LESSON PLAN 5: CODING AND AUTONOMOUS FEATURES IN AUTOMOBILES

– from the Goodman School of Mines at Laurentian University –

DESCRIPTION

In this lesson, students will explore the concept of autonomous robots and their role in technologies like self-driving vehicles. They will learn how bats use echolocation to navigate and how these principles are applied to robotics and autonomous systems. Key topics include spatial awareness, obstacle avoidance, and logical thinking related to robot navigation.

To support teachers, infographics and presentations are provided to guide instruction and enhance understanding. Students will engage in hands-on block coding activities to program robots for obstacle avoidance, testing and refining their code in a controlled environment. Curriculum outcomes related to coding, robotics, and real-world applications of autonomous systems have been clearly identified to ensure alignment with educational goals.

OBJECTIVES

- ☆ Understand the concept of autonomous robots and their application in autonomous vehicles.
- ☆ Learn how bats use echolocation with ultrasonic sound to navigate.
- ☆ Explore the application of echolocation principles in robotics and autonomous vehicles.
- ☆ Understand the importance of spatial awareness and obstacle avoidance in robotics.
- ☆ Develop logical thinking skills related to obstacle avoidance.
- ☆ Gain basic knowledge of block coding and its application in programming robots.
- ☆ Obtain hands-on coding experience to perform obstacle avoidance tasks.
- ☆ Test and refine coded programs in a controlled environment.
- ☆ Connect learned concepts to real-world applications in robotics and autonomous systems.

Note: This activity uses the mBot from MakeBlock, so the term "mBot" will be used instead of "robot".

MATERIALS

mBot kit(s)

- 4 AA Batteries (for the mBot)
- Tablet(s)/Phone(s)/Personal Computer(PC) with mBlock app installed
  - mBlock app for Web/PC/Mac/Chrome: https://mblock.cc/pages/downloads
  - mBlock app for Android (Google Play, mBlock apk)
  - mBlock app for iOS (iOS App Store)

mBlock mobile app
Learn coding on phones and tablets

Android
Android 6.0 +
(ARM-based devices only.
X86 Android not supported)

iOS
iOS 10.0 +

- Small Movable Barriers (eg: Mega Bloks, wooden blocks)
- Perimeter wall (4-6 pieces wooden 2" x 4", 2ft each)
- Clue Card Decks and Challenge Solutions (available at electrifyingthefuture.ca)
- Computer and Projector for presentation
- "Coding and Autonomous Features in Automobiles" presentation (available at electrifyingthefuture.ca)
- Seeing with Sound & Understanding mBlock infographics

ELECTRIFYING THE FUTURE

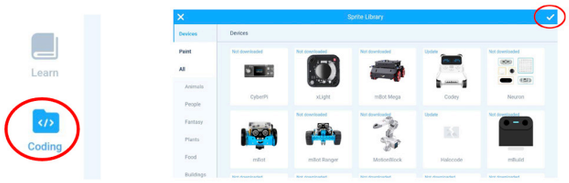
SET-UP

electrifyingthefuture.ca

- Assemble mBot and install batteries.
- Install mBlock app on devices.

  Tablet/PC/Phone :
  - Install mBlock app
  - For Coding
    - » Open mBlock App
    - » Select Coding
    - » Select mBot as device and click the tick on the top-right corner.

WATCH HERE!



Now Tablet/PC/Phone is ready for coding.

- **Perimeter wall:** Place 4 pieces of perimeter wall blocks in a rectangular shape (or six in a hexagonal shape) and attach the corners to prevent movement when hit by the mBot.
- **Small Movable Barriers:** Adjust the height of the barrier structure to be approximately equal to the height of the mBot.

OVERVIEW OF LESSON TASKS

#	Task	Materials
1	Introduction and Welcoming	Instructor PC with slides
2	Scientific Background (Robots and Echolocation)	
3	Echolocation Activity	
4	Scientific Background (Ultrasonic Sensors in Cars)	
5	mBlock Software Introduction	Clue Card Decks
6	Challenge1-Discussion	Clue Card Decks, Task sheet #1
7	Challenge 1-Coding	mBot, mBlock app, Laptop/Tablet, Task sheet #1
8	Challenge 1-Testing	mBot, mBlock app, Laptop/Tablet
9	Challenge 2-Discussion	Clue Card Decks, Task sheet #2
10	Challenge 2-Coding	mBot, mBlock app, Laptop/Tablet, Task sheet #2
11	Challenge 2-Testing	mBot, mBlock app, Laptop/Tablet
12	Challenge 3-Discussion	Clue Card Decks, Task sheet #3
13	Challenge 3-Coding	mBot, mBlock app, Laptop/Tablet, Task sheet #3
14	Challenge 3-Testing	mBot, mBlock app, PC/Laptop/Tablet

*Adjust the lesson timing based on participant experience. If more time is needed for coding, skip certain activities as noted.

