

Using computational thinking to teach foreign languages: A preliminary approach

Apostolos Syropoulos* , Greek Molecular Computing Group, 67133 Xanthi, Greece

Eleni Tatsiou, 2nd Gymnasium of Xanthi, Xanthi, Greece

Renate Wachter, Bundeshandelsakademie, Laa an der Thaya, Austria

Suggested Citation:

Syropoulos, A., Tatsiou, E. & Wachter, R. (2023). Using computational thinking to teach foreign languages: A preliminary approach. *Global Journal of Foreign Language Teaching*. 13(4), 223-235.
<https://doi.org/10.18844/gjft.v13i4.8954>

Received from April 20, 2023; revised from September 22, 2023; accepted from November 12, 2023.

Selection and peer review under the responsibility of Assoc Prof. Dr. Jesus Garcia Laborda, Alcala University, Spain

©2023 by the authors. Licensee Birlesik Dunya Yenilik Arastırma ve Yayıncılık Merkezi, North Nicosia, Cyprus.

This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract

Computational thinking is a teaching methodology that has been successfully applied to STEM education. However, its use outside STEM education seems to be quite limited. This study aims to report a small-scale experiment in which a small group of students were taught elements of the Greek language using STEM education methodology. Since computational thinking is mainly about algorithms and their use in teaching, the study has used some tools to implement the corresponding ideas. The results of this project are encouraging and the study indicates that language learning can greatly benefit from using computational thinking.

Keywords: Abstraction; Algorithm; Computational Thinking; Language Learning; Teaching.

* ADDRESS FOR CORRESPONDENCE Apostolos Syropoulos, Greek Molecular Computing Group, 67133 Xanthi, Greece
E-mail address: asyropoulos@aol.com

1. Introduction

Whether we like it or not, we live in a global village and we need to be able to communicate with our fellow humans. Since people speak different languages, most people opt to communicate in languages that are widely used. Currently, English (in particular American English) is the de facto lingua franca. Thus, most people strive to learn English to be able to communicate with other people even by using simple websites such as YouTube (Alobaid, 2020), programmed Artificial intelligence dialogue systems (Zhai, 2023; Huntington et al., 2023; Hellmich & Vinall 2023), and other mobile technology which helps to improve learners cognitive and individualism also helping teachers to facilitate learning (Togaibayeva et al., 2022). As a result of this, many language teachers are trying to develop and devise different methodologies to effectively teach English to non-English speakers (Brown, 2009). Language teachers that teach different languages to their students usually have adopted these methodologies to their particular teaching subject, since there is no reason to reinvent any wheel!

The Erasmus+ project allows schools to form partnerships and collaborate on the exploration of various subjects. Most of these subjects are, let us say, naive but sometimes they are quite sophisticated. In 2018, six schools from Austria, Greece, Cyprus, Portugal, Serbia, and Romania formed a partnership to explore the use of different teaching methodologies in different contexts. Because of the pandemic, they did not have the chance to work on the full extent of the idea of the project, nevertheless, they managed to apply, to some degree, computational thinking to language teaching. However, to examine the usefulness of this idea, we decided, since we were members of this partnership, to check these ideas by introducing the Greek language using computational thinking to students with no prior knowledge of Greek whatsoever.

The term *computational thinking* was coined by Seymour Papert (1928–2016) in his book entitled “Mindstorms: Children, Computers, and Powerful Ideas” in 1980 (Papert, 1980). In this book, he also introduced the *samba schools for computation*, which are the computational analog of the samba dance schools of Rio de Janeiro. These are not schools since they do not offer instruction in a formal setting. However, people practice and often perform and this is an informal way to teach samba to newcomers. Papert was convinced that since traditional teaching methods have reached their limits, these samba schools could be used to teach programming and, more generally, they could be the starting point for a new teaching methodology. Later on, Papert (1996) in his paper entitled “An Exploration in the Space of Mathematics Educations” noted:

The goal is to use computational thinking to forge ideas that are at least as “explicative” as the Euclid-like constructions (and hopefully more so) but more accessible and more powerful.

Firstmost, Papert was a mathematician, then an educator, and, finally, a computer scientist. Based on his understanding, he designed and implemented the LOGO programming language to facilitate the teaching of mathematics (this should be clear from the previous quotation). Nevertheless, LOGO is not a good teaching vehicle for people who want to introduce programming to novices (Syropoulos & Stavrianos, 2014). Although Papert was the one who coined the term “computational thinking”, still it was Jeannette Marie Wing (Wing, 2008) who defined what is computational thinking [see also (Pollak & Ebner, 2019) for a recent overview of the subject]. In a nutshell, Wing defined computational thinking as a problem-solving methodology, a systems design methodology, and a tool to understand human behavior based on the fundamental ideas of informatics.

1.1. Literature review

1.1.1. What is computational thinking?

One could define computational thinking as a teaching methodology that is based on the way people program computers. In general, when we code to solve any problem with a computer, we try to find ways to represent the components of the problem with data structures. Then, we apply

computational methods to these structures to solve the problem. Going from concrete objects to data structures is a form of *abstraction*. Let us give a concrete example to make clear the process. If one person has five apples and she gets two more apples, then she will have seven apples. Similarly, if she has five pears and then she gets two more pears, in the end, she will have seven pears. From these, one can easily deduce that $5 + 2 = 7$.

In programming, we usually go one step further and abstract structures that are quite useful. For example, a queue and a stack are two “structures” that we find in our everyday life and an abstraction of them is usually quite useful when trying to solve problems that involve them. Unfortunately, it is not possible to add stacks, although sometimes it makes sense to add queues. In addition, it is not possible to multiply either stacks or queues with numbers. Furthermore, it is impossible to merge two algorithms to faster solve any particular problem. Thus, the abstractions we are talking about do not have the clean and easily definable algebraic properties of mathematical abstractions.

