepts to studying this concept, as well as identifying new phenomena and regularities in its development.

In our view, another equally important condition in the development of computational thinking is to ensure continuity in the process of learning that promotes its development at all levels of education. Continuity of the process of learning programming involves the acquaintance of students with programming elements from primary school with the gradual development of computational skills and abilities at subsequent levels of education. The trend of early programming education in school is supported by numerous leading IT vendors. These companies provide accessible programming tools and also widely support the idea of learning programming at schools.

Funding

This research is funded by the Science Committee of the Ministry of Education and Science of the Republic of Kazakhstan (Grant No. AP08856402)

References

- Artym, C., Carbonaro, M. & Boechler, P. (2017). Evaluating pre-service teachers' computational thinking skills in Scratch. *Ubiquitous Learning: An International Journal*, 10(2), 43–65. doi:10.18848/1835-9795/cgp/v10i02/43-65
- Ater-Kranov, A., Bryant, R., Orr, G., Wallace, S. & Zhang, M. (2010). *Developing a Community Definition and Teaching Modules for Computational Thinking: Accomplishments and Challenges*. SIGITE '10 proceedings of the 2010 ACM Conference on Information Technology Education, October 7–9, 2010, Midland, Michigan, USA. doi:10.1145/1867651.1867689
- Barr, V. & Stephenson, C. (2011). Bringing computational thinking to K-12. *ACM Inroads*, 2(1), 48–54. doi:10.1145/1929887.1929905
- Bespal'ko Vladimir Pavlovic. (1989). *Slagaemye pedagogicheskoy tehnologii*. Moskow, Russia: Pedagogika.
- Biggs, J. B. & Collis, K. F. (1982). *Evaluating the quality of learning: the solo taxonomy*. New York: Taxonomy Academic Press.
- Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H. & Krathwohl, D. R. (1956). *Taxonomy of educational objectives: the classification of educational goals*. London, UK: Longman.
- Brennan, K. & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking (2012)*. Annual American Educational Research Association meeting, Vancouver. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.296.6602>.
- Bruner, J. S. & Anglin, J. M. (1974). *Beyond the information given: studies in the psychology of knowing*. Crows Nest, Australia: George Allen & Unwin LTD.
- Denning, P. (2017). Computational thinking in science. *American Scientist*, 105(1), 13. doi:10.1511/2017.124.13
- Scratch (2019). *Statistics*. DR. SCRATCH. Retrieved from <http://www.drscratch.org/statistics>
- Fink, D. G. (1966). *Computers and the human mind*. New York, NY: Doubleday.

