STEAMCITY CURRICULUM

ECOLOGICAL IMPACT OF MOBILITY REGULATION

thematic: sustainable mobility, transport and regulation

sub-theme: environment, well-being and public health

Introduction

This protocol aims to provide guidelines for engaging students in the **collaborative design of self-driving vehicles**, with a minimum of experience in **programming languages**, detailing the proposed steps and practical examples to enable interested teachers to replicate this approach. Beyond technical skills, these activities create valuable opportunities for students to critically examine the **ecological impact** of their **engineering solutions**, promoting **environmental awareness alongside technological innovation**.

The core of this educational experience is a hackathon—a collaborative problem-solving event where students work in teams to develop autonomous vehicle designs. This approach fosters **digital literacy, teamwork, and practical application of STEM concepts.**

Implementation of this protocol requires teachers to assess the prerequisite knowledge needed for successful participation. The technical aspects require **basic C++ programming skills**, necessitating determination of whether students already possess these skills or if a preliminary training phase should be incorporated. **Pairing students with varied skill levels** can also facilitate peer learning and support.

Group formation represents a critical element to address before beginning. Each team should include **at least one student comfortable with understanding, modifying, and writing C++ code**. This strategic grouping helps balance technical expertise across teams and ensures all students can meaningfully contribute regardless of individual programming proficiency.

The physical classroom setup also requires thoughtful planning. Spaces should be arranged to **facilitate team collaboration** while ensuring all groups have adequate access to **necessary technology and materials**. Creation of **stations or pods** allows teams to work closely together while maintaining enough separation to focus on their specific projects.

The detailed protocol that follows assumes **mixed-ability student teams where technical skills can be shared among members.**

Interdisciplinarity

Technology and engineering
Physics & Chemistry
Biology

Sustainable Development Goals

Overview

Protocol Structure

The learning process unfolds through **three complementary phases** that build upon each other to create a comprehensive understanding of the ecological impact of mobility regulation:

Step 1 - The Training Phase
The training phase aims to gain familiarity with the framework and acquire a smattering of the programming language instrumental to understanding the code that determines the behavior of the self-driving vehicle. Please note that students must have a good understanding of the C++ language to undertake this path without a preliminary basis of acquiring programming basics. This preliminary phase is functional to identify the points of insertion and modification to implement the solution relating to the basic mission that will be carried out in the hackathon.

Step 2 - The Collaborative Hands-on Phase
The collaborative hands-on phase is useful for stimulating participation and the development of digital skills through a collaborative and practical approach to solving two challenges:

- **Implementation Challenge**: This involves modifying the code provided to solve a simplified, but real, problem related to the programming of self-driving vehicles. This challenge involves releasing the code so that the autonomous vehicle can overcome the challenge in the city. This challenge assumes that at least one member per group has skills in understanding, modifying, executing and writing C++ code.

- **High-level Design Challenge**: This addresses a more conceptual problem, such as managing the signage of the future capable of regulating both manual and self-driving vehicles. This challenge involves creating a presentation.

Step 3 - The Hackathon Phase
The hackathon verifies the solutions identified to solve the two previously described challenges via two distinct stages:

- **Competition:** Involves the use of an educational smart city environment, allowing all groups to test and validate the correct functioning of the Roobokart in addressing the assigned challenge.

- **Exhibition and Discussion:** Involves the presentation and collective discussion of the design solutions developed during the hands-on phase.

Getting started

Duration: The total duration of the activities is dependent to the complexity of the framework, the availability of support materials and the pre-existing knowledge of the students in programming. By relying on the Roobopoli framework, it requires a total of (at least) 16 hours, distributed as follows:

- The training phase lasts (at least) 3 lessons of 2 hours.
- The collaborative hands-on phase lasts (at least) five hours.

- The hackathon lasts five hours, over a single day.

Level of difficulty:

★★★★★

Material needed: The used tools are digital. We suggest using Roobopoli (https://www.roobopoli.org) as a framework.

