SHATRANJ.AI

WHITE PAPER

# From Players to AI Architects: Using the Intertwined History of Chess and AI to Teach the Future of Computing

Prepared for: High School Directors, STEM Department Heads, Curriculum Coordinators, University Instructors, and Youth Program Leaders

Presented by: Shatranj.ai Team (Lead author: FM FT Tamer Karatekin, MIT EECS alumnus)

Open access curriculum supported by the European Union's Erasmus+ Youth programme; coordinated by Fatih Sultan Mehmet Vakif University (FSMVU).

## 1. Executive Summary

Students live inside AI-powered systems every day, yet most K-12 and early undergraduate pathways still treat Artificial Intelligence as either a black-box tool or an advanced elective. Shatranj.ai is an open-access curriculum designed to close that gap by teaching AI as an intellectual tradition: a lineage of ideas in strategy, logic, search, optimization, and learning. The materials are intended to be shared and reused by schools, clubs, and universities, with clear student artifacts that learners can build and explain.

Our method is straightforward: we turn chess and historic board games into a laboratory for computation. Students do not just play; they build engines, recreate classic algorithms, and then modernize them—moving from puzzles to programs to learning agents. Along the way, they encounter the same milestones that shaped Computer Science: Franklin’s 18th-century argument that chess builds character, Shannon’s formulation of chess as a computational problem, the brute-force era culminating in Deep Blue, the open-source ecosystem for chess AI engines, and AlphaZero’s self-play learning paradigm.

What educators and institutions get:

- A rigorous, project-based AI literacy pathway that students can actually explain.
- A curriculum with deep historical grounding (why algorithms exist), not just tool usage (how to click).
- A measurable portfolio of student artifacts: engines, puzzle notebooks, and learning agents.

## 2. Educational Philosophy: Chess as a Formal Intellectual System

In many classrooms, games are used as metaphors. Shatranj.ai treats games differently: as executable systems with well-defined state spaces that let students test ideas rigorously, observe failure clearly, and iterate quickly. Strategic games repeatedly served as the research testbed for AI for decades—long before today’s Large Language Models—because they are clean, precise, and unforgiving about correctness.

Core principles:

- Algorithmic clarity: puzzles and endgames make abstract ideas concrete (search, constraints, recursion, dynamic programming).
- Explainability by construction: students trace a move back to evaluation, search depth, pruning, or learned policy/value.
- Immediate feedback: code changes alter gameplay quality; learning becomes visible engineering.
- Transferable thinking: the same decision-loop vocabulary applies to robotics, trading, logistics, medicine, and modern AI systems.

## 3. The Intertwined History of Chess and AI

Chess is one of the few cultural artifacts that is simultaneously (1) global heritage, (2) a living competitive ecosystem, and (3) a formal decision problem that drove breakthroughs in computing. Shatranj.ai uses this timeline as the syllabus: students see how enduring questions in strategy and logic became formal algorithms and real software.

Key milestones and the ideas they teach:

**Enlightenment era: Franklin, Philidor, and the "Mechanical Turk":** Debates about mechanical intelligence; discipline, foresight, and decision-making as character training.

**Foundations (1950s): Shannon and Turing:** State spaces, evaluation, minimax-style reasoning, and why brute force is impossible at full scale.

**Early programs (1960s): Kotok–McCarthy and MacHack (MIT):** Engineering constraints, representation, and search under tight compute budgets.

**Master strength hardware (1970s–1980s): Belle (Ken Thompson & team):** Specialized hardware, move generation speed, and search depth trade-offs.

**A watershed event (1997): IBM Deep Blue vs. Kasparov:** Scale, pruning, parallelization, and organizational strategy around computation.

**Engine ecosystem (2000s–2010s): Crafty, Fritz, Rybka, Stockfish:** Evaluation engineering, competitive benchmarking, and systematic improvement.

**Open-source + community (2010s): Stockfish, Leela Chess Zero:** Distributed improvement, reproducibility, and transparent benchmarking.

**Neural self-play (2017+): AlphaZero (DeepMind):** MCTS + policy/value learning, self-play, generalization, and modern AI intuition.

**Beyond chess: solving checkers (Chinook / Schaeffer team):** Proof-by-computation and what it means to solve a game.

## 4. The Logic-Puzzle Tradition: Where CS Concepts Become Intuitive

A missing ingredient in many CS courses is the puzzle tradition that originally trained generations of engineers to think in states, constraints, and proofs. Chess and classic logic problems provide that tradition in a form students enjoy. Shatranj.ai makes these puzzles executable: students do not merely solve them; they write the solver.

Signature puzzle modules include:

- Wheat and the Chessboard: exponential growth, powers of two, and computational scale.

- Horse Tour (Knight’s Tour): recursion, backtracking, heuristics (Warnsdorff), and graph thinking.
- Eight Queens: constraint satisfaction, pruning, and systematic search.
- Dilaram Mate and classic endgame studies: pattern libraries that inspire evaluation functions.
- Suli’s Diamond (historic endgame study): state-space reasoning, tablebase ideas, hashing, and dynamic programming.

