*International Journal of Social Science and Economic Research*

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

# EVALUATING PROCEDURAL VERSUS CONCEPTUAL TEACHING FORMATS FOR BEGINNING AND ADVANCED PROGRAMMING STUDENTS

Lara Mahajan, Ashrita Chadeva and John Leddo

MyEdMaster, LLC, 13750 Sunrise Valley Drive, Herndon, VA, United States of America
John Leddo is the director of research at MyEdmaster.

. DOI: 10.46609/IJSSER.2021.v06i04.016 URL: https://doi.org/10.46609/IJSSER.2021.v06i04.016

## ABSTRACT

As computers and technologies integrate their ways into our current society, a demand for positions in these fields has increased. This demand has led to a rise in computer science and technology courses being offered ranging from elementary school to college. There have been questions raised about what teaching format results in greater mastery of the subject matter. As these courses are the building blocks of many careers, it is imperative to seek an answer. In this experiment, we set out to compare procedural and conceptual teaching formats on beginning and advanced students. 19 beginners were taught for and while loops in either a procedural or conceptual manner. 22 advanced students were taught lambda functions in either a procedural or conceptual format as well. Results showed that beginner-level students responded equally the same to procedural and conceptual teaching formats. On the other hand, advanced students performed statistically significantly better when taught using a procedural teaching method.

**Keywords:** conceptual learning, procedural learning, python programming, learning format

## 1. Introduction

Upon the arrival of new technologies and advancements, there has been a demand for jobs in the technology and computer science field. According to the U.S. Department of Labor Bureau of Labor Statistics, from 2016 leading up to 2026, there has been predicted to be a 16% increase in the computer and information technology field [3]. Even more, an expected 557,100 job opportunities by 2026 will be added. The basis of all learning and students' interest in their futures emerge in middle and high school. Furthermore, more than 90% of parents are advocating that computer science courses are a good use of school resources [9]. Schools must provide and correctly teach interested students the basis of their future.

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

To establish an effective foundation, it is necessary to research which teaching format results in students' understanding of topics. Also, it is important to know whether a student's skill level is a factor while researching the best teaching format. Over the past century, research has been done to find the answer to the long debate and commonly asked question, what teaching format results in higher levels of understanding, procedural or conceptual? Similar studies have been applied to mathematics and science, but there are limited studies in the application of computer science. Research done by Isleyen et al. has discovered more importance in procedural teachings compared to conceptual while teaching mathematics to primary students in Turkey [6]. Furthermore, another study done by Aydin et al. and his colleagues suggests that both conceptual and procedural learning formats are required for a student to effectively learn science [1]. They concluded that learning conceptually first is beneficial, the added procedural learning will only aid the student.

Due to the high demand for computer science courses, College Board, a non-profit organization whose mission is to expand access to higher education, has offered courses namely AP Computer Science Principles and AP Computer Science A. Courses were originally taught in Pascal, then C++, and currently these courses are taught in Java [10]. These AP courses are available to all students in high school of varying skill levels. Although prerequisites for these courses exist, they are only "recommended" and not enforced. This means that a student with absolutely no experience could be taking the course, or a student with mastery in the language could take the course. It is incredibly important for teachers to understand what teaching format is effective for not only that topic but for students of all skill levels. This has the utmost importance in building a solid foundation of understanding. Although this particular research-tested student in Python, the results can be applied to other programming languages.

In our research, we focus on the differences in a procedural versus conceptual teaching format on beginner versus advanced students. Procedural teaching formats refer to the understanding of the algorithm to solve the problem, relying on memorization and surface level connections. Conceptual teaching refers to the understanding of how and why the algorithm works, the material allows students to make profound connections using what they have been taught. After instruction, we evaluated students' skills in three different categories of syntax, logic, and correctness as well as overall performance. This research specifically focuses on the programming language of Python. Students of varying skill levels were sorted into beginners and advanced participants in the study. For the beginners, one group was taught in a procedural manner and the other in a conceptual manner. A similar process was administered for the advanced students.

