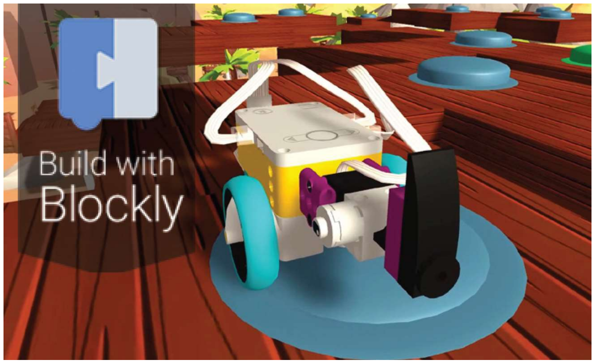
CoderZ™ Content Framework

CoderZ is an engaging self-paced virtual learning platform that provides interactive lessons and activities designed to teach students principles of coding within the exciting and challenging context of robotics.

There are five courses available from which schools/districts may select to implement:

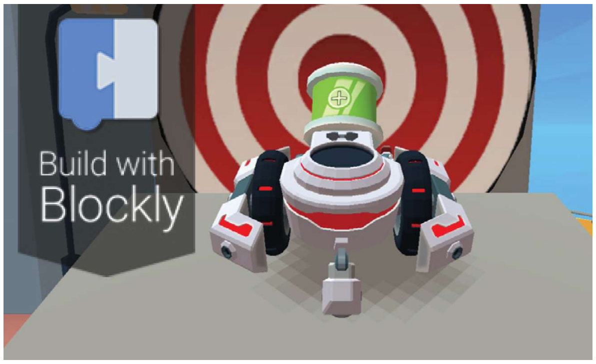
New CoderZ Adventure (Novice)



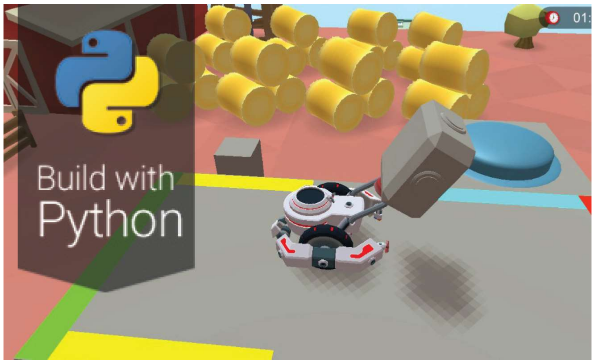
Code Farm (Beginner)

Cyber Robtics 101 (Beginner)

Cyber Robotics 102 (Intermediate)



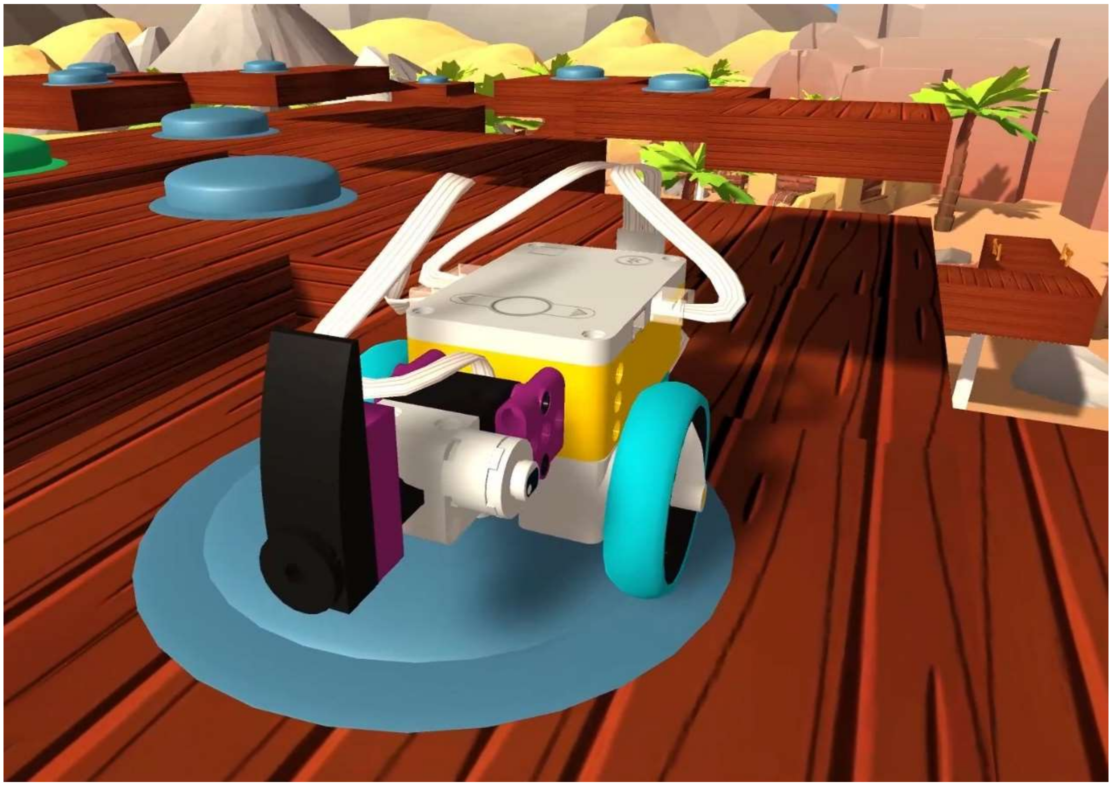
Python Gym (Advanced)



The CoderZ Content Framework provides an overview of the standards, skills, and concepts supported within all CoderZ Materials, including the teacher/course guides and CoderZ's professional development. Each course's content is individually framed for efficient referencing.

TABLE OF CONTENT

The New CoderZ Adventure with LEGO® Education SPIKE™Prime



In this introductory coding course students are introduced to the basic principles of coding within the context of robotics. While participating in a series of increasingly complex tasks students learn about the relationship between hardware and software, the role of mathematical modeling and data in computational problem solving, how basic programming constructs can be combined to create complex algorithms, as well as develop a foundation in key programming practices.