- Instructions to guide the training stage are publicly available (in Italian) at this link: https://www.roobopoli.org/wp-content/uploads/2021/05/Roobokart-il-manuale-operativo-RevA1.pdf
- The implementation challenge stimulates participants to complete the Roobokart behavior starting from the code publicly available on GitHub: https://github.com/Perlatecnica/Roobokart/blob/master/Releases/roobokart_basic_mission.bin
  It is suggested to provide participants with a Roobokart to test the implemented code.
- The hackathon requires a single Roobopoli for all the groups : https://www.roobopoli.org/semaforo

Glossary

Keywords & Concepts	Definitions
Citizen Participation	Involvement of citizens in co-creating urban spaces and decision-making processes for better city planning.
Environmental Sustainability	Strategies aimed at reducing greenhouse gas emissions, promoting renewable energy, and conserving natural resources in urban planning.
Hackathon	A competitive and collaborative event where teams present and test their solutions in a public setting.
Roobokart	A microcontroller-based autonomous vehicle was used as the central element of the programming challenge.
Roobokart Manual	Reference material outlining the Roobokart structure, sensors, and functionality.
Roobokart Sensors	Hardware components that detect environmental conditions and help the Roobokart navigate the track.
Urban Planning	Designing cities to be efficient, sustainable, and accessible, with well-distributed public spaces.
Workshop Approach	A hands-on and interactive method to foster collaboration and innovation among participants.

Bibliography

Mauro D’Angelo and Maria Angela Pellegrino. 2021. Roobopoli: a project to learn robotics by a constructionism-based approach. In Methodologies and Intelligent Systems for Technology Enhanced Learning (MIS4TEL), Workshops.

Gennari, Rosella, Alessandra Melonio, and Mauro D'Angelo. 2023. Engaging Learners in the Collaborative Design of Sustainable Smart Cities. In Sustainable, Secure, and Smart Collaboration (S3C) Workshop @ *CHItaly*.

Mauro D'Angelo. 2023. Engaging Learners in Familiarizing Themselves with Sensors and Actuators. In Methodologies and Intelligent Systems for Technology Enhanced Learning (MIS4TEL), Workshops.

Protocol

Step 1 - Training phase

Background and description of the problem to be solved in this step: This initial phase should be guided by a teacher familiar or self-trained to the Roobopoli framework. The purpose is to create the necessary conditions for students to work effectively during the development phase. Students will learn how to assemble the vehicle (Roobokart) and study the provided code to understand its logic and identify the connection points needed to complete the challenge.

Learning Objectives: Understand the structure and components of the Roobokart. Comprehend the code that will be used during the implementation challenge. Identify potential modification points in the code for the upcoming hackathon. Become familiar with competition rules and evaluation criteria

Conceptualisation

Before beginning the training sessions with students, teachers should familiarize themselves with the Roobopoli framework:

What is Roobopoli?
Roobopoli is an educational smart city platform designed to teach programming and autonomous vehicle concepts in an engaging, hands-on manner. The framework centers around a physical miniature city complete with roads, intersections, and signage that simulates a real urban environment. Within this city, programmable vehicles known as Roobokarts navigate autonomously based on student-modified code. These vehicles incorporate various sensors and microcontrollers that enable them to perceive and respond to their environment, much like actual self-driving cars. The entire system serves as a tangible representation of programming concepts, allowing students to see the immediate physical results of their code modifications.

Key Features of Roobopoli for Classroom Use
The Roobopoli framework offers several pedagogical advantages that make it particularly valuable in educational settings. Its difficulty levels can be scaled to accommodate various student age groups and programming experience, making it accessible for both beginners and more advanced learners. The immediate visual feedback provided when Roobokarts execute programmed instructions helps students grasp abstract programming concepts by connecting code to physical movement and decision-making.

The framework naturally encourages collaborative learning through team-based challenges where students must work together to solve complex problems. This collaboration mirrors real-world engineering environments. Moreover, the focus on autonomous transportation creates meaningful connections between classroom coding activities and relevant technological innovations in society. The interdisciplinary nature of the challenges fosters connections between computer science, engineering, environmental studies, and urban planning, encouraging students to think across traditional subject boundaries.

Technical Requirements
Successfully implementing the Roobopoli framework in a classroom requires careful preparation of the physical environment. Teachers should allocate sufficient space for the Roobopoli city layout, typically requiring at minimum 4 square meters to accommodate the track and provide adequate access for students to observe and make adjustments. Reliable power sources must be available for charging the Roobokarts between testing sessions, as battery failure during testing can impede learning momentum.

Each student group needs access to computers equipped with a C++ development environment compatible with the Roobokart programming requirements. Basic tools for Roobokart assembly and maintenance should be kept on hand to address any mechanical issues that might arise during the sessions. Advance preparation of these technical elements ensures that classroom time remains focused on learning rather than troubleshooting equipment problems.

- **Roobokart operational manual**
  https://www.roobopoli.org/wp-content/uploads/2021/05/Roobokart-il-manuale-operativo-RevA1.pdf
- **Code documentation**
  https://github.com/Perlatecnica/Roobokart/blob/master/Releases/roobokart_basic_mission.bin

Students Investigation

Session 1: Roobokart and Sensors

The first session introduces students to the **physical components of the Roobokart** and establishes foundational knowledge about how these components interact with the controlling microcontroller. Teachers should begin with a comprehensive overview of the **various sensors equipped on the Roobokart**, explaining their individual functions and how they collectively enable the vehicle to perceive its environment. This explanation should include **practical demonstrations that highlight how each sensor responds to different stimuli**, helping students visualize the connection between sensor input and vehicle behavior.

