Cracking Messages: Logic Programming to Secret Communication

Magdalena Maglizhanova[1,†], Laura A. Cecchi[2,*,†], Veneta Tabakova-Komsalova[1,3] and Jorge P. Rodríguez[2,†]

[1]University of Plovdiv "Paisii Hilendarski", Plovdiv, Bulgaria
[2]Grupo de Investigación en Lenguajes e IA, Universidad Nacional del Comahue, Neuquén, Argentina
[3]Centre of Excellence in Informatics and Information and Communication Technologies, Sofia, Bulgaria, Acad. G. Bonchev St., Block 2, 1113 - Sofia, Bulgaria

Abstract

This paper presents a methodological approach for introducing core concepts of Logic Programming through a gamified and communicative activity based on encrypting and decrypting messages. Students take on the role of secret agents who exchange ciphered messages with unknown partners, using Prolog knowledge bases to construct, apply, and interpret simple ciphers. The proposed didactic sequence connects the learning of facts, rules, lists, and recursion with a concrete practical task: designing a cipher, encrypting a message, and decrypting a received one. The methodology is organised as a series of progressively more complex learning moments, in which each new concept emerges naturally as a response to a communicative or technical need. The paper also discusses LP education initiatives in Bulgaria and Argentina and reports on a preliminary international experience involving students from both countries. Initial observations suggest that combining Prolog, cryptography, role-based interaction, and cross-cultural communication may have the potential to foster logical thinking, motivation, and collaboration in school education.

Keywords

Logic Programming, Cryptography, Primary and Secondary School, Logical Thinking, International Collaboration

CEUR-WS.org/Vol-4311/PEG_regular_09.pdf

1. Introduction

Computer science is increasingly present in compulsory education curricula. Numerous initiatives are underway worldwide in this area, including curriculum reforms, the development of tools, the identification of appropriate practices and concepts for integration into schools, and educational research.

Empowering students to engage critically with technology — rather than merely consuming it — is one of the central challenges of contemporary education. Programming education plays a fundamental role in this regard, preparing students not only to use technology but to understand the principles underlying it. In an era defined by the rapid advancement of Artificial Intelligence (AI), fostering logical, analytical, and critical thinking has become an educational imperative.

Logic Programming (LP) is a programming paradigm, knowledge representation framework, and database management approach based on formal logic. This paradigm brings together concepts from computing, logic, and reasoning [1]. In this regard, LP represents a key practice for developing specialised knowledge in STEM and non-STEM disciplines and for promoting both computational and logical thinking [2, 3]. Despite its theoretical depth and demonstrated utility in AI, knowledge representation, and constraint solving, LP is notably underrepresented in Computer Science education. In recent years, however, a growing number of educational initiatives have sought to address this gap by integrating LP

IV PEG Workshop 2026
*Corresponding author.
†These authors contributed equally.
magdalenamaglijanova@uni-plovdiv.bg (M. Maglizhanova); lcecchi@fi.uncoma.edu.ar (L. A. Cecchi); v.komsalova@uni-plovdiv.bg (V. Tabakova-Komsalova); j.rodrig@fi.uncoma.edu.ar (J. P. Rodríguez)
0009-0000-4978-9282 (M. Maglizhanova); 0000-0001-5236-6715 (L. A. Cecchi); 0000-0002-0617-9844 (V. Tabakova-Komsalova); 0000-0002-4697-6477 (J. P. Rodríguez)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-03

into curricula at various levels, drawing on its major paradigms — Prolog, Answer Set Programming (ASP), Datalog, and Constraint Logic Programming (CLP) [4].

While early initiatives focused on algorithms and programming, emphasising the development of computational thinking, programs are progressively incorporating a broader range of computer science knowledge — and, with it, a stronger connection to students' everyday experiences. End-to-end encryption, as seen in widely used platforms such as WhatsApp, exemplifies the kind of concept that bridges technical understanding and daily life.

In this work, we present a methodological approach for introducing core concepts of Logic Programming through a gamified and communicative activity based on encrypting and decrypting messages.

LP languages are often perceived as difficult to learn because they are highly abstract and declarative. This project aims to overcome this challenge by promoting student engagement within the context of ciphered communication. The approach is motivated by a central principle of learning on a "need-to-know" basis. Students are not learning Prolog as an abstract exercise, but are immersed in a communicative, problem-solving environment in which they must encrypt and decrypt messages to achieve a specific goal.

In this context, knowledge related to LP and cryptography is integrated within an educational framework intended to support the development of logical thinking as a skill for solving everyday problems, while encouraging students to engage with scientific knowledge and its applications.

We further present the binational experience between Bulgaria and Argentina that frames this work. The value of this experience is not limited to its technical and cognitive outcomes — it extends to the sociocultural dimension of bringing students and teachers from two different countries into direct, first-person contact with one another. In an increasingly interconnected world, such encounters foster cultural awareness and broaden students' understanding of the realities that exist beyond their own context. We acknowledge, nonetheless, that this is still a developing experience, and the approach we present should be read as an ongoing effort, open to further refinement.