Standards, Skills, and Concepts

CSTA

Computer Systems

1B-CS-01 Describe how internal and external parts of computing devices function to form a system.

1B-CS-02 Model how computer hardware and software work together as a system to accomplish tasks.

Data and Analysis

1B-DA-07 Use data to highlight or propose cause-and-effect relationships, predicate outcomes, or communicate an idea.

Algorithms and Programming

1B-AP-08 Compare and refine multiple algorithms for the same task and determine which is the most appropriate

1B-AP-10 Create programs that include sequences, events, loops, and conditionals.

1B-AP-11 Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.

1B-AP-13 Use an interactive process to plan the development of a program by including others' perspectives and considering user preferences.

1B-AP-15 Test and debug (identify and fix errors) in a program or algorithm to ensure it runs as intended.

Impacts of Computing

1B-IC-18 Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices.

NGSS

3-5-ETS1-1	Define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost.
3-5-ETS1-2	Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.
3-5-ETS1-37	Plan and carry out fair tests in which variables are controlled and failure points are considered to identify aspects of a model or prototype that can be improved.

K-12 Computer Science Framework

Computing Systems

○ Troubleshooting - Check connections and power to resolve common issues. Explain and demonstrate how rebooting a machine is commonly an effective strategy.

Algorithms and Programming

○ Algorithms - Compare algorithms and select the one most appropriate for a specific context/task.

○ Control - Recognize patterns and effectively use loops to enhance the efficiency of writing code.

○ Program Development - Use an iterative process involving design, implementation, and review when developing code.

Impacts of Computing

○ Social Interaction - Analyze how computing technology's facilitation of communication and innovations influences social institutions such as family, education, and the economy.

CC Math

Mathematical Practice

MP1 Make sense of problems and persevere in solving them.

MP2 Reason abstractly and quantitatively.

MP3 Construct viable arguments and critique the reasoning of others.

MP4 Model with mathematics.

MP5 Use appropriate tools strategically.

MP6 Attend to precision.

MP7 Look for and make use of structure.

MP8 Look for and express regularity in repeated reasoning.

21st Century Skills

- Creativity and Innovation
- Critical Thinking and Problem Solving
- Communication
- Collaboration

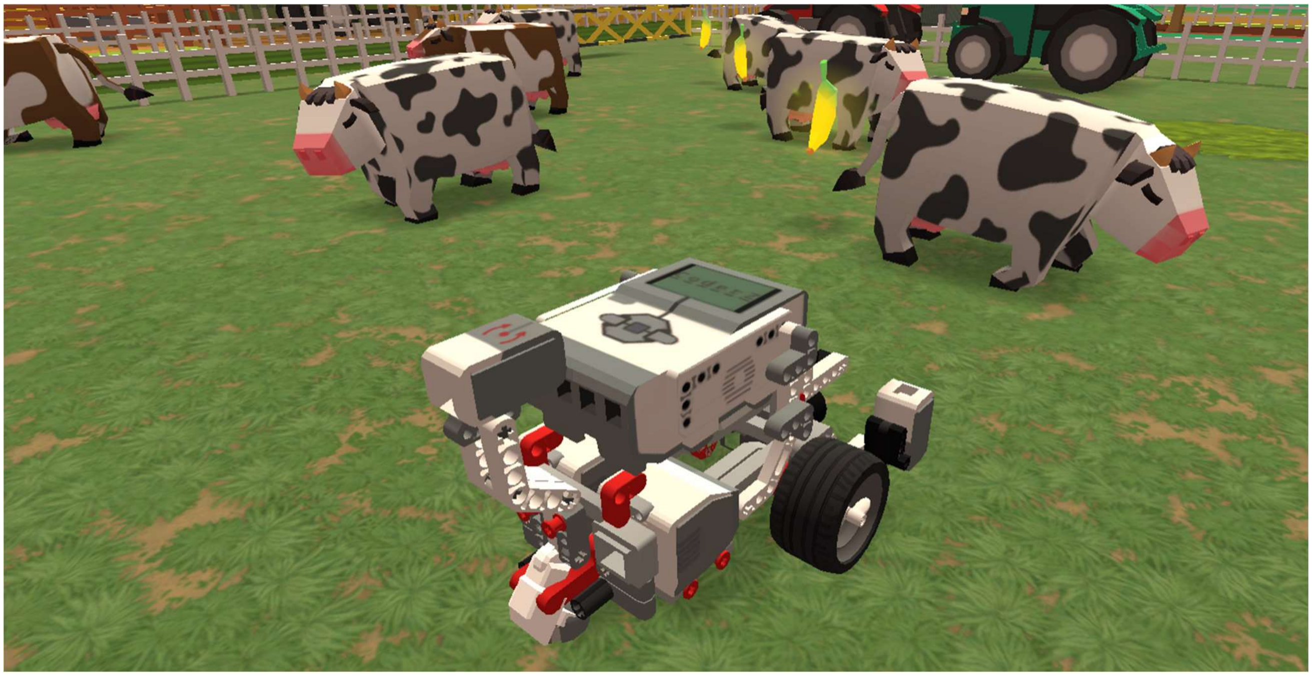
Computational Thinking Skills

- Decomposition
- Pattern Recognition
- Algorithm Design

Robotics Concepts and Skills

- Motion Planning - Direction

Code Farm



The CoderZ **Code Farm** Course is a flexible 40-60 hour course that introduces 5th to 6th grade students to key computer science and robotics concepts through a series of game-like missions and projects. Each lesson is aligned to CSTA, NGSS, and Common Core Standards and includes teacher support for facilitating and assessing student learning.

With accessible videos, built-in classroom activities and discussions, and a discovery-based approach, the course allows students to build their skills and knowledge while engaging in a game-like environment that fosters community in the classroom. Targeted skill-building lessons are paired with open-ended project work, so students can apply what they've learned to a variety of challenging and authentic problems, deepening their understanding while developing cross-cutting cognitive skills.