Following this introduction, teachers should guide students **through the interface** between these sensors and the microcontroller, explaining **how data flows from environmental perception to processing and finally to movement decisions**. This portion of the session bridges the hardware understanding with the programming concepts that will follow in later sessions. The theoretical explanations should be reinforced through hands-on assembly activities where students physically construct their Roobokarts under teacher guidance. This tactile experience helps students develop ownership of their vehicles while gaining practical understanding of the mechanical and electronic components.

The session should conclude with **basic functionality testing**, allowing students to verify that their assembled Roobokarts respond correctly to simple commands. This testing phase serves both as a technical check and as an opportunity for students to experience the satisfaction of seeing their work in action. Teachers should refer to the Roobokart operational manual available online for specific assembly procedures and troubleshooting guidance. Each student group should have access to a complete vehicle assembly kit containing all necessary components for their Roobokart.

Resources: The Roobokart operational manual (available at https://www.roobopoli.org/wp-content/uploads/2021/05/Roobokart-il-manuale-operativo-RevA1.pdf) provides detailed guidance for this session. Vehicle assembly kits should be prepared for each group prior to the session.

Session 2: Code Analysis

The second session transitions from hardware to software, helping students **understand the programming logic** that **controls the Roobokart's behavior**. Teachers should begin by walking through the **provided code framework**, explaining its **overall structure** and how different sections correspond to **specific vehicle functions**. This explanation should emphasize the relationship between code commands and physical actions, reinforcing the connection between programming decisions and real-world outcomes. Students should be encouraged to ask questions and discuss their observations as they explore the code together.

As the session progresses, teachers should help students **identify key functions within the code and understand their specific purposes**. This analysis should highlight how various functions interact to create coherent vehicle behavior, demonstrating the importance of structured programming approaches. Particular attention should be given to discussing potential modification points where students might later adapt the code to meet challenge requirements. These "**entry points**" for modification should be clearly identified and explained in terms of how changes might affect overall vehicle behavior.

The latter portion of the session should include **simple programming exercises** that allow students to **experiment with the Roobokart's code framework in a low-stakes environment**. These exercises might involve making minor adjustments to existing functions and observing the resulting changes in behavior. Through these hands-on coding activities, students build confidence in their ability to work with the framework while developing a practical understanding of programming principles. Teachers should ensure that **all students have access to computers** with the appropriate **programming environment for C++ development**, and provide sample code snippets that students can use as starting points for their practice activities.

Resources: The code documentation serves as the primary reference material for this session. Computers with appropriate programming environments should be available for each group, along with prepared sample code snippets for practice exercises. The documentation is accessible at: https://github.com/Perlatecnica/Roobokart/blob/master/Releases/roobokart_basic_mission.bin

Session 3: Competition Rules

The final training session focuses on establishing **clear expectations for how students' solutions will be evaluated during the hackathon phase**. Teachers should begin by presenting the **competition rules in detail**, explaining the **scoring system** that will be used to assess performance. For the implementation challenge, teachers should explain the point system in detail: Each team begins with an **initial score of 200 points at the start of the race session.** Points are **deducted for any non-compliance** with the mission requirements according to a defined penalty table. Students should understand the procedures for technical difficulties:

Tecnical Difficulties and Failure

In the event of a Roobokart failure during competition, teams may request to move to the end of the race list to gain repair time, though this comes with its own point penalty and resets their race session. The time available for repairs extends only until all other teams complete their sessions, and importantly, teams cannot request more than one repositioning. Battery management is another critical element – Roobokarts must begin with fully charged batteries, and failures due to battery issues will incur penalties. If battery problems make continuation impossible, teams can request repositioning with an additional penalty.

When **penalties** occur, teachers should emphasize the proper **restart procedure**:

Restart Procedure

The Roobokart must be restarted after resetting the Nucleo card and positioned by a team representative at the designated re-start point on the track. This point must be before the location where the penalty occurred, in the Roobokart's direction of travel. At the conclusion of available time, the sum of all accumulated penalties is subtracted from the initial score. If the resulting score falls below zero, a final score of zero will be assigned for the race session.

The **presentation component** carries significant weight in the overall evaluation, with a **maximum possible score of 120 points distributed across four key areas:**

- **75 points** are allocated for content addressing five essential topics: **environmental sustainability, technology and innovation, sustainable mobility, urban planning, and citizen participation**. Each topic carries a maximum of **15 points**, with evaluation focusing on idea originality, solution feasibility, and conceptual clarity.

