Integration of Python into Science Teacher Education, Developing Computational Problem Solving and Using Information and Communication Technologies Competencies of Pre-service Science Teachers

Kaan BATI

Hacettepe University, Faculty of Education,
Department of Mathematics and Science Education
Ankara, Turkey
e-mail: kaanbati@gmail.com

Abstract. This study reports the findings of a program that aims to develop pre-service science teachers' computational problem-solving skills and views on using information and communications technology in science education. To this end, pre-service science teachers were trained on computational thinking, computational problem solving, designing an algorithm, and Python coding, and then they were asked to solve problem situations determined within the science education program using the computational problem-solving process. The study was conducted in a faculty of education in Turkey and carried out conducted in an elective course in the spring semester of the 2019–2020 academic year (in an online platform due to the Covid-19 Pandemic). 38 pre-service science teachers were included in the study. In this process, pre-service science teachers' conceptual development levels regarding computational thinking and their views regarding the use of ICT in schools were collected quantitatively. The development of computational problem-solving skills of pre-service science teachers was scored by a rubric developed in this study. According to the analyzes, pre-service science teachers increased knowledge of computational thinking ($t = -5,969$, $p = .000$), enhanced views regarding the use of ICT in schools ($t = -2,436$, $p = .020$), and developed computational problem-solving skills ($\chi2_{(2) =} 9.000$, $p = 0,011$). These findings have the potential to provide evidence on how computational problem-solving skills can be integrated into science teacher education programs.

Keywords: computational thinking, computational problem solving, science teacher education, information and communications technology, python.

1. Introduction

Today, with the rapid development of information and communications technology (ICT), ICT and computational thinking (CT) skills are integrated with many educational disciplines and this integration creates new areas of discussion. In the literature, it is stated that it is an important matter to provide CT skills at all levels of education starting from kindergarten (Fessakis, Gouli, and Mavroudi, 2013; Sullivan, and Bers, 2016). Wing (2008), as one of the researchers who started the discussions by taking CT out of computer science, defined CT as "solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing" (p. 3717). She also argued that "computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability." (Wing, 2006, p. 33). IEA (International Association for the Evaluation of Educational Achievement) is to evaluate the computer and information literacy (CIL) levels and computational thinking (CT) skills of eighth-grade students through computer-based assessment tools within the scope of the International Computer and Information Literacy Study (ICILS) (Fraillon, Ainley, Schulz, Friedman, & Duckworth, 2019). According to Fraillon *et al.* (2019), CIL refers to a student's ability to use computer technologies to collect and manage information and to generate and change information. CT is defined as the type of thinking used when programming a computer or developing an application for another type of digital device (Fraillon, *et al.*, 2019).

Problem-solving is one of the core concepts of CT, but solving computational problems and computational problem-solving are different concepts and processes. Solving computational problems, especially in computer science education, is one of the necessary skills and includes identifying and solving problems in algorithms or code (Liu, *et al.*, 2011). On the other hand, computational problem solving is the use of computational tools and strategies for solving real-world problems, like a computer scientist (Pellas and Vosinakis, 2017). Computational problem solving is a cognitive thinking process that focuses not only on solving a problem using logical and abstract thinking but also on the application of this process with basic programming concepts (Pellas and Vosinakis, 2017). In addition, a proposed solution to a problem in the computational problem-solving process includes applying basic programming structures such as selection, sequence, or iteration (Pellas and Vosinakis, 2018).

Developing teachers' and pre-service teachers' knowledge and skills about computational thinking and computational problem-solving is very important to integrate CT and CIL into science education programs. Bower *et al.* (2015), in their study which examining the CT perceptions of teachers from different fields in Australia, revealed that many teachers do not have sufficient knowledge and skills in developing students' CT abilities. Corradini, Lodi, and Nardelli (2017) found that Italian primary school teachers did not feel ready to develop CT competencies in their students. Sands, Yadav, and Good (2018) found that pre-service teachers tend to associate CT with more math, computer use, and online games. Similarly, Rich, Yadav, and Schwarz (2019) reported that teachers were reluctant to integrate CT into their teaching plans due to reasons such as time and

content knowledge. As can be understood from these studies, pre-service and in-service teachers' knowledge and skills regarding computational thinking are very low, and they also have difficulties in establishing connections between computational thinking and science education (Bower, *et al.*, 2017; Valerie *et al.*, 2017). Based on these points, this study aimed to increase pre-service science teachers' knowledge of computational thinking, views regarding the use of ICT in schools, and computational problem-solving skills via computational problem-solving based program.

1.2. *Theoretical Framework*

When CT was first conceptualized by Wing in 2006, it was emphasized as a basic skill for everyone, not just computer scientists. According to Barr *et al.* (2011) CT is a problem-solving process that includes the following skills:

- "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"* (p. 21)

After this study, CT was defined as a problem-solving process by many researchers (Grover, and Pea, 2018; Weintrop *et al.*, 2016) and it is claimed that CT skills can help students to choose and use appropriate tools and strategies in the problem-solving process. In summary, CT refers to a set of intellectual skills, practices, and methods that are fundamental to problem-solving and includes a range of sub-skills such as decomposition, abstraction, algorithm design, automation, data collection, data analysis, data representation, simulation, parallelization and generalization (Barr and Stephenson, 2011; Conery *et al.*; 2011a, Conery *et al.*, 2011b; Park and Jeon, 2015).

