Harnessing Auto-Generative Learning Objects in Serious Games

Ciprian-Bogdan CHIRILA

Computer Science and Software Engineering Department, Automation and Computers Faculty, University Politehnica Timişoara, Timişoara, Romania

chirila[at]cs.upt.ro

Abstract: *The IT industry has a shortage problem regarding human resources. High economical projects are rejected due to the human resources shortage. Such practices affect the internal revenue and thus the national economy. Universities developed several solutions in this sense for a faster and better training of students. They developed distance learning programs, MOOCs, etc. The use of auto-generation learning objects (AGLOS) represents a solution in this sense since it has the potential to deliver online variable and dynamic e-learning content based on function compositions and random numbers. A gamification of such objects would motivate and enable students to improve their competences in the field of ITC.*

Keywords: e-learning, auto-generative learning objects, gamification, serious games, computer science disciplines, mobile applications.

1. Introduction

The migration of human resources from the Eastern Europe to the Western European countries puts a constant pressure on the local IT industry. By human resources we mean: software architects, software developers, embedded system developers, web developers, GNOC (global network operations centre) engineers, DevOps engineers, etc. Specifically, in the region around the city of Timişoara periodically, the number of IT specialists is diminished significantly, so the need for specialists becomes more severe affecting the region's development.

In this context, each year software companies reject high value projects because of human resources shortage. Other software companies which prospect the region are reluctant to invest in new subsidiaries to create new jobs for the citizens. In order to tackle this problem universities must train software engineering specialists better, faster and deliver them to the industry to increase regional

https://doi.org/10.58503/icvl-v17y202223

companies' capability of accepting and developing more and larger software projects.

Companies employ students very early, sometimes in the second or third year of study in part time or full-time programs so their study time is quite limited. In this context arise the need for new learning ecosystems that are adapted to this issue where students are motivated more, can learn and exercise several learning objectives efficiently in their limited free time like: lunch breaks, while on / waiting for public transportation, while waiting in queues (e.g., in a bank to meet a teller employee) etc. By learning ecosystems, we understand a combination of technologies and resources that help individuals to learn in an environment.

On the other hand, students tend to be present more and more in online activities, they often use gadgets for information, entertainment (social networks) and also for learning by email, blogging, microblogging, wikis, polls, surveys, collaborative writing tools etc. Universities tend to cope with the increasing IT specialists demand coming from regional industry and benefit from the digitally enabled nature of students so they built several learning environments like: LMSs, MOOCs etc. The learning content offered to students in this environment is mostly static – the content of a learning object does not change.

In this context we propose a new learning ecosystem based on the composition of two concepts: AGLO (Chirila et al., 2015) and gamification (Huotary et al., 2012). Conceptually, we can define the gamified AGLO like:

GamifiedAGLO = AGLO ○ Game

The AGLO is a meta-model for generating dynamic e-learning content that enables several advantages over static learning content. An AGLO has embedded the functionality of a specific learning objective and with the help of the computer one can generate several examples, tests for better understanding and learning assessment. AGLOs are meant to be designed by tutors and to be used independently by students in online platforms.

Using AGLOs the student can benefit automatically from a virtually infinite number of examples when studying a learning objective. In this sense an ecosystem based on AGLOs could be considered a smart learning ecosystem. For example, in the context of IT disciplines when the structure of an array is studied using an AGLO the student can work on virtually an infinite number of memory mappings for that array, variable parameters are element size and array starting address: i) array of integers or floats where the array element has 4 bytes; ii) array of short integers where the array element has 2 bytes; iii) array of doubles where the array element has 8 bytes; iv) array of structures where the element can have an arbitrary size. In the regional automotive industry developers use untypical data types represented on 4, 12, 20 bytes; v) the starting address of the array in the memory is also random; vi) the computation of the array elements addresses can be learned in such an AGLO created context.

Gamification refers to using game design elements and principles (game mechanics) in non-game contexts in order to improve user engagement (Huotari et

al., 2012). Several review researchers find that gamification brings positive effects for their users. We consider that in order to increase even more the attractiveness of AGLOs we can use several gamification techniques. Specifically, to our region, the target users for the gamified AGLOs are as follows. IT students in the first year of study may play gamified AGLOs because of their young age. On the other hand, the biggest dropout rate is found at the end of the first year 50-60% so we consider that such measures have some potential in at least limiting the size of this problem.