2. Logic Programming Education

To illustrate the practical application of Logic Programming in educational contexts, this section presents representative initiatives implemented in different national settings. The experiences developed in Bulgaria and Argentina demonstrate how Prolog and LP can be adapted to different educational contexts through school curricula, teacher collaboration, technological tools, and interdisciplinary projects that connect students with meaningful problems and public forms of interaction.

2.1. Logic Programming Initiatives deployed in Bulgaria

The introduction of Prolog and LP in Bulgarian schools has developed gradually — from interest-based and experimental activities to structured STEM curricula for secondary education. This process is part of the international initiative **"Prolog Education and Thinking"** and the national activity **"Digital Bulgaria in Prolog"**, which aim to introduce students to AI, knowledge representation, and LP through Prolog [5, 6]. Across all of these initiatives, Prolog was selected because of its close connection with classical AI, logical inference, knowledge modelling, and problem solving.

In the Bulgarian context, it has been used not only for teaching programming, but also for developing analytical thinking and interdisciplinary understanding by connecting informatics with biology, chemistry, physics, mathematics, humanities, culture, and art [5, 6]. The first activities involved younger students, including grades IV and V at **Alexander Georgiev-Kodjakafaliyata Primary School in Burgas**. There, Prolog was introduced through joint work between IT and science teachers and connected with topics from Human and Nature. At this stage, the goal was not formal programming training, but the development of logical thinking, classification skills, and structured knowledge representation through simple models such as classifications of Animals and Plants.

Later, the initiative expanded to students from grades V to XII through interest-based classes, experimental courses, and project work within **"Digital Bulgaria in Prolog"**[5]. A more systematic

stage began with a two-year STEM programme at the *Vocational High School of Electrical Engineering and Electronics in Plovdiv*. The first year, for grade VIII, focuses on **"Artificial Intelligence – Logic Programming in Prolog"**, while the second year, for grade IX, develops **"Artificial Intelligence – Software Project Development Using the Prolog Language"** [6, 7].

Through these curricula, students study facts, rules, knowledge bases, expert systems, knowledge processing, and software project development. In the second year, they work in teams on STEM-related projects, passing through topic selection, analysis, modelling, implementation, testing, documentation, and presentation. This supports teamwork, communication, analytical thinking, and the application of knowledge from different school subjects.

An important element of the Bulgarian approach is collaboration between teachers from different disciplines. This is supported by the IDEAS platform — Intelligent Data Educational Analysis System — which provides resources, supports Prolog-based learning, and helps build a network of participating schools [8, 9].

Overall, Prolog education in Bulgaria follows a clear progression: early interest-based activities, broader experimental courses, and structured STEM curricula. It helps students understand AI as a field based on logic, knowledge, and reasoning, while developing digital competence, analytical thinking, and interdisciplinary learning.

2.2. Logic Programming Initiatives deployed in Argentina

The integration of LP and logical thinking in primary and secondary schools in Neuquén and Río Negro, Argentina, is being developed within the context of an initiative involving policymakers, educational institutions, and researchers. Gradual and sustained integration is planned that includes school activities, teacher training programs, and the development of teaching resources and materials. This initiative is part of the activities developed by the Research Group on Languages and Artificial Intelligence to promote the development of logical thinking and computational thinking in compulsory education.

For LP to be successfully integrated into compulsory education, it is necessary to build theoretical and practical frameworks that guide the design of coherent and meaningful learning experiences. In this context, Metagame emerges as a methodological approach to teaching based on gamification, immersive narratives, and collaborative learning to introduce LP content. Through fictional stories, students, organised in groups and assuming leadership roles, solve challenges and puzzles that foster the development of computational and logical thinking, using LP as a problem-solving tool [10].

Examples of Metagames were introduced in classrooms with children aged 8 to 11 attending public primary schools in the city of Neuquén, Argentina. In all cases, the students had no prior programming knowledge, so this experience constituted their first introduction to computer science [10]. Subsequently, the initiative was extended to secondary school students, adapting the narratives to adolescent culture; in this case, the students had prior training in imperative programming.

Furthermore, to consistently integrate LP into compulsory education, it is essential that teachers develop a solid understanding of its concepts and teaching strategies. Co-design is presented as a constructivist approach that fosters collaborative work among researchers, teachers, and developers, enabling the design, prototyping, implementation, evaluation, and improvement of consistent and context-specific educational proposals [11, 12]. Primary school teachers from Neuquén, Argentina, collaborated with university researchers in a co-design workshop to develop LP learning experiences linked to their school curriculum through Metagame-based activities[13].