- Mukasheva, M. & Omirzakova, A. (2021). Computational thinking assessment at primary school in the context of learning programming. *World Journal on Educational Technology: Current Issues*, 13(3), 336-353. <https://doi.org/10.18844/wjet.v13i3.5918>
- Google. (2015). *Google for education: computational thinking*. Retrieved from <https://edu.google.com/resources/programs/exploring-computational-thinking/>.
- Grover, S. & Pea, R. (2013). Computational Thinking in K–12. *Educational Researcher*, 42(1), 38–43. doi:10.3102/0013189x12463051
- Haseski, H. I. & Ilic, U. (2019). An investigation of the data collection instruments developed to measure computational thinking. *Informatics in Education*, 18(2), 297–319. doi:10.15388/infedu.2019.14
- Juskevicius, A. & Dagiene, V. (2018). Computational thinking relationship with digital competence. *Informatics in Education*, 17(2), 265–284. doi:10.15388/infedu.2018.14
- Kalelioglu, F. (2018). Characteristics of studies conducted on computational thinking: a content analysis. *Computational Thinking in the STEM Disciplines*, 11–29. doi:10.1007/978-3-319-93566-9_2
- Knuth, D. E. (1981). *Algorithms in modern mathematics and computer science*. Proceedings, Urgench, Uzbek Ssr, September 16–22, 1979. Berlin, Germany: Springer-Verlag.
- Koh, K. H. & Repenning, A. (2014). *Computational thinking pattern analysis: a phenomenological approach to compute computational thinking* (dissertation). Retrieved from https://scholar.colorado.edu/concern/graduate_thesis_or_dissertations/5h73pw28v
- Kong, S.-C. (2019). Components and methods of evaluating computational thinking for fostering creative problem-solvers in senior primary school education. *Computational Thinking Education*, 119–141. doi:10.1007/978-981-13-6528-7_8
- Kramer, J. (2007). Is abstraction the key to computing? *Communications of the ACM*, 50(4), 36–42. doi:10.1145/1232743.1232745
- Lockwood, J. & Mooney, A. (2018). Computational thinking in secondary education: where does it fit? A systematic literary review. *International Journal of Computer Science Education in Schools*, 2(1), 41–60. doi:10.21585/ijcses.v2i1.26
- Lye, S. Y. & Koh, J. H. (2014). Review on teaching and learning of computational thinking through programming: what is next for K-12? *Computers in Human Behavior*, 41, 51–61. doi:10.1016/j.chb.2014.09.012
- Mukasheva, M. & Zhilbayev, Z. (2016). Continuous and ubiquitous programming: learning in Kazakhstan schools. *Ubiquitous Learning: An International Journal*, 9(2), 13–27. doi:10.18848/1835-9795/cgp/v09i02/13-27
- National Curriculum for Basic Schools. Annex 10. (2014). *Riigi Teataja*. Retrieved from <https://www.riigiteataja.ee/en/eli/524092014014/consolide>.
- National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. Washington, DC: The National Academies Press. doi:10.17226/12840.
- Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. New York, NY: Basic Books.
- Perlis, A. J. (1962). The Computer in the University. In M. Greenberger (Ed.), *Computers and the world of the future*. Cambridge, MA: MIT Press.
- Piaget, J. (1951). *The psychology of intelligence*. London, UK: Routledge & Kegan Paul.
- Selby, C. C. (2015). Relationships: Computational Thinking, Pedagogy of Programming, and Bloom's Taxonomy. *Proceedings of the Workshop in Primary and Secondary Computing Education*, 80–87. <https://doi.org/10.1145/2818314.2818315>
- Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G. & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: a theoretical framework. *Education and Information Technologies*, 18(2), 351–380. <https://doi.org/10.1007/s10639-012-9240-x>
- The International Society for Technology in Education (2011). *Operational Definition of Computational Thinking for K–12 Education*. Retrieved from <https://cdn.iste.org/www-root/ct-documents/computational-thinking-operational-definition-flyer.pdf>

The national curriculum in England (2014). Retrieved from https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/381344/Master_final_national_curriculum_28_Nov.pdf.

Vygotsky, L. S. (1934). *Myshlenie i rech. Psihologicheskije issledovanija*. Moskva – Leningrad, Russia: Gosudarstvennoe socialno jekonmicheskoe izdatelstvo.

Wing, J. (2011). *Research notebook: computational thinking--what and why?* Pittsburgh, PA: Carnegie Mellon School of Computer Science. <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35. doi:10.1145/1118178.1118215

Wirth, N., Wirth, N. & Wirth, N. (1986). *Algorithms and data structures*. Upper Saddle River, NJ, Prentice-Hall.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S. & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, 14(1), 1–16. doi:10.1145/2576872

Appendix A. Generalised interpretation of computational thinking

Components	Brief description	Research sources/authors
Abstract and conceptual perception	Representation of the area which the object (process, task and problem) belongs to; a full understanding of the meaning of the task	Fink (1966), Papert (1980), Knuth (1981), Ershov (1981), Wing (2006), Kramer (2007), Ater-Kranov, Bryant, Orr, Wallace and Zhang. (2010), NRC (2010), Barr and Stephenson (2011), ISTE (2011), Brennan and Resnick (2012), Grover and Pea (2013), Sengupta, Kinnebrew, Basu, Biswas and Clark. (2013), Google (2015), Selby (2015), Scratch (2015), Artym et al. (2017)
Examination and analysis of information	Identification of main and indirect (hidden) attributes (details), properties and causes; identification of incoming (initial data) and outgoing (results) data and their structure; data presentation	Perlis (1962), Knuth (1981), A.P. Ershov (1981), Barr and Stephenson (2011), ISTE (2011), Brennan and Resnick (2012), Grover and Pea (2013), Scratch (2015), Artym et al. (2017)
Decomposition of information (task, situation and processes)	Identification of subtasks and links between subtasks; distribution of data by subtasks	Perlis (1962), Knuth (1981), NRC (2010), Ater-Kranov et al. (2010), Barr and Stephenson (2011), Grover and Pea (2013), Google (2015), Selby (2015), Scratch (2015), Artym et al. (2017)
Forecasting the results and outcomes of actions (processes) and decisions made	Establishing links with other similar (identical) objects (processes and tasks); comparing future results with current results; identifying advantages and disadvantages of the intended result	Perlis (1962), Brennan and Resnick (2012), Google (2015), Selby (2015)
Forecasting implementation or simulation	Intuitive handling available, inaccessible properties and methods of objects (processes and tasks); intuitive possession of various interfaces and digital devices for implementation; offering the best option (tool) for implementation and justification	Perlis (1962), Ater-Kranov et al. (2010), Barr and Stephenson (2011), ISTE (2011), Scratch (2015)
Planning the algorithm	Planning the actions for achievement of the result by taking into account their dependence on each other; determining the order of their implementation; considering alternative actions in exceptional situations	Perlis (1962), Knuth (1981), Ershov (1981), Ater-Kranov et al. (2010), Brennan and Resnick (2012), Sengupta et al. (2013), Scratch (2019), Google (2015), Selby (2015), Artym et al. (2017)
Automation of problem solution search	Availability of automated thinking skills for the search for problem solution	Wing (2006), Barr and Stephenson (2011), ISTE (2011)
Creating template (pattern) solutions based on the generalisation of knowledge	Generalisation of knowledge and ability to sum up and create pattern solutions	Knuth (1981), Ater-Kranov et al. (2010), NRC (2010), ISTE (2011), Wing (2011), Grover and Pea (2013), Google (2015), Selby (2015)
A critical re-evaluation of the decision or result	The rationale for the effectiveness of the decision; an indication of the shortcomings of the decision; proposal for an alternative solution	Ater-Kranov et al. (2010)

