CODING CURRICULUM

Creative Problem Solving, Programming Logic & Digital Skills

NOBEL Courses | 3 Hour Erasmus+ Workshop

Course Overview

This 3-hour hands-on workshop introduces students the fundamentals of programming through creative challenges, collaborative problem solving, and real-world coding exercises. Students will leave with a working mini-project and a solid foundation in computational thinking.

Learning Objectives

- Understand core programming concepts: variables, loops, conditionals, and functions
- Apply computational thinking to break down real-world problems
- Write, test, and debug simple programs independently
- Collaborate on a shared coding challenge and present their solution
- Gain confidence to continue exploring coding independently

Prerequisites & Requirements

- No prior coding experience required
- Basic computer skills (typing, using a browser)
- One laptop or computer per student (or shared pairs)
- Internet access for browser-based coding environments (Scratch, Replit, or Code.org)

Session-by-Session Breakdown

BLOCK 1 | 0:00 – 0:45 | Think Like a Programmer

0:00–0:10 10 min	**Welcome & Icebreaker**
Activities	• Facilitator introduces themselves and the Nobel Courses programme • Quick group icebreaker: 'If you were a robot, what one task would you automate?' • Brief overview of what the next 3 hours will look like
Materials	Whiteboard / projector, slide deck
Outcome	Students are relaxed, introduced to each other, and know the workshop structure

0:10–0:25 15 min	**What Is Coding? Why Does It Matter?**
Activities	• Short video or slide presentation: 'Code is everywhere' — examples from apps, games, healthcare, cars • Group discussion: Where have you seen code in your daily life? • Introduce key vocabulary: algorithm, variable, loop, conditional, function • Real-world analogy: A recipe is an algorithm — step-by-step instructions
Materials	Slide deck, printed vocabulary cards (optional)
Outcome	Students understand what coding is and can identify it in the real world

0:25–0:45 20 min	**Unplugged Activity — Human Algorithm**
Activities	• Students work in groups of 3–4 to write step-by-step instructions for a simple task (e.g., making a sandwich, navigating to a room) • Groups swap instructions and try to follow them EXACTLY — revealing gaps in logic • Debrief: How is this like writing code? What happens when instructions are ambiguous? • Introduce concept of debugging: finding and fixing errors
Materials	Paper and pens, task cards
Outcome	Students experience algorithmic thinking and understand the importance of precision

BLOCK 2 | 0:45 – 1:45 | Code It

0:45–1:00 15 min	**Intro to Coding Environment**
Activities	• Facilitator demo: Open Scratch (scratch.mit.edu) or Replit — tour the interface • Students log in or access the environment (guest/shared accounts if needed) • Mini-challenge: Make the sprite say 'Hello, [your name]!' — first lines of code • Introduce: blocks/text code, the run button, the output window
Materials	Laptops with internet access, Scratch or Replit accounts
Outcome	Every student has a working environment and has written their first program

1:00–1:25 25 min	**Core Concepts — Guided Coding**
Activities	• Facilitator leads live coding with students following along: • Step 1 — Variables: Store a name or score • Step 2 — Loops: Repeat an action 5 times • Step 3 — Conditionals: IF score > 10 THEN show 'You win!' • Students make small changes after each concept is demonstrated • Peer check: Turn to your neighbour and explain what each block does
Materials	Laptops, live coding projector screen
Outcome	Students can use variables, loops, and conditionals in a simple program

1:25–1:45 20 min	**Mini Challenge — Build a Quiz**
Activities	• Students independently (or in pairs) build a 3-question quiz on any topic they choose • Must use: at least one variable (score), a conditional (correct/wrong response), and a loop or repeated structure • Facilitator circulates, offers hints — encourage problem-solving before giving answers • Early finishers: add a timer, a sprite animation, or a high score feature
Materials	Laptops, challenge instruction card
Outcome	Each student has a working mini quiz program with core coding concepts applied

