🧠 Computational Sciences & Engineering

CONSTRUCTOR TALENT SCHOOL

Computational Sciences & Engineering

This program is designed for high school students (aged 15-18) interested in mathematics, informatics, AI, robotics, and engineering in general.

🎯 The objectives of the program are:

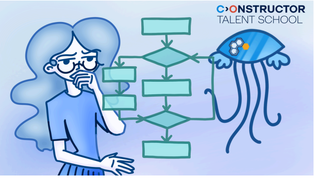
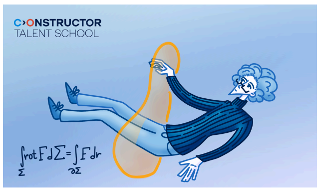
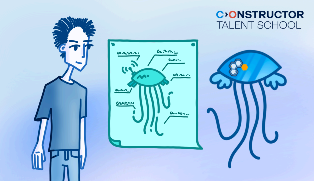
- explaining fundamental mathematical concepts by bridging theory with practical applications;
- developing students' reasoning skills and enhancing their critical thinking abilities;
- teaching the essentials of computer science necessary to effectively address real-life pressing issues and cutting-edge research.

🧑‍🎓 Students enrolling on the program are expected to experience no difficulties with the general school curriculum and feel a need for deepening their knowledge and preparing for future university studies and career.

Courses:

Logic & Problem-Solving Techniques	***(published)***
Combinatorics & Infinity	*(in progress, expected 2026)*
Graph Theory	*(in progress, expected 2026)*
Number Theory	*(expected 2027)*
Calculus & Modeling	***(published)***
Full Introduction to Python	***(published)***
Algorithms & Complexity	***(published)***
Data Structures	*(in progress, expected 2026)*
Neural Networks & AI	*(in progress, expected 2026)*
AI & Robotics	*(expected 2026)*




✅ Logic & Problem-solving Techniques

CONSTRUCTOR TALENT SCHOOL

Logic & Problem-solving Techniques

a.k.a. “Art of Reasoning” ✅

to the program

Description:

An introduction to mathematical language, rigor and ideas. It is designed as a friendly entrance to the world of clear thinking. Students learn how to understand mathematical language, explore logic puzzles, create desired constructions or prove them impossible, and begin to uncover and use patterns. This is the perfect starting point for students new to convincing reasoning, helping them become a confident problem solver from the ground up.

Topics:

- **Boolean operations**

  The language of rigor in mathematics. Learning what a mathematical proof is (and what is not). Main operations (conjunction, disjunction, negation) are considered, along with the main logical laws.

- **Properties and quantifiers**

  Implication and equivalence as logical operations. Discussing the difference between general and particular statements. Formalization mathematical statements using universal and existential quantifiers. Constructing negations to statements with quantifiers.

- **Logical puzzles**

  Construct examples and counterexamples. Discussing when an example is sufficient for a complete solution of a problem and when it is not (and what to do if the example is not sufficient). Logical puzzles about Knights and Knaves.

- **Unordered collections: Sets**

  Introduction to sets and operations on them (intersection, union, complement). Connection between set operations and logical operations. Additionally, the Pigeonhole Principle is explored as a powerful tool for proving surprising results about finite sets.

- **Ordered collections: Sequences**

  Various approaches to dealing with sequences, particularly related to pattern search. Additionally, a number of cases where patterns appear to be false are discussed.

- **Mathematical Induction**

  The fundamental proof technique used to establish statements for all natural numbers by proving a base case and an inductive step, illustrated with numerous cases.

- **Estimating limitations**

  Discussing the arithmetic obstacles, arising in various situations. Calculating the amount of information, which leads to number system base 2.

- **Another point of view**

  Analyzing processes from the end (“rewinding time”). Considering symmetries and using them to solve problems.

