INTERNATIONAL JOURNAL OF ACADEMIC RESEARCH IN PROGRESSIVE EDUCATION & DEVELOPMENT

OPEN ACCESS JOURNAL

Vol 14, Issue 4, (2025) E-ISSN: 2226-6348

Creating Triangle Construction Exercises as Interactive GeoGebra Applets with Automatic Verification, Hints, and Feedback

Peter Vajo

J. Selye University, Faculty of Economics and Informatics, Department of Mathematics, Bratislavská cesta 3322, 945 01, Komárno

Corresponding Author Email: petervajo4@gmail.com

DOI Link: http://dx.doi.org/10.6007/IJARPED/v14-i4/26810

Published Online: 20 November 2025

Abstract

With the growing significance of technology-based education and the advancements in Computer Algebra and Dynamic Geometry systems, automated tools for interactive learning have attracted significant attention. Nonetheless, the scarcity of interactive open-source online tools for geometric construction problems that offer feedback has hindered their wider implementation within the educational system. The primary reason for this absence is that geometric construction problems necessitate a constructed object as a solution rather than a numerical one. This feature renders the evaluation of geometric construction problems to be particularly challenging for machine evaluation (solutions cannot be copied into answer boxes) and time-consuming for teacher evaluation. In this work, the focus will be on triangle construction problems within the broader topic of geometric constructions. Emphasis will be placed on methods for creating interactive triangle construction applets in the widely popular GeoGebra software. The necessary steps will be outlined to enable mathematics teachers to create interactive GeoGebra applets with automatic verification that offer feedback for a diverse range of triangle construction problems.

Keywords: Triangle Construction Problems, Interactive Learning, Feedback, Geogebra, Automatic Verification

Introduction

Due to the rapid development of educational technologies, there is an increasing need for dynamic mathematical software in mathematics lessons (Stols and Kriek 2011; Zengin and Tatar 2017).

The use of various technologies and mathematical software can improve the quality of education, enrich the visualization of geometry, create opportunities for creative thinking, establish a student-centered atmosphere, capture and sustain students' attention, and motivate them to become more deeply involved in the learning process (Sanders, 1998; Bhagat et al. 2016; Hegedus et al. 2015; Stols and Kriek 2011). Mastering geometry through conceptual understanding requires the use of appropriate methods that actively engage

students and support them in becoming independent, critical thinkers and self-directed learners (Gurmu et al. 2024). A tool that facilitates the attainment of such goals can be the widely popular GeoGebra software. GeoGebra is a free educational software with over 100 million users worldwide (Botana et al. 2024). It is available on the web and through the GeoGebra application on computers, tablets, and smartphones; additionally, applets created with GeoGebra can be integrated into various learning management systems (Csiba and Vajo 2024). The Geogebra software creates a lively and dynamic learning environment where conceptual learning and problem-solving-based learning take precedence over memorization (Tatar and Zengin 2016; Suweken 2020).

The aim of this work is to use the possibilities offered by GeoGebra to create interactive applets with automatic verification that would contribute to the teaching and learning of triangle construction problems at different educational levels (secondary and higher education). The necessary steps will be presented, which, when followed and reproduced, will enable us to create well-functioning interactive applets on the topic of triangle construction problems.

Theoretical Basics

In order to examine the process of creating interactive GeoGebra applets with automatic verification, it is essential to emphasize the fundamental mathematical principles upon which the applets are grounded. As we are addressing a mathematical problem, it is necessary that the basic characteristics and properties of triangles are preserved, including:

- The sum of the interior angles of a triangle is 180°.
- In a triangle, the angle opposite the longest side is also the largest (i.e., the sine rule).
- The perpendicular bisectors of the sides of a triangle intersect at a single point. This is the center of the circle that the triangle can be inscribed in.
- The angle bisectors of the interior angles of a triangle intersect at a single point. This is the center of the circle that can be inscribed in the triangle.
- The altitudes of a triangle intersect at a single point. This is called the orthocenter of the triangle.
- The medians of a triangle intersect at a single point. This point is called the centroid of the triangle. The centroid divides the medians in a 2:1 ratio from the vertex.
- The segments connecting the midpoints of the sides of a triangle are called the midsegments of the triangle.
- Any midsegment of a triangle is parallel to the side that does not contain its midpoint and is half as long. The three midsegments divide the original triangle into four congruent triangles.