In computing, formal languages are of paramount importance. For example, the syntax of all programming languages is defined by formal languages. In particular, each formal language consists of an *alphabet* (a set of distinct symbols) whose elements are called *letters*, a set of words (sequences of letters), and a set of words that are *well-formed* according to a set of rules. For example, the ten decimal digits plus the decimal sign may form an alphabet, each positive decimal is a word, and the set of positive decimal numbers is a language. In mathematics, the union of two formal languages is a well-defined operation, nevertheless, the union of two formal programming languages, is a meaningless operation. Thus, it makes no sense to try to define the union of Java with, say, Python or Perl!

A third problem that concerns computational abstractions is that things like stacks and queues are used in programs that are executed by real machines. Practically, this means that although a stack may have an unlimited number of elements, a real-life stack can hold a limited number of elements and this depends on the available computational resources. This is why programmers should take care of abnormal situations (a programmer should take measures to prevent a stack overflow).

When working with rich abstractions (abstractions that are more elaborate than those encountered in mathematics) it is crucial to be able to define the right level of abstraction for each specific case. The abstraction process by which we decide what details we need to highlight and what details we can ignore forms the basis of computational thinking.

When coding a computer program, we usually work with layers. For example, when someone designs a library (i.e., a collection of routines that can be used to solve a set of similar problems), she has to design an interface that will allow users to easily use this library. In computer parlance, this interface is known as *API* (application programming interface). Thus, the programmer does not need to know which people are going to use the library and at the same time, potential users do not need to know the implementation details of the library to use it effectively. In a nutshell, we could say that the basic practical details of computational thinking are the definition of abstractions, working with multiple layers of abstraction, and understanding the relationships among the different layers.

When coding our basic “mental” are abstractions. The power of these “mental” tools is increased by the power of the tools on which we run them. In a final analysis, one could argue that programming is a process by which we automate our abstractions. Automation is deeply connected to computing devices that will interpret abstractions. Since humans are biological machines able to compute, it is possible that a human being can be a computer. Moreover, it is quite possible to have a combination of a conventional computer and a human to act as a computer. Furthermore, a computer network could be used as a computer. To what extent a human can perform difficult computational tasks is an open problem [see (Syropoulos, 2008) for more details].

1.1.2. Computational thinking everywhere

Bundy (2007) has noted that “computational thinking is influencing research in nearly all disciplines, both in the sciences and the humanities.” In addition, he introduced the idea that concepts common in computation can be considered as a new language for describing hypotheses and theories. Furthermore, he goes on to assert that if one wants to understand the 21st Century, then one must first understand computation. To prove his points, he mentioned a series of seminars held at the University of Edinburgh that explored these ideas. At these seminars, one or more experts have discussed the influence of computational thinking on their discipline. The disciplines included physics, biology, medicine, philosophy, architecture, and education. Here we are interested in the application of computational thinking to education, in general, and language learning, in particular.

If we want to train students to be able to use computational thinking in solving problems, we need to first train and educate teachers. It is our understanding that computational thinking is not part of the program planning of curricula in most EU countries. Moreover, it is not clear if educational institutes of these countries push decision-makers to integrate this learning and problem-solving methodology into school curricula. However, Yadav et al. (2014) reported a study on the design of computational thinking modules and their introduction to elementary and secondary education schools in the United States. Not so surprisingly, the results of this effort were quite promising.

The general introduction of computational thinking in elementary and secondary education is a very interesting subject. Jacob et al. (2018) have concluded that the introduction of computational thinking in STEM education is relatively easy. One may argue that this happens because students who are good in STEM lessons are well-versed in programming, but this is not the general case. Although computational thinking has its roots in informatics, still it is not part of informatics like, say, compiler construction theory or database theory. Voogt et al. (2015) explained why computational thinking is more than programming and why it might be helpful to expose students to programming concepts and ideas. Some authors [see (Voogt et al., 2015)] have suggested that a good understanding of the following subjects could be beneficial to students aiming to use computational thinking: data collection, data analysis, data representation, problem decomposition, abstraction, algorithm and procedures, automation, parallelization (i.e., processing data by many different agents at the same time), and simulation. An alternative approach is the use of computational thinking as a problem-solving process that has the following characteristics:

1. formulation of problems in a way that enables us to use a computer and other tools to help solve them.
2. logical organization and analysis of data;
3. representation of data through abstractions such as models and simulations;
4. automation of solutions through algorithmic thinking (a series of ordered steps);
5. identification, analysis, and implementation of possible solutions to achieve the most efficient and effective combination of steps and resources; and
6. generalization and transfer of the problem-solving process to a wide variety of problems.

1.1.3. Computational Thinking and Foreign Language Learning

When computational thinking finds its way into school curricula, it is obvious that foreign language learning will be affected. The real question is how can this happen? To answer this question, we need to understand if and how linguistics has been affected by computational thinking, in general. To start with, let us note that the syntax of a language, in general, or of a linguistic structure, in particular, can be described by Chomsky's (1959) grammar. In addition, today people are using special tree-like diagrams to study and analyze linguistic structures. Furthermore, in recent years some scholars have tried to use the intuitionistic type theory, which was introduced by Per Erik Rutger Martin-Löf, to study linguistic structures (Ranta, 1994). Also, many scholars have used categorical grammars and the calculus of Jim Lambek in the semantic analysis of languages (Carpenter, 1997). Of course, one should not forget computational linguistics, natural language processing, and the recent developments in chatbot technology (i.e., programs like ChatSpot, Chat GPT, Bing Chat, and Bard). Naturally, all these

imply that computational thinking can be used in foreign language teaching. But let us try to be concrete.