Another segment of the target users are the students and graduates from non-IT faculties that easily can be trained into IT because of their technical backgrounds and of their digital user experiences. In our region such students easily become software developers mostly in embedded systems of the automotive industry.

A different segment of potential target users are students from non-technical universities that need basic programming skills. The regional industry searches for such students and even started to develop informal schools for professional conversion. For example, in the regional automotive industry such students / graduates are employed as software testers.

A younger segment of target users are middle school and high school students who intend to learn the first steps in programming and want to follow a career in the IT field and to act in the regional industry.

The paper is structured as follows. In section 2 we analyse related works. In section 3 we present a short definition of AGLOs. Section 4 presents an AGLO example at work. Section 5 analyses several gamification ideas in the area of searching and sorting algorithms, while section 6 deals with gamification of trees and graphs algorithms. Section 7 describes the implementation of the current prototype. Section 8 reflects on the strengths and weaknesses of the approach. Section 9 concludes and sets future work.

2. Related Works

Generally, learning objects (LO) are considered deliverable learning content and have been under several standardisation processes (IEEE LTS, 2016). AGLOs may be considered as specialisations of GLOs which are used in several learning contexts. GLOs are pedagogical patterns that can be reused in several contexts (Boyle, 2003; Boyle, 2006; Jones at al., 2007) through instantiation. They are considered to be the second generation of learning objects (Boyle, 2006). The instantiation can be implemented by metaprogramming or other methods like manual content fulfilment. In (Boyle & Bradley, 2009) is presented a GLO maker tool that assists the creation and instantiation of GLOs using a graphical user interface.

(Chirila, 2013; Chirila, 2014) present an AGLO model used in primary and secondary schools e learning platforms. The model is based on a state machine, random values, interpreters etc. in order to generate a variable dialog where competences are developed in the context of a competence and skills system build.

(Chirila, 2014b; Chirila, 2015; Costea et al., 2018) present generative models of learning objects for computer science disciplines like data structures, algorithms and operating systems.

In (Damasevicius et al., 2008; Stuikys et al., 2013) are presented elaborated models (feature models and others) that generate GLOs operating LEGO robots in order to teach computer programming in a very intuitive manner.

In (Shorn, 2018) is presented an experiment about teaching computer programming based on team-based competitive games and individual competitive activities. The learning gains were positive but of small effect size and lack of statistical significance compared to the traditional approach.

In (Zhan et al., 2022) is described a meta-analysis on gamification in programming compiled from 21 empirical studies using cross-tabulation analysis. Several aspects were analysed like: students cognitive load, game types, reasoning strategies, gamification applications, teaching tools, pedagogical agents and programming types.

In (Mubin et al., 2020) is presented a gamification literature review in programming language learning. The study reveals that gamified solutions for website development programming languages should be developed.

ADL (ADL xAPI, 2016) researches infrastructures, environments and tools in the domain of serious games based on virtual reality and simulations. They contributed to web distributed games that train policy and procedure in the defence area. Other approaches developed by ADL in this sense are: i) game interactions to teaching environments; ii) maths and programming in a story line; iii) Virtual World Sandbox for simulations delivered and stored online.

3. The Basic Concepts of Auto-Generative Learning Objects

The AGLO model is based essentially on function composition and random numbers generation being able to deliver generated e-learning content embedded with interactivity and feedback. The development of free gaming frameworks running also on mobile phones, like Unreal Engine 4, Unity, etc. enables the creation of gamified AGLOs which can enhance even more the attractiveness of the e learning materials.