Student Learning and Standards Alignment in Code Farm

In order to facilitate alignment to a wide range of standards, the student learning in CoderZ’s Code Farm has been organized into a freestanding framework that can be mapped to a wide range of standards relevant in the local context. The expected learning outcomes have been mapped to CSTA and NGSS-ETS and selected CC-ELA standards. These mappings focus on the 5th grade level, but the objectives and associated learning activities can be easily adapted to be more appropriate for older students.

Code Farm Expected Learning Outcomes

Student learning outcomes fall into six main categories. The first two categories, Robotics and Software Development, include discipline-specific content knowledge aligned closely to the CSTA K12 standards. The last four categories, Ethics, Creative Design, Collaboration, and Communication, are cross-cutting skills that map to CSTA standards as well as other commonly used standards such as NGSS, ISTE, 21st Century Skills, and Common Core ELA.

The curriculum also includes multiple opportunities to address key standards in Math and Science. While not formally part of the expected learning outcomes, these opportunities have been separately identified to allow teachers to highlight content appropriate for their classes.

Robotics		
	Modeling Computing Systems Identify the different hardware and software components of a computing system, and explain how they work together to perform tasks	**CSTA 1B-CS-01** **CSTA 1B-CS-02**
	Sending and Receiving Information Model how computing systems collect data from sensors, send commands to hardware components, and share information across different computing systems in order to coordinate a task.	**CSTA 1B-CS-04**
	Using computing systems Use appropriate troubleshooting strategies for when the system is not working properly, as well as employing safe practices around password security and responsible use.	**CSTA 1B-CS-03** **CSTA 1B-NI-05**
Software Development		
	Software Development Process Use a structured and iterative software development	**CSTA 1B-AP-11** **CSTA 1B-AP-13**

	process that includes a goal, a step-by-step plan, and opportunities for feedback and improvement.	
	Writing Algorithms Use sequencing, loops, conditionals, events, and variables to control a program's behavior.	**CSTA 1B-AP-09** **CSTA 1B-AP-10**
	Debugging Identify bugs in a program and use appropriate strategies to investigate their causes and correct them.	**CSTA 1B-AP-15**
Ethics		
	Using Others' Work Integrate the work of other creators into a digital artifact, respecting the creator's copyright. Identify common licenses and how they allow media to be used.	**CSTA 1B-AP-12** **CSTA 1B-AP-14** **CSTA 1B-IC-21**
	Accessibility and Inclusion Recognize the barriers to access and inclusion in a given technology ,and design products that are accessible and inclusive for a wide range of users.	**CSTA 1B-IC-19**
	Social Impact Discuss the potential impacts of a technology and how those impacts vary from person to person.	**CSTA 1B-IC-18**
Creativity and Design		
	Criteria and Constraints Identify appropriate criteria and constraints for a project, and evaluate and improve products according to how well they conform to those requirements.	**NGSS 3-5-ETS1-1** **CSTA 1B-AP-08**
	Using resources Use a wide range of resources when designing a product, and combine ideas from those resources in new and effective ways.	**CSTA 1B-AP-12**
	Generating Ideas Generate a wide range of relevant and distinct design solutions that are new or unique, and elaborate on them with specific detail.	**NGSS 3-5-ETS1-2**
Collaboration		
	Shared Understanding Constructively and respectfully manage disagreement, actively seek input from all members, and maintain clear	**CCSS.ELA-LITERACY.SL.5.1** **CSTA 1B-AP-17**

	documentation of team decisions	
	Managing Team Work Establish and honor collective goals and timelines, distribute tasks equitably, while taking into account each member's unique traits and resources	**CSTA 1B-AP-12** **CSTA 1B-AP-16**
	Exchanging ideas Seek out and provide help or feedback from others when appropriate, and evaluate and incorporate other's ideas into a project	**CSTA 1B-AP-13** **CSTA 1B-IC-20**
Communication		
	Explaining a Process Use presentations, demonstrations, and written reports to explain design choices, and maintain appropriate documentation, including code comments.	**CSTA 1B-AP-17** **CCSS.ELA-LITERACY.SL.5.4** **CCSS.ELA-LITERACY.SL.5.5**
	Supporting an Idea Justify a claim or point of view with evidence, organize data to highlight a relationship or claim, or make a prediction.	**CSTA 1B-NI-06** **CSTA 1B-NI-07**
	Understanding Others Find and use relevant information from text, make inferences and connections, and compare information from multiple sources.	**CCSS.ELA-LITERACY.RI.5.1** **CCSS.ELA-LITERACY.RI.5.3** **CCSS.ELA-LITERACY.RI.5.10**

CSTA Standards Alignment

Code Farm fully addresses all CSTA standards in the 1B (3-5) grade band, as shown in the following heat map.

CSTA Grades 3-5

	L0	L1	L2	L3	L4	C1	P1	L5	L6	L7	L8	C2	P2	P3
1B-CS-01		🟩	▨	▨	▨	🟩	🟩	▨	▨	▨	▨	▨		▨
1B-CS-02		▨	▨	▨	🟩	🟩	🟩	🟩	▨	▨	▨	🟩		▨
1B-CS-03	🟩	▨	▨	▨	▨	▨		▨	▨	▨	▨	▨		▨
1B-NI-04													🟩	
1B-NI-05	🟩												🟩	
1B-DA-06						🟩						🟩		🟩
1B-DA-07						🟩						🟩		🟩
1B-AP-08						▨		▨	🟩	▨	▨	🟩		🟩
1B-AP-09											🟩	🟩		🟩
1B-AP-10	▨	▨	▨	▨	▨	▨		▨	🟩	🟩	▨	🟩		🟩
1B-AP-11				🟩	▨	🟩		▨	▨	▨	▨	▨		▨
1B-AP-12														🟩
1B-AP-13												🟩		🟩
1B-AP-14							🟩						🟩	🟩
1B-AP-15			🟩	▨	▨	▨		▨	▨	▨	▨	▨		▨
1B-AP-16						▨						▨		🟩
1B-AP-17					🟩	🟩		▨	▨	▨	▨	▨	🟩	🟩
1B-IC-18							🟩						🟩	
1B-IC-19							🟩						🟩	
1B-IC-20							🟩						🟩	🟩
1B-IC-21							🟩						🟩	

NGSS Alignment

Code Farm fully addresses all NGSS Engineering Design performance expectations in the 3-5 grade band, as shown in the following heat map. Extensions to support the 6-8 grade band are also available.