## 5. Curriculum Topics at a Glance

Shatranj.ai is modular. Programs can run as a semester elective, an after-school program, an intensive bootcamp, or a STEM club series. The content below reflects our standard 21-lesson spine, with explicit extensions for AlphaZero-style learning in historic board games such as Qirkat and checkers.

| Phase | What students build and learn |
|---|---|
| A. Computing Foundations | Python + Jupyter, data types, control flow, functions, files, testing, OOP, and small games (TicTacToe). |
| B. Game Representation | Board encodings for chess & Shatranj, move generation, legality, terminal conditions, and visualization. |
| C. Search & Heuristics | DFS/BFS/UCS/A*, minimax, alpha-beta pruning, evaluation functions, and performance enhancements. |
| D. Puzzles & Proof-by-Code | Horse Tour, Eight Queens, Wheat puzzle, and historic endgame studies implemented as search/DP problems. |
| E. Modern Learning | Q-learning in gridworlds/endgames, self-play principles, and the AlphaZero loop (MCTS + policy/value learning). |
| F. Historic Board Game AI Extensions | Apply AlphaZero-style MCTS + learning loops to smaller domains and historic board games (Qirkat, checkers, Shatranj variants). |

Core student deliverables:

- A working rules engine (move generation + legality) for chess/Shatranj
- A minimax + alpha-beta engine with tunable evaluation
- A puzzle-solver notebook (Queens / Horse Tour / Wheat / endgame study) with visualizations
- A reinforcement learning agent (Q-learning) that improves across episodes
- A mini AlphaZero-style system (MCTS + policy/value) with an extension path to Qirkat/checkers

## 6. AI Systems We Study (and What Each One Teaches)

Students learn faster when they can point to real systems and name the engineering ideas inside them. Shatranj.ai uses a curated set of historic and modern engines as case studies (conceptual and, where appropriate, code-level).

**Turochamp (Turing):** Paper algorithms and what it means to specify a procedure before you have hardware.

**Shannon Type A / Type B:** Brute force vs. selective search; why evaluation matters.

**MacHack (Greenblatt, MIT):** Efficiency, representation, and tournament constraints.

**Belle (Thompson):** Speed, specialized hardware, and search depth trade-offs.

**Deep Blue (Hsu, Campbell, Hoane):** Parallelization, pruning, and scaling search under constraints.

**Rybka / Fritz / Crafty:** Evaluation engineering and competitive benchmarking.

**Stockfish:** Open-source iteration, testing, and distributed improvement.

**Leela Chess Zero:** Neural evaluation with MCTS in a community-driven system.

**AlphaZero:** Self-play reinforcement learning with policy/value networks guided by MCTS.

**Chinook (checkers):** Solving a game and proof-by-computation; relevance to verification and correctness.

## 7. Academic Foundations and Pedagogical Lineage

Shatranj.ai is shaped by the MIT tradition of teaching computation as both craft and intellectual history: build systems, trace the origins of ideas, and learn to explain mechanisms. It also reflects the accessible rigor of Harvard’s CS50 and the broader movement to democratize AI literacy in K–12.

Coursework influences (selected):

- MIT: Algorithms (6.006), Computation Structures (6.004), and the problem-decomposition style common to MIT computing courses.
- SICP tradition (Abelson & Sussman): abstraction, interpretable systems, and deep conceptual clarity.
- Programming for the Puzzled (MIT): puzzle-driven reasoning and rigorous decomposition.
- History/classics of Computer Science (Harry Lewis): ideas as a lineage, not a pile of tools.
- Democratizing AI education in K–12 (Hal Abelson and Cynthia Braezeal): making AI concepts understandable and accessible.
- Harvard CS50: project-based confidence, debugging literacy, and portfolios.

## 8. Implementation Options

Shatranj.ai can be used in multiple formats depending on local constraints:

- After-school club (8–12 weeks): Python basics + puzzle modules + an engine demo.
- Semester elective (12–18 weeks): full 21-lesson spine with assessments and capstone.
- Intensive bootcamp (1–2 weeks): rapid build path to minimax/alpha-beta + intro RL.
- Teacher PD + turnkey modules: train STEM teachers to deliver the curriculum with notebooks and rubrics.

Minimum requirements:

- Basic algebra readiness (no prior programming required for introductory track).
- Laptop or school lab access (Python/Jupyter).
- Chess familiarity helpful but not required; chess concepts are taught as needed.

## 9. Outcomes and Evaluation

Shatranj.ai is designed for measurable progress. We recommend a simple evaluation stack: pre/post programming checks, notebook rubrics, and capstone demos. Students should be able to explain the difference between search vs. learning, heuristics vs. evaluation, and brute force vs. generalization—using their own code as evidence.

Suggested measurable outcomes:

- Code literacy: writing, reading, and debugging Python functions/classes.
- Algorithmic thinking: modeling problems as states/actions/goals; implementing and comparing search methods.
- AI literacy: minimax, alpha-beta, MCTS, reinforcement learning, and self-play at a conceptual level.
- Communication: explaining system behavior with diagrams, logs, and short write-ups.
- Ethics and responsibility: discussing what game AIs can and cannot tell us about real-world intelligence.