To teach LP effectively, specialised educational resources are needed. Prolog Web App Creator[1] was developed with this goal in mind: an integrated development environment designed to transform the teaching of LP through the creation of web applications. This educational tool prioritises ease of use, collaboration, and project publishing, allowing students to share their work on social media. By consolidating technical learning and fostering technological democratisation, the platform empowers users to move beyond being consumers and become creators of innovative solutions based on LP [14].

[1]Prolog Web App Creator is available at https://prologforkids.fi.uncoma.edu.ar/.

3. Methodology

The proposed methodology follows a gamified, incremental teaching approach designed to introduce core LP concepts across different educational levels. Rather than presenting these concepts in isolation, students engage in a role-based communication activity in which they assume the personas of secret agents exchanging ciphered messages through an unknown counterpart.

In this activity, students work in groups and take on the role of cryptographers first. Each group is tasked with designing a cipher by constructing a Prolog knowledge base, which they then use to encrypt a short message.

Then they transmit both the encrypted message and the cipher to their peer group –trusting that the other side will do the same in return.

Upon receiving a ciphered message, students switch roles and act as decoders, interpreting the knowledge base provided to recover the plain text and formulate a reply. This alternation between encryption and decryption is repeated over two to three cycles of progressively increasing complexity, challenging students to refine and extend their logic programs at each stage. This design grounds abstract Prolog concepts in a concrete, collaborative task while naturally incorporating secondary learning outcomes such as basic cryptography and communicative skills.

Each group of students assumes two roles — agents that encrypt and decrypt — and interacts exclusively through ciphered messages, without direct knowledge of who is on the other side. This anonymity reinforces the narrative: students are agents exchanging classified information through a channel that must remain secure and interpretable only to those who possess the right knowledge base.

A key structural element of the activity is the definition of an overarching goal — an abstract piece of information that one group holds and the other must ultimately infer through the exchange of messages. This goal drives the narrative forward and gives students a concrete purpose beyond the ciphering mechanics. The instructor selects the goal according to the educational level and context of the session, and it fundamentally shapes the dynamics of the entire exchange: it determines what information needs to be encrypted/decrypted and how many cycles of communication are necessary. Suggested goals include: guessing where the other agent lives, identifying a secret meeting location, inferring a hidden identity, or determining the time and place of a covert event. These scenarios are intentionally open-ended: the choice of goal is not merely a narrative device, but a pedagogical tool that the instructor can adjust to match the desired level of conceptual complexity.

The gamification element relies primarily on role immersion rather than points or rewards. Neither group has prior knowledge of the other's encryption strategy, which creates a natural sense of challenge and discovery. This exchange is repeated with complexity increasing progressively: early rounds involve simple LP structures, while later rounds may introduce more elaborate relational structures. Through this escalating difficulty, the activity provides repeated opportunities to engage with core Prolog concepts in a communicative context where meaning must be constructed through interaction.

3.1. Didactic Proposal

The following didactic sequence organises the activity into seven progressive moments, each building upon the concepts and challenges introduced in the previous one. The sequence is designed so that no concept is presented in isolation: instead, every new idea — from facts and constants to rules, lists, and recursion — emerges as a natural response to a concrete limitation that students encounter during the communication exchange. This incremental structure ensures that the complexity of the logic programs grows in tandem with the communicative demands of the secret agent narrative, keeping students simultaneously engaged as learners and as participants in the game. The sequence culminates in a closing reflection that invites students to transfer what they have learned to other domains, consolidating the experience beyond the boundaries of the activity itself.

M 0. *Students are immersed in the narrative of the activity during their first class session. This moment has no technical content: its sole purpose is to establish the fictional frame that will give meaning to*

everything that follows. By encountering the mission before having the tools to carry it out, students experience a felt need that motivates the entire didactic sequence.

Step 1. The instructor introduces the mission context and presents the overarching goal that students must achieve through their encoded exchanges — such as discovering where the other agent lives, identifying a secret meeting location, or inferring a hidden identity. This immersion can be implemented through any of the following alternatives, depending on the available resources and educational context:

- **Written briefing.** A classified mission-style document establishes the narrative and goal without technical guidance.
- **Audiovisual presentation.** A short video or slide deck sets the scene and introduces the communication challenge.
- **Instructor narration.** The instructor presents the mission in character, inviting questions about the goal but not yet about its implementation.

Step 2. Following the mission briefing, the instructor introduces a set of basic cryptography concepts to provide students with the conceptual vocabulary they will need throughout the activity. This includes the notions of *plaintext* and *ciphertext*, and two elementary cipher schemes: substitution ciphers, in which each symbol is replaced by another according to a fixed mapping, and shift ciphers, in which each symbol is displaced by a constant value within an alphabet. These concepts are presented informally and without mathematical formalism, as their purpose at this stage is to orient students within the narrative rather than to provide a rigorous treatment of cryptography.