PRESENTATION GUIDE, ACTIVITIES & CHALLENGES

This lesson plan is designed to be completed alongside the "Coding and Autonomous Features in Automobiles" presentation (Powerpoint or Google Docs versions are available for download at electrifyingthefuture.ca).

▶ Introduction (Tasks 1-2)

Presentation Slides #: 1-4, 6

Objectives: Introduce participants to the concept of robots by examining how self-driving cars function as robotic systems, with a focus on obstacle detection technology. Investigate the similarities between how self-driving cars and bats use echolocation, highlighting the parallels between natural and technological systems.

Variation: For older participants, material from the Scientific Background section can be utilized to offer more in-depth information.

▶ Echolocation Activity (Tasks 3-4)

Presentation Slide #: 5

Materials: Blind folds (optional)

Objective: To help participants understand how sound can be used for navigation and environmental awareness when vision is not available.

Description: One participant is blindfolded (or closes their eyes), while two others stand randomly in the room. The two standing participants take turns clapping their hands, and the blindfolded participant uses the sound of the claps to determine who and where the others are. They will then point in the direction of the sound and identify the participants. This activity is designed to help participants understand how sound can be used for navigation and environmental awareness, similar to how bats use echolocation when vision is not possible.

ELECTRIFYING THE FUTURE

Variation: For older participants, incorporating blindfolds and a few obstacles enhances the activity. Set up a simple obstacle course using small objects like chairs, cones, or boxes in a large, open space. Pair up participants, with one blindfolded and the other acting as the "echolocator." The blindfolded participant can only move forward, turn right, or stop based on the echolocator's click sounds. The echolocator will make clicking sounds to guide the blindfolded participant around obstacles.

The blindfolded participant begins by standing still and saying, "sensor starts on count 1, 2…5," signaling the echolocator and starting a countdown from 1 to 5. If the blindfolded participant does not hear a click sound within the count, they will move forward. If an obstacle is in their path, the echolocator clicks before the count finishes, causing the blindfolded participant to stop and turn right. After turning right, the blindfolded participant starts counting again from 1 to 5. If another click is heard, indicating a new obstacle, the participant stops and turns right again. This process of counting, listening for clicks, and turning right continues until no click is heard within the count, signaling it is safe to move forward.

▸ Introduction to mBlock Software (Task 5)

Presentation Slide #: 8-18

Materials: Clue card decks - Hand out to groups during this part of the presentation.

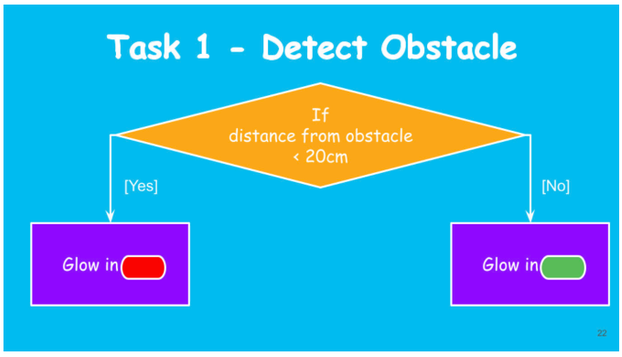
Objective: To help participants understand how sound can be used for navigation and environmental awareness when vision is not available.

Description: The main objective of the robot coding activity is to develop a step-by-step obstacle avoidance program. Begin by introducing the block coding style, a visual and user-friendly approach to programming the mBot. Walk participants through the mBlock software interface, ensuring they understand how to navigate and utilize the available tools. Highlight the block categories that will be essential for the coding tasks: Show, Actions, Sensing, Events, Controls, and Operators. Provide explanations and examples for each category to demonstrate the types of blocks they contain. These categories have been selected because they will be crucial in the upcoming coding activities.