According to literature, CT skills can be gain to students in K-12 levels with different instruments such as educational robots (Gordon *et al.* 2015; Martinez *et al.* 2015; Sullivan *et al.* 2015), and visual block-based programming software (Horn *et al.*, 2012). According to Grover and Pea (2018), the formulation of the solution of a problem is an important part of the problem-solving process and since the process of formulating the solution does not have to be computer-based, CT can be taught without using a computer. For this reason, CT skills are taught using computer-based (plugged-in) and unplugged methods in K-12 training (Lee, Junoh, 2019).

CT involves formulating problems with tools and methods such as computers and data analysis, finding possible solutions, and using these solutions to solve other similar problems (Barr, Harrison, and Conery, 2011; Wing, 2008). This process also defined as computational problem solving (Pellas, Vosinakis, 2017). According to Dierbach (2012), two things are needed to solve a problem computationally: a solution proposal

that covers all relevant aspects of the problem and an algorithm that can solve the problem using this solution proposal. Dierbach (2012) defines the computational problem-solving process as follows:

1. Analyze problem:
    a. Clearly understand problem.
    b. Know what constitutes a solution.

3. Describe data and algorithms:
    a. Determine what time of data is needed.
    b. Determine how data is to be structured.
    c. Find and/or design appropriate algorithms.

4. Implement program:
    a. Represent data within programming language.
    b. İmplement algorithms in programming language.

3. Test and debug:
    a. Test the program on a selected set of problem instances.
    b. *Correct and understand causes of any errors found"* (p. 18).

2. Method

In this current study, one group pretest-posttest design (Fraenkel and Wallen, 1993) was used to determine how the computational problem-solving-based program affected the pre-service science teachers' knowledge of computational thinking, views regarding the use of ICT in schools, and computational problem-solving skills. Before and after the application, a conceptual development of computational thinking test and a scale on views regarding the use of ICT in schools were applied to pre-service science teachers as pre-test and post-test. To examine the development of computational problem-solving skills of pre-service science teachers, the computational problem-solving studies of pre-service science teachers were scored with a rubric developed within the scope of this current study.

2.1. *Participants*

The study was conducted in a faculty of education in Turkey and carried out conducted in an elective course in the spring semester of the 2019–2020 academic year (in an online platform due to the Covid-19 Pandemic). This elective course was conducted online for two hours a week, an additional hour of study was planned each week, these additional studies were composed of small coding assignments belonging to the studies covered in the online course. 38 pre-service science teachers (32 females, 6 males) were included in the study. The convenience sampling method was used in the selection of the participants (Fraenkel, Wallen, 1993). The participants were in the 3rd and 5th semester

(age range 20–21) and it was determined in the preliminary interviews and participation approval forms that none of the pre-service teachers had prior knowledge and experience about computational thinking or computational problem solving before the application. For this reason, basic information such as coding logic, algorithms, the use of Google Colab, variables, loops, and functions in Python was added to the program.
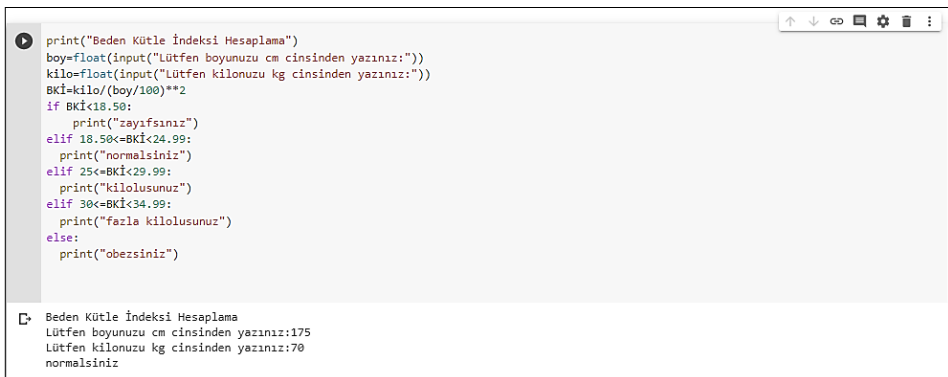
2.2. *Implementation*

Before the Covid-19 pandemic, it was planned that the applications of the program would be carried out in the computer laboratory with pairs. The coding and algorithm design studies planned in this process would be under the supervision of the lecturer, and the computational problem-solving studies would be completed in the classroom environment by pairs. In this way, it was aimed to be able to closely follow the progress of the pre-service science teachers during the process and to give instant feedback and corrections. As of the second week of the term, with the transition to online education, all studies and applications were transferred to a MOODLE-based online platform provided by the university and the planned applications were updated in accordance with distance education. All of the computational problem-solving applications and Python training planned in the program were conducted online on the students' personal computers via Google Colab, and all students were asked to share the Python coding files they created on Google Drive with the instructor.

