Journal of Informatics, Computer Science, Data Science And Arificial Intelligence
Volume 4 No. 1 June 2026, ISSN : 3031-9145




# IMPLEMENTATION OF THE LINEAR CONGRUENTIAL GENERATOR IN A WEB-BASED MATHEMATICS EDUCATIONAL GAME

**Mhd Arpandi[1], Khairuddin Nasution[2], Darjat Saripurna[3]**

[1,2,3]Informatics Engineering Study Program, Faculty of Engineering, Universitas Islam Sumatera Utara, [1)]arpandiharahap05@gmail.com, [2)]khairuddin_nst@gmail.com, [3)]darjatsaripurna@gmail.com

***ARTICLE INFO***

***ARTICLE HISTORY:***
*Received : 6 April 2026*
*Revised : 5 Mei 2026*
*Accepted : 8 June 2026*

***Keywords:*** *Linear Congruential Generator; Educational Game; Web Application; Mathematics; Pseudorandom Numbers*

***ABSTRACT***

*Junior high school mathematics is often taught with monotonous methods, and digital practice needs randomized questions so that sessions do not repeat. This study develops MathQuest, a web-based mathematics game for junior high school students that uses the Linear Congruential Generator (LCG) to randomize questions. The research followed the Research and Development method with the Waterfall model; the system was built with PHP and MySQL and tested by black box testing in two iterations. The first iteration found five failures, concerning login messages, repeated question sequences, timer expiry, non-numeric input, and session protection; after debugging, all fourteen scenarios of the second iteration passed. In a one-group pretest–posttest with 30 seventh-grade students, the mean score was 35.97 before and 35.13 after use, with three students improving and nineteen declining. An analysis of the LCG parameters (a = 7, c = 3, m = 10, 20, 50) shows cycles of at most four values, so they do not give a full period. MathQuest meets its functional requirements, but the evidence does not show a learning gain, and the generator parameters need revision.*

## INTRODUCTION

Mathematics is the foundation of cognitive development for junior high school students, and basic arithmetic is the base for the more complex topics of later grades. Observation at the research site showed that many students find mathematical concepts hard to grasp because the teaching method is monotonous and not very interactive. In the Programme for International Student Assessment (PISA) 2022, Indonesia scored 366 in mathematics, ranked 70th among 81 participating countries and economies, and was far below the OECD average of 472 (OECD, 2023). The main problems are low motivation and an abstract approach without attractive visualization.

Students today are digital natives who are used to technology and tend to prefer interactive digital media (Prensky, 2001a). Educational games integrate learning content into game mechanics (Gee, 2003; Prensky, 2001b), and game elements such as scores, levels, and rewards can be used to motivate learners (Deterding et al., 2011). A web platform is a practical choice because it runs on many devices and operating systems without installation, so students and teachers can use it at school or at home.

An important design aspect of an educational game is randomization. Randomized numbers and questions make every session different, which avoids boredom, while the difficulty must still match the chosen level. This task is performed by a pseudorandom number generator (PRNG). The Linear Congruential Generator (LCG), introduced by Lehmer (1951), is one of the simplest and most widely used PRNGs. It produces a sequence by the recurrence $X_{n+1} = (aX_n + c) \bmod m$, where $a$ is the multiplier, $c$ the increment, $m$ the modulus, and $X_0$ the seed. It needs only multiplication, addition, and one stored value, and the same seed always reproduces the same sequence, which helps testing (Knuth, 1997). Table 1 places LCG among other well-known generators.

**Table 1.** Comparison of LCG with other pseudorandom number generators

| Generator | Introduced by | Period | State | Remark |
|---|---|---|---|---|
| Middle-square | von Neumann (1951) | Very short; may collapse to zero | One number | Unreliable in practice |
| **LCG** | Lehmer (1951) | Up to $m$ (full period under Hull–Dobell conditions) | One value | Simple, fast, linear correlation |
| Multiplicative ($c$ = 0) | Variant of LCG | At most $m - 1$; seed not zero | One value | Slightly faster than LCG |
| Xorshift | Marsaglia (2003) | $2^{64} - 1$ (64-bit variant) | One 64-bit word | XOR and shifts, no modulo |
| Mersenne Twister | Matsumoto & Nishimura (1998) | $2^{19937} - 1$ | 624 32-bit words | Passes most tests; more memory |

This study develops MathQuest, a web-based mathematics game for junior high school students in which the LCG randomizes the questions. The research questions are: (1) how to design and develop an attractive and interactive web-based educational mathematics game for junior high school students; and (2) how to implement the LCG as an efficient question randomizer in such a game. The system is evaluated by black box testing and by a one-group pretest–posttest with 30 seventh-grade students of

UPTD SMP Negeri 1 Rahuning, Asahan, North Sumatra. Multiplayer, social media integration, and in-app payment are outside the scope.