NGSS Grades 3-5

	L0	L1	L2	L3	L4	C1	P1	L5	L6	L7	L8	C2	P2	P3
3-5-ETS1-1							🟩					🟩	🟩	🟩
3-5-ETS1-2							🟩					🟩	🟩	🟩
3-5-ETS1-3						🟩						🟩		🟩

CC-Math Alignment - Grade Specific Standards

CCSS.MATH.CONTENT.5.OA.A.
Write and interpret numerical expressions.

As students code they must represent mathematical relationships (e.g. addition, multiplication) as expressions. Within the context of code, these expressions use parentheses to indicate the order of operations.

CCSS.MATH.CONTENT.5.OA.B
Analyze patterns and relationships.

As students make use of structure to simplify their programming, they must abstract out the particular values of numbers to identify patterns. For example, students may realize that the value three must be added to a variable in each iteration of a loop without calculating the variable's specific value each time.

CCSS.MATH.CONTENT.5.NBT.A
Understand the place value system.

In order to create algorithms with sufficient precision to complete a task, students must work with decimal values as specific as one thousandth, using place value to compare the magnitude of different choices and calibrate their robots to move as intended.

CCSS.MATH.CONTENT.5.MD.A
Convert like measurement units within a given measurement system.

Students are presented with multiple options for specifying measurements (e.g. seconds versus milliseconds) and must be able to both convert between them and determine the most appropriate option for their task.

CCSS.MATH.CONTENT.5.MD.B
Represent and interpret data.

As students test different algorithmic solutions, they must collect and represent data, then use their interpretations of that data to make and defend a decision for which algorithm is most appropriate.

CC-Math Alignment - Practices

Beyond the grade specific standards addressed in the curriculum, Code Farm provides extensive support for the CCSS Standards for Mathematical Practice.

CCSS.MATH.PRACTICE.MP1 Make sense of problems and persevere in solving them.
In each lesson, students are tasked with problems of increasinging complexity that they must analyze and overcome using the algorithmic tools at their disposal. Within the curriculum, 'failure' normalized, and students are encouraged to see setbacks and debugging as a normal part of the programming process.

CCSS.MATH.PRACTICE.MP2 Reason abstractly and quantitatively.
Programmers must move fluidly between the abstract and concrete, and Code Farm prompts students to use the concrete quantitative problems in the sample simulations to abstract out general algorithmic rules and structures. In doing so, students learn to think flexibly about problems at various layers of abstraction.

CCSS.MATH.PRACTICE.MP3 Construct viable arguments and critique the reasoning of others.
As student code becomes more complex, they are encouraged to evaluate various approaches according to given criteria and constraints. In the culminating projects, students must present their reasoning, explaining the differences between approaches and the advantages of the one they chose.

CCSS.MATH.PRACTICE.MP4 Model with mathematics.
Understanding the relationships between speed, momentum, and other physics concepts is key to navigating with robots. In Code Farm, students use the data they collect to model these relationships, using these models to make predictions about robot movement and justifying algorithmic choices.

CCSS.MATH.PRACTICE.MP5 Use appropriate tools strategically.
Open problems and projects give students opportunities to choose from multiple algorithmic constructs and overall design solutions. Students justify these choices based on quantitative criteria that align to the overall goals of the problem at hand, and explain how different approaches can optimize for different priorities.

CCSS.MATH.PRACTICE.MP6 Attend to precision.
The debugging process demonstrates how small errors in code can have a big impact on how the program runs. Students learn to ensure their code is error free, debugging it when necessary, and to pay attention to hardware limitations that affect the robot's ability to maneuver precisely through the simulation.

CCSS.MATH.PRACTICE.MP7 Look for and make use of structure.
Computing highlights the role that structure plays in problem solving, providing algorithmic constructs designed to clarify and simplify the coding process. Students recognize and use patterns in code, then apply and combine the appropriate programming constructs to develop solutions to complex problems.

CCSS.MATH.PRACTICE.MP8 Look for and express regularity in repeated reasoning.
Leveraging repetition is at the core of programming. In Code Farm, students must use repeated reasoning to solve similar challenges, then abstract out the general programming 'rules' for the various situations they encounter. In addition, programming constructs such as the repeat loop reinforce the power of repetition in computing.

CC-ELA Alignment

Computing and robotics provide students with opportunities to engage with many varieties of text. Whether reading user profiles, looking up technical information, or researching previous innovations, students must find, interpret, and use relevant information. These samples provide models when students document and present their own designs.

Key standards addressed at the Grade 5 level include:

CCSS.ELA-LITERACY.RI.5.1
Quote accurately from a text when explaining what the text says explicitly and when drawing inferences from the text.

CCSS.ELA-LITERACY.RI.5.2
Determine two or more main ideas of a text and explain how they are supported by key details; summarize the text.

CCSS.ELA-LITERACY.RI.5.3
Explain the relationships or interactions between two or more individuals, events, ideas, or concepts in a historical, scientific, or technical text based on specific information in the text.

CCSS.ELA-LITERACY.SL.5.1
Engage effectively in a range of collaborative discussions (one-on-one, in groups, and teacher-led) with diverse partners on grade 5 topics and texts, building on others' ideas and expressing their own clearly.

CCSS.ELA-LITERACY.SL.5.2
Summarize a written text read aloud or information presented in diverse media and formats, including visually, quantitatively, and orally.

CCSS.ELA-LITERACY.SL.5.4
Report on a topic or text or present an opinion, sequencing ideas logically and using appropriate facts and relevant, descriptive details to support main ideas or themes; speak clearly at an understandable pace.

CCSS.ELA-LITERACY.SL.5.5
Include multimedia components (e.g., graphics, sound) and visual displays in presentations when appropriate to enhance the development of main ideas or themes.