Step 3. Regardless of the format chosen, students discuss the mission within their group and formulate their first questions: *How do we encode a message? How will the other side understand us?* These questions are left deliberately unanswered, as they serve to motivate the start of Moment 1.

Step 4. Groups are assigned their initial role — encoder — and are informed that roles will alternate across the subsequent moments of the activity.

M 1. *In this first moment, students are introduced to the idea of representing knowledge explicitly. By building a dictionary and an encryption system as a Prolog knowledge base, they encounter facts and constants in a meaningful context: communication.*

Knowledge Structures: Constants. Variables. Facts. Atomic queries.

Step 1. To get acquainted with the notion of facts in Prolog, students construct a small bilingual dictionary mapping words between their native language and a common language, formalised as a knowledge base that can be queried for translations. At this stage, the dictionary may contain nouns only, as the primary goal is to familiarise students with the structure of a Prolog fact rather than to achieve full linguistic coverage.

Step 2. The dictionary is then tested against a set of short messages to assess whether its vocabulary is sufficient for basic communication. Any missing words are identified and added to the knowledge base before proceeding.

Step 3. **Becoming cryptographers.** Students design their first encryption system, referred to as C1, by choosing between two elementary cipher schemes: a shift cipher, in which each character is displaced by a fixed value within the alphabet, or a substitution cipher, in which each character is mapped to another according to a predefined correspondence. The chosen scheme is then formalised as a Prolog knowledge base using the binary relation `code/2`, where each fact encodes a mapping between a plaintext character and its ciphertext counterpart.

Step 4. Using their newly designed encryption system, students compose a short message in their native language, translate it into the common language using the bilingual dictionary built

in the previous step, and encrypt the result using cipher C1. Sending the ciphertext and the cipher through separate channels is a deliberate choice, simulating a basic principle of secure communication: an eavesdropper who intercepts only one channel gains nothing without access to the other.

M 2. *Students now switch to the decoder role, experiencing the activity from the other side. This shift in perspective is pedagogically deliberate: by having to interpret someone else's knowledge base, students consolidate their understanding of facts and begin to see how knowledge can be reused and inverted. The introduction of rules emerges naturally from the need to automate a task that would otherwise be done manually.*

Knowledge Structures: Simple rules.

Step 1. Each group receives an encrypted message and the cipher C2 designed by their peer group, along with the corresponding knowledge base needed to interpret it.

Step 2. Students begin decoding manually by querying `code(A,B)` directly, treating `B` as the known ciphertext character and `A` as the plaintext character to be retrieved. While functional, this approach quickly proves tedious, motivating the introduction of a dedicated decoding rule.

Step 3. To automate the decoding process, students define the rule `decode/2`, which inverts the `code/2` relation:

```
decode(A,B) :- code(B,A).
```

This rule allows students to recover the plaintext character `A` from any ciphertext character `B` without querying `code/2` in reverse manually.

Step 4. Having decoded the incoming message, students compose a reply in their native language, translate it, encrypt it using their own cipher C1, and transmit it back.

Step 5. This exchange is repeated two to three times, depending on the pace of the group, progressively consolidating the students' ability to switch between the encoder and decoder roles.

Problem motivating Moment 3 and Moment 4:
"Imagine that decoding a long text could take a very long time — by the time we manage to decode it, it may already be too late."

M 3. *Having established a working encoder and decoder, students now face the challenge of making their solution interactive. This moment shifts the focus from what the program knows to how it communicates: reading and writing predicates.*

Knowledge Structures: Input/Output predicates.

Step 1. Students transmit their encrypted message and receive a new ciphered message from their peer group.

Step 2. To decode the incoming message, students first define an output-only predicate that takes a ciphertext character `P` as input, retrieves its plaintext counterpart `X` via `code/2`, and prints the result:

```
decodeO(P) :- code(X,P), write('The decoding of '),
              write(P), write(' is '), write(X).
```

Step 3. Students then extend the predicate to handle input as well, allowing the program to read a ciphertext character interactively from the user before decoding and printing it:

```
decodeIO :- read(P), code(X,P),
            write('The decoding of '), write(P),
            write(' is '), write(X).
```

This version makes the decoding process fully interactive, eliminating the need to pass the ciphertext character as an argument and instead prompting the user for input at runtime.

M 4. *Students discover that decoding messages one character at a time is impractical for real communication. This limitation motivates the introduction of lists as a way to represent and process sequences of data. The use of a fixed message header gives students a concrete anchor for understanding how structure in a message can guide its interpretation.*

Knowledge Structures: Lists.

Step 1. Inspired by the Known-Plaintext Attacks[15] used in the cryptanalysis of the Enigma machine, each message is required to begin with a fixed header — a short sequence of characters known to both parties in advance. The students choose a string of 3 to 5 characters, for example: HELLO, 123, ##S##.

Step 2. Students send a message beginning with that keyword and receive another message starting with the agreed keyword — for example, a 5-letter word such as HELLO.