## RESEARCH METHODS

### Research design

The study uses the Research and Development (R&D) method, which produces a product and tests its effectiveness (Sugiyono, 2019). The product is a web-based mathematics game that uses the LCG to generate questions. Development follows the Waterfall model in four stages. Requirement analysis lists the functional needs. System design prepares the use case diagram, flowcharts, user experience, and database. Implementation builds the system with HTML5, CSS3, and JavaScript on the client side and PHP with MySQL on the server side. Testing uses black box testing of the input and output of each function.

The learning effect is explored with a One Group Pretest–Posttest Design, $O_1 \rightarrow X \rightarrow O_2$, in which 30 seventh-grade students (aged 12 to 14 years, chosen by purposive sampling with parental permission) took a pre-test, used the game, and took a post-test. The test has ten items on basic arithmetic rated on a scale of 1 to 5. Scores are summarized with descriptive statistics. Because each student is measured twice, the difference is also examined with a paired-samples *t*-test and a Wilcoxon signed-rank test at a significance level of 0.05.

### Tools and system design

The tools are Visual Studio Code as the editor, XAMPP (Apache and MySQL) as the local server, MySQL Workbench and phpMyAdmin for the database, Figma for interface mockups, and Google Chrome and Mozilla Firefox for testing, on Windows 10 or 11 with at least 4 GB of RAM. The functional requirements are that the system generates arithmetic questions with the LCG; validates answers and gives real-time feedback; records scores and progress; authenticates students and teachers; offers three difficulty levels (easy, medium, hard); shows a leaderboard; gives teachers a dashboard to monitor students; and shows learning history and analytics as charts.

The architecture follows the Model–View–Controller pattern with a responsive, mobile-first interface. Passwords are hashed with bcrypt, database access uses prepared statements, and every protected page checks the user role stored in the session. The database (Figure 1) holds the student and teacher accounts, three question tables for grades VII, VIII, and IX (*soal7*, *soal8*, *soal9*), and a score table that stores, for every session, the level, the number of correct and wrong answers, the duration, and the LCG seed.

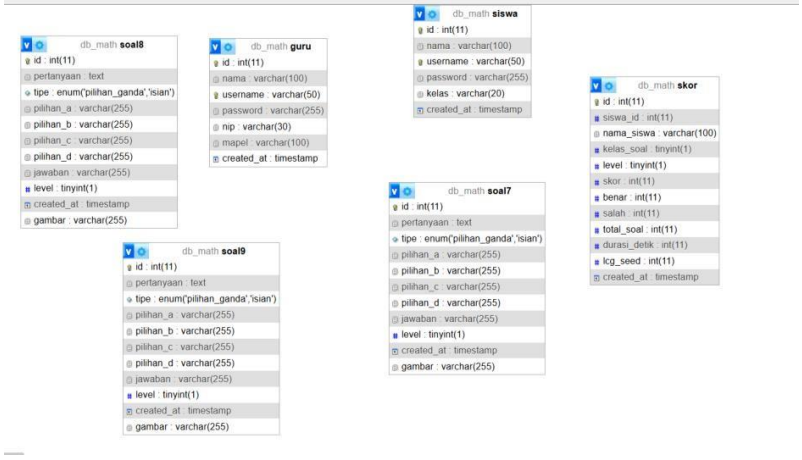



**Figure 1.** Database structure of MathQuest

## Question bank and difficulty levels

Each grade has 45 questions, 15 for each of the three levels. The levels are not arbitrary: they follow five pedagogical criteria based on Bloom's taxonomy (Bloom, 1956) and cognitive load theory (Sweller, 1988), namely the number of solution steps, the type of numbers, the context of the question, the number of concepts combined, and the type of answer. An easy question needs one step and one concept with small integers, a medium question needs two or three steps and two concepts, and a hard question needs more steps, a story context, or three concepts. A correct answer earns one, two, or three points at the easy, medium, and hard level, respectively.