✅ Full Introduction to Python

Full Introduction to Python

a.k.a. “Elementary, My Dear Python” ✅

to the program

Description:

This course provides a strong base for understanding essential programming concepts and hands-on experience, setting you on the path to becoming a confident coder. We'll explore the basics of Python, a beginner-friendly programming language, starting from basic operations and ending with classes. For those who are new to programming, this course is the perfect starting point for a coding journey.

Topics:

- **The First Python Experience**
  Syntax and code structure. Variables and data types. Input/output operations. Type conversion. Arithmetic operations. Operators and expressions. Reading and writing data from/to a file.
- **Logic in Python**
  Comparison operations and logical operators. Conditional statements (if, elif, else). While Loop. Iterations. Conditional loops and exit conditions. Infinite loops.
- **Working with a Large Number of Things**
  For Loop. Iterating through sequences. Working with ranges. Creating and using lists and tuples. Indexing and slicing. List manipulation methods.
- **Conquering the World of Text: Strings and Chars**
  Strings in Python. String concatenation and formatting. String operations and substrings. Operations on characters.
- **The Art of Data Organization: Dictionaries and Sets**
  Creating and using dictionaries and sets. Iterating through dictionaries and sets. Dictionary and set methods.
- **Crafting Code Recipes: Functions**
  Function: definition and calls. Working with parameters and return values. Scope of variables.
- **Standing on the Shoulders of Giants: Libraries**
  What a Library is. The NumPy Library. Multi-dimensional arrays. Practice with different Libraries.
- **Crafting New Structures: Object-Oriented Programming in Python**
  Classes and Objects. Class Attributes and Methods. Real-Life Case Studies

✅ Algorithms & Complexity

Algorithms & Complexity

a.k.a. “Taming the Algorithms” ✅

to the program

Description:

This course provides a structured exploration of algorithmic problem-solving and computational complexity. Students will learn to analyze the efficiency of algorithms, optimize problem-solving techniques, and understand key concepts such as Big-O notation, recursion, sorting, searching, and problem reduction. The course emphasizes both theoretical understanding and practical strategies for designing efficient algorithms.

Topics:

- **Elementary Functions**
  Understanding fundamental mathematical functions that appear in algorithm analysis, including logarithmic and polynomial growth. Comparison of different function growth rates to understand algorithm efficiency.
- **Complexity in Terms of Big-O**
  Formalizing algorithm efficiency using Big-O notation and learning to ignore insignificant terms. Applying Big-O analysis to different approaches for solving the same problem.
- **Complexity of working with lists**
  Analyzing the efficiency of list manipulations such as searching and sorting. Binary search and Bubble Sort considered in detail.
- **Reduction**
  A special form of Mathematical Induction: transforming problems into simpler or smaller versions to solve them efficiently. Iterative problem reduction techniques and their applications.
- **Descent**
  Using mathematical and algorithmic descent techniques to solve problems efficiently. Specifically, Euclidean Algorithm and Recursion are considered.
- **Recursion**
  Exploring recursion as a key computational tool for breaking down problems. Special attention dedicated to limitations of recursion. Comparing merging and bubbling sorting algorithms.
- **Sorting**
  Sorting techniques and their efficiency, from simple to advanced algorithms. Insertion sorting and Quicksort considered in detail.
- **Compressing and searching**
  Efficient compressing (Huffman tree) and searching (interpolation search) techniques and data structures.Special attention paid to interpolation search and Huffman algorithm.

✅ Calculus and Modeling

Calculus and Modeling

(Finalized syllabus ✅)

to the program

Description:

This course introduces the core concepts of calculus and their role in mathematical modeling. It covers functions, derivatives, integrals, differential equations, data analysis, and optimization, with applications in natural, social, and technical sciences. Emphasis is placed on clarity of interpretation and relevance to real-world problems.

Topics:

- **Functions as a tool for mathematical modeling**
  Overview of mathematical modeling and its applications. Representation of processes and phenomena using elementary functions.