Python is an open-source scripting language with a simple syntax, easy to read and write. At the same time, as it offers powerful programming features, it is widely used and offers wide library support (Dierbach, 2012). Using Python as an introductory programming language helps novice learners to easily incorporate key features of CT (Grandell *et al.*, 2006). Compared to languages such as Java or C ++, Python is very useful for novice users, with features such as its structure that facilitates dynamic writing, provides immediate feedback for potential errors, and mandatory structural design that leads to an indented and structured writing path (Buitrago Flórez, *et al.*, 2017). Collaboratory, or shortly "Colab", one of the services provided by Google, is a system that allows writing and executing code in Python language over the browser without any pre-configuration. Colab runs on Jupyter Notebook and allows writing code and text in a single document. Colab notebooks created by users are stored in Google Drive accounts and shared with desired people and provide the opportunity to collaborate. In this study, all the procedures performed by pre-service science teachers were recorded and the developments in computational thinking skills could be followed throughout the process.

Within the scope of the program, three problem situations were determined from different grade levels of the K-12 science teaching program. These problem situations were given to pre-service science teachers in the process and they were asked to design an algorithm for the solution and write Python code for this algorithm. While determining the problem situations, the criteria of "suitability to the context" and "suitability to the level" were taken into consideration. Suitability to the context refers to the determination of problem situations directly concerning science subjects and to be solved by

creating an algorithm. In this way, it was aimed that pre-service science teachers could associate computational problem-solving skills with the science program. The suitability to the level refers to pre-service science teachers who do not have any prior knowledge of algorithms, coding, and computational problem-solving. It was planned that pre-service science teachers would be able to solve problems with the basic knowledge they gained in this process. Problem situations consist of two stages. In the first stage, a work file was presented to the teacher candidates, explaining the problem situation, and containing the steps to follow for a solution. This document consists of decomposition, collecting and organizing data, and algorithm design, which are the steps of the computational problem-solving process. In the second stage, pre-service teachers were asked to write the Python codes suitable for their algorithm designs on Google Colab. At this stage, pre-service science teachers were provided with individual feedbacks to debug the codes that did not work and concretize the solution (abstraction). The computational problem-solving studies conducted in this study are detailed below.

- **Study 1:** Study 1 is about body mass index. In this study, science teacher candidates were told that the Ministry of Health will initiate obesity screening across the country, so an obesity calculation tool should be placed on the web page. Pre-service science teachers were asked to write an algorithm and a program to print the obesity status of people on the screen after taking the height and weight values from people and making the necessary calculations. Study 1 was given after the instruction of *"if", "else", "elif"* commands, and prospective science teachers designed their algorithms and codes using these commands. It was aimed to make research on body mass index so that science teacher candidates could create their algorithms and convert the body mass index formula into an algorithm and Python code.

- **Study 2:** Study 2 is about thermometer conversions. In this study, pre-service science teachers were asked to design an algorithm to convert a temperature value (C or F) to another thermometer value and write the Python code for this algorithm. Study 2 was given after teaching *"for"* and *"while"* loops. Although they were

```python
print("Beden Kütle İndeksi Hesaplama")
boy=float(input("Lütfen boyunuzu cm cinsinden yazınız:"))
kilo=float(input("Lütfen kilonuzu kg cinsinden yazınız:"))
BKİ=kilo/(boy/100)**2
if BKİ<18.50:
    print("zayıfsınız")
elif 18.50<=BKİ<24.99:
    print("normalsiniz")
elif 25<=BKİ<29.99:
    print("kilolusunuz")
elif 30<=BKİ<34.99:
    print("fazla kilolusunuz")
else:
    print("obezsiniz")
```

```
Beden Kütle İndeksi Hesaplama
Lütfen boyunuzu cm cinsinden yazınız:175
Lütfen kilonuzu kg cinsinden yazınız:70
normalsiniz
```

Fig. 1. Example of study 1.

```
print("""<<<SICAKLIK DÖNÜŞTÜRÜCÜ>>>
1. Celsius'u fahrenheit' a dönüştür
2. Fahrenheit'ı celsius'a dönüştür""")
n= input("lütfen 1 ve 2 arasından bir seçim yapınız: ")
if n=="1":
    celsius=float(input("celsius değerini giriniz:"))
    fahrenheit=(celsius*1.8)+32
    print(fahrenheit, "fahrenheit" )
elif n=="2":
    fahrenheit=float(input("fahrenheit değerini giriniz:"))
    celsius=(fahrenheit-32)/1.8
    print(celsius, "celsius")
else:
  print("geçersiz sayı girdiniz")
```

```
<<<SICAKLIK DÖNÜŞTÜRÜCÜ>>>
1. Celsius'u fahrenheit' a dönüştür
2. Fahrenheit'ı celsius'a dönüştür
lütfen 1 ve 2 arasından bir seçim yapınız: 1
celsius değerini giriniz:36
96.8 fahrenheit
```

Fig. 2: Example of study 2.

expected to use these loops in algorithm designs, it was seen that some pre-service science teachers solved the problem with if-else commands. For the pre-service science teachers to solve this problem, they were expected to figure out thermometer types and how the values measured in thermometers can be converted to each other and to create their algorithms.