Appendix B. Collation and comparison of Bloom's, SOLO and Bepalko's educational taxonomies

Bloom's taxonomy (Bloom, 1956)	Structure of the Observed Learning Outcomes (SOLO) taxonomy (Biggs & Collis, 1982)	The Methodology of Diagnostic Description of the Purpose of Forming the Experience of Students (Bespalko, 1989)	
		Levels of information assimilation (α)	Levels of scientific knowledge abstraction (β)
<p><i>Knowledge</i> Remembering and reproducing specific facts and content of the learning material (information).</p> <p><i>Comprehension</i> Interpretation of learning material; the assumption about the further course of phenomena, events, the transformation of learning material from one form of expression into another.</p> <p><i>Application</i> Using the knowledge assimilated in solving specific situational problems and stories.</p> <p><i>Analysis</i> Dividing the whole into parts, while maintaining a connection or structural relationship between them.</p> <p><i>Synthesis</i> Creating a whole consisting of individual components or subsystems based on principles or concepts that are new to students.</p> <p><i>Evaluation</i> Evaluation of the meaning of information, results and conclusions</p>	<p><i>Prestructural</i> An initial idea of the situation or object; just an understanding of the information.</p> <p><i>Unistructural explanation</i> Identification of at least one cause or fact or investigative link to explain the task, situation or information.</p> <p><i>Multistructural Explanation</i> Identification of several grounds, facts and investigative connections (without connections, i.e. unrelated to each other in any way) for an explanation.</p> <p><i>Correlation and interaction (Relationship)</i> Systemic and complex understanding of the task, identification of interrelated causes and facts and relationships; a unification of ideas.</p> <p><i>Extended abstract (extended comprehension)</i> Abstracting the knowledge acquired; applying it to other tasks; generalisation; drawing conclusions and new hypotheses.</p>	<p><i>Comprehension</i> Comprehension; meaningful perception of new information.</p> <p><i>Identification</i> Recognition of the objects and processes under study or actions with them; separation of the object under study from some presented different objects.</p> <p><i>Reproduction</i> Reproduction of previously acquired knowledge and its application in known situations (or for solving typical problems).</p> <p><i>Application</i> Reproduction and transformation of the assimilated information for discussion of known objects and its application in various unfamiliar situations (not typical tasks).</p> <p><i>Creative activity</i> Creation of objectively new information (previously unknown to anyone).</p>	<p><i>Phenomenological level ($\beta = 1$)</i> Description of facts and phenomena; classification of objects; stating their properties and qualities (a certain number of homogeneous factors are known); using mainly natural language and worldly concepts.</p> <p><i>Analytical and synthetic level ($\beta = 2$)</i> Explanation of the nature and properties of objects and regularities of phenomena (qualitative or semi-quantitative signs), prediction of possible outcomes in observed phenomena.</p> <p><i>Prognostic level ($\beta = 3$)</i> Explanation of phenomena of the given area and creation of their quantitative theory; modelling of the basic processes; analytical representation of laws and properties. Prediction of terms and quantities in the outcomes of processes and phenomena.</p> <p><i>Axiomatic level ($\beta = 4$)</i> Explanation of phenomena using a high degree of commonality of description. The wide use of scientific language. Deep penetration into the essence of phenomena. Accurate and long-term prediction and explanation. Creation of interdisciplinary scientific language for an explanation of phenomena and processes.</p>