- Another **10 points** are awarded for **teamwork**, assessing how well group members express their ideas, accommodate others' perspectives, and collaborate toward common goals.

When assessing the teamwork component, teachers can use the following guiding questions:

- **How are presentation turns distributed within the team?** Are slides presented by only one student (potentially negative approach) or by multiple team members (positive approach)?

- **How do students handle potential errors or additions from teammates?** Do students correct their teammates (potentially negative approach) or complement what their teammates have said (positive approach)? Do students step in when a teammate is struggling (positive approach)?

- **What is the attitude of each speaker?** Does the presenter refer to the group effort (positive approach) or speak in the first person singular (negative approach)? Do students deliver content promptly and directly without hesitation (positive approach)?

- The **presentation quality** itself accounts for **20 points**, with emphasis on clear communication, logical structure, visual presentation, information completeness, and full participation from all team members.

- The final **15 points** reward **innovative thinking**, focusing on novel, forward-looking ideas that can meaningfully improve urban environments from both technological and environmental perspectives.

The session should conclude with the **formation of student groups** for the next phase, ensuring that each group includes at least one student with **sufficient C++ programming skills** to lead the technical implementation aspects of the challenge. Teachers should provide students with a written handout detailing all scoring criteria to reference throughout the remaining phases of the project.

Resources: A prepared handout detailing the point system, penalties, and presentation evaluation criteria should be distributed to all students. Teachers might also consider providing a checklist or rubric that students can use to self-assess their progress throughout the subsequent phases.

Conclusion & Further Reflexion

The training phase prepares students for the hands-on collaboration phase by ensuring they understand both the hardware and software components of the challenge. The knowledge gained about the Roobokart's structure, code framework, and competition rules will be essential for developing innovative solutions in the implementation and design challenges that follow.

By the end of the training phase, students should be organized in groups so that at least one student per group has skills in understanding, modifying, executing, and writing C++ code. This ensures all teams can successfully participate in the implementation challenge.

The Training Phase establishes the essential foundation for students to engage meaningfully with the autonomous vehicle design challenge. Through structured exploration of the Roobopoli framework, code analysis, and competition rules, students develop the technical understanding and contextual knowledge necessary for success in subsequent phases. This initial investment of classroom time yields significant returns as students transition to more independent work in the hands-on and hackathon phases.

Teachers should observe student engagement closely during this phase to identify potential challenges early. Pay particular attention to how students interact with the technical components, noting which aspects generate excitement and which might require additional support. The way students organize themselves during these sessions often provides valuable insights for effective group formation. While technical skills are important, remember that balanced groups benefit from diverse strengths including creative problem-solving, systematic thinking, and effective communication.

Step 2 - Collaborative hands-on phase

Background and description of the problem to be solved in this step: This phase invites students to work collaboratively on two parallel challenges that address both technical implementation and conceptual design. Building on the knowledge gained during the training phase, students now apply their understanding to program a self-driving Roobokart and develop innovative ideas for sustainable urban environments. Through these hands-on activities, students enhance their programming skills while reflecting on the ecological impact of future cities.

Learning Objectives: Develop programming skills through practical application in a real-world context. Apply understanding of the Roobokart system to solve specific navigation challenges. Explore concepts of environmental sustainability in urban planning and technological innovation. Collaborate effectively in diverse teams to address complex problems. Create connections between technical solutions and their ecological impacts.

Conceptualisation

Before beginning the collaborative hands-on activities, teachers should understand the dual challenge approach that characterizes this phase and how it supports comprehensive learning outcomes.

Team Formation and Peer Learning
This phase requires thoughtful organization of student teams to maximize learning opportunities. Each team should include students with **diverse skills and knowledge areas, creating natural opportunities for peer learning**. This deliberate mixing of **technical abilities, creative thinking, and communication skills** enriches the learning experience for all participants. While ensuring that each team has **at least one student proficient in C++ programming** remains important, teachers should also consider **balancing other strengths** such as design thinking, presentation skills, and environmental knowledge.

The peer model embedded in this approach naturally develops **digital competencies** as students share their expertise with teammates. Those more confident in programming guide others through technical challenges, while students with strengths in other areas take **leadership roles in the conceptual design aspects**. This collaborative structure mirrors **real-world project environments** where **multidisciplinary** teams must integrate diverse perspectives to develop comprehensive solutions.

