ROBOT DRONE LEAGUE
2024 Challenge: MINESHAFT

Standards Alignment with the Virginia Computer Science Standards of Learning, STEM, and Engineering

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.

Science, Technology, Engineering & Mathematics (STEM)

What is STEM Education?

Science, Technology, Engineering and Mathematics (STEM) education entails authentic learning experiences for all students with an interdisciplinary and applied approach where all fields connect in complex relationships. In today’s economy, problems are not solved in isolation of a specific discipline, but are solved through multiple approaches and perspectives. A strong STEM educational foundation helps to prepare our students for tomorrow’s world by emphasizing collaborative, innovative, quantitative and logical analysis rooted in a solid understanding of the interdisciplinary nature of science, technology, engineering and mathematics.

How is STEM Education being refocused?

Recently there has been a shift in beliefs about the purpose of STEM education. Traditionally a STEM education focused on creating a pipeline of students whose educational backgrounds prepared them for a STEM-specific workforce. Today, the focus is on developing STEM-literate citizens necessary for success in any 21st century profession. STEM literacy is the ability to identify and acknowledge science, technology, engineering and mathematics concepts and processes in everyday life.

STEM literacy comes from an understanding that it takes

1. a scientific approach to observe and interpret the world;
2. technology to serve as a tool to solve problems or reach a goal;
3. engineering to design, test and solve a problem through the creation of products or processes; and

4. mathematics to help quantify, comprehend and evaluate the problem and solution's success.

As students become STEM literate citizens, they have the foundational content and the discipline processes to allow them make informed decisions and to participate in public/civil discourse concerning future STEM issues and technologies.

Engineering Design

The engineering design process is iterative. Although there are steps to the process, movement between the various steps may not be sequential and may be repeated as the student or engineer develops potential solutions to the problem. The process includes:

- Define the problem and determine the parameters
- Brainstorm potential solutions
- Research the problem (this can be done before the brainstorm session)
- Pick one solution, plan a prototype, and determine what data to collect to determine its effectiveness
- Build the prototype
- Test the prototype
- Improve the prototype and test it again
- Communicate the results

Grade 6 2024 Standards of Learning
Essential Knowledge and Skills (EKS)

Algorithms and Programming: Algorithms, a sequence of steps designed to accomplish a specific task, are translated into programs, or code,to provide instructions for computing devices. Creating programs involves choosing information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

Algorithms and Programming (AP)
6.AP.1 The student will apply computational thinking to identify patterns, make use of decomposition to break down problems or processes into sub-components, and design algorithms.
a. Identify patterns and repeated steps in an algorithm, problem, or process.
b. Decompose an algorithm, problem, or process into subcomponents.
c. Abstract relevant information to identify essential details.
d. Design algorithms using abstraction to accomplish a task or express a computational process.

6.AP.2 The student will plan and implement algorithms that include conditional control structures and collection of numeric data using a block-based or text-based tool.

a. Create a decision tree diagram to illustrate the decisionmaking process.
b. Read and write programs that initialize Boolean, integer, and decimal number variables.
c. Read and write programs that collect numeric data from users.
d. Read and write programs that contain nested conditional control structures.
e. Predict the results of logic expressions that use Boolean operators: and, or, and not; including expressions that use relational expressions as one or more operands.

6.AP.3 The student will use the iterative design process to create, test, and debug programs using a block-based or text-based programming language.
a. Create and test programs that uses multiple conditional control structures.
b. Incorporate existing code, media, or libraries into original programs
c. Trace and predict outcomes of programs.
d. Analyze and describe program results to assess validity of outcomes.
e. Analyze the outcomes of programs to identify logic and syntax errors.
f. Incorporate feedback from others to refine program.
g. Revise and improve programs to resolve errors and produce desired outcomes.

6.AP.4 The student will demonstrate proper attribution when incorporating ideas and works of others.
a. Identify and give proper attribution of information and assets from the Internet and other sources.

