ROBOT DRONE LEAGUE
2024 Challenge: MINESHAFT

Standards Alignment with the Alabama Course of Study: COMPUTER SCIENCE

RDL Introduction
Creativity and innovation are key elements to advancing the fields of science, technology, engineering, and mathematics (STEM) into the future. Robot Drone League (RDL) has been designed to provide students with open-ended challenges that allow for creation and innovation by engaging in hands-on design, engineering, and programming of interactive robots and drones. Students are presented with the opportunity to develop real-world connections to classroom learning. Working with robots in a collaborative game format can be a very powerful tool to engage students and enhance math and science skills through hands-on, student-centered learning. Through participation in RDL, students can develop the essential life skills of teamwork and collaboration, as well as critical thinking, project management, and communication required to become the next generation of innovators and problem-solvers in our global society.

Grade 6
During sixth grade, students will continue to develop the foundation of computer science. They will expand their problem-solving skills and progress toward independence while continuing to collaborate on local and global issues. Students must be creators, not just consumers, who can effectively utilize digital tools and understand the influence of technology. These standards are written to encourage student-centered learning through innovative and engaging activities.

Computational Thinker
Abstraction
1. Remove background details from an everyday process to highlight essential properties.
Examples: When making a sandwich, the type of bread, condiments, meats, and/or vegetables do not affect the fact that one is making a sandwich.
2. Define a process as a function.
Example: Functions or sets of steps combined to produce a process: turning off your alarm + getting out of bed + brushing your teeth + getting dressed = morning routine.
Algorithms
3. Create pseudocode that uses conditionals.
Examples: Using if/then/else (If it is raining then bring an umbrella else get wet).
5. Identify algorithms that make use of sequencing, selection or iteration.
Examples: Sequencing is doing steps in order (put on socks, put on shoes, tie laces); selection uses a Boolean condition to determine which of two parts of an algorithm are used (hair is dirty? True, wash hair; false, do not); iteration is the repetition of part of an algorithm until a condition is met (if you're happy and you know it clap your hands, when you're no longer happy you stop clapping).
Programming and Development
6. Identify steps in developing solutions to complex problems using computational thinking.

7. Describe how automation works to increase efficiency.
8. Create a program that initializes a variable.

Global Collaborator
Creative Communications
16. Communicate and/or publish collaboratively to inform others from a variety of backgrounds and cultures about issues and problems.

Innovative Designer
Design Thinking
30. Discuss and apply the components of the problem-solving process.

Grade 7
During seventh grade, students will become independent thinkers while developing their global online presence. Students must be creators, not just consumers, who will effectively utilize digital tools, understand technology's impact on a global society, and integrate principles of computer science. These standards are written for student-centered learning with teacher mentoring.

Computational Thinker
Abstraction
1. Create a function to simplify a task.
Algorithms
2. Create complex pseudocode using conditionals and Boolean statements.
Example: Automated vacuum pseudocode – drive forward until the unit encounters an obstacle; reverse 2"; rotate 30 degrees to the left, repeat.
3. Create algorithms that demonstrate sequencing, selection or iteration.
4. Design a complex algorithm that contains sequencing, selection or iteration.
Programming and Development
5. Solve a complex problem using computational thinking.
6. Create and organize algorithms in order to automate a process efficiently.
7. Create a program that updates the value of a variable in the program.
8. Formulate a narrative for each step of a process and its intended result, given pseudocode or code.

Global Collaborator
Creative Communications
16. Construct content designed for specific audiences through an appropriate medium.
17. Publish content to be available for external feedback.

Innovative Designer
Design Thinking
30. Apply the problem-solving process to solve real-world problems.

Grade 8
During eighth grade, students will expound upon computer science and global collaboration experiences. Students will be designers, not just consumers, who will effectively utilize digital tools and articulate the impact of technology on a global society. These standards are written to provide student-centered learning with minimal guidance from the teacher.

Computational Thinker
Abstraction
1. Design a function using a programming language that demonstrates abstraction.
Example: Create a program that utilizes functions in an effort remove repetitive sequences of steps.
2. Explain how abstraction is used in a given function.
Example: Examine a set of block-based code and explain how abstraction was used.
Algorithms
3. Create an algorithm using a programming language that includes the use of sequencing, selections, or iterations.
4. Create a function to simplify a task.
Programming and Development
5. Discuss the efficiency of an algorithm or technology used to solve complex problems.
6. Describe how algorithmic processes and automation increase efficiency.
7. Create a program that includes selection, iteration, or abstraction, and initializes, and updates, at least two variables.
Examples: Make a game, interactive card, story, or adventure game.
Global Collaborator
Creative Communications
16. Present content designed for specific audiences through an appropriate medium.
17. Communicate and publish individually or collaboratively to persuade peers, experts, or community about issues and problems.

Innovative Designer
Design Thinking
29. Create an artifact to solve a problem using ideation and iteration in the problem-solving process.
Examples: Create a public service announcement or design a computer program, game, or application.

Grades 9 – 12

RDL will help students in grades 9-12 meet the following learning goals:

- As **Computational Thinkers**, students demonstrate how to simplify complex problems by developing algorithms that define the systematic processes.
- As **Global Collaborators**, students utilize digital tools to collaborate and communicate with others to solve problems presented in today's technical world.
- As **Innovative Designers**, students make decisions and create solutions using the

various digital tools available in today's technical environments.

Computational Thinker

Abstraction

1. Decompose problems into component parts, extract key information, and develop descriptive models to understand the levels of abstractions in complex systems.
2. Explain how computing systems are often integrated with other systems and embedded in ways that may not be apparent to the user.

Examples: Millions of lines of code control the subsystems within an automobile (e.g., antilock braking systems, lane detection, and self-parking).

Algorithms

3. Differentiate between a generalized expression of an algorithm in pseudocode and its concrete implementation in a programming language.

a. Explain that some algorithms do not lead to exact solutions in a reasonable amount of time and thus approximations are acceptable.

b. Compare and contrast the difference between specific control structures such as sequential statements, conditional, iteration, and explain the benefits and drawbacks of choices made.

Examples: Trade Offs involving implementation, readability, and program performance.

c. Distinguish when a problem solution requires decisions to be made among alternatives, such as selection constructs, or when a solution needs to be iteratively processed to arrive as a result, such as iterative "loop" constructs or recursion.

d. Evaluate and select algorithms based on performance, reusability, and ease of implementation.

e. Explain how more than one algorithm may solve the same problem and yet be characterized with different priorities.

4. Use and adapt classic algorithms to solve computational problems.

Examples: Sorting, searching, shortest path, and data compression.

Programming and Development

5. Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using current events.
6. Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects, with parameters, and which return a result.
7. Compare and contrast fundamental data structures and their uses.

Examples: Strings, lists, arrays, stacks, queues.

8. Demonstrate code reuse by creating programming solutions using libraries and Application

Programming Interfaces.

9. Demonstrate the ability to verify the correctness of a program.

a. Develop and use a series of test cases to verify that a program performs according to its design specifications

b. Collaborate in a code review process to identify correctness, efficiency, scalability and readability of program code.

10. Resolve or debug errors encountered during testing using iterative design process.

Examples: Test for infinite loops, check for bad input, check edge-cases.

Innovative Designer

Design Thinking

40. Use an iterative design process, including learning from mistakes, to gain a better understanding of a problem domain.