Dual Challenge Structure
The hands-on phase encompasses two complementary challenges that together address both practical implementation and conceptual innovation. The "**City Challenge**" focuses on technical programming skills, requiring students to apply their understanding of the Roobokart framework to solve **specific autonomous navigation problems**. Simultaneously, the "**Ecological Impact of the City of Tomorrow**" challenge invites **creative thinking** about sustainable urban design and environmental implications of emerging technologies.

This parallel approach helps students recognize the **interconnections between technical capabilities and their broader societal applications**. By working on both challenges concurrently, students develop a more complete and integrated understanding of **how programming and design decisions influence ecological outcomes**. The dual-challenge structure also accommodates different learning preferences and strengths, allowing all students to contribute meaningfully to their team's efforts.

Workshop Approach
The collaborative hands-on phase employs a workshop methodology where teachers serve primarily as **facilitators** rather than instructors. This approach encourages **student agency** and **problem-solving** while creating space for **authentic collaboration**. Teachers should monitor group dynamics and provide support when needed, but avoid **directing specific solutions or approaches**. The role shifts from presenting information to asking guiding questions, suggesting resources, and helping teams overcome obstacles when they become genuinely stuck.

Students Investigation

The City Challenge: Implementation

The first challenge, called "City Challenge," represents the technical implementation component where students **program their Roobokart to respond to a specific mission**. This hands-on activity requires students to build upon the skeleton code provided during the training phase, adding functionality to meet detailed mission requirements.

The base mission is defined as follows:

> The Kart moves on the streets of a standard Roobopoli, within its own lane, reading the horizontal road signs that provide information about intersection crossing possibilities. It must respect traffic lights, crossing on green and stopping on red, and navigate through intersections by randomly choosing available paths. If an obstacle is present, the Kart must stop and resume the mission when the obstacle is removed.

The starting point for this programming challenge is the basic mission code available at: https://github.com/Perlatecnica/Roobokart/blob/master/Releases/roobokart_basic_mission.bin. Teachers should emphasize to students that **this code serves only as a reference and does not implement the entire mission**. Students must appropriately complete it to fulfill the assigned mission requirements. **Each team should have access to a Roobokart for testing and refinement throughout the session.**

Each group works autonomously and independently, managing the challenge resolution as they see fit. Teachers should emphasize that each team is responsible for their own materials and must ensure that everything complies with the regulations when competition time arrives.

Teachers should encourage an **iterative approach** where students implement features incrementally, **testing each addition before moving to the next**. This methodology helps teams manage the complexity of the challenge while ensuring steady progress. While supporting students' technical questions, teachers should also prompt teams to document their programming decisions and reflect on how their implementations might address real-world autonomous vehicle scenarios.

The Ecological Impact Challenge: Design

The second and final challenge, called "Challenge for the Ecological Impact of the City of Tomorrow," invites students to reflect on how we might rethink tomorrow's cities by improving ecological impact through the use of innovative technological and digital tools.

> For this challenge, each group is asked to produce a presentation that addresses the following aspects:

1. **Environmental Sustainability**: A city of tomorrow should be environmentally sustainable, with reduced greenhouse gas emissions, efficient use of natural resources, and promotion of renewable energy. Examples might include introducing more green spaces within cities, using alternative energy sources for transportation, or transitioning from non-renewable energy sources (e.g., coal, oil) to renewable ones (e.g., increased use of solar energy through the installation of solar panels on buildings).

2. **Technology and Innovation**: A city of tomorrow should use technology and innovation to improve citizens' quality of life, make public services more efficient, and optimize resource use. Some ways technology and innovation can create "smarter," more sustainable, and livable cities include:
    - *Internet of Things (IoT)*: IoT technology connects devices and objects throughout a city to collect data and information that improves citizens' quality of life. For example, IoT can ensure proper waste management and disposal or improve public transportation.
    - *Smart City*: A smart city uses technology to enhance citizens' quality of life through technological solutions for managing public services such as lighting, energy, and water distribution.

3. **Sustainable Mobility**: A city of tomorrow should promote sustainable mobility with low-emission transportation, car or bike sharing programs, and pedestrian/cycling spaces. Examples might include more intensive use of public transportation with a reduction in private vehicles (which pollute more), or alternatively, greater use of more sustainable transportation within urban spaces, such as bicycles, with corresponding allocation of space for such means.

4. **Urban Planning**: A city of tomorrow should be planned rationally, with effective land management and more sustainable urban densification, building a human-scale city with well-distributed public spaces and easily accessible services. However, planning must occur sustainably, for example, with low-energy and smarter buildings and accessible green spaces.