CCSS.ELA-LITERACY.SL.5.6
Adapt speech to a variety of contexts and tasks, using formal English when appropriate to task and situation. (See grade 5 Language standards 1 and 3 here for specific expectations.)

CCSS.ELA-LITERACY.RI.5.10
By the end of the year, read and comprehend informational texts, including history/social studies, science, and technical texts, at the high end of the grades 4-5 text complexity band independently and proficiently.

ISTE Alignment

As a robotics and computing curriculum, the focus of Code Farm is on the creation of new technologies, rather than their use. However, as students must be sophisticated users of technology to innovate, there is extensive coverage of the ISTE standards.

Empowered Learner
In creating and implementing new digital artifacts, students have the freedom to set their own goals and decide how to achieve them. Self-differentiation is facilitated as students set their own goals in mission, deciding what ‘success’ can mean for them, in addition to the extra challenge missions available. Rubric-based projects allow students to decide how to demonstrate learning in a way that is personally meaningful to them.

Digital Citizen
Safety and responsibility are integrated throughout the curriculum. Students discuss ethical issues that arise with new technologies and reflect on the impact of their own creations. They are explicitly assessed on responsible and ethical use of intellectual property, as well as their reflections on how physical and digital security measures maintain safety and privacy online.

Knowledge Constructor
Use of resources is an authentic practice in both computing and robotics, and students are asked to synthesize knowledge from documentation, tips, instructional videos, as well as their own explorations. They then collaborate with others to test and refine this new knowledge as they work to achieve various goals.

Innovative Designer
Multiple projects challenge students to design technological solutions to various problems. Students are pushed to innovate as they consider the needs of different users, develop criteria, and work within the given constraints to create products that integrate and refine various technological features.

Computational Thinker
As they work through the various missions and projects, students engage in a wide variety of programming tasks, analyzing, testing, and comparing various algorithmic solutions. They break problems into manageable pieces, then use patterns and abstraction to identify and combine appropriate programming constructs.

Creative Communicator
Students express their ideas in multiple ways throughout the curriculum: in informal discussions, design presentations, and digital artifacts. These works include individual, collaborative, and repurposed creations, and students are expected to consider audience, purpose, and context as they communicate.

Global Collaborator
Students collaborate on multiple projects throughout the curriculum, taking on a variety of roles and responsibilities. They exchange feedback from those without and outside of their teams, and are expected to consider the perspectives of users with a wide variety of perspectives.

Developing 21st Century Skills

Collaboration, creative problem solving, social awareness are integral to robotics. Students work together on projects that challenge them to think in new ways, understand the different needs and perspectives of others, and work to respond to those needs effectively, considering the social impact of existing technology and of the products they develop.

Collaboration

Students collaborate on multiple projects throughout the curriculum, taking on a variety of roles and responsibilities. They exchange feedback from those without and outside of their teams, and are expected to consider the perspectives of users with a wide variety of perspectives. Collaboration is integrated into most lessons, but is also explicitly taught and scaffolded within the context of complex projects. Students are expected to collaborate with peers in solving problems, creating digital artifacts, and refining knowledge.

Communication

Robotics and computing provide a unique lens on communication, as students are expected to communicate not only with natural language in interactions with others, but also with computing language, to instruct a robot. This differentiation prompts reflection on the complexity of communication, and how communication styles change in response to different audiences and purposes. Students must communicate their ideas both in code and natural language, as well as give formal multimedia presentations on their design process.

Critical Thinking

As students encounter various problems and perspectives, they must analyze complex systems, consider tradeoffs between solutions, and use various resources to determine their path forward. In defending their design choices, they articulate their priorities and how those are reflected in a final product. These types of problem solving skills are further honed as students consider the perspectives of those different from themselves, and how those perspectives might impact a design solution’s effectiveness or usability.

Creativity

Creativity is at the heart of robotics, as engineers design and create innovative products that push technology forward. Design challenges and missions provide a wide range of opportunities for students to practice creativity, and the creative process is scaffolded through exposure to sample designs for inspiration, criteria and constraints for success, and a step-by-step design process. Creative thinking is framed as a skill that can be learned, and students are given clear instructions for its development.

Standards, Skills, and Concepts

CSTA

Computer Systems

1B-CS-01 Describe how internal and external parts of computing devices function to form a system.

1B-CS-02 Model how computer hardware and software together as a system to accomplish tasks.

1B-CS-03 Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies.

Algorithms and Programming

1B-AP-08 Compare and refine multiple algorithms for the same task and determine which is the most appropriate.

1B-AP-10 Create programs that include sequences, events, loops, and conditionals.

1B-AP-11 Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.

1B-AP14 Observe intellectual property rights and give appropriate attribution when creating or remixing programs.

1B-AP-15 Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.

1B-AP-17 Describe choices made during program development using code comments, presentations, and demonstrations.

Impacts of Computing

1B-AP-17 Describe choices made during program development using code comments, presentations, and demonstrations.

1B-IC-19 Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users.

1B-IC-20 Seek diverse perspectives for the purpose of improving computational artifacts.

1B-IC-21 Use public domain or creative commons media, and refrain from copying or using material created by others without permission.

Networks & the Internet

1B-NI-04 Model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination.

1B-NI-05 Discuss real-world cybersecurity problems and how personal information can be protected.

NGSS

3-5-ETS1-1 Define a simple design problem that can be solved through the development of an object, tool, process, or system and includes several criteria for success and constraints on materials, time, or cost.

3-5-ETS1-2 Generate and compare multiple solutions to a problem based on how well they meet the criteria and constraints of the design problem.

K-12 Computer Science Framework

Computing Systems

- Devices - Examine the interaction between humans and computing devices. Evaluate and analyze the advantages, disadvantages, and recognize the unintended consequences.
- Troubleshooting - Deploy a structured process to troubleshoot problems within a system to ensure potential solutions are not overlooked and simple issues are resolved.