## Design of the LCG question generator

The generator runs on the server with the recurrence in Equation (1). The multiplier and increment are fixed at $a = 7$ and $c = 3$, and the modulus depends on the level: $m = 10$ for easy, 20 for medium, and 50 for hard. The seed $X_0$ is derived from the PHP functions *session_id()* and *microtime()*, so each session starts from a different value. The generator is run twice to obtain the two operands of a question, and the operator follows the chosen category. The new question is compared with the questions already shown in the session; if it is a duplicate, the generator is run again. The correct answer is computed and kept on the server and only the question is sent to the browser.

$$X_{n+1} = (a \cdot X_n + c) \bmod m \tag{1}$$

Table 2 gives a worked example for the medium level ($m = 20$) with seed $X_0 = 5$.

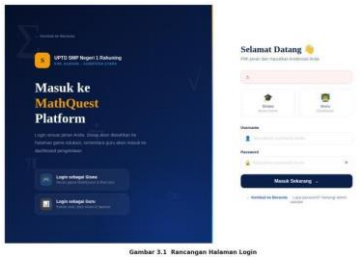
**Table 2.** Worked LCG example for the medium level (a = 7, c = 3, m = 20, X0 = 5)

| Step | Computation | Value | Use |
|---|---|---|---|
| 1 | (7 × 5 + 3) mod 20 = 38 mod 20 | 18 | First operand, question 1 |
| 2 | (7 × 18 + 3) mod 20 = 129 mod 20 | 9 | Second operand: 18 + 9 = ? |
| 3 | (7 × 9 + 3) mod 20 = 66 mod 20 | 6 | First operand, question 2 |
| 4 | (7 × 6 + 3) mod 20 = 45 mod 20 | 5 | Second operand: 6 + 5 = ? |
| 5 | (7 × 5 + 3) mod 20 = 38 mod 20 | 18 | Repeats step 1 |

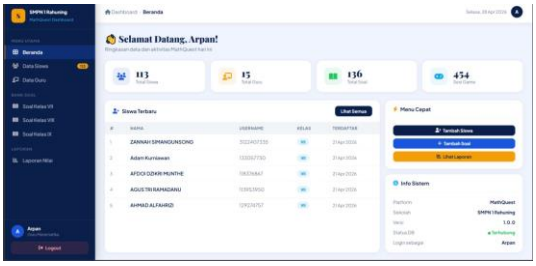
## RESULTS AND DISCUSSION

### Implementation

MathQuest was implemented in PHP with a MySQL database and a responsive interface. Figure 2 shows the main screens. The login page (Figure 2a) lets the user choose the role of student or teacher before entering the username and password; students go to the game and teachers go to the management dashboard (Figure 2b), which shows class statistics, daily activity, student performance, a mini leaderboard, and shortcuts to the question bank, student data, and algorithm settings. Students choose a grade and a level on the home screen (Figure 2c). The game screen (Figure 2d) shows the question that the server generated, a progress bar, a countdown timer, the running score, and the answer options, and the next question appears without reloading the page. After ten questions the result page shows the total score, accuracy, duration, and a per-question breakdown, and the history page shows weekly score charts and the record of all sessions.




(a) Login page




(b) Teacher dashboard




(c) Grade and level selection




(d) Gameplay screen

**Figure 2.** MathQuest interface: (a) login, (b) teacher dashboard, (c) grade and level selection, and (d) gameplay

## Black box testing

Table 3 shows the two test iterations. The first iteration, on the early prototype, found five failures: the login page redirected without an error message; the question sequence repeated because the seed did not include a timestamp; the page stopped when the timer reached zero; non-numeric answers produced an empty submission and a PHP error; and a protected page could be opened directly by URL without a valid session. The fixes were to pass an error variable to the login view; to build the seed from *session_id()* and *microtime()*; to call *form.submit()* from a *setInterval()* countdown at zero; to add *type="number"* with server-side *is_numeric()* validation; and to check *$_SESSION["role"]* on every protected page. In the second iteration all fourteen scenarios passed.