5. **Citizen Participation**: A city of tomorrow should involve its citizens in city planning and management through participatory and consultation processes. Future citizen participation should be a central element in urban planning, involving citizens in co-creating the cities where they live. This can happen, for example, through digital platforms, online public consultations, workshops, and meetings with citizens. At the same time, the inclusion and participation of all citizens in decision-making processes should be guaranteed so that everyone can express their needs, thus developing strategies suitable for meeting their needs, always from a sustainable perspective.

Teachers should encourage teams to use presentation tools they are comfortable with, such as **PowerPoint, Google Slides, Prezi, or Mentimeter**. The focus should be on developing thoughtful content that demonstrates understanding of both technological possibilities and environmental considerations rather than on presentation aesthetics.

Material Management: Each team should be responsible for managing their own materials and ensuring everything meets requirements when competition time arrives. Teachers should prepare one Roobokart per group and verify that all vehicles are functioning properly before the session begins. Additionally, ensure that computers with appropriate programming environments are available for each team, along with presentation creation tools.

Time Management: With a total duration of five hours, this phase requires careful time management. Consider dividing the session into segments with clear transitions between activities. While the suggested allocation gives equal time to both challenges (2.5 hours each), teachers may adjust based on their specific classroom context and student needs. Some teams might benefit from a more fluid approach where they alternate between the challenges, while others might prefer to tackle one challenge before moving to the next.

Conclusion & Further Reflexion

During this collaborative hands-on phase, teachers should actively **observe how teams distribute tasks and establish collaborative patterns**. These observations provide valuable insights into student learning and group dynamics. Consider taking notes on:

- How effectively teams distribute technical and conceptual tasks?
- Which students emerge as leaders in different aspects of the challenges?
- How teams resolve disagreements or technical obstacles?
- The strategies teams develop for integrating programming solutions with ecological considerations

By the end of the session, teachers should collect both the code produced for the City Challenge and the presentations created for the Ecological Impact Challenge. These artifacts serve as evidence of learning and preparation for the final hackathon phase.

The collaborative hands-on phase prepares students for the hackathon by developing both their **technical solutions and conceptual presentations**. The code created during the City Challenge will be **tested during the competition portion of the hackathon**, while the presentations developed for the Ecological Impact Challenge will be shared during the exhibition portion. This phase represents the core development work that will be evaluated and refined in the final phase.

Thanks to this step, students gained engagement with both the technical and conceptual dimensions of autonomous vehicle design. Through practical programming activities and thoughtful consideration of environmental impacts, students develop a more nuanced understanding of how technology shapes urban environments. This phase builds critical connections between abstract programming concepts and their real-world applications.

Teachers should pay particular attention to how the dual challenges influence each other in students' thinking. Do programming decisions reflect environmental considerations? Do design ideas account for technical limitations? The most successful teams will demonstrate integration between these two aspects, recognizing that technological solutions must address ecological imperatives.

Step 3 - Hackathon

Background and description of the problem to be solved in this step: The final phase of the protocol is dedicated to the public exposition of solutions developed during the hands-on collaboration phase. This hackathon serves as both a competition and a presentation forum where students demonstrate their programming solutions and share their ideas for cities of tomorrow. Through this culminating experience, students refine their ability to communicate technical concepts, improve their programming skills, and engage in collective brainstorming that enriches everyone's understanding.

Learning Objectives: Develop public speaking and presentation skills for technical and conceptual ideas. Refine programming solutions through competitive testing. Participate in collective knowledge sharing through public discussion. Apply feedback in real-time to improve solutions. Experience authentic assessment in a collaborative environment.

Conceptualisation

Before conducting the hackathon, teachers should understand its dual nature as both a competitive event and a collaborative learning opportunity. Unlike traditional testing, the hackathon format provides authentic assessment while fostering a spirit of community and shared discovery.

Physical Setup Requirements

The hackathon requires a Roobopoli city assembled and ready for testing. This smart city should be set up before the hackathon begins, as one city will serve all participating groups. Teachers should follow the construction instructions available online at: https://www.roobopoli.org/wp-content/uploads/2021/05/Roobokart-il-manuale-operativo-RevA1.pdf in the "La Roobopoli" section.

The city layout should include clear starting positions, intersections with traffic lights, and space for obstacle placement as described in the mission parameters.

In addition to the physical city, the classroom should be arranged to accommodate both the competition area and a presentation space. The competition area should provide good visibility for all students to observe the Roobokarts in action, while the presentation space should support digital presentations and group discussions. Technical equipment including projectors, computers, and charging stations for Roobokarts should be prepared in advance.

Assessment Framework

The hackathon integrates assessment of both technical implementation and conceptual design. The competition portion evaluates students' programming solutions against objective criteria established during the training phase, while the presentation portion assesses their understanding of broader implications through the ecological impact challenge. This comprehensive approach ensures that students with different strengths can demonstrate their learning and contribute to their team's success.