Step 3. As a first step toward full-text decoding, students decode the fixed five-letter header by representing it as a Prolog list and applying the `decode/2` rule to each of its elements:

```
decode_header([A,B,C,D,E|MessageTail]) :-
    decode(A,PA), decode(B,PB),
    decode(C,PC), decode(D,PD), decode(E,PE),
    write(PA), write(PB), write(PC), write(PD),  write(PE).
```

M 5. *This moment centres on recursion as the natural mechanism for processing messages of any length. Students work through a progression of increasingly refined solutions, each one addressing a concrete limitation of the previous. This iterative refinement mirrors the way programmers think: identifying a problem, proposing a solution, and improving it. By the end of this moment, students have built a fully functional text decoder and used it to complete the overarching goal of the activity.*

Knowledge Structures: Recursive rules.

Step 1. Having decoded isolated characters and fixed headers, students now face the challenge of processing a message of arbitrary length. They approach this by traversing the text sequentially, character by character, working with slightly longer phrases such as *"Hello, my name is…"*. This step introduces recursion as the natural mechanism for handling sequences of arbitrary length, motivated by the practical need to decode full messages rather than isolated characters.

```
traverse([]).
traverse([Letter|Rest]) :-
    write(Letter),
    traverse(Rest).
```

Step 2. Building on the traversal predicate, students extend the recursive definition to incorporate the encoding scheme: each time the first character of the list is retrieved, it is mapped through `code/2` and printed before the recursion proceeds over the remaining characters.

```
codeMessage([]).
codeMessage([PlainLetter|Rest]) :-
    code(PlainLetter, CipherLetter),
    write(CipherLetter),
    codeMessage(Rest).
```

Step 3. A more elegant solution: each time the first letter is retrieved, encrypt it, append it to a new list, and process the remainder of the text:

```
codeMessage([],[]).
codeMessage([PlainLetter|RestPlainText],
    [CipherLetter|RestCipherText]) :-
            code(PlainLetter, CipherLetter),
            codeMessage(RestPlainText, RestCipherText).
```

Step 4. Although the previous solution is functionally correct, it requires the input to be provided as a list, which is impractical for real message exchange. Students address this limitation by wrapping the predicate in a higher-level definition that converts the original text into a list, applies the encoding, and converts the result back into a readable string:

```
codeText(PlainText, CipherText) :-
    string_list(PlainText, PlainList),
    codeMessage(PlainList, CipherList),
    string_list(CipherText, CipherList).
```

Step 5. These steps are repeated across multiple cycles of exchange, with each group alternating between encoder and decoder roles, until the overarching mission goal is reached.

M 6. Closing Moment.

The closing moment serves as a space for reflection and consolidation. Having completed the full exchange, student groups meet synchronously for the first time, revealing their identities and reconstructing the arc of the communication they carried out. This debriefing allows students to articulate what they learned, compare the knowledge bases they constructed, and identify the Prolog concepts — facts, rules, lists, and recursion — that underpinned each stage of the activity. The session concludes with a knowledge transfer exercise, in which students and the instructor collaboratively explore how the same concepts apply to problems and domains beyond cryptography, reinforcing the generality of LP as a problem-solving tool.

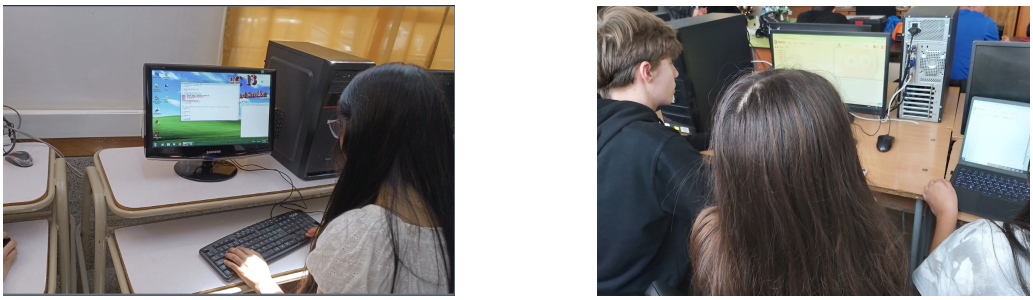
Taken together, the seven moments of this didactic sequence trace a coherent arc from first contact with Prolog facts to the construction of fully recursive text-processing predicates, driven at every stage by the communicative demands of the secret agent narrative. The methodology is designed to be adaptable: the goal, the cipher scheme, and the number of exchange cycles can all be adjusted by the teacher to suit different educational levels and contexts, making the sequence applicable beyond the specific setting in which it was conceived.

4. Preliminary Experience in Secondary Schools in Argentina and Bulgaria

This section reports on an initial implementation of the proposed methodology carried out with secondary school students in Argentina and in Bulgaria. The experience is ongoing and exploratory in nature; the results presented here are preliminary and anecdotal, intended to illustrate the feasibility of the approach rather than to provide conclusive empirical evidence.