It is also worth mentioning that, from the perspective of constructibility, certain essential conditions and properties must be fulfilled, such as the triangle inequality (the sum of the lengths of any two sides must be greater than the length of the third side), and that the length of a median can never be less than the height corresponding to the same side.

Furthermore, the role of geometric loci is essential in the creation of the triangle construction applets, as each newly constructed object is essentially a set of points that meet certain criteria—a geometric locus. Geometric loci include circles, parallel lines, perpendicular bisectors, angle bisectors, Thales' circles, ellipses, parabolas, hyperbolas, and the Apollonius circle. In construction exercises, as Varga (1969) writes, geometric loci are

often defined implicitly (e.g., determine the geometric loci of points 4 cm from the given line!).

Steps for Creating Interactive Triangle Construction Applets with Automatic Verification and Feedback

The following section describes the steps for creating interactive triangle construction applets with hints and feedback in GeoGebra. Figure 1 shows the appearance of a finished applet. The applet contains the text of a chosen exercise, including the data necessary for a successful construction. This data (length of the altitude and median) is generated randomly so that the solver is presented with new data after each update. In addition, the applets contain a modified toolbar and three dedicated buttons, from left to right: the Verification Button, the Help Button, and the Refresh Button.

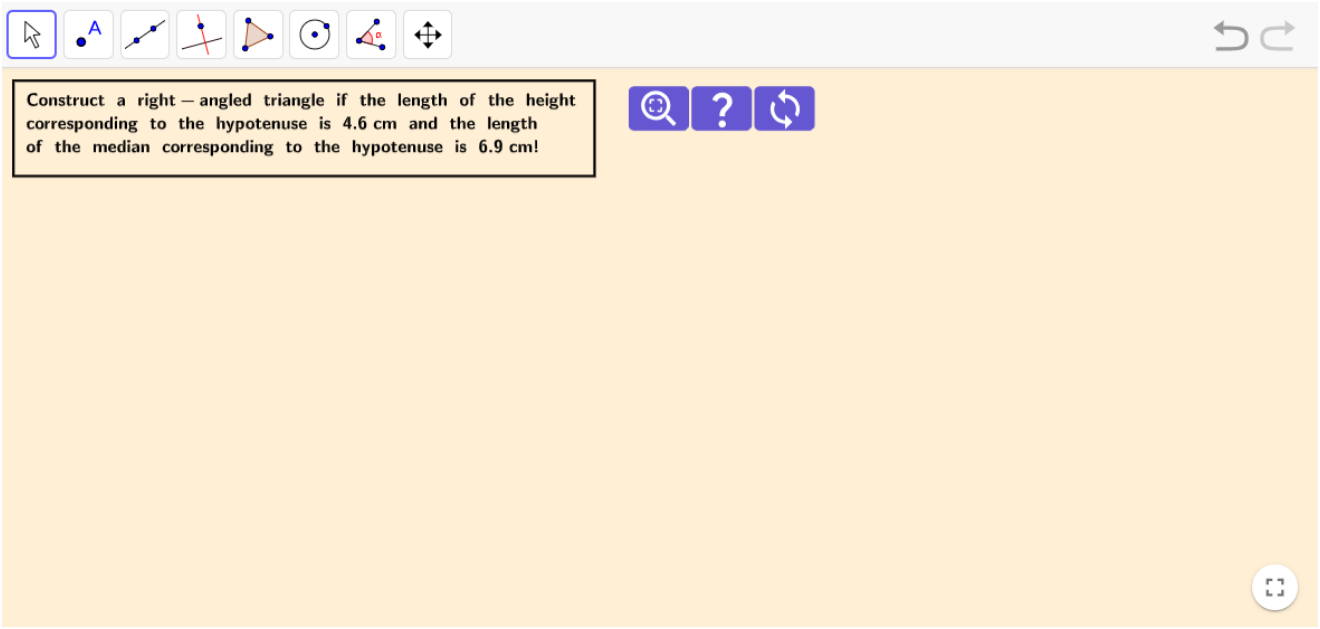