● **Study 3:** Study 3 is about projectile motion. In this study, firstly, a problem situation related to the security problems of fireworks was given. Then, pre-service science teachers were asked to design an algorithm that would calculate the time to reaching the fireworks peak, the maximum height it could go out, and the longest path it could take on the horizontal when the angle and mass variables were entered before it was launched, and to write the Python code suitable for this algorithm. Study 3 is given after teaching the subject of functions, and they are expected to integrate and use what they have learned to date.

```
#PROJE3
import math
from math import sin
açı=float(input('lütfen havai fişeğin yerle yaptığı açıyı giriniz.'))
kütle=float(input('lütfen havai fişeğin kütle bilgisini yazınız.'))
açı=sin(math.radians(açı))
def formül(Vo,açı,g):
    tçıkış=(Vo*sin(açı))/g
    hmaxsimum=(Vo*sin(açı)*Vo*sin(açı))/(2*g)
    xmenzil=(Vo*Vo*sin(2*açı))/(2*g)
    print('tçıkış=',tçıkış)
    print('hmaxsimum=',hmaxsimum)
    print('xmenzil=',xmenzil)
if(kütle==250):
    formül(6,açı,10)
else:
    if(kütle==400 or kütle==650):
        formül(8,açı,10)
```

```
lütfen havai fişeğin yerle yaptığı açıyı giriniz.30
lütfen havai fişeğin kütle bilgisini yazınız.250
tçıkış= 0.28765532316252174
hmaxsimum= 0.41372792471867414
xmenzil= 1.5146477726542136
```

Fig. 3: Example of study 3.

2.3. *Data Collection and Data Analysis*

To examine the computational thinking conceptual development levels of pre-service science teachers, Questionnaire of Computational Thinking (QCT) developed by Alfayez and Lambert (2019) was translated into Turkish. The test consists of 22 multiple-choice questions on computational thinking and Alfayez and Lambert (2019) reported the Cronbach Alpha reliability coefficient of the test as 0.70, in this study it was found 0.59. The questionnaire includes multiple-choice questions asking definitions and examples related to decomposition, data collection and analysis, abstraction and algorithm design, automation, data collection, data analysis, data representation, simulation and parallelization, and generalization, which are defined as sub-dimensions of CT in the literature. Descriptive statistics values related to the test are reported under the title of validity and reliability. To determine teachers' views regarding the use of ICT in schools, views regarding the use of ICT in the teacher survey used in the ICILS 2018 application were selected and translated into Turkish. The scale consists of four-point Likert-type responses such as strongly disagree, disagree, agree, strongly agree. There are a total of 13 statements on the scale as shown below:

Using ICT at school …

"... impedes concept formation by students."
"... helps students develop greater interest in learning. "
"... helps students to work at a level appropriate to their learning needs."
"... results in students copying material from Internet sources."
"... helps students develop problem solving skills."
"... distracts students from learning."
"... results in poorer written expression among students."
"... results in poorer calculation and estimation skills among students."
"... limits the amount of personal communication among students."
"... enables students to collaborate more effectively."
"... helps students develop skills in planning and self-regulation of their work."
"... improves academic performance of students."
"... enables students to access better sources of information."

This scale was developed by ICILS to collect teachers' views regarding using ICT in schools. Within the scope of this current research, it is used to gather information about perceptions of preservice science teachers about using ICT in schools. Confirmatory factor analysis and item analysis were performed by reaching 121 participants for the Turkish adaptation of this scale. The Cronbach Alpha reliability coefficient of the data collected in this study was calculated as 0.80. The confirmatory factor analysis results of the scale were reported under the title of validity and reliability.

The three studies described above were given to pre-service teachers to determine whether they developed computational problem-solving skills. At this stage, pre-service science teachers were first expected to formulate the problem (decomposition, abstraction, organizing, and analyzing data), then design an algorithm for the solution (algorithm design), and finally write a Python code for this algorithm. This process organized

based on the programming description of Webb, *et al.*, (2017) as; *"a process involving: analysis and understanding of problems; identifying and evaluating possible solutions; generating algorithms; implementing solutions in the code of a particular programming language; testing and debugging, in order to formulate solutions into executable computer programs"* (p. 449). To gather more evidence about the development of computational problem-solving skills of pre-service science teachers, a rubric was developed in accordance with the work of Moreno-León and Robles (2015) on the scoring of Scratch projects. The scoring tool used in the study of Moreno-León and Robles (2015) focuses on the use of computational thinking skills in Dr. Scratch practice. The rubric developed in this current study focuses on the computational problem-solving process. As seen in Table 1, the computational problem-solving rubric was developed in accordance with the steps of Dierbach's (2012) framework of computational problem solving. However, although it is not included in Dierbach's (2012) framework, generalization was added to the scoring key as an important skill in the computational problem-solving process. The rubric developed is presented in Table 1.