By the end of this session, participants should have a clear understanding of the mBlock software interface and the relevant block categories, setting a strong foundation for the coding tasks ahead.

Variation: For participants who are already familiar with block coding, the instructor can modify the activity by offering a brief overview of the relevant block categories instead of an in-depth explanation. Each group can then be assigned a category to explore independently. After a few minutes, the groups will share their findings or prior knowledge with the class. This approach encourages peer learning and allows more advanced participants to lead the discussion.

▸ Coding Challenge 1: Obstacle Detection (Tasks 6-8)



Presentation Slide #: 22-23

Materials: mBot, mBlock software, Building block, Task sheet 1 (optional)

Objective: Teach participants how to program the mBot to detect obstacles using its ultrasonic sensor and change the mBot's LED color according to the distance from the obstacle.

Description: Start the activity with a discussion on the logic of the program, explaining why the LED color changes according to the distance detected by the ultrasonic sensor. Provide a step-by-step guide through the coding process, ensuring participants understand the function of each block. Participants will have time to code their program independently based on the discussed logic. Once coding is complete, they can test their program by placing small objects (e.g., building blocks) at varying distances in front of the mBot to observe its response.

Note: To test the code, turn on the mBot, connect it to the mBlock app via Bluetooth, and click the green flag to start the program. Participants can evaluate their obstacle avoidance code by placing a block both within 20 cm and beyond 20 cm of the mBot.

Important Considerations: Instructors should be aware that the mBot's ultrasonic sensors have a limited range and can only detect flat surfaces. It is advisable to position the block directly in front of the mBot rather than at an angle. After testing, ensure all groups turn off their mBots to conserve battery life.

Extensions:

- **Understanding Ultrasonic Sensor Range:** If time permits, the instructor can explain how ultrasonic sensors detect distances. After testing the 20 cm range, participants can adjust the distance to 10 cm. They should move the block to find where the mBot's LED changes from green to red. When it turns red, moving the block to the right should eventually turn the LED green again, marking the end of the sensor's range.

ELECTRIFYING THE FUTURE

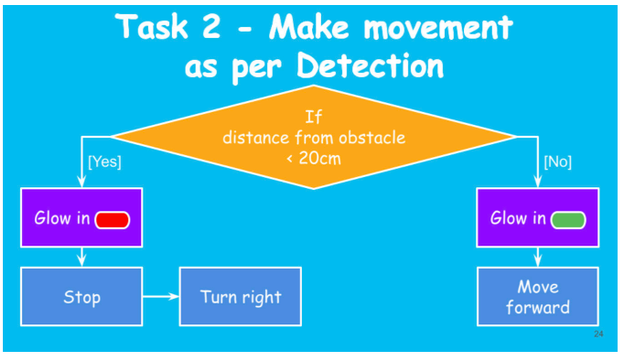
- **Real-World Applications:** Following this exploration, the instructor can discuss how this limitation is addressed in cars. For instance, vehicles often use multiple ultrasonic sensors (typically four at the rear) to extend detection range and improve obstacle detection during reversing.

Variations:

For Beginners: Provide a task sheet with the solution included. Participants will follow the sheet to build their code, learning through example.

For Advanced Participants: If participants are experienced with block coding and confident in their coding skills, focus on discussing the underlying logic. Give them time to code the program independently. Afterward, present the solution for them to compare with their own work.

▸ Coding Challenge 2: Single Obstacle Avoidance (Task 9-11)



Presentation Slide #: 24-25

Materials: mBot, mBlock software, Building block, Task sheet 2 (Optional)

Objective: Teach participants how to program the mBot to stop upon detecting an obstacle, turn right, and then continue moving forward when no obstacle is present.

Description: Participants will enhance the obstacle detection logic from Challenge 1 by incorporating movement commands into the mBot's programming. When an obstacle is detected, the mBot will stop, then turn right at 30% speed. If no obstacle is detected, it will continue moving forward.

Start with a discussion on how this logic controls the mBot's movement based on obstacle detection. Participants will then code the program, either independently, with guidance, or using a task sheet, depending on their skill level.

For testing, have participants bring their mBots to a designated testing area with perimeter walls to contain movement. The mBot should be tested in this area rather than on a table.

Important Considerations:

- To code the turn right command, use the same block code as for moving forward, but select the direction from the first drop-down menu in that block code.
- The mBot will not move if the speed is set to 20%. It is recommended to use a speed of 30% or higher.
- Before testing, make sure the distance in the condition block is set to 20 cm, not less.