Suppose that a teacher has to introduce her students to conditional/hypothetical sentences of the English language. The first form of the sentences has the following general form: if + present simple, ... will + infinitive. Clearly, one can express this form with grammars, syntax trees, etc. Going from a linguistic pattern to a computational structure is a form of abstraction. In case one has to form a conditional sentence of this form, she has to know what is the present simple and what is the infinitive. Note that here we break the original problem into subproblems to solve it. What is intriguing, at least for us, is that here we “subconsciously” use computational thinking to solve a problem. From this simple example, and many more, we have concluded that people who are knowledgeable in computational thinking can use it in foreign language teaching. Also, teachers who, in addition, have some programming skills, can use their skills to create games and tools that would make learning really fun (Amukune et al., 2022; Bakan et al., 2022).

There is something that every teacher who attempts to use computational thinking in her classes should do—under no circumstances a teacher should make use of computational terms or, even worse, of mathematical formalism in her classes. Students should not be aware that they are using an algorithm or that they are using abstractions. However, there are cases where we describe a method that can tackle a particular problem. Since in most cases, these methods are algorithms, a teacher must describe them using simple everyday terms (Jacob et al., 2018). Thus, we can use words like plan of action or strategy instead of the word algorithm. To explain what is a “plan of action,” a teacher can use a morning routine to demonstrate the idea. For example, someone has to wake up at a specific time during work days, she must have breakfast, use the bathroom, etc. All these actions make a (fuzzy) algorithm and describe easily what it means to apply a “plan of action” to achieve something. The morning routine is fuzzy because we generally describe what it involves whereas algorithms tend to be quite precise. We are convinced that this is not a real problem, at least when one tries to explain the notion of an algorithm.

1.2. Purpose of study

It seems that most European educational systems are not open to innovation in the sense that teachers are expected to strictly follow the curricula and adhere to the general teaching methodology and goals. Of course, there is a rationale behind this philosophy but still, we believe there should be room for innovation. This is the “bad” news. The “good” news is that schools can experiment and innovate by forming strategic partnerships for schools only in the framework of the Erasmus+ project (of course schools should apply for funding and if their application is successful, then they can proceed with their experiments). We teach in two different schools that participated in such a strategic partnership whose title was “Teach Me Differently.” This partnership aimed to use different teaching methodologies and to examine their usability and usefulness. One of the schools, the 2nd Gymnasium of Xanthi, Greece, according to the original plan, had to investigate the use of computational thinking in foreign language teaching. Since these partnerships involve trips where a small group of teachers and students from each participating school visit one school and participate in activities that are implemented and sometimes designed by the host school

2. Method and materials

The mini-course was a four-day event and on the fifth day everybody who attended the course answered a questionnaire about the course. In what follows, we will describe a mini course where we tried to introduce the Greek language to teachers and students from different countries who had no prior knowledge of the language.

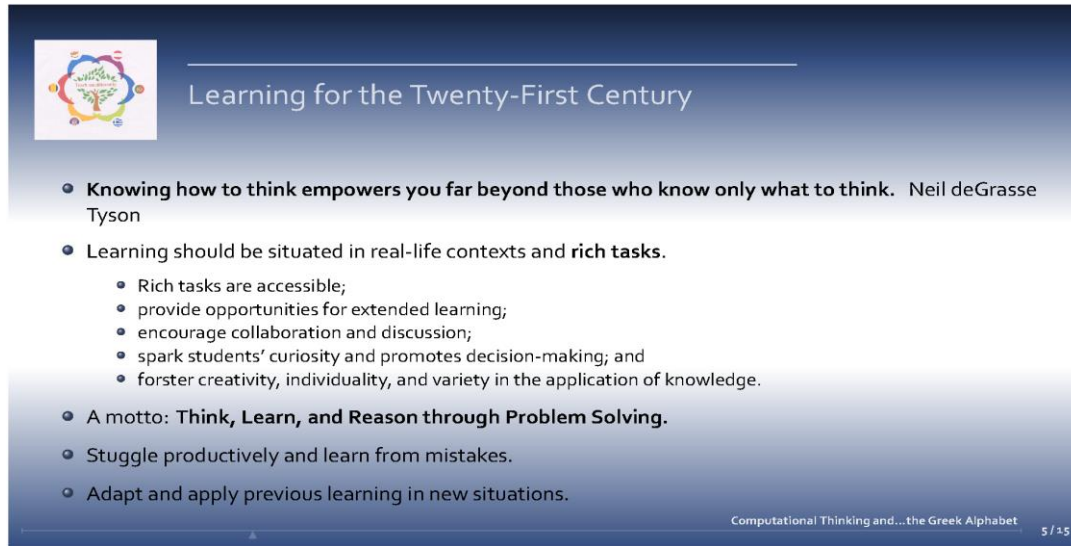
The study and its findings follow the ethical standards of conducting experimental research, and the study’s findings pose no harm to the participants, environment, or organizations involved.

3. Results

The first day started by introducing computational thinking and giving a summary of what is the essence of learning in the 21st century. This overview is shown in Figure 1. Note that instead of explicitly stating what we taught our audience, in what follows, we will just show the screens of the presentations that we have used during the lectures.