The Computational Problem-Solving Rubric consists of four subdimensions:

(a) Analyze problem.
(b) Describe data and algorithms.
(c) Implement program / test and debug.
(d) Generalizing and transferring.

Analyze problem subdimension includes dividing problem into manageable small pieces, identifying and removing unnecessary details in the problem, and determining and collecting required information required to solve problem. Describe data and algorithms subdimension includes collecting and organizing necessary data. this subdimension is also related to how the algorithm is designed. The third sub-dimension, implement program / test and debug, is the most comprehensive subdimension and includes designing an error-free algorithm for problem-solving, representing this algorithm with a flowchart, and write a python code that works in line with this algorithm. The last sub-dimension, generalizing and transferring, is not included in Dierbach's (2012) framework, but since it is considered one of the core components of CT (Barr and Stephenson, 2011; Conery *et al.*, 2011a, Conery *et al.*, 2011b; Park and Jeon, 2015) is included in the rubric. In this sub-dimension, pre-service science teachers' ability to apply their previous knowledge to new situations was scored. Pre-service teachers were not informed about which structures to use in computational-problem solving studies, they decided on their own according to the nature of the problem. Those who used the appropriate structures to solve the problem and used their pre-learning for this, got 1 point.

2.4. *Validity and Reliability*

In this study, three types of data were collected from pre-service science teachers. The QCT and Pre-Service Teachers' views regarding the use of ICT in schools scale were

Table 1

Computational Problem-Solving Rubric

CPS SKILLS	SCORES		
	1 point	2 points	3 points
Analyze Problem *a. Clearly understand problem* *b. Know what constitutes a solution*	The problem is divided into manageable small pieces.	The problem is divided into manageable small pieces. Unnecessary details in the problem were identified and removed.	The problem is divided into manageable small pieces. Unnecessary details in the problem were identified and removed. The information required to solve the problem has been determined.
Describe data and algorithms *a. Determine what time of data is needed* *b. Determine how data is to be structured* *c. Find and/or design appropriate algorithms*	Necessary data were collected for problem-solving.	Necessary data were collected for problem-solving. Collected data are organized for problem-solving.	Necessary data were collected for problem-solving. Collected data are organized for problem-solving. Collected data were analyzed for problem-solving.
Implement program *a. Represent data within programming language* *b. Implement algorithms in programming language* Test and Debug *a. Test the program on a selected set of problem instances* *b. Correct and understand causes of any errors found*	An error-free algorithm has been created for problem-solving.	An error-free algorithm has been created for problem-solving. The generated algorithm is represented by a flowchart.	An error-free algorithm has been created for problem-solving. The generated algorithm is represented by a flowchart. A python code that works in line with the algorithm has been written.
Generalizing and transferring *a. Problem-solving process to other problems.*	Previous experience and knowledge were transferred in problem-solving.		

Table 2

Descriptive statistics and reliability analysis of scales

Scale	α	Mean	SD
Pre-service teachers' views regarding the use of ICT in schools	.80	38,50	4.4
Questionnaire of Computational Thinking (QCT)	.59	9.26	3.42

Table 3

Confirmatory factor analyses

	χ^2	p	df	90 Percent Confidence Interval for RMSEA	CFI
Pre-Service Teachers' views regarding the use of ICT in schools	741.610	0.0	317	0.0952–0.115	0.871

applied to pre-service teachers at the beginning and end of the process as a pre-test and a post-test. The results of the reliability analysis carried out with the data obtained from the pre-test applications of these tests are given in Table 2.

As seen in Table 2, while the Cronbach α reliability coefficient of the scale of pre-service science teachers' views regarding the use of ICT in schools was 0.80, the Cronbach α reliability coefficient of the QCT was found to be 0.59. The low-reliability coefficient in the QCT results should be taking into consideration. Table 3 shows the results of the confirmatory factor analysis of the Views on to use of ICT in schools.

Kline (2016) recommends reporting RMSEA at a 90% confidence interval, χ^2, degree of freedom (df), significance value, and CFI values in confirmatory factor analysis studies. In this study, the RMSEA value was calculated between 0.0952–0.115 at a 90% confidence interval, and the CFI value of the scale was calculated as 0.871. MacCallum, Widaman, Preacher, and Hong, (2001) stated that the RMSEA value in the range of 0.08–0.10 shows a mediocre fit. According to this view, it was thought that the factor structure of this scale was appropriate and could be used in this study.

In the development of the rubric, which was designed to examine the computational problem-solving skills of pre-service science teachers, the opinions of two field experts were taken and the scoring key was revised in line with the feedback. Besides, a randomly selected sample from the pre-service teachers' studies was scored together with a field expert. The Kappa value was calculated for the agreement between raters (Landis, & Koch, 1977) and substantial agreement (κ = 0.681, p = 0.000) was found between raters.

3. Results

The data collected in this study to determine the competencies of prospective science teachers regarding the computational problem-solving can be collected under the fol-

lowing headings; conceptual development of CT, views regarding the use of ICT in schools, and computational problem-solving skills. The analyzes made are presented under these headings.