## 2. Materials and Methods

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

## Participants

Participants in this study included 19 students at a beginner level and 22 students at an advanced level in Python programming. These students were middle through high schoolers located in Northern Virginia in the United States. These classes were advertised as free Python classes for beginner and advanced students. An email with information about the classes was sent out to parents on the MyEdMaster emailing list, the email included a Google Form for the participants to fill in to sign up. Neither the email nor the form, included the topics being taught in either class to avoid the student's research about the topics beforehand. A list of prerequisites for the beginner and the advanced class was listed on the form. Prerequisites for beginners were the understanding of parameters, conditional statements, variables, print statements, and knowledge of operators in Python. Prerequisites for advanced participants were the understanding of lists, arrays, parameters, arguments, conditional statements, variables, print statements, functions, and operators in Python. Based on the participants' self-evaluation of the given prerequisites, the participants had to indicate the class that they would like to attend, beginner or advanced, and the time slot that they preferred.

## Materials

There were four different Google Presentations made to represent the different formats of teaching and the skill level of the students. Beginners were taught for and while loops, and advanced students were taught lambda functions. The conditions were Loops Procedural, Loops Conceptual, Lambda Function Procedural, and Lambda Function Conceptual. For both the loop presentations, they included the same two sets of three practice problems followed with the same final problem students were asked to solve. Similar to the loop presentation, both conceptual presentations included the same two practice problems with the same final problem. But the content for each topic, depending on whether it was procedural or conceptual, was different. Procedural presentations focused on the basics which are needed to know for each topic. These presentations mainly focused on the syntax aspect of the topics. Learning the syntax could allow a student to write a correct answer, but did not give the student a complete understanding of the logic behind it. Concept presentations, for both topics, included syntax, real-world applications, and additional information on keywords included in the syntax of loops and lambda functions. This information in addition to the syntax in conceptual presentation allows the student to fully understand the logic and reasoning behind the algorithm. The loop's conceptual slides also included flow charts that explain the decision process for each line of code in the loop, as seen in Table 1 and Table 2.

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

**Table 1: For-Loops FlowChart**

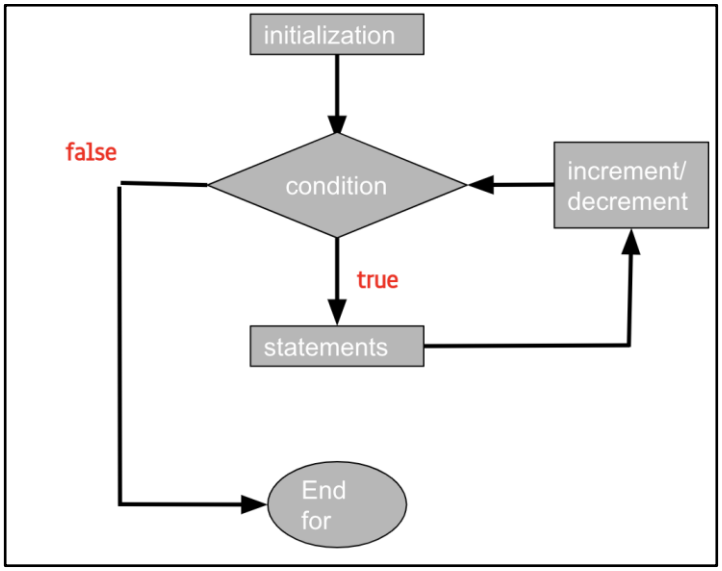
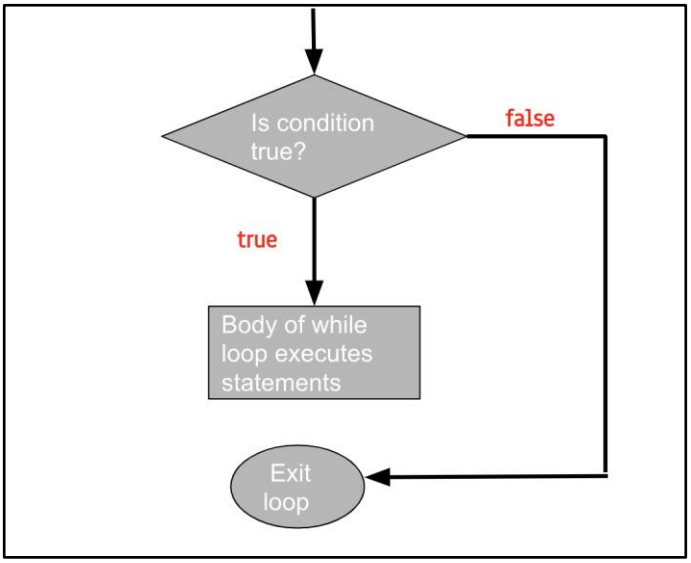