Figure 1: Interface of a selected triangle construction exercise

An intriguing observation is the blankness of the applet's construction area—it does not contain a pre-constructed starting object from which the solver must continue the construction. This implementation gives the solver greater freedom in constructing the correct solution, as we do not limit the initial step of the construction process to our ideas. The students may arrive at the correct solution using different construction steps or a different order of steps. Furthermore, such an approach can be more challenging, especially in exercises where the first step of the construction is not clear or the three data required for the construction are not self-evident (see Figure 1).

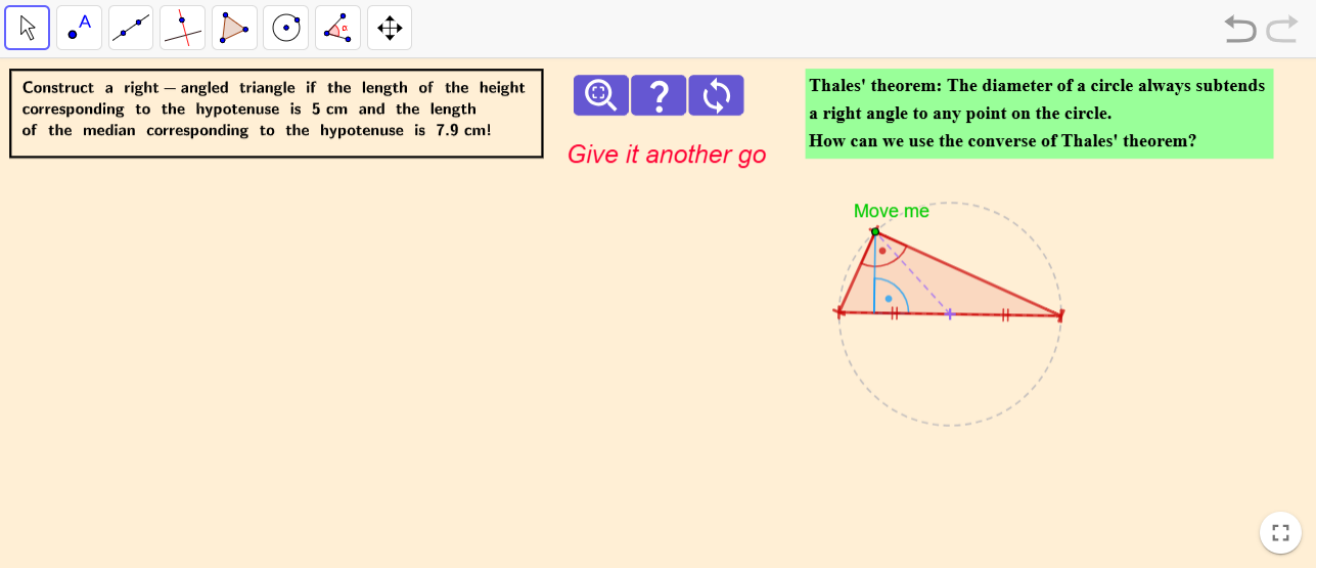

Figure 2: Interface of a selected triangle construction exercise with hints and feedback

The *Verification Button* provides textual feedback for the solver about the correctness of their solution. As shown in Figure 2, since there is no correct solution constructed (the solver has not drawn anything yet), the textual feedback indicated no correct solutions found to the problem. The *Help Button* reveals a hint for the solver, often about the nature of visual representations and highlighting key geometric properties. The *Refresh Button* creates new values for the given data appearing in the description of the problem, providing the solver with a new set of values for the same problem.

Constructing the Solution

Regardless of the triangle construction problem, the first step in the creation of applets with automatic verification is to construct the correct solution or solutions (see Figure 3).