Figure 1

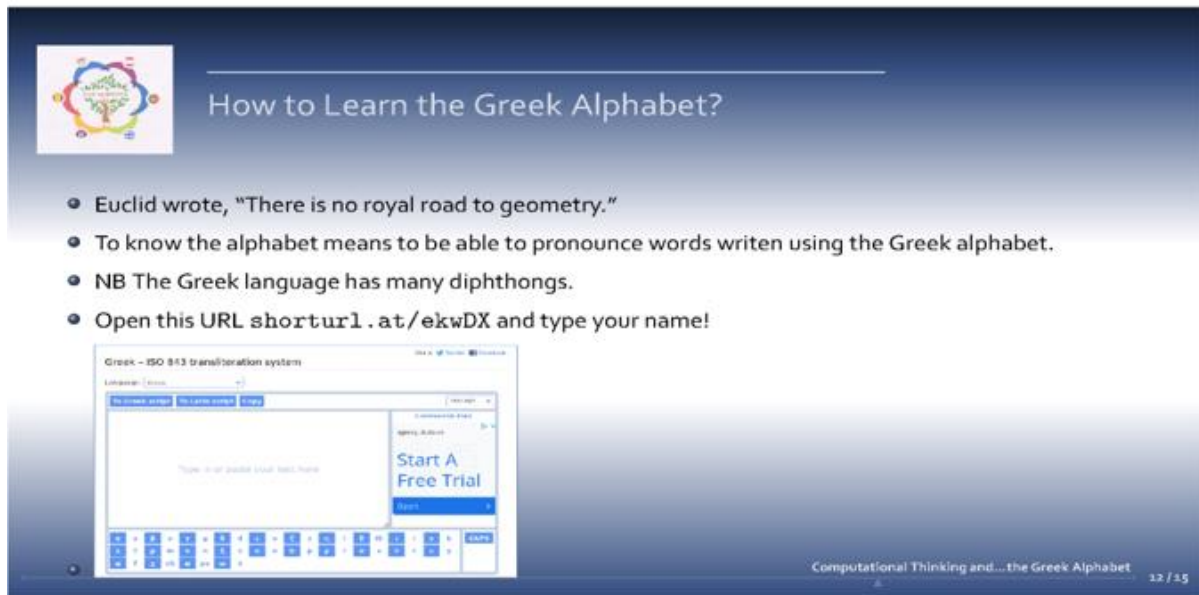
The essence of learning in the 21st century



The next important task was to make the audience familiar with the Greek alphabet. A good idea was to introduce them to the alphabet through a mapping. The Greek state uses a standard transliteration to “translate” foreign names into Greek and Greek names into English. Some sites implement this transliteration and allow users to write their names using Greek letters. Thus, we asked students and teachers to write their names in their language. Most participants used the Latin alphabet to write their names, but the Serbians used a version of the Cyrillic alphabet, nevertheless, they used the English transliteration that was printed on their passports. Figure 2 shows the complete slide of the presentation that was designed to introduce people to the Greek alphabet. In addition, we briefly presented some diphthongs and the tonos, an accent that indicates stress. Here we employed an algorithm that can teach learners the sounds of the Greek letters. One needs to go from familiar things to, let us say, uncharted territories.

Figure 2

How to get to know the Greek alphabet.



How to Learn the Greek Alphabet?

- Euclid wrote, "There is no royal road to geometry."
- To know the alphabet means to be able to pronounce words written using the Greek alphabet.
- NB The Greek language has many diphthongs.
- Open this URL `shorturl.at/ekwDX` and type your name!

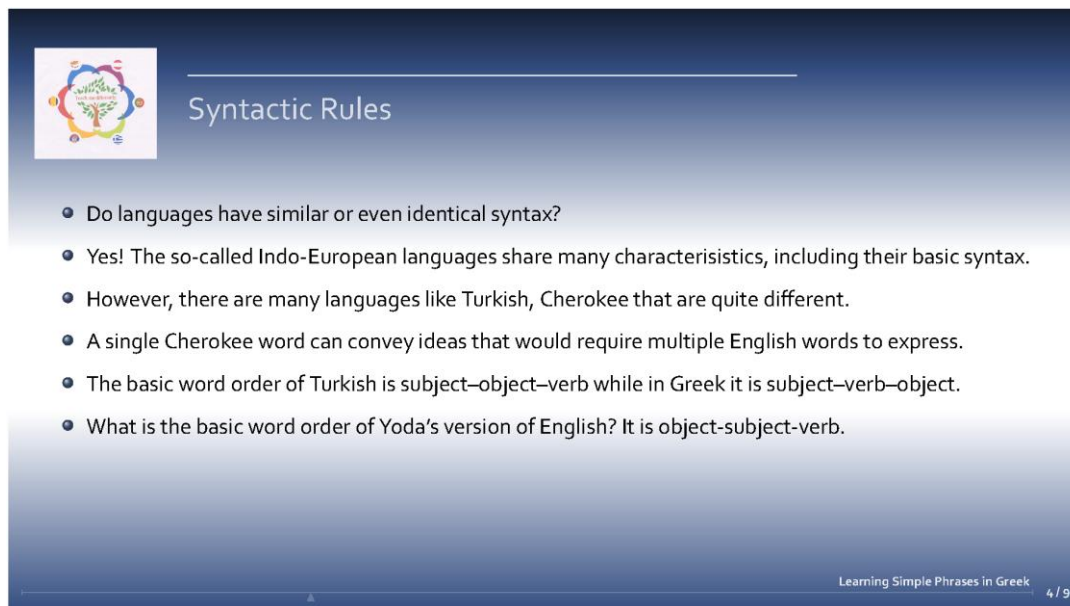
The screenshot shows a web interface for a "Greek - ISO 843 transliteration system". It features a text input field with the placeholder "Type in a word (or words) from here", a "Start A Free Trial" button, and a keyboard layout at the bottom. The interface is titled "Greek - ISO 843 transliteration system" and includes a "Language" dropdown menu.