**Table 3.** Black box test results over two iterations

| No. | Test scenario | Iteration 1 | Iteration 2 |
|---|---|---|---|
| 1 | Open the login page | – | Pass |
| 2 | Student login with valid credentials | – | Pass |
| 3 | Teacher login with valid credentials | – | Pass |
| 4 | Login with wrong credentials (error message shown) | Fail | Pass |
| 5 | LCG questions differ in every new session | Fail | Pass |
| 6 | Submit a correct answer (score increases) | – | Pass |
| 7 | Submit a wrong answer (score unchanged) | – | Pass |
| 8 | Timer expires without an answer (auto-submit) | Fail | Pass |
| 9 | Non-numeric input is rejected | Fail | Pass |
| 10 | Open a protected page without login (redirect) | Fail | Pass |
| 11 | Teacher views student data | – | Pass |
| 12 | Teacher views leaderboard | – | Pass |
| 13 | Student views game history | – | Pass |
| 14 | Logout (session destroyed) | – | Pass |

– = not tested in that iteration.

## Pre-test and post-test results

Table 4 and Figure 3 compare the scores of the 30 students before and after using the game. The mean score changed from 35.97 to 35.13, a mean difference of −0.83 points (SD of the difference 1.58). Three students improved, eight were unchanged, and nineteen scored lower. The paired-samples $t$-test gives $t(29) = -2.89$, $p = 0.007$, and the Wilcoxon signed-rank test gives $p = 0.002$, so the small difference is a decrease and not an improvement.

**Table 4.** Descriptive statistics of the pre-test and post-test scores (n = 30)

| Test | Mean | Median | SD | Min–Max |
|---|---|---|---|---|
| Pre-test | 35.97 | 36.0 | 3.53 | 24–42 |
| Post-test | 35.13 | 35.0 | 3.33 | 26–42 |

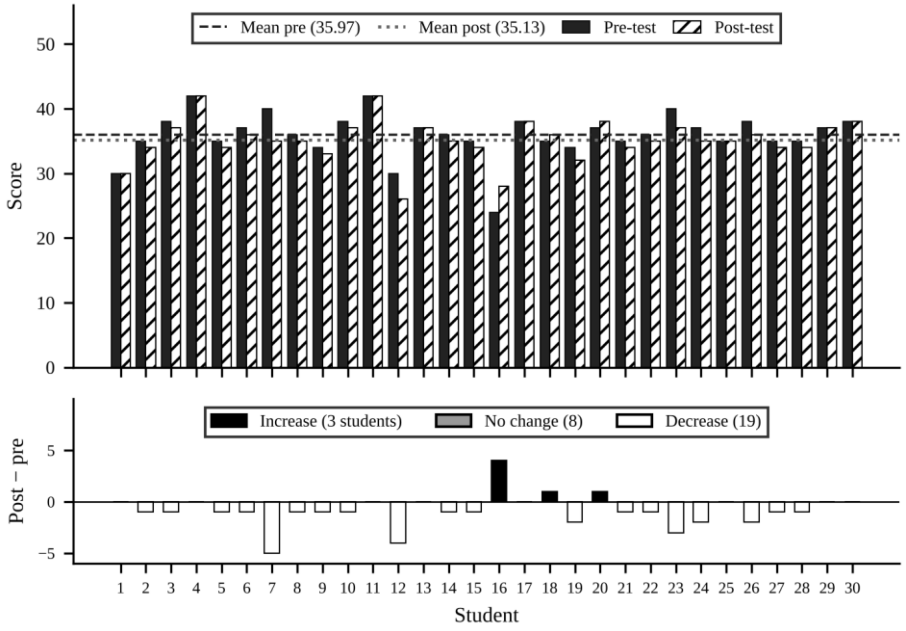



**Figure 3.** Pre-test and post-test scores of each student (top) and the change in score (bottom)

## Discussion

**Functional results.** MathQuest meets the functional requirements it was designed for. Authentication, question generation, answer validation, the automatic timer, session protection, the leaderboard, and the learning history all passed the second test iteration, and the first iteration shows that black box testing found real defects, such as the unprotected pages and the repeated question sequence, that were then repaired. The tests, however, check functions and not learning, and no usability instrument was reported.