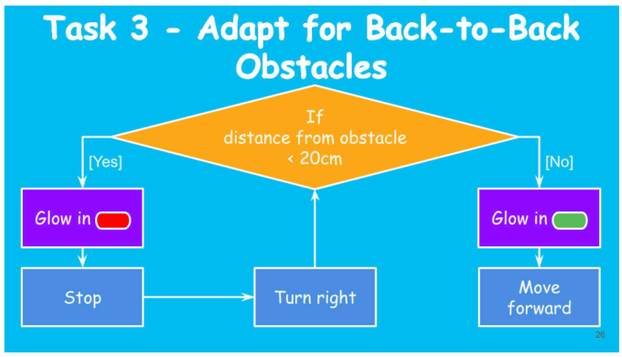
Extensions: Participants can adjust the distance to observe the effects. The instructor should explain that the distance is set to 20 cm because the mBot needs enough space to stop completely before hitting an obstacle. Just like real-world vehicles need to maintain a safe distance from the car in front to avoid sudden stops, the mBot requires a sufficient distance to stop safely.

Variations:

For Beginners: Provide additional task sheets or detailed instructions to support them in completing the coding. Offer closer guidance through each step to ensure they grasp the basics of movement control in block coding.

For Advanced Participants: Encourage them to experiment with different movement patterns, such as making the mBot turn in various directions or adjusting speeds for different scenarios. Allow them to work more independently, with the instructor offering guidance as needed.

▸ Coding Challenge 3: Multiple Obstacle Avoidance (Task 12-14)



Presentation Slide #: 8-18

Materials: Clue Card Decks

Objective: Improve the mBot's ability to navigate by programming it to detect and avoid multiple obstacles in a row. The mBot should continuously make right turns until it finds a clear path.

Description: Participants can enhance the mBot's obstacle avoidance abilities by programming it to navigate around multiple obstacles. If the mBot detects an obstacle (Obstacle 1) and turns right to avoid it but then encounters another obstacle (Obstacle 2) in the new direction, it should make another right turn. The mBot will keep turning right until it finds a clear path to move forward.

Begin by discussing this improved logic, explaining how participants will program the mBot to handle multiple obstacles without getting stuck. The coding process

ELECTRIFYING THE FUTURE

will involve adding a loop to the existing logic from Challenge 2, which checks for obstacles after each turn and makes additional turns as needed until a clear path is found.

Participants will use control block codes, operation block codes (such as loops and conditionals), and the ultrasonic sensor block code to implement this enhanced obstacle avoidance. The approach will be consistent with previous challenges, allowing participants to code independently, follow guided instructions, or use a task sheet, depending on their skill level.

For testing, participants should bring their mBots to the designated testing area, which will have perimeter walls. The instructor should set up the area with multiple obstacles to simulate a complex environment.

Important Considerations:

- To code the turn right command, use the same block code as for moving forward, but select the direction from the first drop-down menu in that block code.
- The mBot will not move if the speed is set to 20%. It is recommended to use a speed of 30% or higher.
- Before testing, make sure the distance in the condition block is set to 20 cm, not less.

Extensions:

Turn Right, Then Left: Instead of always turning right, participants can program the mBot to first turn right, then check for obstacles again. If an obstacle is detected after the right turn, the mBot should make a 180-degree turn to the left (relative to the initial direction) and then proceed.

Turn at Specific Angles: Challenge participants to program the mBot to turn at specific angles rather than just turning right. They can achieve this by programming one wheel to move while the other remains stationary, using a wait block to control the turning duration, and then adding a move forward block once the turn is complete. This enhancement will enable participants to control the mBot's movement with greater precision and explore more advanced coding techniques.

SCIENCE BACKGROUND

Echolocation in Bats

Bats use a fascinating method called echolocation to navigate and hunt in the dark. By emitting ultrasonic sound waves that are beyond the range of human hearing, bats can create a mental map of their surroundings. When these sound waves bounce off objects and return as echoes, bats analyze the timing and characteristics of the echoes to determine the location, size, and shape of objects around them. This ability allows bats to fly swiftly and accurately, even in complete darkness.