☕ **BREAK — 10 minutes**

BLOCK 3 | 1:55 – 3:00 | Create & Share

1:55–2:35 40 min	**Creative Project — Build Something of Your Own**
Activities	• Students choose ONE of three project types: • Option A: Interactive Story — A choose-your-own-adventure with conditionals • Option B: Simple Game — Catch the falling object, avoid obstacles, score points • Option C: Useful Tool — A calculator, a random name picker, or a daily tip generator • Facilitator provides a starter template for each option • Students build, test, and iterate — debugging is encouraged as part of the process • Facilitator prompts: 'What's one thing you'd add if you had more time?'
Materials	Laptops, project starter templates (printed or digital), facilitator circulating
Outcome	Each student has a personalised creative project that demonstrates core concepts

2:35–2:50 15 min	**Showcase & Peer Feedback**
Activities	• Gallery walk format: students leave their screen open and walk around to see 3–4 other projects • Leave a sticky note (or verbal feedback) with: one thing you liked, one question you have • 3 volunteers share their screen with the group and explain what they built (2 min each) • Facilitator highlights creativity, problem-solving, and debugging moments observed
Materials	Sticky notes or digital feedback form
Outcome	Students gain confidence presenting their work and learn from peers

2:50–3:00 10 min	**Wrap-Up, Reflection & Next Steps**
Activities	• Exit ticket: Each student writes on paper — one thing they learned, one thing they want to explore next • Facilitator shares free resources: freeCodeCamp, Khan Academy, CS50, Scratch community • Group reflection: What was harder than you expected? What surprised you?

	• Certificate or digital badge issued (Nobel Courses / Erasmus+) • Group photo for Erasmus+ documentation
Materials	Exit ticket cards, resource handout, certificates
Outcome	Students leave with a completed project, a reflection, and a clear path to continue coding

Key Vocabulary

- Algorithm — a precise sequence of instructions to solve a problem
- Variable — a named container that stores data
- Loop — a set of instructions that repeats
- Conditional — IF/THEN logic that makes decisions
- Function — a reusable block of code with a name
- Debug — find and fix errors in your code

3D PRINTING CURRICULUM

Digital Fabrication, 3D Modelling & Turning Ideas into Prototypes

NOBEL Courses | 3 Hour Erasmus+ Workshop

Course Overview

This 3-hour practical workshop immerses students in the complete design-to-print pipeline. Using industry-grade Bambu Lab A1 and P1S printers, students will move from idea to physical object — learning CAD design basics, slicing software, material science, and the engineering mindset of iterate-and-improve.

Learning Objectives

- Understand the four-stage 3D printing process: Design → Slice → Print → Finish
- Create a simple 3D model using beginner-friendly CAD software (Tinkercad or Bambu Studio)
- Understand how slicing settings affect print quality, speed, and material use
- Safely operate and monitor Bambu Lab A1 and P1S printers
- Apply iterative design thinking: prototype, evaluate, and improve

Equipment & Materials

- Bambu Lab A1 printer (open frame, ideal for demonstrations and observing the print process)
- PLA filament (standard, easy to work with, safe for classroom use)
- Laptops with Tinkercad (free, browser-based) or Bambu Studio installed
- Callipers and rulers for measuring printed objects
- Printed reference sheets: layer height guide, infill patterns, support types

Session-by-Session Breakdown

BLOCK 1 | 0:00 – 0:45 | From Digital to Physical

0:00–0:10 10 min	**Welcome & Hook**
Activities	• Facilitator introduces the Nobel Courses / Erasmus+ context • Show physical printed objects: a gear, a phone stand, a medical device replica, a bridge model • Ask: 'How do you think this was made? What does a 3D printer actually do?' • Quick poll: Who has seen or used a 3D printer before?
Materials	Physical printed objects for handling, slide deck
Outcome	Students are curious and engaged; have a physical intuition for what 3D printing produces

0:10–0:30 20 min	**The 3D Printing Process — Theory**
Activities	• Slide/visual walkthrough of the four-stage process with real examples: • 01. DESIGN — CAD software (Tinkercad, Fusion 360, Blender) — create the 3D model • 02. SLICING — Slicing software (Bambu Studio) converts model to printable layers (G-code) • 03. PRINTING — Printer reads G-code and deposits material layer by layer (FDM explained) • 04. FINISH — Remove supports, sand, paint, assemble • Key concepts: layer height (resolution vs speed), infill (density vs strength), supports (overhangs) • Show a time-lapse of a 3D print completing
Materials	Projector, slide deck, time-lapse video, sample prints showing different infill/layer settings
Outcome	Students understand the complete pipeline and the key variables that affect a print