Teachers should approach the hackathon as a culminating learning experience rather than simply a test. The competitive element serves primarily to motivate engagement and provide authentic context for applying skills. Throughout the event, teachers should emphasize that the ultimate goal is collaborative learning and skill development rather than winning at all costs.

Teacher Role During Hackathon

During the hackathon, teachers transition to the role of facilitators and judges. As facilitators, they maintain the event's structure, ensure fair competition, and guide productive discussions following each presentation. As judges, they apply the established evaluation criteria consistently across all teams. This dual role requires balancing encouragement with objective assessment, celebrating creative solutions while maintaining consistent standards.

Students Investigation

Competition Phase: Roobokart Testing

The competition phase follows a structured format to ensure fair and consistent assessment of each team's programming solution. Teachers should explain the competition rules to all students before beginning:

- The teacher will call teams to compete in an arbitrary order. Teams must respond promptly, as any team that does not appear **at the competition table within 60 seconds of being called** will forfeit their race session. If a team is not ready to race when called, they may request, within the same 60-second window, to be moved to the end of the competition order by accepting the corresponding point penalty.

- Each competition consists of **two separate two-minute sessions per team**. Importantly, a team's sessions are not consecutive, allowing time for code adjustments between sessions. Teams may modify their code between sessions, even for simple calibration purposes, but they cannot test modified code on the official competition table before their next session.

- At the beginning of each session, the Roobokart must be positioned **at the starting reference point** indicated by the teacher and highlighted on the competition table. A representative from the competing team must start the Roobokart from the designated starting point on the track. During each session, the Roobokart must navigate **according to the rules of the basic mission** described in the previous phase.

- Timing begins when the Roobokart leaves the starting point and continues during actual movement, traffic light stops, obstacle management, and intersection crossings. The timer is **temporarily paused whenever the Roobokart is off-track due to penalties** or stopped because of an obstacle placed by teachers during the competition.

Teachers should **position obstacles at strategic points** during the competition to test the Roobokarts' ability to detect and respond appropriately. These obstacles should be **consistent across teams to ensure fair evaluation**.

After each team completes both sessions, points are calculated based on the initial allocation (200 points) minus any penalties incurred during the competition.

Presentation Phase: Ecological Impact Solutions

Following the competition phase, teams present their **solutions for the ecological impact challenge**. Each team delivers their presentation to the entire class, explaining their vision for tomorrow's sustainable city. Teachers should establish a **consistent time limit for presentations (typically 5-10 minutes)** and ensure that all team members participate in the presentation.

After each presentation, teachers should **facilitate a brief discussion** period where **other students can ask questions** and offer feedback. This collaborative exchange enriches everyone's understanding and provides valuable practice in both giving and receiving constructive criticism. Teachers should guide these discussions to focus on the five key areas outlined in the challenge: **environmental sustainability, technology and innovation, sustainable mobility, urban planning, and citizen participation**.

The presentation phase offers an opportunity for students to demonstrate their understanding of complex interconnections between technology and ecology.

Teachers should evaluate presentations using the established criteria: content addressing the five essential topics (75 points), teamwork (10 points), presentation quality (20 points), and innovative thinking (15 points).

Maintaining Engagement: While only one team competes or presents at a time, it is important to keep all students engaged throughout the hackathon. Teachers can assign specific observation tasks to teams waiting their turn, asking them to note effective strategies or interesting approaches they observe from other teams. This active observation helps maintain focus and promotes learning from peers.

Real-time Feedback: The hackathon format allows for immediate feedback and iteration. Between their two competition sessions, teams have the opportunity to adjust their code based on their first performance. Similarly, teams presenting later in the order can refine their presentations based on feedback provided to earlier groups. Teachers should emphasize this iterative improvement as a valuable part of the learning process.

Celebrating Success: While the hackathon includes competitive elements, teachers should create an atmosphere that celebrates all forms of success. This might include recognition for most innovative solution, best teamwork, most improved performance, or most creative presentation, alongside the overall point-based evaluation. This broader recognition ensures that different types of achievement are valued.

Conclusion & Further Reflexion

The Hackathon Phase represents the **culmination of students' learning journey** through the autonomous vehicle design protocol. By publicly testing their programming solutions and presenting their ideas for sustainable cities, students demonstrate both technical mastery and conceptual understanding. This authentic assessment creates natural opportunities for reflection and growth.

The competition element of the hackathon provides **immediate, concrete feedback on programming solutions**. When a Roobokart successfully navigates complex challenges or fails to respond as expected, students witness the real-world consequences of their code decisions. This tangible feedback often proves more meaningful than abstract evaluation and motivates thoughtful reflection on improvement opportunities.