Ultrasonic Sound and Human Hearing

Ultrasonic sound refers to sound waves with frequencies higher than the upper limit of human hearing, which is around 20,000 Hz. Humans cannot hear ultrasonic sounds, but many animals, including bats, dolphins, and certain insects, use these high-frequency sounds for communication and navigation. In technology, ultrasonic sound is utilized in various applications, such as medical imaging and obstacle detection, due to its ability to provide detailed information without being intrusive.

Nature's Influence on AI Systems

Scientists and engineers often look to nature for inspiration when developing advanced technologies. The concept of echolocation, for example, has been adapted to create artificial systems that mimic this natural ability. By studying how bats use sound to navigate, researchers have developed ultrasonic sensors that enable machines to perceive their environment similarly. This bio-inspired approach helps improve the efficiency and functionality of artificial intelligence (AI) systems.

Autonomous Cars

Autonomous cars, or self-driving cars, are a prime example of how AI and sensor technology come together to create intelligent systems. These vehicles are equipped with a variety of sensors that allow them to operate without human intervention. Key sensors include cameras, radar, lidar, and ultrasonic sensors, each providing different types of data to help the car understand its surroundings.

Ultrasonic Sensors in Autonomous Cars

Ultrasonic sensors play a crucial role in the safe operation of autonomous cars. These sensors emit ultrasonic waves that reflect off nearby objects, and the car's system analyzes the returning echoes to detect obstacles and measure distances. This technology is particularly useful for short-range detection, such as during parking and low-speed maneuvers. By incorporating ultrasonic sensors, autonomous cars can accurately navigate complex environments, avoid collisions, and ensure passenger safety.

ELECTRIFYING THE FUTURE

ONTARIO CURRICULAR OUTCOMES

(Intermediate) The following outcomes are projected to directly relate to the following lesson. Detail can be added to the lesson to match a specific unit plan or extend the learning of the experience. Please note that this list may not include all relevant outcomes.

Grades 1-8

Coding and Emerging Technologies

A2.1 write and execute code in investigations and when modeling concepts

- G1 with a focus on creating clear and precise instructions for simple algorithms
- G2 with a focus on decomposing problems into smaller steps
- G3 with a focus on testing, debugging, and refining programs
- G4 with a focus on producing different types of output for a variety of purposes
- G5 with a focus on using different methods to store and process data for a variety of purposes
- G6 with a focus on obtaining input in different ways for a variety of purposes
- G7 with a focus on planning and designing programs
- G8 with a focus on automating large systems in action

A2.2 identify and describe impacts of coding and of emerging technologies

- G1-3 on everyday life
- G4-8 on everyday life, including skilled trades

Grade 9

STEM Investigation Skills (SNC1W)

A1.3 apply an engineering design process and associated skills to design, build, and test devices, models, structures, and/or systems

A1.4 apply coding skills to investigate and to model scientific concepts and relationship

Applications, Careers, and Connections (SNC1W)

A2.1 design an experiment or a prototype to explore a problem relevant to a STEM-related occupation, such as a skilled trade, using findings from research

Problem Solving and Project Management (TIJ10)

B1.1 apply the steps of a design process or other problem-solving process to plan and develop products and services (e.g., define the problem or challenge, taking into account relevant contextual or background information; gather information [about criteria, materials, constraints]; generate possible solutions, using techniques such as brainstorming; choose the best solution; develop and produce a model or prototype; test the model or prototype; incorporate improvements or redesign and retest; report on results)

B1.5 demonstrate the ability to work cooperatively in a group environment to solve problems (e.g., share tools, tasks, materials, and resources)

Grade 10 (ICD20)

Applications, Careers, and Connections (ICD20)

A3.1 investigate how digital technology and programming skills can be used within a variety of disciplines in real-world applications

A3.2 investigate ways in which various industries, including those that involve skilled trades, are changing as a result of digital technology and programming innovations

Understanding Hardware and Software (ICD20)

B1.1 describe the functions and features of various core components of hardware associated with digital technologies they encounter in their everyday life

B1.2 describe the functions and features of various connected devices associated with digital technologies they encounter in their everyday life

Innovations in Digital Technology (ICD20)

B4.1 investigate current innovations, including automation and artificial intelligence systems, and assess the impacts of these technologies on everyday life

B4.3 investigate emerging innovations related to hardware and software and their possible benefits and limitations with reference to everyday life in the future