3.1. *Conceptual Development of Computational Thinking*

Questionnaire of Computational Thinking (QCT) developed by Alfayez and Lambert (2019) was used as pre-test and post-test to examine the effect of the application conducted in the study on the computational thinking conceptual development levels of pre-service science teachers. Independent groups t-Test results were made to determine whether a significant difference occurred between pre-test and post-test scores of science teacher candidates are given in Table 4.

When Table 4 is examined, it is seen that there is a significant difference between the computational thinking conceptual development scores of pre-service science teachers in favor of posttest application (t = −5.969, p = .000). Possible reasons for this development may be the instructions used in the three studies given to the pre-service science teachers and the experiences of the pre-service teachers in the computational problem-solving process.

3.2. *Pre-service Science Teachers' Views Regarding the Use of ICT in Schools*

The pre-Service teachers' views regarding the use of ICT in schools scale developed by ICILS (Fraillon *et al.*, 2019) was applied as a pre-test and a post-test to determine the opinions of pre-service science teachers. Independent groups t-Test results of the scores obtained from pre-service science teachers are given in Table 5.

When Table 5 is examined, it is seen that computational problem-solving practices increase the opinions of pre-service science teachers about ICT use in schools (t = −2.436,

Table 4

QCT t-test results

	Mean	N	Std. Dev.	t	df	p
Pre-test of QCT	9.26	38	3.422	−5.969	37	.000
Post-test of QCT	12.18	38	2.502			

Table 5

Pre-service teachers' views regarding the use of ICT in schools scale t-test results

	Mean	N	Std. Dev.	t	df	p
Pre-test	38.50	38	4.404	−2.436	37	.020
Post-test	40.26	38	4.864			

p = .020). ICILS findings (Fraillon, *et al.*, 2019) show that teachers who are confident in their use of ICT have positive views on ICT and are more likely to include CIL and CT in their teaching processes. This may be one of the possible causes of this development achieved in this study. The inclusion of pre-service science teachers in computational problem-solving practices may have increased their self-confidence about ICT use, and consequently, they may have developed positive opinions about ICT use in schools.

3.3. *Computational Problem-Solving Skills*

Three computational problem-solving studies given to pre-service science teachers during the process were scored with the rubric created within the scope of this study, and the progression of pre-service science teachers was examined. Since the data are not suitable for parametric tests, non-parametric statistics were applied. Descriptive statistics values of the scores that the teacher candidates got from computational problem-solving applications are given in Table 6.

The result of Friedman's chi-square test conducted to determine whether there is a significant difference between the scores of pre-service science teachers in computational problem-solving practices shows that there is a significant difference between the students' study 1, study 2, and study 3 scores ($\chi2_{(2)} = 9.000$, p = 0.011). Wilcoxon Signed Ranks Test was applied in pairs to determine which studies differed between. Post-hoc test results are given in Table 7.

According to the Wilcoxon signed sum of ranks test, there is a statistically significant difference between the students' study 1 and study 2 scores (T = 38, p = 0.04, z = −2.916, r = −0.47). However, no statistically significant difference was found between study 2 – study 3 and study 1 – study 3 scores. These findings were interpreted as pre-service teachers' computational problem-solving skills developed in the process.

Table 6

Nonparametric descriptive statistics

	N	Mean	Std. Dev.	Min.	Max.	Percentiles		
						25th	50th(Median)	75th
Study 1	38	8,66	1,632	0	10	8,00	9,00	9,00
Study 2	38	9,08	1,746	0	10	9,00	10,00	10,00
Study 3	38	8,92	1,746	0	10	9,00	9,00	10,00

Table 7

Wilcoxon Signed Ranks test scores

	T	p	z	r
Study 1 – Study 2	38	0.04	−2.916	−0.47
Study 2 – Study 3	62.50	0.21	−1.255	−0.20
Study 1 – Study 3	96	0.97	−1.661	−0.26

4. Discussions and Conclusions

Today, computational thinking has become the focal point of STEM fields, especially science and mathematics education. STEM or STEAM approach can lead to the education of individuals equipped with the needed 21st-century skills. On the other hand, it is claimed that the STEM approach requires a shift towards more student-centered learning environments to equip students with basic 21st-century skills such as complex problem solving and teamwork (Salonen, *et al.*, 2017; Struyf, *et al.*, 2017). Based on this point of view, a program based on computational problem solving was applied to pre-service science teachers, and the conceptual development levels of CT, views regarding the use of ICT in schools, and the development of computational problem-solving skills were examined. According to the analyzes, it was determined that the applied program developed computational thinking conceptual development levels (t = −5,969, p = .000), views regarding the use of ICT in schools (t = −2,436, p = .020) and computational problem solving skills ($\chi2_{(2)}$ = 9.000, p = 0,011) of the pre-service science teachers. ICILS (Fraillon *et al.*, 2019) findings show that teachers who are confident in their use of ICT have positive views on ICT and are more likely to include CIL and CT in their teaching processes. This may be one of the possible reasons for this development achieved in this study. The inclusion of pre-service science teachers in computational problem-solving practices may have increased their self-confidence about ICT use, and consequently, they may have developed positive opinions about ICT use in schools. Additionally, the results of this study reveal that computational problem-solving practices in pre-service science teacher education can increase pre-service science teachers' CT conceptual development. Increasing their conceptual development regarding CT may also help reduce concerns about integrating CT into science education (Rich, Yadav, and Schwarz, 2019) and understand the links between CT and science education (Bower *et al.*, 2017; Valerie *et al.*, 2017)