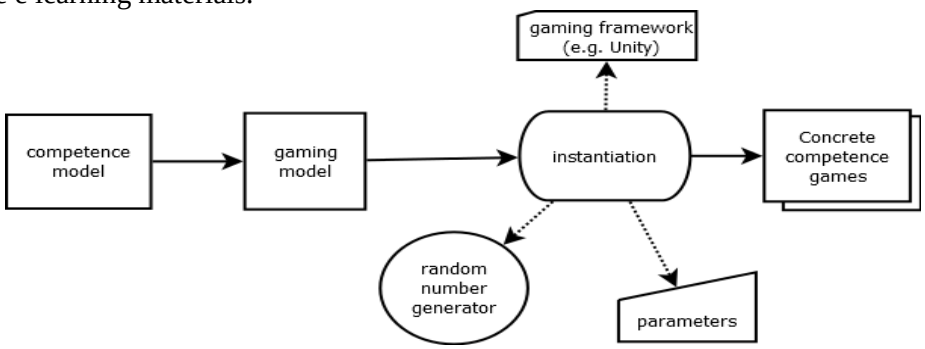

Figure 1. The AGLO Gamification Approach

In this paper we explore the first steps of creating AGLO models based on gamifications in order to increase the attractiveness of learning objects. Figure 1 presents the framework in which AGLO gamification will work. We start from a competence model which can be an informal description of the competences we want to achieve. Next, we design a generic gamification model which is instantiated based on random numbers, local parameters and animation facilities from different frameworks. Finally, after the instantiation we obtain a concrete playable gamification. Re-instantiating the model will produce different concrete gamification. The AGLO model (Chirila, 2015) is based on several sections like: i) name – the identification section, small descriptions, and other metadata can be set; ii) scenario – the symbols definition section using formulas and random numbers; iii) theory – a section for explaining theoretical fundamentals necessary to be accessed in order to perform actions in the other sections; iv) questions – a section where the questions are formulated based in symbols; v) answers – a sections where answers are assessed relating on precomputed symbols; vi) feedbacks – a section for explanations related to the previously asked questions and based on the computed symbols. The number of sections is not limited to the previously listed ones, only the symbols definition section is mandatory and other sections may or not depend on the defined symbols.

In this paper we want to add the advantages of gamification like status and achievements to the AGLO model in order to better motivate students to achieve their learning goals. The challenges of our approach are to make first steps towards the design of the gamification generic model in order to be able to apply it to different learning objectives from different disciplines. The first step of our research is to identify the gamification principles in concrete examples, namely searching, sorting and graph related algorithms.

4. AGLO Based Serious Games for Arrays

Revisiting the array memory mapping example from the first section we can enter in the anatomy of that AGLO where its functionality can be explained into details. Firstly, in the scenario section we define the model of the learning objective. In this AGLO the proposed learning objective is to learn the memory mapping of an array and to exercise the computation of the array's elements addresses. We imagine our model following the next strategy: i) to generate randomly a base memory address denoted by b; ii) to convert the memory address into hexadecimal representation sb; iii) to create an array of array elements sizes, for this example we propose to have [1,2,3,4,8,16] ; iv) to randomly select one on the sizes from the array using an index named id; v) to get the value from the sizes array d=tab[id]; vi) to generated a random index i; vii) to compute the result address of the array element r=b+i*d; viii) to convert the result address into hexadecimal representation sr. All these parameters will be defined as symbols in the context of an AGLO. The formalisms that implement the previously explained steps are:

```
<symbol name="b" type="integer">random(100,200,0); </symbol>
<symbol name="sb" type="string">(v("b")).toString(16); </symbol>
<symbol name="tab" type="array">[1,2,3,4,8,16]; </symbol>
<symbol name="id" type="integer">random(0,5,0); </symbol>
<symbol name="d" type="integer">v("tab")[v("id")]; </symbol>
<symbol name="i" type="integer">random(0,20,0);</symbol>
<symbol name="r" type="integer">v("b")+v("i")*v("d"); </symbol>
<symbol name="sr" type="string">(v("r")).toString(16); </symbol>
```

The implementation is a prototype in JavaScript with a few wrapper functions used to implement a simple running environment. The random(…) function is based on the Math.random(…) library function. The v(…) function is used to access the values of previously defined symbols in the dynamic of symbols definition. For example, symbol d uses previously set symbols tab and id, etc. The dynamic content shown the student is based in the following text pattern:

Given the memory address of the beginning of an array <value name="sb"/>h, knowing that the element size is of <value name="d"/> bytes, compute the element address at index <value name="i"/> from the array.

The generative content of the AGLO uses only 3 symbols sb, d and i. The correct answer was computed and can be compared to the student answer so points, medals, cups or other gaming mechanics can be awarded to the student.

Next, we show three different instantiations of the model:

#1

Given the memory address of the beginning of an array 10h, knowing that the element size is of 4 bytes, compute the element address at index 3 from the array.