Algorithms and Programming

- Algorithms - Design, test, and debug algorithms that are readable and easy to follow.
- Variables - Understand and use variables to represent and process data to produce varying outputs.
- Control - Use loops, conditions, and other control structures to create more complex programs.
- Modularity - Use functions to make code easier to reuse and read.

Networks and the Internet

- Cybersecurity - Explain the importance of protecting information sent and received across networks from unauthorized access and modification (encryption and Hypertext Transfer Protocol Secure - HTTPS).

Data and Analysis

- Collection - Discuss how using sensors with a robot to collect data regarding its environment and make decisions about the next steps to take is similar to the human process and how robots and humans interact.

CC Math

Mathematical Practice

MP1 Make sense of problems and persevere in solving them.

MP2 Reason abstractly and quantitatively.

MP3 Construct viable arguments and critique the reasoning of others.

MP4 Model with mathematics.

MP5 Use appropriate tools strategically.

MP6 Attend to precision.

MP7 Look for and make use of structure.

MP8 Look for and express regularity in repeated reasoning.

21st Century Skills

Learning and Innovation Skills

- Creativity and Innovation
- Critical Thinking and Problem Solving
- Communication and Collaboration

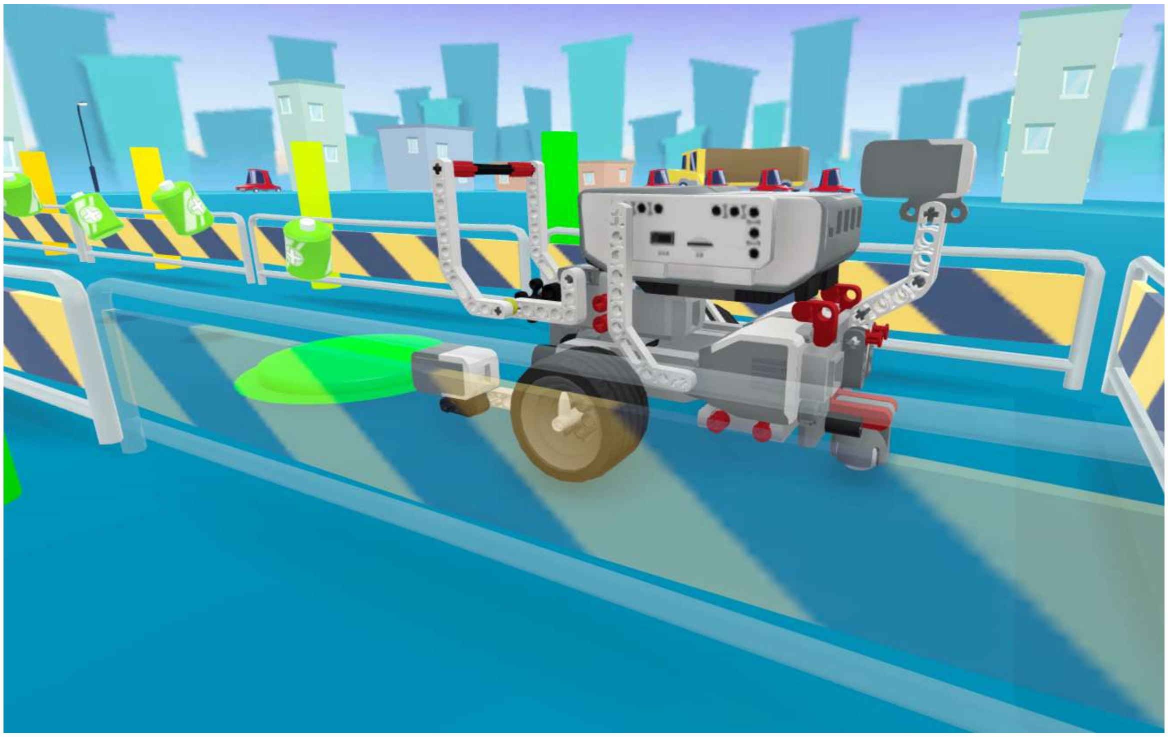
Information, Media and Technology Skills

- Information Literacy
- Information and Communications Technology (ICT)

Life and Career Skills

- Flexibility and Adaptability
- Initiative and Self-direction
- Social and Cross-Cultural Skills
- Productivity and Accountability
- Leadership and Responsibility

Cyber Robotics 101



In this beginner’s coding course students are introduced to the principles of coding within the context of robotics. While engaged in a series of activities with increasing complexity students learn the essential principles of robotics including motion planning, motion planning with sensors, the relationship between hardware and software, the role of mathematical modeling and data in computational problem solving, the importance of quality planning, as well as develop the ability to execute good programming practices. CoderZ and supporting integrated content is sequenced to meet STEAM objectives and is designed to provide students authentic learning opportunities to increase interest and engagement while moving from acquisition of knowledge to transference of learning.

Standards, Skills, and Concepts

CSTA

Computer Systems

2-CS-02 Design projects that combine hardware and software components to collect and exchange data.

Algorithms and Programming

2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-14 Create procedures with parameters to organize code and make it easier to use.

2-AP-17 Systematically test and refine programs using a range of test cases.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

2-AP-19 Document programs in order to make them easier to follow, test, and debug.

Impacts of Computing

2-IC-20	Compare tradeoffs associated with computing technologies that affect people’s everyday activities and career options.
2-IC-21	Discuss issues of bias and accessibility in the design of existing technologies.
2-IC-23	Describe tradeoffs between allowing information to be public and keeping information private and secure.

NGSS

MS-ETS1-2	Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.
MS-ETS1-3	Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.

K-12 Computer Science Framework

Computing Systems

- Devices - Examine the interaction between humans and computing devices. Evaluate and analyze the advantages, disadvantages, and recognize the unintended consequences.
- Troubleshooting - Deploy a structured process to troubleshoot problems within a system to ensure potential solutions are not overlooked and simple issues are resolved.

Algorithms and Programming

- Algorithms - Design, test, and debug algorithms that are readable and easy to follow.
- Variables - Understand and use variables to represent and process data to produce varying outputs.
- Control - Use loops, conditions, and other control structures to create more complex programs.
- Modularity - Use functions to make code easier to reuse and read.