**Table 2: While-Loops FlowChart**




## Procedure

Each student was taught in a single session. Participants did not get a choice in the teaching format; they simply just chose a time slot, and we assigned a teaching method to each time slot.

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

Each class was held on a Google Meet session for an hour. Participants were given a Google Document to record their answers to the practice and final problems. The teacher who taught these classes was a master's student in computer science who regularly teaches computer science at MyEdMaster. The teacher was given the teaching materials but was not told that the classes were part of an experiment, but rather part of a free programming class being offered by MyEdMaster. The teacher taught from the provided teaching material, and students were encouraged and welcomed to ask any questions they had. In the procedural teaching conditions, the instructional portion of the session was devoted exclusively to teaching students how to do loops or lambda functions. In the conceptual teaching conditions, the instructional portion of the session was split 75% for teaching the concepts behind the loops or lambda functions and 25% for teaching how to implement loops or lambda functions.

When the practice problems were presented to the students, before the students solved them, the teacher walked through the general idea of the solution to the problem. The students were given as much time as they needed, about 10-15 minutes. After each student finished, the teacher then walked through and demonstrated the solution to the practice problems. Once the teaching material was completed, students were presented with the final problem. The final problem was the same for each topic regardless of what teaching method they were receiving. The final problems are shown in Tables 3 and 4.

**Table 3: Final Problem for Loops Procedural and Conceptual Students**

| |
|---|
| Make a program that finds the sum of the series 3 + 33 + 333 + 3333… for n terms.<br><br>You will need to find the sum of this<br>3 + 33 + 333 + 3333 + 33333 + 333333 + 3333333<br>(1) (2) (3) (4) (5) (6) (7)<br><br>Expected Output:<br>370371 |

**Table 4: Final Problem for Lambda Function Procedural and Conceptual Students**

| |
|---|
| Write a program that goes through each element in the array called **array_nums,** and determines which numbers are odd and which are even. Then, sort and add the odd and even numbers to new arrays called **odd_nums** and **even_nums**. Next, multiply each number in **odd_nums** by the number 123 and make a new array called **new_odd_nums**. Finally, print |

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

out **even_nums** and **new_odd_nums.**

array_nums = [2,6,3,9,12,33,55,68,45,67,22,34,70,99]
Expected Output:
even_nums = [2, 6, 12, 68, 22, 34, 70]
new_odd_nums = [369, 1107, 4059, 6765, 5535, 8241, 12177]

As can be seen in Table 3, the students were presented with a final problem in which they would have to choose between using a for-loop, or while loop and apply the concepts they learned in the lesson. Table 4 describes the problem given to the advanced students, they would have to choose between map and filter functions within the lambda function to complete the task. Both of these problems were created so that the solution provided by the students would be reflected if the student truly understood the concepts. Students wrote down their solutions, which we then collected. The exact process was repeated for each session we had for each topic. The students' solutions to these final problems for loops and lambda functions were the data we used in this experiment.

Once we had collected all the data, the solutions written by the students were sent over to the teacher to be graded. The solutions were sent with students' names and teaching conditions removed to avoid bias. As shown in Table 5, the teacher was provided with a rubric to grade the solutions on. The rubric consisted of four categories of syntax, logic, correctness, and an overall category of the total.