0:30–0:45 15 min	Live Printer Demo & Safety Briefing
Activities	• Facilitator starts a print job on the Bambu Lab A1 — students observe bed levelling, first layer adhesion • Safety rules: hot end temperature (up to 280°C), do not reach inside during print, ventilation • Explain the Bambu Lab P1S enclosed system — why enclosed = better for certain materials and faster prints • Students receive a checklist: what to check before starting a print
Materials	Bambu Lab A1 and P1S powered up and ready, safety checklist handout
Outcome	Students can safely interact with and monitor the printers

BLOCK 2 | 0:45 – 1:50 | Design & Slice

0:45–1:10 25 min	Tinkercad Design Workshop
Activities	• Students open Tinkercad (tinkercad.com) — free account or facilitator-provided login • Facilitator-led tutorial: navigate the workspace, add shapes, resize, combine, cut holes • Guided project: Design a personalised keyring/tag — include their initials or a simple symbol • Key skills practiced: extruding shapes, boolean operations (hole tool), grouping objects • Save and export as STL file for slicing
Materials	Laptops with Tinkercad access, step-by-step printed guide
Outcome	Every student has a completed STL design ready to slice

1:10–1:30 20 min	Slicing with Bambu Studio
Activities	• Import student STL files into Bambu Studio • Facilitator explains slicing parameters: layer height (0.2mm standard), infill (15% for keyring), supports (on/off), print speed • Students adjust one parameter each and observe the preview change: estimated print time, material used • Facilitate discussion: Why might you choose 10% infill vs 50% infill? Speed vs strength trade-off • Preview the layer-by-layer sliced view — students can 'watch' the print before it happens • Export G-code to SD card / Bambu cloud
Materials	Laptops with Bambu Studio, projector for live demo
Outcome	Students understand how slicing decisions affect the final print

1:30–1:50 20 min	**Print Queue & Observation**
Activities	• Facilitator queues 2–3 student designs to print on the Bambu Lab P1S (fast, reliable) • While printing: students complete a 'Print Observation Log' — noting layer appearance, speed, any issues • Class discussion: What would happen if the first layer doesn't stick? (warping, failed prints — common challenges) • Introduce concept of iterative design: engineers rarely get it right on the first try • Show examples of print failures and what caused them — a normalisation of mistakes
Materials	Bambu Lab P1S running prints, observation log worksheet, example failed prints
Outcome	Students observe real printing in progress and understand quality factors

☕ **BREAK — 10 minutes**

BLOCK 3 | 2:00 – 3:00 | Prototype, Evaluate & Present

2:00–2:20 20 min	**Design Challenge — Solve a Real Problem**
Activities	• Students are given a brief: 'Design a small object that solves a real problem at school, home, or in your community' • Examples: a cable organiser, a phone stand, a door stopper clip, a plant pot marker • Students sketch their idea on paper first (5 min) — encourage rough, fast sketches • Begin modelling in Tinkercad — focus on one key feature, keep it printable within constraints • Facilitator circulates: ask 'What is this for? Who is it for? What would make it better?'
Materials	Paper and pens for sketching, laptops with Tinkercad, idea prompt cards
Outcome	Students apply design thinking to a real problem and produce a new STL model

2:20–2:35 15 min	**Collect Completed Prints & Post-Processing**
Activities	• Retrieve completed prints from the Bambu Lab P1S — facilitator supervises careful removal • Students inspect their print: measure with callipers, compare to digital model • Post-processing demo: removing support material, light sanding, optional painting • Discuss: Is it exactly what you designed? What would you change in version 2?
Materials	Completed prints, callipers, sandpaper (120 grit), optional paint/markers
Outcome	Students hold their finished object and evaluate it critically

2:35–2:50 15 min	**Showcase — Design Pitches**
Activities	• Each student (or pair) presents their design to the group in 90 seconds: • What problem did you solve? Show your digital model and (if printed) physical object • What did you change or improve during the process? • Peers ask one question each • Facilitator highlights engineering thinking, creativity, and problem-solving moments
Materials	Projector or passing laptop around, physical prints
Outcome	Students build confidence presenting technical work and practise communication skills