#2

Given the memory address of the beginning of an array 22h, knowing that the element size is of 8 bytes, compute the element address at index 7 from the array.

#3

Given the memory address of the beginning of an array ACh, knowing that the element size is of 16 bytes, compute the element address at index 9 from the array.

5. AGLO Based Serious Games for Searching and Sorting Algorithms

In this section we will present a few ideas and principles for the gamification of basic computer science algorithms like searching and sorting. We consider that linear search and sentinel based linear search algorithms are quite trivial to gamify so we focused on other algorithms like simple sorting algorithms. In order to achieve the gamification effect, we will include the following dimensions: i) points – to model a score in the game economy, achieving correct algorithm steps will

increase the score, while wrong actions will decrease the score; ii) badges – each passed algorithm gamification will be rewarded with a specific badge, they are be implemented as parameterized vector graphic images; iii) top performers board – this facility will be used for the promotion of competition between students, such boards could be created locally at school or university level or at national and international levels after the gamification are internationalised; iv) levels – the same gamification can be parameterized with the size of the problem, e.g., a sorting algorithm can be instantiated for 7, 15 and 25 elements representing different difficulty levels; v) unlocking higher levels – gamification of algorithms may have prerequisite relations between them, e.g., in order to access interpolation search algorithm gamification you must first fulfil the binary search algorithm gamification.

5.1. AGLO Serious Games for Binary Search Algorithm

For the binary search algorithm, we will consider for each step a set of actions to be executed by the player. Firstly, he needs to point the left and right boundaries of the array where the search is done. This can be implemented by dragging and dropping a set of parentheses like "[" and "]" or other computer graphics sprite decorations at the right location. Secondly, he needs to points to the pivot element. This step can be implemented by dragging and dropping the "m" letter, or other sprite over the median element. Variations can be imagined at this step by allowing the player to set the pivot at his own will. Allowing the median to be at an index different than the median will possibly trigger more or less algorithm steps in the search. Thirdly, the player has to select the sub-array where the search should continue. This can be implemented by highlighting differently the two zones, to be able to click on one of them. Next, the player has to adjust the left boundary or the right boundary by moving the left or right parenthesis. Finally, he has to repeat the previous steps until the search is over. The total number of points are computed by adding each action point. The rewarding badge as a certified binary searcher may look like a military distinction with an inscription of the name of the algorithm inside. The top performers' board will list the fastest players (total no of steps completed / total time). The levels can be achieved by setting the array length to 7, 15 and 25. The unlocking of higher levels after completing the binary search may involve revealing the gamification of interpolation search and quick sorting algorithm.

5.2. AGLO Based Serious Games for Sorting by Insertion Algorithm

The philosophy of the algorithm is to split the array into two parts sorted and unsorted and at each step to increase the size of the sorted part thus diminishing the size of the unsorted one. This philosophy can be visually implemented by using two different background colours for the two array zones. Firstly, the player will see the initial array with one element already sorted, belonging to the growing

zone. Secondly, the player is invited to extract the first element from the unsorted zone.

Thirdly, the player will have to repeatedly move the greater elements one position to the right using the drag and drop technique. Optionally, the player could select all elements that have to be transposed and transpose them as a whole block. Fourthly, the player is invited to insert the extracted element at the right location between the other two elements. The steps are repeated until the array is sorted. The points are obtained for each element move. The rewarding badge for the sorting by insertion algorithm will be a decoration with one symbolic element to be inserted between the other two. On the top performers board will be the players which will make the largest number of correct moves. Levels can be created varying the number of elements in the array. The unlocking of higher levels may be related to the binary insertion sorting algorithm or any algorithms based on insertions.

6. AGLO Based Serious Games for Trees and Graph Data Structures

6.1. AGLO Based Serious Games for the Binary Tree Creation Algorithm

The gamification of the binary tree construction starts with displaying a random list of integers. Firstly, the player must set its root by dragging the first element from the list to the binary tree construction area. Secondly, the player must compare the next element in the list with the root, this can be implemented by positioning it on the left side or right side near the root. Thirdly, the player must attach the new element to the root on the left or on the right depending on the case. The points are obtained for each correct comparison and node placement. The granted badges will have miniature binary trees on them. The top performers' board will hold the names of the fastest and most accurate players of this gamification. The levels will be generated according to the numbers of nodes like: 7, 15 and 20. Regarding the unlocking of higher levels, we can trigger the release of the binary tree searching algorithm gamification. Similarly, a binary tree searching algorithm gamification can be imagined.