Computational Thinking and...the Greek Alphabet 12 / 15

On the second day, we introduced our audience to the meaning of syntactic rules. We wanted to give them a brief overview and then allow them to create simple sentences in Greek. In Figure 3, we show the basic ideas regarding syntactic rules that we wanted to teach to our audience whereas, in Figure 4, we show the basic ideas regarding syntax that we thought our audience needed to know.

Figure 3

An overview of syntax and syntactic rules



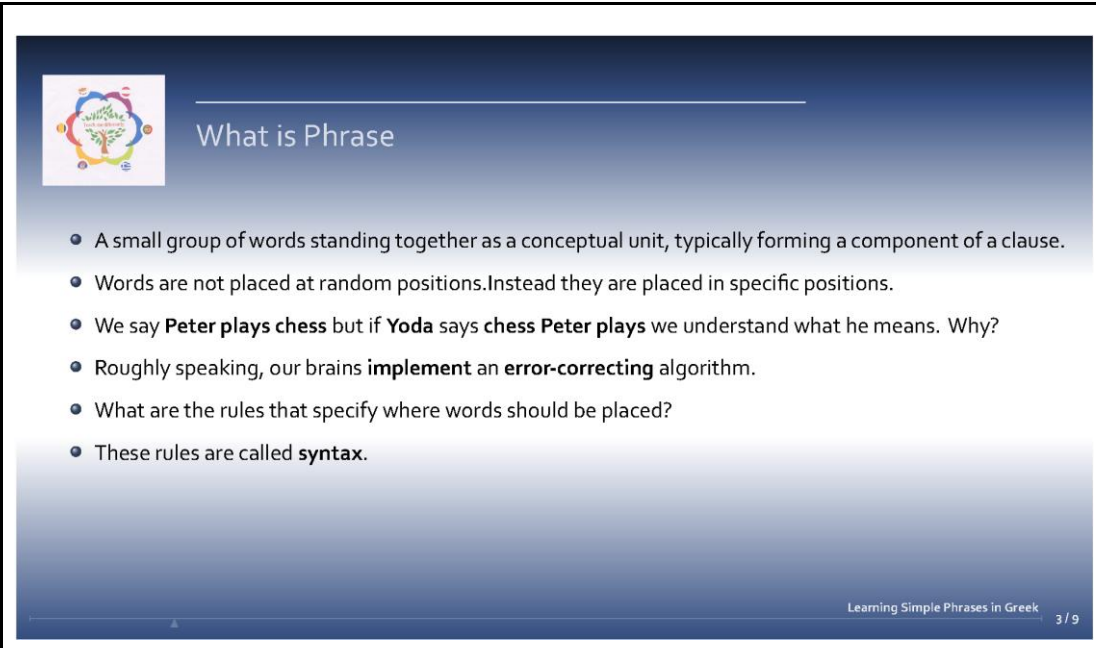
Syntactic Rules

- Do languages have similar or even identical syntax?
- Yes! The so-called Indo-European languages share many characteristics, including their basic syntax.
- However, there are many languages like Turkish, Cherokee that are quite different.
- A single Cherokee word can convey ideas that would require multiple English words to express.
- The basic word order of Turkish is subject-object-verb while in Greek it is subject-verb-object.
- What is the basic word order of Yoda's version of English? It is object-subject-verb.

Learning Simple Phrases in Greek 4 / 9

Figure 4

A brief overview of what is syntax.



What is Phrase

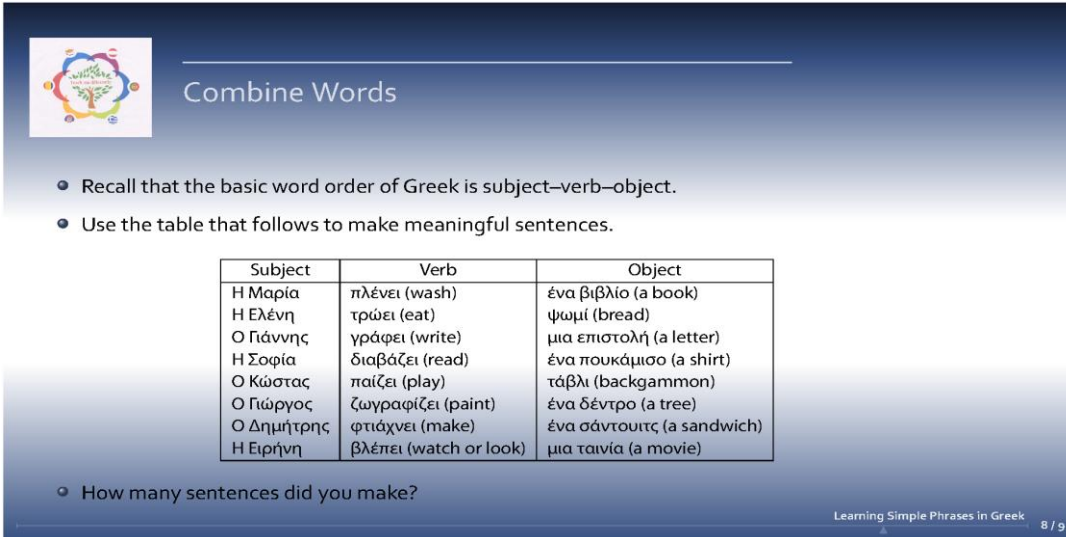
- A small group of words standing together as a conceptual unit, typically forming a component of a clause.
- Words are not placed at random positions. Instead they are placed in specific positions.
- We say **Peter plays chess** but if **Yoda says chess Peter plays** we understand what he means. Why?
- Roughly speaking, our brains **implement** an **error-correcting** algorithm.
- What are the rules that specify where words should be placed?
- These rules are called **syntax**.