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

**Table 5: Grading Rubric for Python Final Problem**

| | **Advanced 4** | **Proficient 3** | **Approaching Proficiency 2** | **Beginning 1** |
|---|---|---|---|---|
| **Syntax** Ability to understand and follow the rules of the programming language. | Program compiles and contains no evidence of misunderstanding or misinterpreting the syntax of the language. | Program compiles and is free from major syntactic misunderstandings, but may contain non-standard usage or superfluous elements. | Program compiles, but contains errors that signal misunderstanding of syntax. | Program does not compile or (in a dynamic language) contains typographical errors leading to undefined names. |
| **Logic** Ability to specify conditions, control flow, and data structures that are appropriate for the problem domain. | Program logic is correct, with no known boundary errors, and no redundant or contradictory conditions. | Program logic is mostly correct, but may contain an occasional boundary error or redundant or contradictory condition. | Program logic is on the right track with no infinite loops, but shows no recognition of boundary conditions (such as < vs. <=) | Program contains some conditions that specify the opposite of what is required (less than vs. greater than), confuse Boolean AND/OR operators, or lead to infinite loops. |
| **Correctness** Ability to code formulae and algorithms that reliably produce correct answers or appropriate results. | Program produces correct answers or appropriate results for all inputs tested. | Program produces correct answers or appropriate results for most inputs. | Program approaches correct answers or appropriate results for most inputs, but can contain miscalculations in some cases. | Program does not produce correct answers or appropriate results for most inputs |

The provided rubric is based on the Computer Programming Grading Rubric from California State University Long Beach, College of Engineering. The teacher then sent back the graded solutions for each of the students. The highest possible score possible for this final problem was 12.

## 3. Results and Discussion

**Table 6: Mean Scores for Python Final Problem**

| | Procedural Group | Conceptual Group |
|---|---|---|
| Beginner | **Syntax:** 3.4<br>**Logic:** 3.6<br>**Correctness:** 2.7<br>**Total:** 9.7 | **Syntax:** 3.7<br>**Logic:** 3.6<br>**Correctness:** 3.1<br>**Total:** 10.4 |
| Advanced | **Syntax:** 4<br>**Logic:** 3.8<br>**Correctness:** 3.5 | **Syntax:** 2.9<br>**Logic:** 3.4<br>**Correctness:** 2.3 |

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

| | | |
|---|---|---|
| | **Total:**11.3 | **Total:** 8.7 |

As observed, Table 6 compares the mean scores for each subgroup in the rubric between the procedural and conceptual groups based on skill level. T-test values were calculated for the procedural and conceptual categories for each topic. The outcome for the loops is that $t<1$, for all grading categories, indicates not statistically significant. It is understood that teaching formats do not play a significant role in learning programming at the beginning level. On the other hand, results for the lambda functions varied based on teaching format. Categories of syntax and total performance for lambda functions had t-test values of $t=2.79$ and $t=2.42$, with $p < .05$ for each t value. This suggests that the differences between the categories of syntax and total performance are statistically significant. The correctness category had a t-test value of $t=1.96$ and a p-value of $p=0.64$. This approached statistical significance. The category of logic was concluded as not statically significant for lambda functions with $t<1$. It is thus concluded that for advanced programmers teaching syntax, correctness, and the overall total in a procedural manner can result in better scores. Teaching procedurally for syntax and correctness is preferable, but the same can not necessarily be derived for the category of logic.

## 4. Conclusions and Recommendations

It can be concluded that for beginner-level Python students, there is no preference for teaching format. Educators could take these results and implement either a procedural and conceptual teaching manner when teaching beginners. In contrast, the advanced Python students performed with overall higher scores when taught procedurally. One explanation for this difference may be that the beginning topic (loops) was easy enough to learn that students in the conceptual condition were able to understand how to implement loops even though only 25% of the instructional time was devoted to teaching actual implementation procedures. One the other hand, because lambda functions are more difficult to master, students may have benefited from the extra time learning programming procedures for lambda functions.