Networks and the Internet

- Cybersecurity - Explain the importance of protecting information sent and received across networks from unauthorized access and modification (encryption and Hypertext Transfer Protocol Secure - HTTPS).

Data and Analysis

- Collection - Discuss how using sensors with a robot to collect data regarding its environment and make decisions about the next steps to take is similar to the human process and how robots and humans interact.

CC Math

Mathematical Practice

MP1 Make sense of problems and persevere in solving them.

MP2 Reason abstractly and quantitatively.

MP3 Construct viable arguments and critique the reasoning of others.

MP4 Model with mathematics.

MP5 Use appropriate tools strategically.

MP6 Attend to precision.

MP7 Look for and make use of structure.

MP8 Look for and express regularity in repeated reasoning.

21st Century Skills

Learning and Innovation Skills

- Creativity and Innovation
- Critical Thinking and Problem Solving
- Communication and Collaboration

Information, Media and Technology Skills

- Information Literacy
- Information and Communications Technology (ICT)

Life and Career Skills

- Flexibility and Adaptability
- Initiative and Self-direction
- Social and Cross-Cultural Skills
- Productivity and Accountability
- Leadership and Responsibility

Computational Thinking Skills

- Decomposition
- Pattern Recognition
- Algorithm Design
- Abstraction

Robotics Concepts and Skills

- Motion Planning
  - Direction
  - Power/Speed
  - Distance
  - Duration
  - Arm
- Sensor-based Motion Planning
- Motors and Optical Encoders
- Sensors and Controllers
  - Touch - collision
  - Gyroscope - rotation
  - Ultrasonic - distance
  - Light/Color - surface color/brightness

Cyber Robotics 102

Cyber Robotics 102 (CR102) is a continuation of the Cyber Robotics 101 course. This course teaches STEM and coding topics using gamification in a physics-based environment. Through a series of activities with increasing complexity CR102 introduces autonomous systems, teaches scanning and mapping an environment, error correction methods, and different system control algorithms. Cyber Robotics 102 provides a deeper coding experience with more opportunities to use conditionals, variables, functions, etc. CoderZ and supporting integrated content is sequenced to meet STEAM objectives and is designed to provide students authentic learning opportunities to increase interest and engagement while moving from acquisition of knowledge to transference of learning. By the end of the course students will understand the forces of physics acting on robots and the influence those forces have on a robot’s performance. Students will effectively control and program a robot to interact with objects around it and safely navigate through changing environments.

Standards, Skills, and Concepts

CSTA

Computer Systems

2-CS-02 Design projects that combine hardware and software components to collect and exchange data.

Algorithms and Programming

2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-14 Create procedures with parameters to organize code and make it easier to use.

2-AP-17 Systematically test and refine programs using a range of test cases.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

2-AP-19 Document programs in order to make them easier to follow, test, and debug.

Impacts of Computing

2-IC-20 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

2-IC-21 Discuss issues of bias and accessibility in the design of existing technologies.

2-IC-23 Describe tradeoffs between allowing information to be public and keeping information private and secure.

NGSS

MS-ETS1-2 Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.

MS-ETS1-3 Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.

K-12 Computer Science Framework

Computing Systems

- Devices - Examine the interaction between humans and computing devices. Evaluate and analyze the advantages, disadvantages, and recognize the unintended consequences.
- Troubleshooting - Deploy a structured process to troubleshoot problems within the system to ensure potential solutions are not overlooked and simple issues are resolved.
- Algorithms and Programming
- Algorithms - Design, test, and debug algorithms that are readable and easy to follow.
- Variables - Understand and use variables to represent and process data to produce varying outputs.
- Control - Use loops, conditions, and other control structures to create more complex programs.
- Modularity - Use functions to make code easier to reuse and read.

Networks and the Internet

- Cybersecurity - Explain the importance of protecting information sent and received across networks from unauthorized access and modification (encryption and Hypertext Transfer Protocol Secure - HTTPS).

Data and Analysis

- Collection - Discuss how using sensors with a robot to collect data regarding its environment and make decisions about the next steps to take is similar to the human process and how robots and humans interact.

CC Math

Mathematical Practice

MP1 Make sense of problems and persevere in solving them.

MP2 Reason abstractly and quantitatively.

MP3 Construct viable arguments and critique the reasoning of others.

MP4 Model with mathematics.

MP5 Use appropriate tools strategically.

MP6 Attend to precision.

MP7 Look for and make use of structure.

MP8 Look for and express regularity in repeated reasoning.

21st Century Skills

Learning and Innovation Skills

- Creativity and Innovation
- Critical Thinking and Problem Solving
- Communication and Collaboration

Information, Media and Technology Skills

- Information Literacy
- Information and Communications Technology (ICT)

Life and Career Skills

- Flexibility and Adaptability
- Initiative and Self-direction
- Social and Cross-Cultural Skills
- Productivity and Accountability
- Leadership and Responsibility