Learning Simple Phrases in Greek 3 / 9

Figure 5 shows the simple exercise that both students and teachers had to do. The exercise is about the formation of simple sentences using the word order of Greek, where one can choose from a set of objects, sussubjects, and verbs.

Figure 5

An exercise on sentence formation.



Combine Words

- Recall that the basic word order of Greek is subject–verb–object.
- Use the table that follows to make meaningful sentences.

Subject	Verb	Object
Η Μαρία	πλένει (wash)	ένα βιβλίο (a book)
Η Ελένη	τρώει (eat)	ψωμί (bread)
Ο Πάννης	γράφει (write)	μια επιστολή (a letter)
Η Σοφία	διαβάζει (read)	ένα πουκάμισο (a shirt)
Ο Κώστας	παίζει (play)	τάβλι (backgammon)
Ο Γιώργος	ζωγραφίζει (paint)	ένα δέντρο (a tree)
Ο Δημήτρης	φτιάχνει (make)	ένα σάντουιτς (a sandwich)
Η Ειρήνη	βλέπει (watch or look)	μια ταινία (a movie)

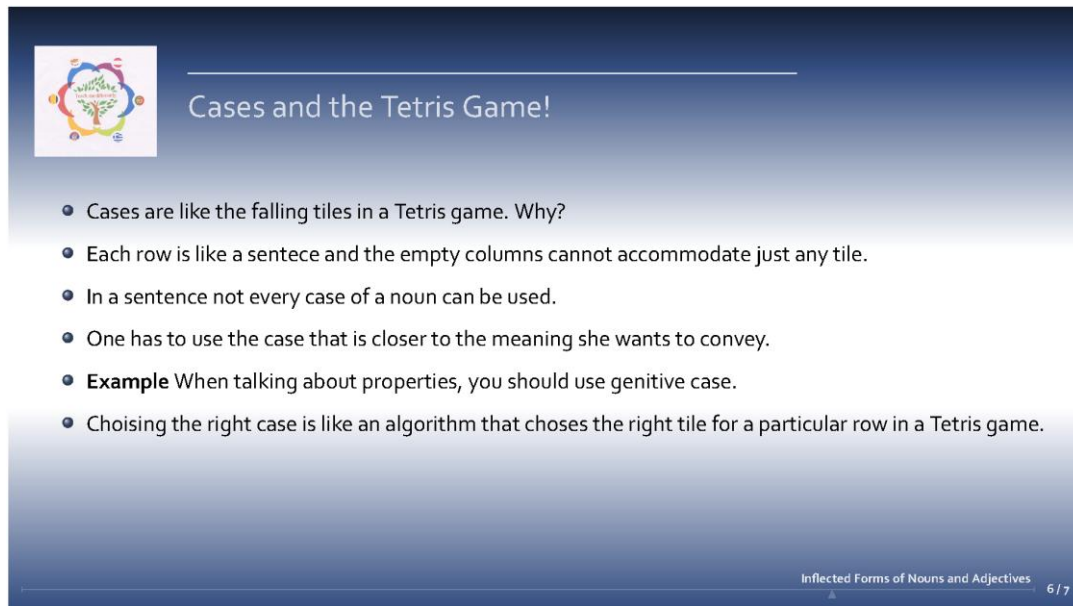
- How many sentences did you make?

Learning Simple Phrases in Greek 8 / 9

The next subject we tried to introduce to our audience was the notion of inflected forms of nouns and adjectives. In the Greek language each noun has eight different forms and an adjective has twenty-four different forms. The reason is there are four cases in the singular form and four in the plural form. In addition, there are three grammatical genders and each of them has four cases for the singular and four cases for the plural form. We have managed to find an association between the Tetris game and the procedure to choose the right case of a noun or an adjective. Figure 6 explains the association. In addition, we introduced our audience to the meaning of the cases in the Greek language and tried to make sure they understood them by asking them to do an exercise. The exercise is shown in Figure 7.

Figure 6

An association between Tetris and cases.



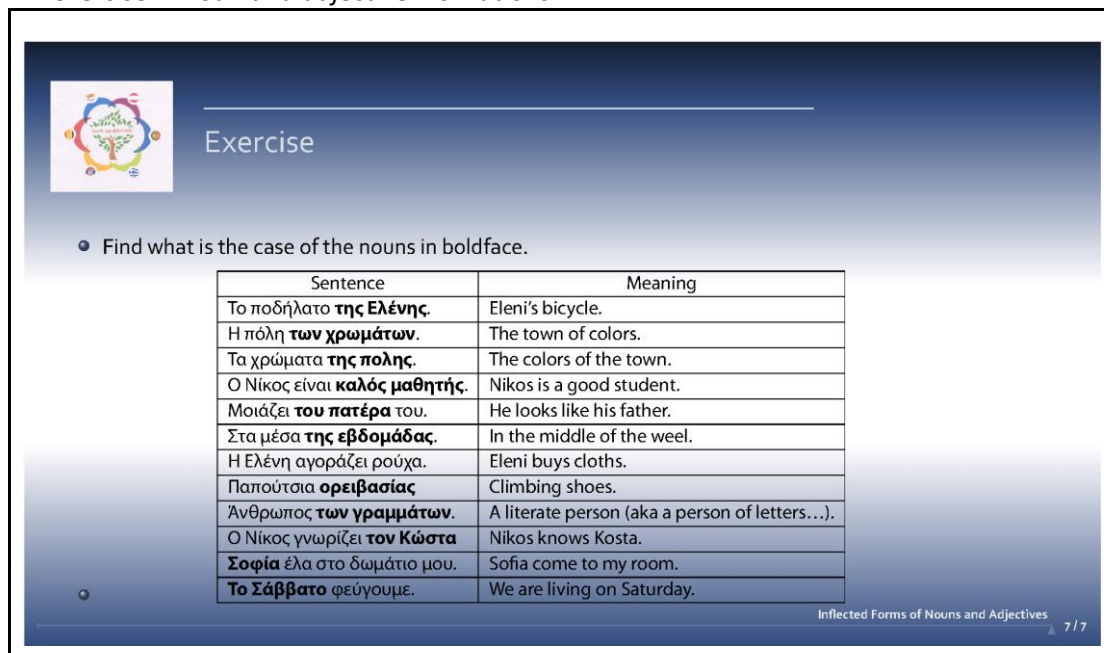
Cases and the Tetris Game!