- **Derivatives: how everything changes**
  Derivatives as a measure of change. The relationship between derivatives and function behavior, including monotonicity and extrema. Application of derivatives to analyze process dynamics and solve basic optimization problems.

- **Integrals and their application**
  Describing processes based on their rate of change. Introduction to the concept of integral. Differential equations as an inevitable stage of mathematical modeling. Applications of integral to solving or analyzing solutions of differential equations.

- **From observations to models: how a mathematical picture of the world is born**
  Identifying key parameters from real-world observations. Construction of basic models. Acquaintance with the logistic model, its assumptions, purpose, and scope of applicability.

- **Statistics: how to work with data**
  Exploration of data characteristics such as mean, median, and spread. Introduction to distributions (normal, binomial, Pareto) and deviations from idealized models. Use of spreadsheets for basic statistical processing.

- **Mathematical models and optimization**
  Use of mathematical models to allocate limited resources efficiently. Examples include rinsing, packing, and transportation problems, with attention to structure and solvability.

- **Mathematical models and decision making**
  Formalization of selection and decision-making scenarios. Thorough analysis of the fussy suitor problem. Imitative behaviour models.

- **Mathematical models and forecasting**
  Modeling future outcomes based on current data. Application of linear regression and logistic regression. Basic scoring models for classification tasks, such as creditworthiness assessment.

⏳ Combinatorics & Infinity

Combinatorics & Infinity

CONSTRUCTOR TALENT SCHOOL

(Finalized syllabus ✅)

to the program

Description:

Combinatorics originated from problems of counting elements in various sets, and then branched out to study structures of sets. This course provides an introduction to combinatorial mathematics and problem-solving techniques, essential for reasoning about options, arrangements, and structures. Students will explore fundamental combinatorial principles, designs, ideas, as well as strategies to tackle a wide range of problems by logical thinking and the application of combinatorial methods.

Topics:

- **Foundations of counting**

  Understanding how to count the number of options, particularly the number of ordered (permutations) and unordered (combinations) selections of objects, with applications in probability and problem-solving. Two-way counting approach in reasoning. Applications: password security and the Birthday Paradox.

- **Counting information**

  The Pigeonhole Principle as a tool for understanding limitations and guarantees in counting problems. Applications to information theory concepts, including optimal strategies for weighing problems. Why certain sorting methods are fastest possible, and introduction to data compression principles.

- **Counting the infinity**

  Introduction to comparing sizes of infinite sets through one-to-one correspondence. Understanding countable sets and exploring which familiar number systems (natural numbers, integers, rationals) have the same "size," contrasted with discovering our first truly larger infinity.

- **Uncountable infinity**

  Exploring uncountable sets and the continuum. Surprising results about higher infinities, including why the plane has the same size as a line, and fundamental theorems for comparing infinite sets. Introduction to the rich hierarchy of infinite sizes.

- **A linear recurrent growth: Fibonacci sequence**

  Introduction to recursive sequences through the famous Fibonacci numbers. Discovering the surprising algebraic properties and identities that emerge from this simple counting rule, with applications to growth patterns in nature and mathematics.

- **A non-linear recurrent growth: Subsets and triangulations**

  Exploring Pascal's Triangle and the binomial theorem through a recursive lens. Introduction to Catalan numbers, which count surprisingly diverse mathematical objects from parentheses arrangements to geometric divisions.

- **Games and strategies**

  Mathematical analysis of games and competitions. Understanding winning and losing positions, constructing simple game trees, and introduction to Nim and related games that reveal deep mathematical structures in strategic thinking.

- **Paints on the pavement: Graph colorings**

  Introduction to one of the most elegant problems in graph theory: how to color vertices or edges with the minimum number of colors. Exploring methods for counting the number of valid colorings and introduction to chromatic polynomials as powerful counting tools.