2:50–3:00 10 min	**Reflection & Next Steps**
Activities	• Exit ticket: 'One thing I made, one thing I learned, one thing I'd do differently' • Resources shared: Thingiverse (free model library), Printables, Bambu Lab community, Tinkercad tutorials • Highlight real-world careers: product designer, biomedical engineer, architect, manufacturing • Certificate or digital badge issued (Nobel Courses / Erasmus+) • Group photo with printed objects
Materials	Exit ticket cards, resource handout, certificates, completed prints
Outcome	Students leave with a physical creation, reflective awareness, and a roadmap to continue

ROBOTICS CURRICULUM

Build Smart Robots with micro:bit and Arduino — Code. Connect. Create.

NOBEL Courses | 3 Hour Erasmus+

Course Overview

This 3-hour intensive workshop brings together hardware, software, and systems thinking. Students will program micro:bit sensors and Arduino-powered servos to build a working robot prototype. Using MakeCode, Arduino IDE, and C++, they will experience the full loop of real-world engineering: design, build, code, test, and iterate.

Learning Objectives

- Understand the difference between micro:bit and Arduino and their real-world applications
- Write and upload code to a physical microcontroller using MakeCode and Arduino IDE
- Connect sensors, motors, and servos to build a responsive robotic system
- Apply a test-and-iterate engineering approach to improve robot behaviour
- Collaborate in teams to solve an open-ended robotics challenge

Equipment & Materials

- micro:bit v2 boards (one per student or pair) — includes accelerometer, compass, Bluetooth, speaker
- Arduino Uno or Nano boards with servo motors and ultrasonic sensors (HC-SR04)
- Breadboards, jumper wires, USB cables, battery packs
- Laptops with MakeCode (makecode.microbit.org) and Arduino IDE installed
- Chassis kits (2-wheel drive robot frames, if available) or cardboard chassis option
- Reference card: pinout diagrams for micro:bit and Arduino Uno

Session-by-Session Breakdown

BLOCK 1 | 0:00 – 0:45 | Robots in the Real World

0:00–0:10 10 min	**Welcome, Hook & Team Formation**
Activities	• Facilitator introduction and Erasmus+ Nobel Courses context • Video hook: 30-second clips of robots in surgery, warehouses, Mars rovers, drones • Ask: 'What do all of these robots have in common? What do they need to function?' • Form teams of 2–3; teams choose a robot team name
Materials	Projector, video clips, team name cards
Outcome	Students are motivated, teams formed, and understand what robots fundamentally are

0:10–0:30 20 min	**Hardware Introduction — micro:bit vs Arduino**
Activities	• Hands-on first: pass around micro:bit and Arduino boards — students examine them • Side-by-side comparison on slides: • micro:bit: built-in sensors (buttons, accelerometer, compass), LED grid, Bluetooth, MakeCode blocks — ideal for rapid prototyping and wireless control • Arduino: more I/O pins, works with motors/servos, C++ coding, real-world physical systems • Analogy: micro:bit is your robot's BRAIN (senses and decisions), Arduino is your robot's MUSCLES (movement and power) • Show the code snippet from the poster: explain #include, void setup(), void loop() — the structure of an Arduino program
Materials	micro:bit and Arduino boards for handling, slide deck, printed pinout cards
Outcome	Students can distinguish between the two platforms and explain their complementary roles

0:30–0:45 15 min	micro:bit First Program — Sensors & Inputs
Activities	• Connect micro:bit to laptop via USB • Open MakeCode (makecode.microbit.org) in browser • Guided activity: program the micro:bit to display a message when Button A is pressed • Then: display a heart when the board is shaken (accelerometer input) • Extend: use the compass — show direction on LED display • Flash the code and test immediately — instant feedback loop
Materials	micro:bit boards, USB cables, laptops with MakeCode
Outcome	Every student has flashed their first program to physical hardware