The session took place under different academic circumstances in each country, which is itself noteworthy. In Argentina, the activity was carried out after the end of the school term, meaning that students participated entirely on a voluntary basis outside of regular class hours. In Bulgaria, the session was integrated into the students' regular academic schedule as part of their coursework. Active participation was observed during the activity, which suggests that the approach may be applicable both as a curricular component and as an extracurricular initiative.

During this session, the didactic sequence advanced through Moment 1, in which students constructed their first knowledge base to define a cipher and used it to encrypt a short message. A distinctive feature of this implementation was the international dimension of the exchange: Argentine students encoded messages and transmitted them to their peer groups in Bulgaria, who acted as decoders; simultaneously,



(a) Students participating in the activity in Argentina. (b) Students participating in the activity in Bulgaria.

Figure 1: Students participation in each country.

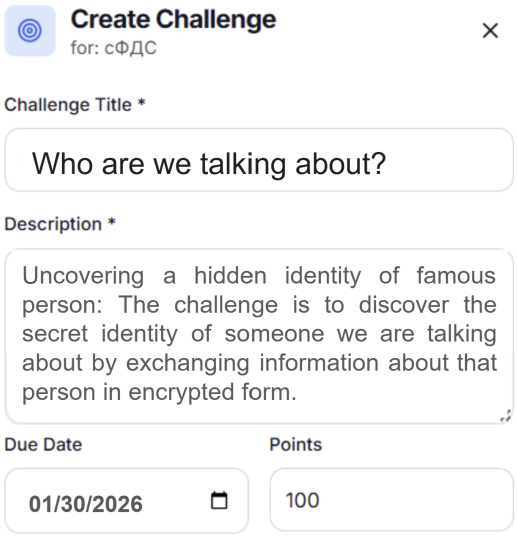
Bulgarian students encoded their own messages and sent them back to Argentina for decoding. English served as the common language for the exchange, providing a shared linguistic ground that made the bilingual dictionary a meaningful and practical component of the activity. This bidirectional exchange meant that both groups experienced the encoder and decoder roles within the same session, giving the activity a fully symmetric and genuinely international character. The first encrypted messages exchanged between both countries, together with their corresponding knowledge bases, are included in Appendix A.

This cross-cultural communication added an additional layer of meaning to the activity, as students were not only motivated by the secret agent narrative but also by the genuine novelty of exchanging coded information with unknown peers in another country. The reception was enthusiastic: students engaged actively with the encoding task and showed genuine curiosity about both the cryptographic mechanics and the identity of their distant counterparts.

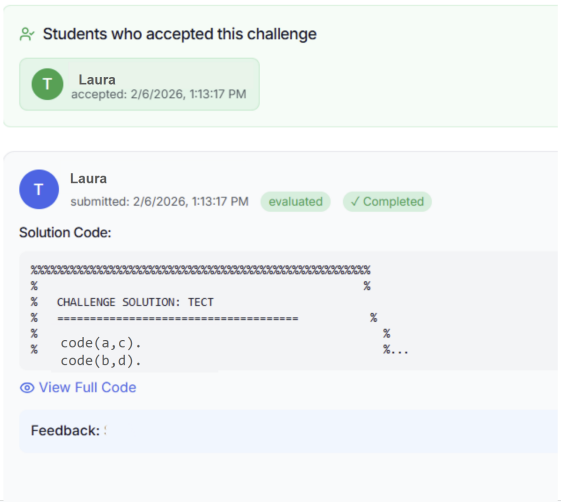
Figures 1a and 1b document the students' participation in each country, serving as qualitative evidence of the activity's reception. Although the sequence was not carried beyond Moment 1 in this instance, the experience validates the opening stages of the methodology and informs the design of future sessions, in which the remaining moments of the sequence will be implemented and more systematic feedback will be collected from both students and instructors. The message exchange was initially conducted between instructors via email, serving as a first proof of concept for the cross-cultural communication dynamic. In future sessions, it is planned to conduct the activity using the IDEAS platform [8, 9], which will provide a structured digital environment for managing peer-to-peer challenges, supporting international collaboration, and collecting more systematic evidence of students' engagement and learning processes.

The IDEAS platform supports structured peer interaction through challenge-based activities. In the context of our methodology, the instructor creates a challenge that frames the overarching mission goal of the activity, as illustrated in Figure 2a. The challenge creation interface allows the instructor to define the task in a structured way: a title, a description of what students must accomplish, a deadline, a point value, and optionally a difficulty level. In our example, the challenge is titled *"Who are we talking about?"* and tasks students with uncovering the hidden identity of a famous person by exchanging encrypted information with their peer group — an instance of the abstract goal described in Section 3.