ICILS 2018 (Fraillon *et al.*, 2019) findings reveal that providing ICT equipment to students or teachers alone is not enough to develop digital skills. For this, teacher training becomes a very important issue, and an effective understanding of how to train pre-service teachers in CT integration needs to be developed (Yadav, Stephenson, Hong, 2017). The findings of this study are very important in terms of providing concrete suggestions to overcome these deficiencies. It is thought that the significant improvement in the positive views of the pre-service science teachers determined in this study regarding the use of ICT in schools may also support their motivation to use CT when they become teachers. One of the difficulties that can be encountered especially in programming teaching is expressed as women's low interest in computer science courses and programs. Especially the lack of peer support (Kelleher, Paushc, and Kiesler, 2007), a lack of motivation, and gender stereotypes (Doube and Lang, 2012) are considered as the main causes of this problem. However, approximately 84% of the participants included in this study are females (32 females, 6 males). The significant improvement observed in the computational problem-solving skills of pre-service science teachers throughout the process reveals the groundlessness of these concerns. In this respect, it is thought that the findings of the study support the transferability of this program to teacher education.

Another obstacle in the dissemination of the findings of the study is that it is difficult to implement this in groups and schools with low technology access. Although the age we are in is called the technology and information age, access to such technology is not always possible in many rural areas. Related to this, many studies are showing that access to technology resources, the internet, and professional development opportunities in rural schools is limited (Aduwa-Pgiegbaen and Iyamu, 2009; Wood and Howley, 2012). This situation can create two types of obstacles. The first obstacle is that teachers who receive this training will not be able to implement these practices in schools with technology deficiencies. Because Kale, Akcaoglu, Cullen, and Goh (2018) found that students have more advantages in acquiring CT skills in schools where teachers have a basic technology level. The second obstacle arises when science teacher candidates have low access to technology, in this case, the application of the proposed program within the scope of the study will be difficult.

Ethical Statement

At the beginning of the semester, volunteer participation forms were distributed to the pre-service science teachers, and they were informed that participation in the study and surveys were entirely voluntary. All selected pre-service science teachers signed the consent form.

References

Aduwa-Pgiegbaen, S.O., & Iyamu, E.S. (2009). Availability and utilization of classroom computers across urban and rural schools in southwestern Nigeria. *International Journal of Information & Communication Technology Education, 5*, 74–87.

Alfayez, A.A., & Lambert, L. (2019). Exploring Saudi computer science teachers' conceptual mastery level of computational thinking skills, *Computers in the Schools,* 36(3), 143–166, DOI:10.1080/07380569.2019.1639593

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology,* 38(6), 20–23.

Bower, M., Wood, L.N., Lai, J.W., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the Computational Thinking Pedagogical Capabilities of School Teachers. *Australian Journal of Teacher Education,* 42(3). `http://dx.doi.org/10.14221/ajte.2017v42n3.4`

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research,* 87(4), 834–860.

Conery, L., Stephenson, C., Barr, D., Barr, V., Harrison, J., James, J., & Sykora, C. (2011a). *Computational thinking in K–12 education: Teacher resources*. (2nd ed.). Portland, OR: Computers in the Schools 17 International Society for Technology in Education. Retrieved from `https://id.iste.org/docs/ct-documents/ct-teacher-resources_2ed-pdf.pdf?sfvrsn=2`

Conery, L., Stephenson, C., Barr, D., Barr, V., Harrison, J., James, J., & Sykora, C. (2011b). *Computational thinking: Leadership toolkit* (1st ed.). Portland, OR: International Society for Technology in Education. Retrieved from `https://id.iste.org/docs/ct-documents/ctleadershipt-toolkit.pdf?sfvrsn=4`

Corradini, I., Lodi, M., & Nardelli, E. (2017). Conceptions and misconceptions about computational thinking among Italian primary school teachers. Proceedings of the *2017 ACM Conference on International Computing Education Research – ICER* '17*,* Tacoma, WA, USA (pp. 136–144).

Dierbach, C. (2012). *Introduction to Computer Science using Python: A Computational Problem-Solving Focus.* Wiley Publishing.

Doube, W., & Lang, C. (2012). Gender and stereotypes in motivation to study computer programming for careers in multimedia. *Computer Science Education,* 22, 63–78. Retrieved from `http://eric.ed.gov/?id=EJ958969`

Fraenkel, J. R., & Wallen, N. E. (1993). *How to design and evaluate research in education* (Vol. 7). New York: McGraw-Hill.