A similar relationship in the syntax category of the rubric can be noticed between the beginner and advanced students. Research performed by Bhandarkar et al. suggests that syntactic lesson plans for beginner Java students result in higher scores than semantic lesson plans [2]. It is noticed that for advanced Python students, the category of syntax taught procedurally is statistically significant. Based on the combined results of both the present study and the Bhandarkar et al. study, we suggest that, in most circumstances involving students of varying skill levels, initially teaching the topic in a procedural manner results in the best scores. This enables students to understand how to implement these concepts before they proceed to a more conceptual level of understanding. The implication is that students need to be strong in their

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

procedural knowledge before they are ready to advance to a more conceptual level of understanding.

Skills needed to be successful in mathematics and computer science are quite similar. They are similar in the sense that one must master the basics of math or computer science before one can master the next levels of that subject. This means that it is integral that one must have a solid foundation of basics to be successful. A study done by Nadhi et al. studies students' understanding of fractions in elementary school at a conceptual vs a procedural level [7]. Through their research, they have found that students with both procedural and conceptual knowledge will be able to develop useful knowledge to learn math effectively. They conclude that the deep understanding of the basics of fractions in math gives the student the ability to be successful in higher levels. If the results are taken from the beginners in our research to use both procedural and conceptual lesson plans, this will allow students to build a solid foundation and be set up for success when learning next-level skills.

The testing of procedural or conceptual lesson plans is not limited to mathematics or programming languages like Python or Java. This can be tested on virtually any subject that can be taught to a student. If this procedural instruction was applied to every course provided at a school, would an improvement of student's scores and understanding be noticed? Are there other methods that may work better for different subjects and/or different skill levels? These questions would be useful to ask and even apply to school systems, as this could greatly improve the quality of education and the students' performance.

**References**

1. Aydin, Suleyman & Ural Keleş, Pinar & Akif, Mehmet & Aydın, Leyla. (2016). Academicians' Vıews on Conceptual and Procedural Learning in Science Educatıon. Participatory Educational Research. spi16. 121-129. 10.17275/per.16.spi.2.13.

2. Bhandarkar, Shaan & Leddo, John & Lakkoju, Siddharth & Somayyajula, Surya. (2016). Syntactic versus conceptual lesson plans: Towards improving middle school computer science curricula. International Journal of Humanities and Social Science Research. 2. 59-62. 10.1145/2160000/2157319.

3. "Computer and Information Technology Occupations: Occupational Outlook Handbook." *U.S. Bureau of Labor Statistics*, U.S. Bureau of Labor Statistics, 1 Sept. 2020, www.bls.gov/ooh/computer-and-information-technology/home.htm.

ISSN: 2455-8834

Volume:06, Issue:04 "April 2021"

4. Computer Programming Grading Rubric. California State University, Long Beach, assessment.fiu.edu/resources/rubrics-and-curriculum-maps/_assets/rubrics/Computer%20Programming%20Grading%20Rubric%20-%20California%20State%20University%20Long%20Beach.pdf.

5. Fawcett, Amanda. "How to Use Python Lambda Functions: a 5 Minute Tutorial." *Educative*, 1 Dec. 2020, www.educative.io/blog/python-lambda-functions-tutorial.

6. Isleyen, Tevfik & Işık, Ahmet. (2003). Conceptual and Procedural Learning in Mathematics. Research in Mathematical Education. 7.

7. Nahdi, Dede & Jatisunda, Mohamad. (2020). Conceptual Understanding And Procedural Knowledge: A Case Study on Learning Mathematics of Fractional Material in Elementary School. Journal of Physics: Conference Series. 1477. 042037. 10.1088/1742-6596/1477/4/042037.

8. "Python Course." *Python Tutorial: Lambda Operator, Filter, Reduce and Map*, www.python-course.eu/lambda.php.

9. "Trends in the State of Computer Science in U.S. K-12 Schools." *Google*, Google, 2016, services.google.com/fh/files/misc/trends-in-the-state-of-computer-science-report.pdf.

10. Wikipedia contributors. (2021, January 15). AP Computer Science. In *Wikipedia, The Free Encyclopedia*. Retrieved 22:36, April 6, 2021, from https://en.wikipedia.org/w/index.php?title=AP_Computer_Science&oldid=1000468208