Programming Concepts and Algorithms (ICS20), ICD20

(B1.5) use appropriate terminology to describe programming concepts and algorithms

C1.1 use appropriate terminology to describe programming concepts and algorithms

C1.2 describe simple algorithms that are encountered in everyday situations

C1.3 identify various types of data and explain how they are used within programs

C1.4 determine the appropriate expressions and in structions to use in a programming statement, taking into account the order of operations

C1.5 identify and explain situations in which conditional and repeating structures are required

ELECTRIFYING THE FUTURE

Writing Programs (ICD20)

C2.2 write programs that use and generate data involving various sources and formats

C2.4 write programs that include sequential, selection, and repeating events

C2.5 write programs that include the use of Boolean operators, comparison operators, text operators, and arithmetic operators

C2.6 interpret program errors and implement strategies to resolve them

Modularity and Modification (ICD20)

C3.1 analyze existing code to understand the components and outcomes of the code

C3.2 modify an existing program, or components of a program, to enable it to complete a different task

Programming Concepts and Algorithms (ICS20)

B1.5 use appropriate terminology to describe programming concepts and algorithms

B1.6 describe the function of Boolean operators (e.g., AND, OR, NOT), comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to), and arithmetic operators (e.g., addition, subtraction, multiplication, division, exponentiation, parentheses), and use them correctly in programming

Writing Programs (ICS20)

B2.1 use a visual problem-solving model (e.g., IPO [Input, Process, Output] chart; HIPO [Hierarchy plus Input, Process, Output] chart and diagram; flow chart; storyboard) to plan the content of a program

B2.4 write a program that includes a decision structure for two or more choices (e.g., guessing game, rock-paper-scissors game, multiple-choice quiz, trivia game)

B2.5 write programs that use looping structures effectively (e.g., simple animation, simple board games, coin toss)

Computer Programming (TEJ20)

B5.3 use a decision structure and a repetition structure in a program (e.g., determine if a user is old enough to drive, run a high-low guessing game, count from a starting value to an end value)

B5.4 use a design process (see pp. 18–19) to plan, write, and test a computer program to control a simple robot or peripheral device (e.g., servo motor, LED display)

Technology and Society (TEG20)

C2.2 describe how computers are used in various occupations (e.g., engineering calculations, architectural drawings, customer tracking and business data collection, navigation of airplanes and ships), and what work in these occupations would be like without computers

Grade 11

Data Types and Expressions (ICS3U)

A1.4 demonstrate the ability to use Boolean operators (e.g., AND, OR, NOT), comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to), arithmetic operators (e.g., addition, subtraction, multiplication, division, exponentiation, parentheses), and order of operations correctly in computer programs

Control Structures and Simple Algorithms (ICS3U)

A2.2 use sequence, selection, and repetition control structures to create programming solutions

A2.3 write algorithms with nested structures (e.g., to count elements in an array, calculate a total, find highest or lowest value, or perform a linear search)

Problem-solving Strategies (ICS3U)

B1.1 use various problem-solving strategies (e.g., step wise refinement, divide and conquer, working backwards, examples, extreme cases, tables and charts, trial and error) when solving different types of problems

B1.2 emonstrate the ability to solve problems independently and as part of a team

Designing Software Solutions (ICS3U)

B2.1 design programs from a program template or skeleton (e.g., teacher-supplied skeleton, Help facility code snippet)

B2.2 use appropriate vocabulary and mode of expression (i.e., written, oral, diagrammatic) to describe alternative program designs, and to explain the structure of a program

B2.4 represent the structure and components of a program using industry-standard programming tools (e.g., structure chart, flow chart, UML [Unified Modeling Language], data flow diagram, pseudocode)

Exploring Computer Science (ICS3U)

D2.1 demonstrate an understanding of emerging areas of research in computer science (e.g., cryptography, parallel processing, distributed computing, data mining, artificial intelligence, robotics, computer vision, image processing, human– computer interaction, security, geographic information systems [GIS])

D2.2 demonstrate an understanding of an area of collaborative research between computer science and another field (e.g., bioinformatics, geology, economics, linguistics, health informatics, climatology, sociology, art)

ELECTRIFYING THE FUTURE

Grade 12

Data Structures

A1.2 use Boolean operators (e.g., AND, OR, NOT), comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to), arithmetic operators (e.g., addition, subtraction, multiplication, division, exponentiation, parentheses), and order of operations correctly in programming