Similarly, the presentation component encourages students to **articulate connections between technical capabilities and broader societal implications**. By explaining their vision for tomorrow's cities, students practice important communication skills while demonstrating their understanding of how technology shapes our environment. The public discussion format promotes idea cross-pollination, where insights from one team inspire thinking in others.

Throughout the hackathon, teachers should **observe how students respond to challenges, adapt to feedback, and support their teammates**. These observations provide valuable insights into students' growth not only in technical and conceptual understanding but also in collaboration and resilience. The skills developed through this comprehensive protocol—**from programming and design thinking to teamwork and communication**—prepare students for success in an increasingly interconnected and technology-driven world.

Appendix A: Scoring Tables for the City Challenge

Penalties

PENALTY	DESCRIPTION	POINTS
Lane Invasion	Each time the Roobokart invades the opposite lane.	-2
Leaving the Roadway	Each time the Roobokart goes off the roadway.	-3
Vehicle Breakdown	Each time the Roobokart stops without reason and requires driver intervention to restart.	-5
Collision	When the Roobokart hits an obstacle. This penalty can be assigned at most three times per game session.	-5
Noise Pollution	When the Roobokart sounds the horn without reason. This penalty does not include sounds emitted before the vehicle starts or those required by objectives. This penalty can be assigned at most three times per game session.	-2
Incorrect Intersection Crossing Maneuver	When the Roobokart does not correctly resume navigation after completing the intersection crossing phase.	-7
Crossing on Red	When the Roobokart crosses the intersection with a red traffic light.	-5
Low Battery	The Roobokart stops due to a low battery. Battery replacement during the race is possible, and teams may request to be moved to the end of the queue.	-20
Withdrawal from Race	The team withdraws from the race session. This penalty can be assigned only once per game session.	-150
Moving to End of Queue	The team requests to be moved to the end of the queue due to a serious technical problem. Approval of this request is at the discretion of the game referee. This request can only be made before the race session has begun and only once throughout the entire competition.	-20
Non-compliant Roobokart Behavior	Each time the Roobokart commits irregularities not covered by this table.	-10
Non-compliant Team Behavior	Each time the team does not behave in accordance with the regulations or does not adhere to the fundamental principles of fair play during the entire competition.	-20

Note: In case of incorrect intersection crossing (when the Roobokart does not correctly return to its lane to resume navigation), the intersection crossing penalty is applied rather than the lane departure penalty. In this case, the Roobokart resumes the race from the re-start point immediately after the intersection corresponding to the road that the Roobokart unsuccessfully attempted to enter.

Bonuses

BONUS	DESCRIPTION	POINTS ADDED
Right Turn	Each time the Roobokart makes a right turn at an intersection.	+1
Left Turn	Each time the Roobokart makes a left turn at an intersection.	+5
Straight Ahead	Each time the Roobokart continues straight at an intersection.	+3
Acoustic Signal	At least once during the game session, the Roobokart sounds its horn if it detects an obstacle for more than five seconds. This objective can be achieved only once per game session.	+8
Obstacle Detection	The Roobokart stops when an obstacle is detected and resumes when the road is clear. This objective can be achieved at most three times per game session.	+2

Appendix B: Scoring Table for the Ecological Impact Challenge

TOPIC	POINTS
Environmental Sustainability	15 pts
Technology and Innovation	15 pts
Sustainable Mobility	15 pts
Urban Planning	15 pts
Citizen Participation	15 pts
Teamwork	10 pts
Presentation	20 pts
Innovation of Ideas	15 pts
TOTAL	**120 pts**

Appendix C: Competition Procedure

- [ ] Teams will be called to compete in an arbitrary order determined by the teacher.
- [ ] Teams must report to the competition table within 60 seconds of being called or forfeit their race session.
- [ ] If a team is not ready to race, they may request to be moved to the end of the queue (accepting the -20 point penalty) within 60 seconds of being called.
- [ ] Each team participates in two separate two-minute sessions.
- [ ] A team's sessions are not consecutive, allowing for code modifications between sessions.
- [ ] Modified code cannot be tested on the official competition table before the next session.
- [ ] At the beginning of each session, the Roobokart must be positioned at the starting reference point.
- [ ] Timing begins when the Roobokart leaves the starting point and continues during movement, traffic light stops, obstacle management, and intersection crossings.
- [ ] Timing is paused when the Roobokart is off-track due to penalties or stopped because of an obstacle placed by teachers.
- [ ] Final score is calculated as initial points (200) minus penalties plus bonuses across both sessions.