- Cases are like the falling tiles in a Tetris game. Why?
- Each row is like a sentence and the empty columns cannot accommodate just any tile.
- In a sentence not every case of a noun can be used.
- One has to use the case that is closer to the meaning she wants to convey.
- **Example** When talking about properties, you should use genitive case.
- Choosing the right case is like an algorithm that chooses the right tile for a particular row in a Tetris game.

Inflected Forms of Nouns and Adjectives 6/7

Figure 7

An exercise in noun and adjective inclinations.



Exercise

- Find what is the case of the nouns in boldface.

Sentence	Meaning
Το ποδήλατο της Ελένης .	Eleni's bicycle.
Η πόλη των χρωμάτων .	The town of colors.
Τα χρώματα της πόλης .	The colors of the town.
Ο Νίκος είναι καλός μαθητής .	Nikos is a good student.
Μοιάζει του πατέρα του.	He looks like his father.
Στα μέσα της εβδομάδας .	In the middle of the week.
Η Ελένη αγοράζει ρούχα.	Eleni buys cloths.
Παπούτσια ορειβασίας .	Climbing shoes.
Άνθρωπος των γραμμάτων .	A literate person (aka a person of letters...).
Ο Νίκος γνωρίζει τον Κώστα .	Nikos knows Kosta.
Σοφία έλα στο δωμάτιο μου.	Sofia come to my room.
Το Σάββατο φεύγουμε.	We are living on Saturday.

Inflected Forms of Nouns and Adjectives 7/7

Once the audience had a “working” understanding of elements of the Greek language, we asked them to answer a questionnaire whose purpose was the evaluation of the mini course.

3.1. Evaluating the Mini Course

To evaluate the mini-course, we designed a questionnaire (see Appendix) and asked all participants to answer it. The questions are quite simple as we wanted to be sure that the participants understood the basic things they were introduced to. In general, most participants had no problem

correctly answering most questions. We have noticed that they did not understand the notion of word order as most answers were wrong. What was interesting was the answers to the last question. The time constraints did not allow us to elaborate on many things. So, students expected something people learn in normal language classes (e.g., useful, daily phrases) and not about inflection or word order. Of course, we wanted to experiment and so we had to make some rather “strange” but completely justified choices. All in all, we think this was a somehow successful experiment.

4. Conclusions

Teaching students a subject with a new methodology is not an easy task and certainly not a straightforward task. One needs to carefully see how the new methodology can be used in the classroom. Ideally, one should do some experiments where students will be exposed to the new methodology, and from this interaction, the teacher must draw her conclusions.

The researchers have performed such an experiment on a small scale but we understood that it works. Students got a basic understanding of computational thinking and they managed to use the new tools that we introduced to them. We are convinced that computational thinking can be fully used in foreign language learning classes and we are sure the outcome will be quite beneficial to both teachers and students.

Acknowledgment

The work described in this paper has been co-funded by the Erasmus+ Program of the European Union.

References

- Alobaid, A. (2020). Smart multimedia learning of ICT: role and impact on language learners' writing fluency—YouTube online English learning resources as an example. *Smart Learning Environments*, 7(1), 24. <https://link.springer.com/article/10.1186/s40561-020-00134-7>
- Amukune, S., Caplovitz Barrett, K., & Józsa, K. (2022). Game-Based Assessment of School Readiness Domains of 3-8-year-old-children: A Scoping Review. *Journal of New Approaches in Educational Research*, 11(1), 146-167. <https://eric.ed.gov/?id=EJ1325543>
- Bakan, U., Han, T., & Bakan, U. (2022). Learner Perceptions and Effectiveness of Using a Massively Multiplayer Online Role-Playing Game to Improve EFL Communicative Competence. *Knowledge Management & E-Learning*, 14(3), 286-303. <https://eric.ed.gov/?id=EJ1364887>
- Brown, A. V. (2009). Students' and teachers' perceptions of effective foreign language teaching: A comparison of ideals. *The Modern Language Journal*, 93(1), 46-60. <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-4781.2009.00827.x>
- Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, 1(2), 67-69. <https://core.ac.uk/download/pdf/28961399.pdf>
- Carpenter, B. (1998). *Type-logical semantics*. MIT Press. [https://books.google.com/books?hl=en&lr=&id=-GwbT2HMI2wC&oi=fnd&pg=PR13&dq=Carpenter,+B.+\(1997\).+Type-Logical+Semantics.+The+MIT+Press.&ots=S8qwbyfk7C&sig=O3_P9WrJ6Y4sEGho7ppfobZioPW](https://books.google.com/books?hl=en&lr=&id=-GwbT2HMI2wC&oi=fnd&pg=PR13&dq=Carpenter,+B.+(1997).+Type-Logical+Semantics.+The+MIT+Press.&ots=S8qwbyfk7C&sig=O3_P9WrJ6Y4sEGho7ppfobZioPW)
- Chomsky, N. (1959). On certain formal properties of grammars. *Information and control*, 2(2), 137-167. <https://www.sciencedirect.com/science/article/pii/S0019995859903626>
- Hellmich, E. A., & Vinall, K. (2023). Student use and instructor beliefs: Machine translation in language education. *Language Learning & Technology*, 27(1), 1–27. <https://hdl.handle.net/10125/73525>
- Huntington, B., Goulding, J., & Pitchford, N. J. (2023). Expert perspectives on how educational technology may support autonomous learning for remote out-of-school children in low-income contexts. *International Journal of Educational Research Open*, 5, 100263. <https://www.sciencedirect.com/science/article/pii/S2666374023000389>