Fraillon, J., Ainley, J., Schulz, W., Duckworth, D., & Friedman, T. (2019). *IEA International Computer and Information Literacy Study 2018: Assessment Framework.* Amsterdam: IEA.

Grandell, L., Peltomaki, M., Back, R.-J., & Salaskoski, T. (2006). Why complicate things? Introduction programming in high school using phyton. Paper presented at the proceeding of the *8th Australasian Conference on Computing Education,* Hobart, Australia. Retrieved from `http://dl.acm.org/citation.cfm?id=1151880`

Grover, S. & Pea, R. (2018). Computational thinking: A competency whose time has come. In: Sentance, S., Carsten, S., & Barendsen, E. (Eds), *Computer Science Education: Perspectives on Teaching and Learning*. Bloomsbury.

Kale, U., Akcaoglu, M., Cullen, T., & Goh, D. (2018). Contextual Factors Influencing Access to Teaching Computational Thinking. *Computers in the Schools,* 35(2), 69–87, DOI: 10.1080/07380569.2018.1462630

Kelleher, C., Paushc, R., & Kiesler, S. (2007, April). Storytelling Alice motivates middle school girls to learn computer programming. Paper presented at the proceedings of the *SIGCHI Conference on Human Factors in Computing Systems*, San Jose, CA. DOI: 10.1145/1240624.1240844

Kline, R. B. (2016). *Principle and Practice of Structural Equation Modelling* (4. bs.). New York, NY: The Guilford Press.

Landis, J.R., Koch, G.G. (1977). The measurement of observer agreement for categorical data. *Biometrics,* 33, 159–174.

Lee, J., & Junoh, J. (2019). Implementing unplugged coding activities in early childhood classrooms. *Early Childhood Education Journal,* 47(6), 709–716.

Liu, C.C., Cheng, Y.B., & Huang, C.W. (2011). The effect of simulation games on the learning of computational problem solving. *Computers & Education,* 57(3), 1907–1918.

MacCallum, R.C., Widaman, K.F., Preacher, K.J., & Hong, S. (2001). Sample size in factor analysis: The role of model error. *Multivariate Behavioral Research, 36*(4), 611–637.
DOI: 10.1207/S15327906MBR3604_06

Moreno-León, J., & Robles, G. (2015, August). Analyze your Scratch projects with Dr. Scratch and assess your computational thinking skills. *In Scratch conference* (pp. 12–15).

Park, S., & Jeon, Y. (2015). Teachers' perception on computational thinking in science practices. *International Journal of Education and Information Technologies,* 9(1), 180–185.

Pellas, N., & Vosinakis, S. (2017, April). How can a simulation game support the development of computational problem-solving strategies?. In: *2017 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1129–1136). IEEE.

Pellas, N., & Vosinakis, S. (2018). The effect of simulation games on learning computer programming: A comparative study on high school students' learning performance by assessing computational problem-solving strategies. *Education and Information Technologies,* 23(6), 2423–2452.

Rich, K.M., Yadav, A., & Schwarz, C.V. (2019). Computational thinking, mathematics, and science: Elementary teachers' perspectives on integration. *Journal of Technology and Teacher Education,* 27(2), 165–205.

Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K-12: In-service teacher perceptions of computational thinking. In: M. S. Khine (Ed.), *Computational Thinking in the STEM Disciplines* (pp. 151–164). Cham: Springer.

Salonen, A., Hartikainen-Ahia, A., Hense, J., Scheersoi, A., & Keinonen, T. (2017). Secondary school students' perceptions of working life skills in science-related careers. *International Journal of Science Education,* 39(10), 1339–1352.

Struyf, A., De Loof, H., Boeve-de Pauw, J., & Van Petegem, P. (2019). Students' engagement in different STEM learning environments: integrated STEM education as promising practice?. *International Journal of Science Education,* 41(10), 1387–1407.

Webb, M., Davis, N., Bell, T. *et al.* (2017). Computer science in K-12 school curricula of the 21st century: Why, what and when? *Educ Inf Technol*, 22, 445–468.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining com-

putational thinking for mathematics and science classrooms. *Journal of Science Education and Technology,* 25(1), 127–147.

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

Wing, J. (2008). Computational thinking and thinking about computing. *The Royal Society*, 366(July), 3717–3725. DOI: 10.1098/rsta.2008.0118

Wood, L., & Howley, A. (2012). Dividing at an early age: The hidden digital divide in Ohio elementary schools. *Learning, Media and Technology,* 37(1), 20–39.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM,* 60, 55–62.

K. Bati received undergraduate education from Hacettepe University Faculty of Education Science Teaching program in 2005. After working as a science teacher and administrator in different private schools for three years, he started to work as a research assistant at Hacettepe University in 2008. He completed master's degree in 2010 and finished my doctorate in science education in 2014. In 2018, he completed post-doctoral studies on computational thinking in science education and developing classroom materials for STEM education at Ohio State University, College of Education and Human Ecology, Teaching and Learning Department with a TÜBİTAK 2219 scholarship. He received the title of associate professor in 2020, still continue to work at Hacettepe University, Department of Mathematics and Science Education. He works on computational thinking in science education and STEM.