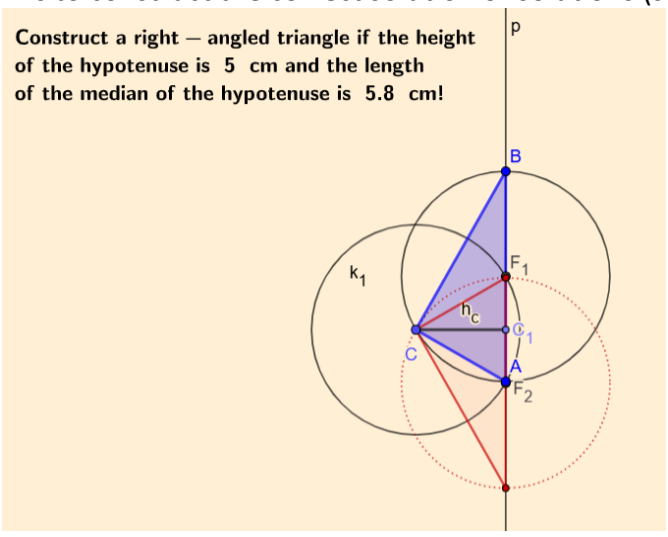

Figure 3: Constructing the solution to a given problem

After constructing the correct solution, we hide it along with all the auxiliary constructions from the applet interface—we can do this in the Show object setting. This way, the solution will not be visible to the solver. The process of constructing and hiding the correct solution to the problem can be considered the first step in creating interactive applets with

automatic verification, as this constructed and hidden solution is essentially the basis for comparison for the later constructed objects by the solver.

Using JavaScript

An important function of interactive applets is automatic verification, which checks the correctness of the solution constructed by the solver. This verification is performed by a pre-written JavaScript code, so it is not necessary (nor expected) for mathematics teachers to have this level of IT or coding knowledge. For the verification function to work, the JavaScript code below must be inserted into the appropriate place in the applet. This process can be done in the settings of any element or object placed on the applet interface by following the Script→Global JavaScript menu items. The pre-written JavaScript is used in the interface found under Global JavaScript:

```
function ggbOnInit() {
   ggbApplet.debug("ggbOnInit");
   ggbApplet.registerAddListener("newObjectListener");
   ggbApplet.registerUpdateListener("myUpdateListenerFunction");
}
function myUpdateListenerFunction(obj) {
   if (obj == "result") {
      var finished = ggbApplet.getValueString("finished");
      var res = ggbApplet.getValueString("result");
      if (res.indexOf("true") > -1) {
         if (finished.indexOf("true") > -1) {
            ggbApplet.setVisible("result1", true);
         } else {
                  ggbApplet.setVisible("result2", true);
           }
      } else {
         ggbApplet.setVisible("result1", false);
         ggbApplet.setVisible("result2", false);
      }
   }
}
function newObjectListener(obj) {
   var finished = ggbApplet.getValueString("finished");
   if (finished.indexOf("true") == -1) {
     if (obj != "finished") {
        var cmd = "finished = AreCongruent(" + obj + ",solution)";
        ggbApplet.debug(cmd);
        ggbApplet.evalCommand(cmd);
        finished = ggbApplet.getValueString("finished");
      }
   }
}
```

Since these are already complete, well-functioning scripts, they do not require any additional IT or coding knowledge on the part of the mathematics teacher. By copying them,

the functionality of the applets can be reproduced in other exercises as well. To explain the above JavaScript in more detail, the script counts all the constructed objects and compares them with the correct solution we constructed in the first step. If it finds that they are congruent, then the object constructed by the solver is evaluated as true. This is done by the script line ***var cmd = "finished = AreCongruent(" + obj + ",solution)";*** which compares all newly constructed objects with the correct solution named "solution" (see Figure 4—it compares the newly constructed triangle z_1 with the correct solution, and since they are not congruent, it evaluates the solution as false, meaning that no correct solution has been found). If we do not want to change the term "solution" in the script (which refers to the correct solution we have constructed and hidden), then we need to rename our correct solution from the first step to "solution" in the script. This way, the name of the hidden solution will be "solution," and the JavaScript-controlled verification function will work.