## 10. Culture, Character, and Equity

Chess is global cultural heritage. Shatranj.ai connects the journey from Chaturanga to Shatranj to modern chess and invites students to see computing as a global human project. We also connect chess to character education through classical sources such as Benjamin Franklin's essay on the morals of chess, emphasizing foresight, circumspection, caution, and perseverance.

## Appendix A: 25-Lesson Spine (compact map)

Schools can adopt this spine as-is or select modules to match scheduling constraints.

1. Course scope: Shatranj.ai context, platforms, roles, outcomes.
2. Computing basics and Python/Jupyter setup (CPU/RAM/I-O; bits/bytes).
3. Python data types and operations.
4. Conditionals and loops; interactive programs.
5. Functions, parameters, and scope.
6. Files, exceptions, libraries, and testing.
7. OOP via TicTacToe and debugging.
8. Board representation for chess and Shatranj.
9. Move generation, legality, and terminal conditions.
10. Search problems: DFS, BFS, UCS; graph visualization.
11. Heuristics and adversarial search: A*, minimax, alpha-beta, evaluation.
12. Horse Tour: recursion/backtracking + heuristics.
13. Eight Queens: constraint satisfaction and pruning.
14. Wheat & Chessboard: exponential growth and math puzzles.
15. Minimax/alpha-beta in checkmate logic and strategy motifs.
16. Suli's Diamond: historic endgame study; dynamic programming and hashing.
17. Engine evolution (Deep Blue, Rybka, Stockfish) and modifying Stockfish for Shatranj.
18. RL foundations: agent-environment loop; dynamic programming in gridworld.
19. Q-learning on Frozen Lake: Why RL: limits of alpha-beta at scale; exploration vs exploitation; policy vs value.
20. RL for two-rook checkmate with Q-learning: environment design, reward shaping, coordination.
21. Deep Q-Networks: From Q-Tables to Neural Networks
22. MonteCarlo Tree Search on Qirkat (ancestor of checkers)
23. AlphaZero on Reversi/Othello
24. AlphaZero on Qirkat: MCTS + policy/value networks; self-play training loop; mini-AlphaZero demos; extension path to Qirkat/checkers.
25. AlphaZero on Checkers (Turkish checkers variant)

## Appendix B: Selected References (starter list)

- Franklin, B. The Morals of Chess (essay).

- Shannon, C. E. (1950). Programming a Computer for Playing Chess.
- Turing, A. M. (1953). Digital Computers Applied to Games.
- Campbell, M., Hoane Jr., A., & Hsu, F.-h. (2002). Deep Blue.
- Silver, D., et al. (2017). Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm.
- Schaeffer, J. et al. (checkers / Chinook): solving checkers and proof-by-computation.

## About the Author

Tamer Karatekin is a lead author and contributor to Shatranj.ai. He is an MIT-trained engineer (Electrical Engineering and Computer Science) and holds master, trainer, and school instructor titles awarded by the World Chess Federation. He is also a youth chess coach whose students have earned medals at World Youth and World School Chess Championships.

His work connects competitive training habits to computer science practice: students learn to model games as formal systems and then implement search, evaluation, and learning algorithms in Python and Jupyter notebooks. The curriculum emphasizes explainability by construction: learners can trace system behavior back to representation choices, search depth, pruning, or learned policy/value estimates.

He has delivered public talks on the intertwined history of chess and artificial intelligence and brings a global, historically grounded perspective to AI education, including work with early chess literature such as Kitab ash-Shatranj and Libro de Axedrez. The goal of Shatranj.ai is to help learners become creators of intelligent systems, not just users of AI tools, by understanding AI as a human story of ideas, engineering choices, and measurable improvement.

## How to Get Started (Open Access)

All core curriculum materials and whitepapers are openly accessible for educational use at https://shatranj.ai/publications/ and https://lms.shatranj.ai Materials are shared under an open access no commercialization principle; please refer to shatranj.ai for the current license terms and attribution guidance. Institutions can run Shatranj.ai as a club hour, an after-school program, or as modular units inside an existing CS or AI course. Suggested starting path: (1) open the LMS at https://lms.shatranj.ai to follow the lesson sequence, (2) use https://play.shatranj.ai to play and analyze positions with engines for chess and shatranj, and (3) visit https://shatranj.art to connect the technical work to the cultural history of games and puzzles. Open-source historic chess and shatranj fonts and related design assets are available on GitHub: https://github.com/TamerKaratekin/shahi-chess-shatranj-font.

Contact for collaboration and educator support: tamer@shatranj.ai | WhatsApp: +1 (617) 430-5047

Optional add-on for facilitated delivery: ChessCoding.ai offers teacher trainings and student workshops (online or in-person) for institutions that want guided onboarding. These services support long-term sustainability while keeping Shatranj.ai materials open access. Learn more: https://chesscoding.ai

Also watch our TEDxBoston talk:
Chess: Bridging Cultures, Inspiring AI, and Redefining Education
https://www.youtube.com/watch?v=BDLiXo_acLg