- Syropoulos, A., Tatsiou, E. & Wachter, R. (2023). Using computational thinking to teach foreign languages: A preliminary approach. *Global Journal of Foreign Language Teaching*, 13(4), 223-235. <https://doi.org/10.18844/gjflt.v13i4.8954>
- Jacob, S., Nguyen, H., Tofel-Grehl, C., Richardson, D., & Warschauer, M. (2018). Teaching computational thinking to English learners. *NYS TESOL Journal*, 5(2). <https://par.nsf.gov/servlets/purl/10073683>
- Papert, S. (1996). An exploration in the space of mathematics educations. *Int. J. Comput. Math. Learn.*, 1(1), 95-123. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=d774d954adcca78e0c4>
- Papert, S. A. (2020). *Mindstorms: Children, computers, and powerful ideas*. Basic books.
- Pollak, M., & Ebner, M. (2019). The missing link to computational thinking. *Future Internet*, 11(12), 263. <https://www.mdpi.com/1999-5903/11/12/263>
- Ranta, A. (1994). *Type-theoretical Grammar*, Clarendon.
- Syropoulos, A. (2008). *Hypercomputation: computing beyond the Church-Turing barrier*. Springer Science & Business Media.
- Syropoulos, A., & Stavrianos, A. (2014). Using Scripting Languages to Teach Programming. *arXiv preprint arXiv:1404.5870*. <https://arxiv.org/abs/1404.5870>
- Togaibayeva, A., Ramazanova, D., Yessengulova, M., Yergazina, A., Nurlin, A., & Shokanov, R. (2022). Effect of mobile learning on students' satisfaction, perceived usefulness, and academic performance when learning a foreign language. In *Frontiers in Education* 7, p. 946102. <https://www.frontiersin.org/articles/10.3389/feduc.2022.946102/full>
- Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and information technologies*, 20, 715-728. <https://link.springer.com/article/10.1007/s10639-015-9412-6>
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725. <https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2008.0118>
- Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 1-16. <https://dl.acm.org/doi/abs/10.1145/2576872>
- Zhai, C. (2023). A systematic review on artificial intelligence dialogue systems for enhancing English as foreign language students' interactional competence in the university. *Computers and Education: Artificial Intelligence*, 100134. <https://www.sciencedirect.com/science/article/pii/S2666920X23000139>

APPENDIX



Co-funded by the
Erasmus+ Programme
of the European Union

May 20, 2022

Questionnaire for the participants of the Erasmus+ *Teach Me Differently* Training Event for Pupils

Host Organization: 2nd Gymnasium of Xanthi, Greece.
Xanthi, Greece from May 16 till May 20, 2022

1. What is Computational Thinking?
 - A programming methodology.
 - A way to advance cooking skills.
 - A teaching methodology.
 - A method to advance one's typing skills.
2. Who "invented" Computational Thinking?
 - Seymour Papert.
 - Jeannette Marie Wing.
 - Ayrton Senna.
 - Stephen Glenn Martin.
3. What one cannot do with Computational Thinking?
 - Solve problems.
 - Explore new worlds.
 - Design systems.
 - Understand the human behavior.
4. What is not important for Computational Thinking?
 - Knowing how to generalize from specific instances.
 - Be able to understand that certain situations may have multiple outcomes.
 - Be able to build something large by putting together collections of smaller parts.

<p><input type="radio"/> Understanding how and why something has to be performed an infinite number of times.</p> <p>5. How many letters does the Greek Alphabet have?</p> <p><input type="radio"/> 33.</p> <p><input type="radio"/> 24.</p> <p><input type="radio"/> 26.</p> <p><input type="radio"/> 28.</p> <p>6. What is Syntax?</p> <p><input type="radio"/> A strange Greek word.</p> <p><input type="radio"/> The structure of a sentence.</p> <p><input type="radio"/> Rules that specify where words should be placed.</p> <p><input type="radio"/> All the above.</p> <p>7. The basic word order of Greek is</p> <p><input type="radio"/> verb-subject-object.</p> <p><input type="radio"/> subject-verb-object.</p> <p><input type="radio"/> object-verb-subject.</p> <p><input type="radio"/> subject-object-verb.</p> <p>8. What is inflection?</p> <p><input type="radio"/> An action of a muscle.</p> <p><input type="radio"/> A grammatical phenomenon.</p> <p><input type="radio"/> A change to the form of a word that shows a change in the way it is used in sentences.</p> <p><input type="radio"/> A way to specify that something will happen in the future.</p> <p>9. How many cases does the Modern Greek language have?</p> <p><input type="radio"/> 12</p> <p><input type="radio"/> 6</p> <p><input type="radio"/> 5</p> <p><input type="radio"/> 4</p> <p>10. Write down with a few words what you think about this training event.</p>