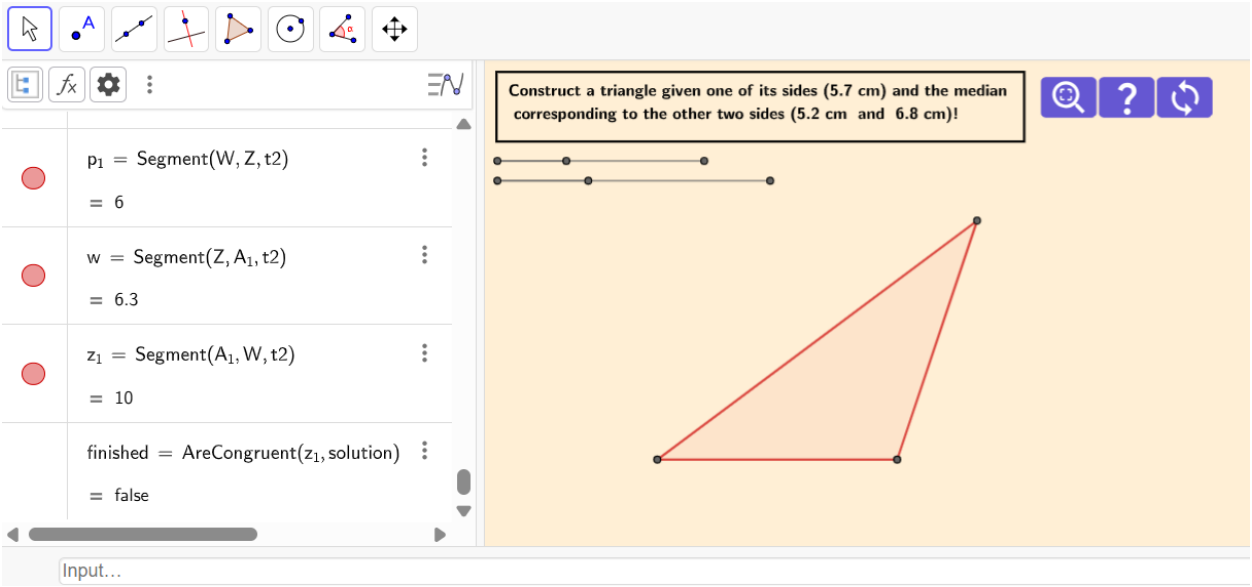

Figure 4: The finished Boolean value is false, since the constructed triangle is not congruent to the solution

Incorporating all Solutions into the Script

All possible solutions can be understood as all congruent (symmetrical) correct solutions (see Figure 5) and non-congruent correct solutions. Since symmetrical solutions follow the same line of thought and construction steps, in this work we devote our time to the incorporation of non-congruent correct solutions into the applets' script.