6.2. AGLO Based Serious Games for the Kruskal's Algorithm

In the preparation stage a weighted graph will be generated based on control parameters like: i) no of nodes; ii) number of edges; iii) degree; iv) minimum and maximum weight values. In the first stage of the gamified algorithm, we have a drag and drop episode where the player will sort the graph edges. The edges list will have the possibility of accepting other edges between two already existing ones. In the second stage the player will select and rebuild a greyed shape of the graph with the minimal edges avoiding cycles. The points are obtained for each action in the process of ordering and in the process of filling the graph without making cycles. The badge gained will depict the name of the author, namely Kruskal. The top performer's board will contain the fastest players playing with

graphs having a great number of edges. The levels can be obtained by changing the parameter values with 5, 10 and 15 for the number of edges in order to increase difficulty. Regarding unlocking higher levels, we can unlock other minimum spanning tree algorithms like Prim's algorithm for example or other complex algorithms.

7. Web Application Implementation

The implementation is based on a rapid prototyping of a client web-based application. The code is written in JavaScript and runs on the client side, namely in browsers. The used libraries are JQuery and its drag and drop extension. Each algorithm implementation is based on a state machine with the unrolling of the execution steps. When the player acts according to the correct step then the number of points is increased. The common elements of the prototypes are the fact that all work with arrays, all need: i) element selection facilities; ii) inserting facilities and iii) elements swapping facilities (distant or neighbour). Dynamically, each phase is played in one virtual line and after its completion a new line is created so the player can see the progress of its actions and eventual error not to be repeated in the near future. The first two lines the player is assisted with indication and afterwards it is left to play alone. If errors occur then the assistance is set back on in order to help him understand the behaviour of the studied algorithm. When assisted the player will get no points. With xAPI each action can be stored in a standardised format for later analysis or publication. We consider that only the full completion of an algorithm gamification should be stored in the database and not each phase's result.

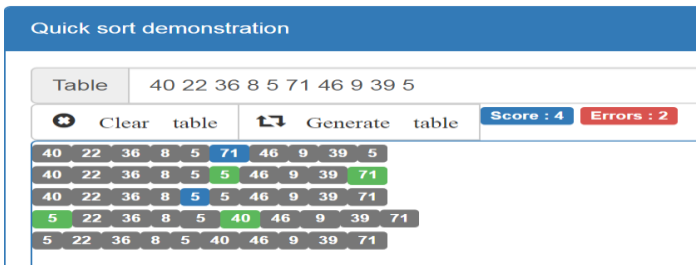

Figure 2. Gamification of Quicksort Algorithm

In Figure 2 we show a snapshot picture of the prototype where the student plays the gamification of the quick sorting algorithm. Using drag and drop, clicking, he will have: i) to select the first pivot; ii) to swap the pairs of elements around the first pivot; iii) to select the left pivot; iv) to swap the pairs of elements around the left pivot; v) to select the right pivot; vi) to swap the pairs of elements around the right pivot; vii) etc. For each correct move the score is increased and for each bad move the number of errors will grow. In this example the AGLO instance provided the learning context which is the input integer array. In this case it may seem trivial to generate a random integer array. But, for example, in the context of graphs the graph generation models from AGLOs are not trivial. The generation of

a random graph involves several iterations for generating the nodes, the edges, the weights etc., which are features offered by AGLOs.

8. Conclusions and Future Work

In this paper we presented ideas towards the gamification of AGLOs dedicated to the learning of computer science basic algorithms. We imagined and experimented with several gamified AGLO models in this sense. Gamified AGLOs can embed not only variable text and variable images to form variable questions with attached answers and feedback but also gamification of concepts, namely data structure specific algorithms. The drawback of the approach stands in the complexity of the learning content which involves advanced knowledge of programming. Still some levels of abstraction can be set to favour accessibility e.g., a level controllable by parameters only accessible to non-programmer content authors. We consider that the union of benefits obtained by the proposed composition can help to train faster and better students benefiting from the advantages of both concepts, thus supporting regional development. As future work we intend to reuse and refine the AGLO gamification model to other computer science basic programming disciplines like Programming Techniques where the majority of universities have a large number of students they have to train, especially in the first years of study. Nevertheless, such LOs could be used in high school learning programs.