Appendix C. Level of characteristics of structural components of computational thinking

Levels of computational thinking development	Components of computational thinking								
	Abstract and conceptual perception of information (object and processes)	Examination and analysis of information	Decomposition of information (task, situation and processes)	Forecasting results and outcomes of actions (processes) and decisions	Forecasting the implementation	Planning the algorithm	Problem-solving automation	Creating template (pattern) solutions based on the generalisation of knowledge	A critical re-evaluation of the decision or result
Phenomenological	Partial understanding of the area, which the object (process, task and problem) belongs to and partial understanding of the task	Identifying explicit details, properties and causes and identification of explicit incoming and outgoing data	Identifying some subtasks	Recognising identical objects (tasks and processes) and some of their common properties and methods	Intuitive possession of the best-known interfaces and digital devices for implementation	Planning the actions to achieve the result	Low speed of solution search and achieving the result by <i>t</i> (time)	Identification of the scope of application; classification of the task according to certain features; selection of repetitive data (elements), functions (methods, fragments and actions)	-
Analytical and synthetic	Understanding of the area, which the object (process, task and problem) belongs to and understanding of the task	Identifying main details, properties and causes, and partially identifying incoming and outgoing data	Identifying main subtasks and connections between main subtasks	Establishing links with other similar (identical) objects (processes and tasks), comparing future results with current results; identifying advantages and disadvantages of the intended result, anticipating risks and exceptional situations (minimum number)	Intuitively handling the available properties and methods of objects (processes and tasks); intuitive possession of the best-known interfaces and digital devices for implementation	Planning the actions and determining the order of their implementation, planning the alternative actions in exceptional situations (minimum)	The average speed of solution search and achieving the result by <i>t</i> (time)	Identification of the scope of application; classification of the task according to certain features; identification of repeated data (elements), functions (methods, fragments and actions), repeated application of the fragments for another task	Specifying some shortcomings of the decision

Attitudinal and prognostic	Quite good understanding of the area, which the object (process, task, problem) belongs to and full understanding of the task	Identifying main and partially indirect details, properties and causes and partially identifying incoming (initial data) and outgoing (results) data	Identifying all subtasks and connections between main subtasks, distribution of data by subtasks	Establishing links with other similar (identical) objects (processes and tasks), comparing future results with current results; identifying advantages and disadvantages of the intended result, anticipating risks and exceptional situations (sufficient number)	Intuitively handling the available properties and methods of objects (processes and tasks); intuitive possession of the best-known interfaces and digital devices for implementation, the proposal of several options (tools) for implementation	Planning the actions for achievement of the result taking into account their dependence on each other; planning alternative actions in exceptional situations	Quite high-speed solution search and achieving the result by t (time)	Identification of the scope of application, classification of the task according to certain features; identification of repeated data (elements), functions (methods, fragments and actions), repeated application of the algorithm or its fragments for another task, application of templates	Attitudinal-prognostic level
Axiomatic	Good understanding of the area which the object belongs to (process, task, problem), full understanding of the task	Identifying main and indirect (hidden) features (details), properties and causes, determination of all incoming (initial data) and outgoing (results) data	Identifying all subtasks and connections between subtasks; distribution of data by subtasks	Establishing links with other similar (identical) objects (processes and tasks), comparing future results with current results; identifying advantages and disadvantages of the intended result, anticipating risks and exceptional situations (maximum number)	Intuitively handling the available; unavailable properties and methods of objects (processes and tasks); intuitive possession of the various interfaces and digital devices for implementation, the proposal of the best option (tool) for implementation and justifying the choice	Planning the actions for achievement of the result taking into account their dependence on each other; determining the order of their implementation; planning alternative actions in exceptional situations	High-speed solution search and achieving the result by t (time)	Identification of the scope of application; classification of the task according to certain features; identification of repeated data (elements), functions (methods, fragments and actions), repeated application of the algorithm or its fragments for another task, application of templates, creation of the template library	The rationale for the effectiveness of the decision; an indication of the shortcomings of the decision, proposal for an alternative solution