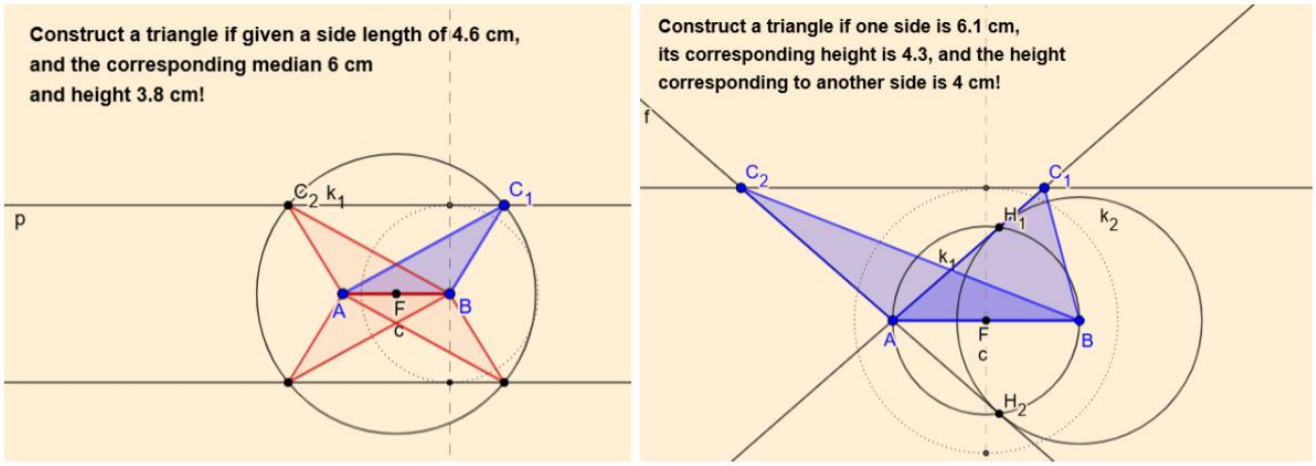

Figure 5: Multiple congruent and non-congruent solutions to the same problem

A typical cause of non-congruent solutions is the position of the height line in a triangle. Figure 5 illustrates the two non-congruent correct solutions to the exercise: one with the height inside the triangle and the other with the height outside the triangle. We can achieve the detection of these solutions and the correct evaluation of the exercise by extending our JavaScript. After constructing, renaming (solution1 and solution2), and hiding both correct solutions, we extend the next line of the script as follows:

var cmd = "finished = (AreCongruent(" + objName + ", solution1) || AreCongruent(" + objName + ", solution2))";

This allows us to provide different types of feedback to the solver depending on the number of solutions constructed, including different types of point-based or textual feedback and evaluation. Within GeoGebra applets, textual feedback can be provided, which appears depending on the value of the finished variable (feedback indicating a correct, incorrect, or partially correct solution).

When the applets are integrated into various learning management systems, the system recognizes the finished variable and scores the solver based on its value.

Randomization

Another important step in creating triangle construction applets is randomization. Since these applets can be used not only for visualization but also mainly for practicing exercises on this topic and for homework and assignments, it is necessary to randomize the data appearing in the exercise text. A good way to achieve such randomization is to use random sliders (see Figure 6), keeping in mind the constructability and appearance of the solution (e.g., no internal angle should be less than 10°) and avoiding certain intersection points from being "too close" to each other, which would cause inconvenience during construction.