References

Advanced Distributed Learning (2016). The Experience API (xAPI), www.adlnet.gov.

Boyle, T. (2003). Design principles for authoring dynamic, reusable learning objects. *Australian Journal of Education Technology*, 19(1), 46-58.

Boyle, T. (2006). The design and development of second generation learning objects. Invited talk at Ed Media 2006, *World Conference on Educational Multimedia, Hypermedia and Telecommunications, Orlando, Florida*.

Boyle, T., Bradley, C. (2009). *User Guide for the GLO Maker 2 Authoring Tool*, http://www.glomaker.org.

Burbaite, R., Bespalova, K., Damasevicius, R., Stuikys, V. (2014). Context Aware Generative Learning Objects for Teaching Computer Science, International Journal of Engineering Education, 30(4), 929-936.

Chirila, C. B. (2013). A Dialog Based Game Component for a Competencies Based E-Learning Framework. In *Proceedings of SACI 2013 8-th IEEE International Symposium on Applied Computational Intelligence and Informatics*, Timisoara, Romania, pp. 055-060.

Chirila, C. B. (2014). Educational Resources as Web Game Frameworks for Primary and Middle School Students. In *Proceedings of eLSE 2014 International Scientific Conference eLearning and Software Education, Bucharest, Romania*.

Chirila, C. B. (2014b). Generative Learning Object Assessment Items for a Set of Computer Science Disciplines. In *Proceedings of SOFA 2014 6-th International Workshop on Soft Computing Applications - Advances in Intelligent and Soft Computing, Timisoara, Romania*, Springer Verlag, ISSN 1867-5662.

Chirila, C. B., Ciocarlie, H., Stoicu-Tivadar, L. (2015). Generative Learning Objects Instantiated with Random Numbers Based Expressions. *BRAIN - Broad Research in Artificial Intelligence and Neuroscience*, vol. 6, no. 1-2, Bacau, Romania.

Chirila, C. B., Raes, R., Roland, A. (2016). Towards a Generic Gamification of Sorting Algorithms, In *Proceedings of 12-th International Symposium on Electronics and Telecommunications, Timisoara, Romania*.

Costea, F. M., Chirila, C. B., Cretu, V. (2018). Towards Auto-Generative Learning Objects for Industrial IT Services. *Proceedings of the IEEE 12-th International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania*.

Damasevicius, R., Stuikys, V. (2008). Specification and Generation of Learning Object Sequences for e-Learning Using Sequence Feature Diagrams and Metaprogramming Techniques. In *Proceedings of 2009 9-th International Conference on Advanced Learning Technologies*.

Florea, A., Burghelea, E., Gellert, A. (2015). MiniGL: Game and Learning. *The International Scientific Conference eLearning and Software for Education, volume 1, 180-187, Bucharest: "Carol I" National Defense University*.

Huotari, K. & Hamari, J. (2012). Defining Gamification - A Service Marketing Perspective. *Proceedings of the 16th International Academic MindTrek Conference 2012, Tampere, Finland*.

IEEE Learning Technology Standards Committee, (2016). *LOM working draft v4.1*. Available: http://ltsc.ieee.org/doc/wg12/LOMv4.1.htm.

Jones, R., Boyle, T. (2007). Learning Object Patterns for Programming. *Interdisciplinary Journal of Knowledge and Learning Objects*, vol. 3.

Mubin, S. A., Poh, M. W. A., Jantan, A. H. (2020). Gamification in Programming Language Learning: A Review and Pathway. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 11(12), 1148-1155.

Shorn, S. P. (2018). Teaching Computer Programming Using Gamification. *Proceedings of the 14th International CDIO Conference, Kanazawa Institute of Technology, Kanazawa, Japan*.

Stuikys, V., Burbaite, R., Damasevicius, R. (2013). Teaching of Computer Science Topics Using Meta-Programming-Based GLOs and LEGO Robots. *Journal of Informatics in Education*, 12(1), 125-142, Institute of Mathematics and Informatics, Vilnius.

Zhan, Z., He, L., Tong, Y., Liang, X., Guo, S., Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence*, 3.