⏳ Neural Networks & AI

Neural Networks & AI

(Finalized syllabus ✅)

to the program

Description:

This course is a practical introduction to artificial intelligence with a focus on image understanding. You'll start by building simple models and gradually learn the math and core algorithms that power modern AI. As you progress, you'll explore powerful tools like convolutional neural networks, object detection, and image generation. The course balances theory with hands-on projects to help you apply what you learn right away. By the end, you'll understand how AI sees the world — and how to build systems that do the same.

Topics:

- **Getting started: Building a basic AI model**
  Introductory content and fundamental concepts. Basic AI models. Core problem categories such as classification, regression, and clustering.
- **The math we actually need**
  Essential mathematical concepts. Matrix operations and relevant libraries. Probabilities of independent events. Learning algorithms using gradient descent.
- **Core AI algorithms**
  Linear models, including logistic regression, and decision trees. Enhancing performance (ensembles) using a combination of algorithms.
- **Teaching machines to see: CNNs and image basics**
  Computer vision and convolutional neural networks (CNNs). Metrics and errors in model performance.
- **Smarter vision: ResNet, VGG & custom models**
  Advanced image models such as ResNet or VGG. Adapt pre-trained networks (customizing models) for specific image tasks.
- **Finding and understanding objects in images**
  Tasks beyond classification: object detection, image segmentation.
- **Generating images and creative AI**
  AI-powered creativity. Autoencoders. Image generation with Stable Diffusion. Using deep learning for style transfer, upscaling, and synthetic media.
- **The big picture: Evolution and real-world impact of AI**
  Evolution of AI from early rule-based systems to today's deep learning breakthroughs. Real-world applications across industries like healthcare, art, transportation, and security. AI-powered reshaping of the world and challenges that still lie ahead.

🏗️ Robotics & AI

Neural Networks & Robotics

(Preliminary syllabus 2025-12-05)

to the program

Description:

An introduction to the foundations of embodied artificial intelligence through hands-on work with a fully functional robotic platform. Instead of studying AI in the abstract, intelligent behavior is explored as it emerges from the interaction between algorithms, sensors, actuators, and the unpredictability of the real world. The course focuses on intuitive understanding, practical experimentation, and the engineering mindset needed to build and refine autonomous systems. By the end of the program, a fully trained, AI-driven robot is assembled, tested, and ready to navigate a physical rail environment and can be taken home as a personal example of real-world AI.

Topics:

- **Introduction to AI through interactive RL**

  Running ready-to-use RL (reinforcement learning) models like CartPole and tweaking reward values to watch behavior instantly change. Experimenting with different setups to see how algorithms adapt.

- **Robot hardware**

  Assembling the robot by mounting the motors, attaching the camera and sensors, installing the magnet switch and emergency stop, and completing the wiring. Powering the robot on for the first time and verifying that basic movement and the safety cutoff work correctly.

- **Perception: how robots sense the world, and sensor calibration**

  Analyzing raw sensor signals from the camera, distance sensor, and magnet detector to understand noise, range limits, and sampling behavior. Calibrating each sensor through simple experiments adjusting thresholds, testing response curves, and observing how environmental changes affect signal quality.

- **Action: motion, control, and feedback**

  Tuning system robustness by adjusting speed and braking commands from a laptop and observing the robot's response to external disturbances. Building tiny scripts to automate behaviors like "slow down when an object is close.

- **Behavior: policies in the real world**

  Loading several pre-trained policies ("cautious," "smooth," "aggressive") and comparing how each one drives the robot. Running short tests to understand strengths and weaknesses of different behaviors.

- **Reward: shaping intelligent behavior**

  Modifying reward parameters (speed reward, collision penalty, station-stop bonus) to influence navigation style. Simulating how different reward sets would change the robot's choices on track.

- **Training loops and model deployment**

  Launching a simplified training loop that uses the modified reward settings to produce a new policy. Exporting the resulting model and loading it onto the robot to evaluate real-world performance.

- **Bot's Journey**

  Setting up a full miniature railway with obstacles and magnetic stations. Running each robot through the course and presenting results based on speed, safety, and consistency.