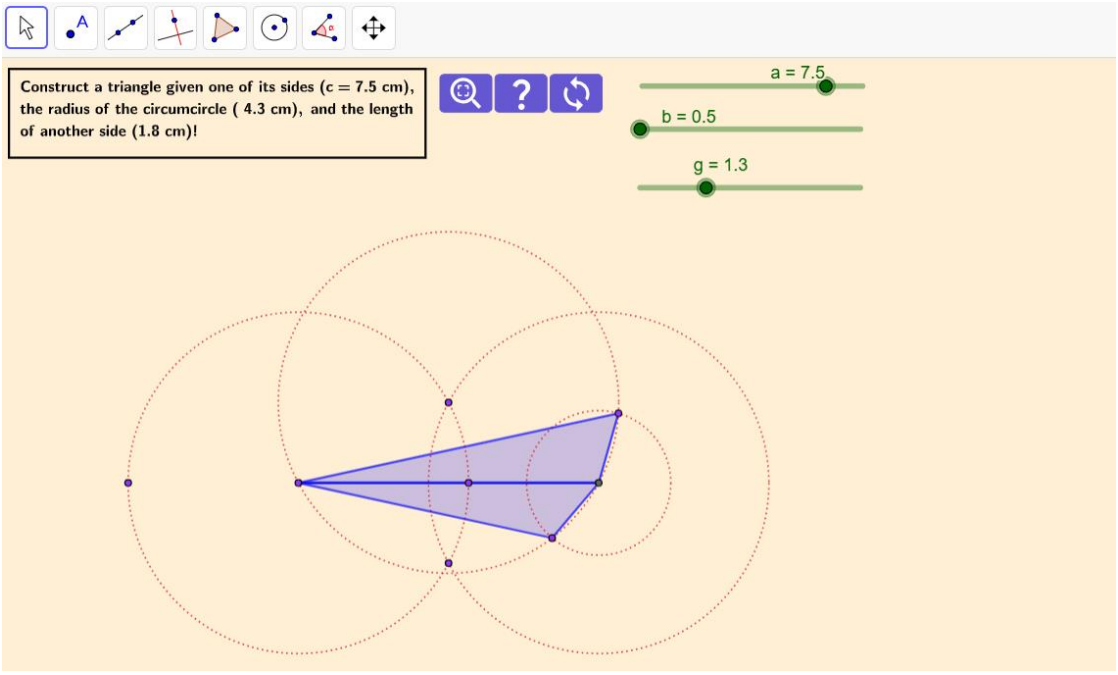

Figure 6: Randomization options using random sliders

By using random sliders and other randomization methods to modify the hidden correct solution during the initial construction, we ensure that students will encounter different values for the given data each time they open the same exercise, presenting them with a new challenge when they solve it again.

Feedback to the Solver

Another role of the JavaScript code used is that it allows the teacher to provide different textual feedback to the solver based on the number of correct solutions constructed. For example, upon finding and constructing the correct solution to a problem, the JavaScript makes pre-made textual feedback visible to the solver. In the creation of the interactive applets presented in the paper, one further step was made to optimize the functionality of this feedback, a Button was introduced to control the visibility of the textual feedback made visible by the JavaScript code. This means the solver can check the correctness of their solution anytime by using the Verification Button (see Figure 7).

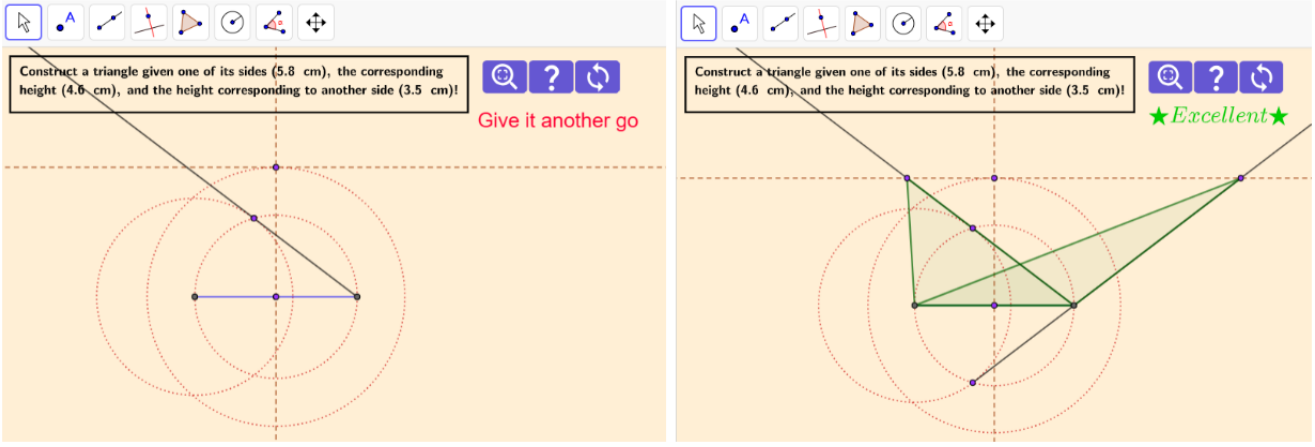

Figure 7: Different types of textual feedback

There can be even a third type of textual feedback indicating the special cases where two different non-congruent solutions can be constructed as correct answers to the problems. This textual feedback, as shown in Figure 8, indicates the correctness of the constructed solution but guides the solver's attention to the fact that there is another required correct solution to the problem.