Once the challenge is published, students can accept it and submit their solutions directly on the platform. Figure 2b shows an example submission by a student named *Laura*, who accepted the challenge and submitted a Prolog knowledge base implementing a simple substitution cipher using the `code/2` relation. Her submission was evaluated and marked as completed. The instructor then reviews the submitted code against the task requirements, assigns a score, and may provide written feedback to help the student understand their mistakes and improve their solution. This review cycle closes the loop between task design, student execution, and formative assessment within a single integrated environment. After the submitted solution is reviewed and validated, students move into the collaborative phase, exchanging ciphers and encrypted messages with their peers.



(a) Creating a challenge.



(b) The student Laura accepted the challenge and uploaded a solution.

Figure 2: IDEAS platform screenshot.

5. Conclusions

This paper presents a gamified approach to introducing LP through secret communication and basic cryptographic tasks. By first acting as cryptographers and later as secret agents responsible for encoding, exchanging, and decoding messages, students engage with a meaningful problem-solving context. This setting gives practical meaning to fundamental Prolog concepts such as facts, rules, lists, input/output predicates, and recursion. Instead of learning these concepts as isolated programming structures, students apply them to address an authentic communication challenge.

The methodology builds on previous LP education initiatives in Bulgaria and Argentina and extends them through an international, peer-based learning scenario. The preliminary implementation with secondary school students in both countries showed that the activity can generate strong engagement, curiosity, and motivation. Although the first experience reached only the initial stages of the didactic sequence, it demonstrated the feasibility of using ciphered message exchange as a meaningful context for learning Prolog and for Computer Science with cryptography, communication, and intercultural collaboration.

An important contribution of the proposed approach is its gradual structure. Each moment of the activity introduces a new LP concept as a response to a concrete limitation encountered by students during the exchange. This progression supports conceptual understanding while maintaining the narrative and collaborative character of the task. The activity also encourages teamwork, problem-solving, abstraction, and logical reasoning, which are essential skills in contemporary STEM education.

This study focuses on the initial stage of the framework; the proposed M0–M6 sequence defines a broader research agenda. Future iterations will implement and evaluate the remaining stages to obtain a comprehensive validation of the approach.

Future work will focus on implementing the complete didactic sequence with larger groups of students and in more diverse educational contexts. The use of the IDEAS platform will make it possible to organise international exchanges more directly, support challenge-based interaction between peer groups, and collect more systematic data about students' participation, reasoning processes, and learning outcomes. Further research will also examine how this methodology influences students' understanding of LP, their motivation to study computer science, and their ability to transfer logical reasoning to other domains.

A. Cipher Messages and Knowledge Bases

The following message and knowledge base were used in the experience.

A.1. Cipher message and knowledge base from Bulgaria

A.1.1. Encrypted Message

```
nkrru lxoktjy, vxurum sgqky kbkxeznotm vuyyohrk!
```

A.1.2. Knowledge Base

```
code(a,g). code(b,h). code(c,i). code(d,j). code(e,k). code(f,l). code(g,m).
code(h,n). code(i,o). code(j,p). code(k,q). code(l,r). code(m,s). code(n,t).
code(o,u). code(p,v). code(q,w). code(r,x). code(s,y). code(t,z). code(u,a).
code(v,b). code(w,c). code(x,d). code(y,e). code(z,f).
```

A.2. Cipher message and knowledge base from Argentina

A.2.1. Encrypted Message

```
umwu ji ucvp, aczgzn ywt oz hwtd jtupcpiujtn umjtni!
```

A.2.2. Knowledge Base

```
code(a,w). code(b,s). code(c,y). code(d,o). code(e,p). code(f,q). code(g,n).
code(h,m). code(i,j). code(j,k). code(k,l). code(l,g). code(m,h). code(n,t).
code(o,z). code(p,a). code(q,r). code(r,c). code(s,i). code(t,u). code(u,v).
code(v,b). code(w,x). code(x,e). code(y,d). code(z,f).
```


Acknowledgments

The authors would like to express their sincere gratitude to the authorities and students of the Vocational High School of Electrical Engineering and Electronics – Plovdiv, Bulgaria and the Centro Provincial de Educación Media N° 71 – Centenario, Argentina. Their collaboration was essential to this work, and we deeply appreciate both the institutional support provided by the authorities and the enthusiasm and willingness of the students to participate.

The authors would like to express their sincere gratitude to Professor Jorge Navarro, whose commitment and support made it possible to put this initiative into practice at the school in Argentina.

Finally, we are especially grateful to the members of PEG for the many stimulating and insightful discussions, particularly Dr. Verónica Dahl, whose ideas and comments were greatly appreciated.

The work was partially supported by Center for Excellence in Informatics and ICT under grant N° BG16RFPR002-1.014-0018-C01, funded by the Research, Innovation and Digitalization for Smart Transformation Program 2021–2027 in Bulgaria and co-funded by the European Union.