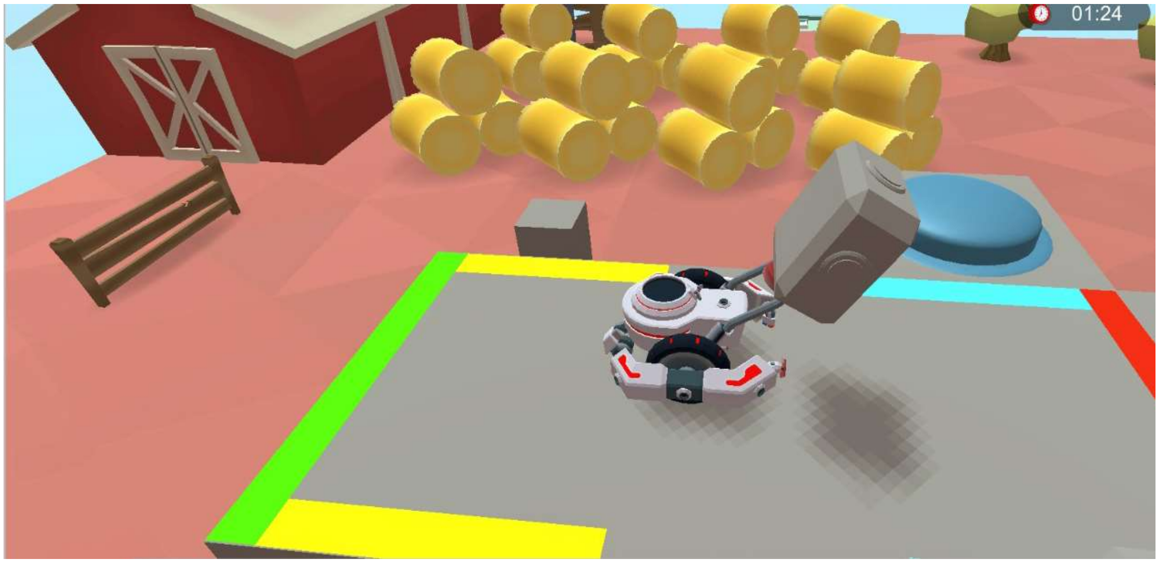
Computational Thinking Skills

- Decomposition
- Pattern Recognition
- Algorithm Design
- Abstraction

Robotics Concepts and Skills

- Motion Planning
    - Direction
    - Power/Speed
    - Distance
    - Duration
    - Brake
    - Arms
- Sensor-based Motion Planning
- Motors and Optical Encoders
- Sensors and Controllers
    - Touch - collision
    - Gyroscope - rotation and tilts (x, y, z)
    - Ultrasonic - distance
    - Light/Color - surface color/brightness

Python Gym



Python Gym is an advanced course providing teachers and students the opportunity to code in Python, a text-based computer language, using the CoderZ physically-based environment introduced in Cyber Robotics 102. While engaged in a series of activities with increasing complexity students learn the essential principles of text-based coding including syntax and debugging and continue to develop best programming practices for robotics including; motion planning, motion planning with sensors, physics, computational problem solving, and the importance of quality planning. Python Gym provides opportunities to use conditionals, variables, functions, modules and more.

CoderZ and supporting integrated content is sequenced to meet STEAM objectives and is designed to provide students authentic learning opportunities to increase interest and engagement while moving from acquisition of knowledge to transference of learning.

Standards, Skills, and Concepts

CSTA

Computer Systems

2-CS-02 Design projects that combine hardware and software components to collect and exchange data.

3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.

Algorithms and Programming

2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-14 Create procedures with parameters to organize code and make it easier to use.

2-AP-17 Systematically test and refine programs using a range of test cases.

2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

2-AP-19 Document programs in order to make them easier to follow, test, and debug.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

3B-AP-11 Evaluate algorithms in terms of their efficiency, correctness, and clarity.

3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

3B-AP-23 Evaluate key qualities of a program through a process such as code review.

Impacts of Computing

2-IC-20 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

2-IC-21 Discuss issues of bias and accessibility in the design of existing technologies.

2-IC-23 Describe tradeoffs between allowing information to be public and keeping information private and secure.

3A-IC-24 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

3A-IC-28 Explain the beneficial and harmful effects that intellectual property laws can have on innovation.

3A-IC-30 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics.

3B-IC-27 Predict how computational innovations that have revolutionized aspects of our culture might evolve.

NGSS

MS-ETS1-2	Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.
MS-ETS1-3	Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.

K-12 Computer Science Framework

Computing Systems

- Devices – Examine
- e the interaction between humans and computing devices. Evaluate and analyze the advantages, disadvantages, and recognize the unintended consequences.
- Troubleshooting - Deploy a structured process to troubleshoot problems within the system to ensure potential solutions are not overlooked and simple issues are resolved.
- Hardware and Software - Explore the interactions between hardware, software and user of a computing system and evaluate the flow of information.

Algorithms and Programming

- Algorithms - Design, test, and debug algorithms that are readable and easy to follow. Evaluate and select algorithms based on performance, reusability, and ease of implementation.

- Variables - Understand and use variables to represent and process data to produce varying outputs.
  Use data structures to effectively manage program complexity (based on functionality, storage, and performance tradeoffs).

- Control - Use loops, conditions, and other control structures to create more complex programs.
  Evaluate control structures and control structure combinations considering the tradeoffs related to implementation, readability, and program performance.

- Modularity - Use functions to make code easier to reuse and read.
  Use modules to develop and manage complex tasks/programs.

Networks and the Internet

- Cybersecurity - Explain the importance of protecting information sent and received across networks from unauthorized access and modification (encryption and Hypertext Transfer Protocol Secure - HTTPS).

Data and Analysis

- Collection - Discuss how using sensors with a robot to collect data regarding its environment and make decisions about the next steps to take is similar to the human process and how robots and humans interact.

CC Math

Mathematical Practice

MP1 Make sense of problems and persevere in solving them.

MP2 Reason abstractly and quantitatively.

MP3 Construct viable arguments and critique the reasoning of others.

MP4 Model with mathematics.

MP5 Use appropriate tools strategically.

MP6 Attend to precision.

MP7 Look for and make use of structure.

MP8 Look for and express regularity in repeated reasoning.

21st Century Skills

Learning and Innovation Skills

- Creativity and Innovation
- Critical Thinking and Problem Solving
- Communication and Collaboration

Information, Media and Technology Skills

- Information Literacy
- Information and Communications Technology (ICT)

Life and Career Skills

- Flexibility and Adaptability
- Initiative and Self-direction
- Social and Cross-Cultural Skills
- Productivity and Accountability
- Leadership and Responsibility

Computational Thinking Skills

- Decomposition
- Pattern Recognition
- Algorithm Design
- Abstraction

Robotics Concepts and Skills

- Motion Planning
    - Direction
    - Power/Speed
    - Distance
    - Duration
    - Brake
    - Arms
- Sensor-based Motion Planning
- Motors and Optical Encoders
- Sensors and Controllers
    - Touch - collision
    - Gyroscope - rotation and tilts (x, y, z)
    - Ultrasonic - distance
    - Light/Color - surface color/brightness