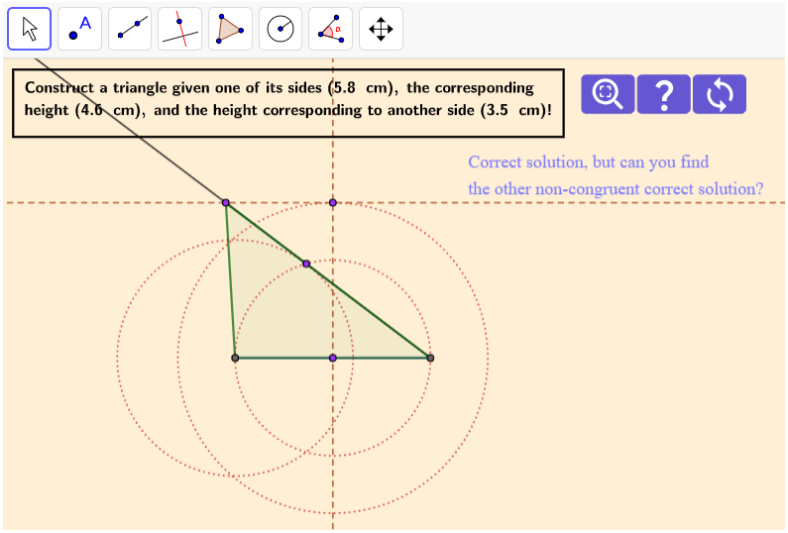

Figure 8: Textual feedback in case of non-congruent correct solutions

Conclusion

The various mathematical software and dynamic geometry environments are not intended to replace teachers but rather to become effective tools in their hands, integrated into the teaching and learning process. The methods presented enable mathematics teachers to create interactive applets with automated verification containing randomly generated data according to their ideas for triangle construction problems. These applets can be used on their own and on the GeoGebra website or can be integrated into learning management systems to facilitate the work of teachers, providing students with entertaining exercises and a learner-centered approach. By utilizing the Classroom feature of the GeoGebra platform, teachers can monitor their students' progress in real time, and by using the Verification Button (if the students have not already done so), they can swiftly confirm the correctness of their solutions. The aforementioned function can significantly assist mathematics teachers in evaluating homework assignments, as well as tests and examinations, in contrast to the traditional, time-consuming evaluation process.

Acknowledgements

This research was supported by KEGA grant (Kultúrna a edukačná grantová agentúra MŠVVaM SR) number 004UJS-4/2025.

References

Bhagat, K. K., Chang, C. N., Chang, C. Y. (2016). The impact of the flipped classroom on mathematics concept learning in high school. *Educational Technology & Society*, 19 (3) , 124–132.

Botana, F., Recio, T., & Vélez, M. P. (2024). On Using GeoGebra and ChatGPT for Geometric Discovery. *Computers*, 13(8), 187. https://doi.org/10.3390/computers13080187

Csiba, P., & Vajo, P. (2024). Problems and challenges of using randomized automatically evaluating geometric construction problems in Moodle LMS[J]. *AIMS Mathematics*, 9 (3), 5234–5249, https://doi.org/10.3934/math.2024253

Gurmu, F., Tuge, C., & Hunde, A. B. (2024). Effects of GeoGebra-assisted instructional methods on students' conceptual understanding of geometry. *Cogent Education*, 11(1). https://doi.org/10.1080/2331186X.2024.2379745

Hegedus, S. J., Dalton, S., Tapper, J. R. (2015). The impact of technology-enhanced curriculum on learning advanced algebra in US high school classrooms. *Educational Technology Research and Development*, 63 , 203–228, https://doi.org/10.1007/s11423-015-9371-z

Sanders, C. V. (1998). Geometric constructions: visualizing and understanding geometry. *Mathematics Teacher*, 91(7), 554-556.

Stols, G., & Kriek, J. (2011). Why don't all maths teachers use dynamic geometry software in their classrooms?. *Australasian Journal of Educational Technology*, 27 (1), 137–151, https://doi.org/10.14742/ajet.988

Suweken, G. (2020). STEM oriented mathematics learning with GeoGebra. In *3rd International Conference on Innovative Research Across Disciplines (ICIRAD 2019)* (pp. 258-263). Atlantis Press.

Varga, T. (1969). *A matematika tanítása*. Budapest: Eötvös Loránd University.

Zengin, Y., & Tatar, E. (2017). Integrating dynamic mathematics software into cooperative learning environments in mathematics. *Journal of Educational Technology & Society*, 20 (2), 74–88, Retrieved from http://www.jstor.org/stable/90002165