Additionally, partial funding was provided by Universidad Nacional del Comahue, Argentina, through Research Projects grants F025 and F026, and Extension Project 53.

Declaration on Generative AI

During the preparation of this work, the authors used Claude and ChatGPT in order to: translate text and, grammar and spelling check. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] R. Kowalski, Computational logic and human thinking: how to be artificially intelligent, Cambridge University Press, 2011.

[2] G. Gupta, E. Salazar, J. Arias, Computational thinking with logic programming, in: Workshop Proceedings of the 40th International Conference on Logic Programming (ICLP-WS 2024) co-located with the 40th International Conference on Logic Programming (ICLP 2024), Dallas, TX, USA, October 12th and 13th, 2024, volume 3799 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024.

[3] C. Jendreiko, The simple generative logic grammar: A tool for teaching logical thinking through visual research in art and design, in: D. Azzolini, L. A. Cecchi, et al. (Eds.), Joint Proceedings of the Workshops and Doctoral Consortium of the 41st International Conference on Logic Programming co-located with 41st ICLP 2025, Rende, Italy, September 9-13, CEUR Workshop Proceedings, CEUR-WS.org, 2025, p. 11. URL: https://ceur-ws.org/Vol-4117/PEG-Regular-5.pdf.

[4] S. O. Semerikov, I. S. Mintii, N. V. Moiseienko, Teaching logic programming: a review, CTE Workshop Proceedings 12 (2025) 399–425. URL: https://acnsci.org/journal/index.php/cte/article/view/838. doi:10.55056/cte.838.

[5] V. Tabakova-Komsalova, S. Stoyanov, A. Stoyanova-Doycheva, L. Doukovska, Prolog Education in Selected Secondary Schools in Bulgaria, in: Prolog: The Next 50 Years, volume 13900 of *Lecture Notes in Computer Science / Lecture Notes in Artificial Intelligence*, Springer, 2023. doi:10.1007/978-3-031-35254-6_12.

[6] V. Tabakova-Komsalova, I. Stoyanov, G. Cholakov, M. Maglizhanova, Introducing Artificial Intelligence to Secondary Schools Through STEM Learning and the Logic Programming Language Prolog, TEM Journal 13 (2024) 3221–3230. doi:10.18421/TEM134-56.

[7] S. Stoyanov, V. Tabakova-Komsalova, Artificial Intelligence – Logic Programming with Prolog, Mathematics and Education in Mathematics 52 (2023) 221–223. doi:10.55630/mem.2023.52.221-223, published by the Union of Mathematicians in Bulgaria.

[8] V. Tabakova-Komsalova, M. Maglizhanova, A. Stoyanova-Doycheva, S. Stoyanov, Platform for Introducing Logic Programming in Bulgarian Secondary Schools – IDEAS, in: D. Azzolini, et al. (Eds.), Joint Proceedings of the Workshops and Doctoral Consortium of the 41st International Conference on Logic Programming co-located with 41st ICLP 2025, Rende, Italy, September 9-13, CEUR Workshop Proceedings, 2025.

[9] M. Maglizhanova, V. Tabakova-Komsalova, V. Dahl, S. Stoyanov, S. Srebrev, T. Srebreva, Web Platform for Symbolic AI and Knowledge, Under review at PEG Workshop 2026 (2026).

[10] J. P. Rodríguez, L. A. Cecchi, Logic programming in primary school: Facing computer science at an early age, in: 2024 L Latin American Computer Conference (CLEI), IEEE, 2024, pp. 1–9.

[11] J. P. Rodríguez, L. A. Cecchi, N. Monjelat, Programación lógica en la educación primaria: análisis sobre un trayecto de formación docente, JAIIO, Jornadas Argentinas de Informática 11 (2025) 123–137.

[12] J. P. Rodríguez, L. A. Cecchi, Constructionist co-design for the learning experience: A work-in-progress on engaging teachers to integrate logic programming in primary school, in: D. A. et al. (Ed.), Joint Proceedings of the Workshops and Doctoral Consortium of the 41st International Conference on Logic Programming co-located with 41st ICLP 2025, Rende, Italy, September 9-13, CEUR Workshop Proceedings, CEUR-WS.org, 2025, p. 10. URL: https://ceur-ws.org/Vol-4117/PEG-Regular-3.pdf.

[13] M. E. Pavón, C. Huentenao, S. Ibarra, Introducción a la Programación Lógica en la Escuela Primaria en Prolog, in: Jornadas Argentinas de la Didáctica de las Ciencias de la Computación - Experiencias Docentes, 2025.

[14] S. Villarroel, C. Gimenez, J. Rodríguez, L. Cecchi, Democratising access to logic programming: A web application design tool for querying prolog code (2024).

[15] J. Katz, Y. Lindell, Introduction to modern cryptography: principles and protocols, Chapman and hall/CRC, 2007.