BLOCK 2 | 0:45 – 1:50 | Build the Robot

0:45–1:10 25 min	Arduino & Servo Control
Activities	• Facilitator demo: connect a servo motor to Arduino Uno using breadboard and jumper wires (power, ground, signal) • Open Arduino IDE — walk through the sketch structure: setup() and loop() • Live code together: • #include <Servo.h> — import the servo library • myServo.attach(9); — connect servo to pin 9 • myServo.write(90); — set angle to 90 degrees • Students replicate the wiring on their own breadboard (or in pairs) • Test: Servo sweeps from 0° to 180° and back — students see physical movement from code • Challenge: Can you make the servo wave hello? (loop from 0 to 90 to 0)
Materials	Arduino boards, servo motors, breadboards, jumper wires, laptops with Arduino IDE
Outcome	Students have a working servo controlled by Arduino code they wrote themselves

1:10–1:30 20 min	**Ultrasonic Sensor — Adding Robot Senses**
Activities	• Introduce the HC-SR04 ultrasonic sensor: emits and detects sound pulses to measure distance • Real-world application: parking sensors, obstacle-avoiding robots, drones • Wire the sensor to Arduino: VCC, GND, Trig (pin 7), Echo (pin 8) • Provide ready-to-use starter code with the pulseIn() function — students upload and test • Serial Monitor: students open it and physically move their hand toward/away from the sensor — watch the distance values change in real time • Discussion: How would you use this to make a robot stop before hitting a wall?
Materials	HC-SR04 sensors, Arduino boards, wiring guide handout, Serial Monitor
Outcome	Students can read sensor data from physical hardware and interpret it in code

1:30–1:50 20 min	**Connecting micro:bit & Arduino — Wireless Control**
Activities	• Facilitator demonstrates wireless control concept: micro:bit transmits a signal, Arduino receives and acts on it • Simplified approach using micro:bit as a remote: pressing buttons on micro:bit sends coded value • Students modify their MakeCode program: Button A sends value 1, Button B sends value 2 over serial/radio • Introduce the combined code structure shown in the workshop poster (from the slide: if buttonA was pressed, angle -= 10) • Teams test: press micro:bit button → servo moves • This is the complete micro:bit + Arduino robot brain-to-muscle system working together
Materials	micro:bit and Arduino boards, USB cables, modified starter code provided
Outcome	Students experience a micro:bit-Arduino integrated system — sensors input, motors output

☕ **BREAK — 10 minutes**

BLOCK 3 | 2:00 – 3:00 | Build, Test & Compete

2:00–2:35 35 min	**Robot Challenge Build Sprint**
Activities	• Teams are given the Robot Challenge brief — choose ONE: • Challenge A (Obstacle Avoidance): Robot detects an object <15cm away and stops or turns • Challenge B (Remote Control): Use micro:bit buttons to control servo direction and speed wirelessly • Challenge C (Reaction Robot): Robot responds differently to different sensor inputs (light, sound, touch) • Teams plan their approach (5 min sketch), then build and code • Facilitator circulates: ask 'What's your hypothesis? What did you test? What will you change?' • Emphasis on iteration: working through problems is the point, not just getting it right first time
Materials	All hardware, laptops, challenge brief cards, extra components available
Outcome	Each team has a functional (or nearly functional) robot demonstrating their chosen challenge

2:35–2:50 15 min	**Robot Showcase — Demo & Debrief**
Activities	• Each team has 2 minutes to demonstrate their robot doing something • Explain: What challenge did you pick? What code did you write? What didn't work at first? • Audience: ask one question to each team • Facilitator awards informal team badges: Most Creative Solution, Best Debugger, Most Iterations, Furthest Reach • Highlight that professional engineers spend more time debugging than writing new code
Materials	Projector (optional), completed robots, team badge cards
Outcome	Students celebrate their work, build confidence, and learn from each other

2:50–3:00 10 min	**Reflection, Resources & Next Steps**
Activities	• Career connections: robotics engineer, embedded systems developer, mechatronics, AI researcher • Certificate or digital badge issued (Nobel Courses / Erasmus+) • Group photo with robots
Materials	Exit ticket cards, resource handout, certificates
Outcome	Students leave with a working robot, a reflective learning record, and clear next steps

Key Vocabulary

- Microcontroller — a small computer on a chip that controls hardware
- GPIO — General Purpose Input/Output pins for connecting components
- Servo — a motor that rotates to a precise angle based on a signal
- Sensor — a device that measures something from the physical world
- Sketch — an Arduino program (setup + loop)
- Serial Monitor — a tool to display data from Arduino in real time
- Iterate — to test, evaluate, and improve a design repeatedly