**The LCG parameters.** The LCG is cheap to run and reproducible, which suits a game on a modest web server, and the stored seed lets a session be replayed for debugging. The chosen parameters, however, do not give a full period. By the Hull–Dobell theorem (Hull & Dobell, 1962), a full period needs $c$ coprime to $m$, $a - 1$ divisible by every prime factor of $m$, and $a - 1$ divisible by 4 when $m$ is divisible by 4. With $a = 7$, the value $a - 1 = 6$ is not divisible by 5, so none of the moduli 10, 20, and 50 qualifies. Enumerating every seed confirms that the cycle length is at most four for all three moduli, and Table 2 shows the cycle 18, 9, 6, 5 for $m = 20$. A session can therefore draw only a few distinct operands from a given seed, which limits the variety of arithmetic questions and makes the duplicate check unable to find new questions after a few rounds. A better design is one generator with a large modulus, for example the

Park–Miller parameters $a = 16807$, $c = 0$, and $m = 2^{31} - 1$ (Park & Miller, 1988), with the output mapped to the range of each level.

**Learning outcome.** The data do not support the expectation that the game improves arithmetic scores. The post-test mean is lower than the pre-test mean, and the decrease is small but statistically detectable. Several explanations are possible and cannot be separated in this design: the intervention was short, the game content (which includes grade VIII and IX topics) may not match the arithmetic test, there is no control group, and test conditions may differ between the two occasions. The result should therefore be read as showing that a short use of the game did not raise scores, not as showing that the approach cannot work.

**Limitations.** The study has one school, one grade, and 30 students, a single-group design, and no reported results from expert review or student usability testing. The pre-test and post-test scores were not compared with a control group, and the length of the intervention is not documented in detail.

## CONCLUSION

This study built MathQuest, a web-based mathematics game for junior high school students with a student and teacher interface, three difficulty levels, a leaderboard, and learning analytics, using PHP, MySQL, and an LCG question randomizer. All fourteen black box scenarios passed after one round of debugging. In the trial with 30 seventh-grade students the mean score did not improve (35.97 before, 35.13 after), so no learning gain can be claimed. The LCG parameters used ($a = 7$, $c = 3$, $m = 10, 20, 50$) give cycles of at most four values and should be replaced by parameters with a long period. Future work should repair the generator, run a longer study with a control group and more schools, and add a validated usability instrument and expert review.

## REFERENCES

Bloom, B. S. (Ed.). (1956). *Taxonomy of educational objectives: Handbook I, Cognitive domain*. New York: David McKay.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification". *Proceedings of the 15th International Academic MindTrek Conference*, 9–15.

Gee, J. P. (2003). *What video games have to teach us about learning and literacy*. New York: Palgrave Macmillan.

Hull, T. E., & Dobell, A. R. (1962). Random number generators. *SIAM Review, 4*(3), 230–254.

Knuth, D. E. (1997). *The art of computer programming, Volume 2: Seminumerical algorithms* (3rd ed.). Boston, MA: Addison-Wesley.

Lehmer, D. H. (1951). Mathematical methods in large-scale computing units. *Annals of the Computation Laboratory of Harvard University, 26*, 141–146.

Marsaglia, G. (2003). Xorshift RNGs. *Journal of Statistical Software, 8*(14), 1–6.

Matsumoto, M., & Nishimura, T. (1998). Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation, 8*(1), 3–30.

OECD. (2023). *PISA 2022 results (Volume I): The state of learning and equity in education*. Paris: OECD Publishing. https://doi.org/10.1787/53f23881-en

Park, S. K., & Miller, K. W. (1988). Random number generators: Good ones are hard to find. *Communications of the ACM, 31*(10), 1192–1201.

Prensky, M. (2001a). Digital natives, digital immigrants part 1. *On the Horizon, 9*(5), 1–6.

Prensky, M. (2001b). *Digital game-based learning*. New York: McGraw-Hill.

Sugiyono. (2019). *Metode penelitian kuantitatif, kualitatif, dan R&D*. Bandung: Alfabeta.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12*(2), 257–285.

von Neumann, J. (1951). Various techniques used in connection with random digits. In *Monte Carlo method* (Applied Mathematics Series 12, pp. 36–38). Washington, DC: National Bureau of Standards.