Grade 7 2024 Standards of Learning
Essential Knowledge and Skills (EKS)

Algorithms and Programming (AP)

7.AP.1 The student will apply computational thinking to design programs to accomplish a task as a means of creative expression or scientific exploration.
a. Identify patterns and repeated steps in an algorithm, problem, or process.
b. Decompose an algorithm, problem, or process into subcomponents.
c. Abstract relevant information to identify essential details.
d. Contrast various algorithms to solve reasoning problems when accomplishing a task.

7.AP.2 The student will plan and implement algorithms that include sequencing, loops, variables, user input, conditional control structures, and functions using a block-based or text-based programming tool.
a. Describe the concept of functions for use in a computer program.
b. Plan an algorithm using plain language, pseudocode, or diagrams.
c. Read and write programs that collect and use numeric and text data from users.
d. Read and write programs that contain nested conditionals and nested loops.

7.AP.3 The student will use the iterative design process to create, test, and debug programs using a block-based or text-based programming language.

a. Create and test programs that contain multiple control structures.
b. Trace and predict outcomes of programs.
c. Analyze the outcomes of programs to identify logic and syntax errors.
d. Analyze and describe the results of a program to assess validity of outcomes.
e. Revise and improve an algorithm to resolve errors or produce desired outcomes.

7.AP.4 The student will apply proper attribution when incorporating other sources into original work.
a. Apply proper methods of attribution when using work from the Internet and other sources.
b. Incorporate information or assets from the Internet into a program with proper attribution.

Grade 8 2024 Standards of Learning
Essential Knowledge and Skills (EKS)
Algorithms and Programming (AP)

8.AP.1 The student will apply computational thinking to construct programs to accomplish a task as a means of creative expression or scientific exploration.
a. Identify patterns and repeated steps in an algorithm, problem, or process.
b. Decompose an algorithm, problem, or process into subcomponents.
c. Abstract relevant information to identify essential details.
d. Use pseudocode, decision tree diagrams or flowcharts to illustrate complex problems as algorithms.

8.AP.2 The student will plan and implement algorithms that include sequencing, loops, variables, user input, conditional control structures, functions, and various data types.
a. Describe the concept of input and output of various data types for use in a computer program.
b. Plan an algorithm using plain language, pseudocode, or diagrams.
c. Write and test algorithms expressed using block-based or text based programming languages.

8.AP.3 The student will use the iterative design process to create, test, and debug programs using a block-based or text-based programming language.
a. Create and test programs that contain multiple control structures.
b. Trace and predict outcomes of programs.
c. Analyze the outcomes of programs to identify logic and syntax errors.
d. Analyze and describe the results of a program to assess validity of outcomes.
e. Revise and improve algorithms to resolve errors or produce desired outcomes.

8.AP.4 The student will incorporate work from others into programs and projects.
a. Explain the role of Creative Commons licensing for the use and modification or “remixing” of information.
b. Utilize Creative Commons assets in a programming project.
c. Use and remix code from other projects within a programming project and provide proper attribution.

Middle School Computer Science Elective Computer Science Standards of Learning: 2024

Algorithms and Programming: Algorithms, a sequence of steps designed to accomplish a specific task, are translated into programs, or code, to provide instructions for computing devices. Creating programs involves choosing information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

9 Weeks

• Apply computational thinking to evaluate and solve a problem• Use iterative design process to create a program

• Plan and implement algorithms that include loops, variables, user input, compound and nested conditional control structures and procedural definitions that accept parameters using block-based or text-based programming

• Use an interactive approach to trace, predict, test, and debug to improve existing programs and solve problems

18 Weeks

• Improve existing solutions to problems to create new programs

• Systematically use multiple test cases to verify the accuracy of a program

• Work collaboratively in an iterative design process to solve problems, including peer review and feedback

• Analyze solve and document a problem-solving process for others to duplicate the solution

• Investigate different coding languages

36 Weeks

• Develop a solution to a real-world problem using programming

• Decompose problems and subcomponents into parts to facilitate the design, implementation, and review of programs

• Design and create algorithms using a text-based programming language

2024 Standards of Learning
Knowledge and Skills (KS)
Essential Knowledge and Skills (EKS)
9 – Week Module

MSCSE -9.AP.1 The student will apply computational thinking to evaluate and solve a problem.
a. Decompose a problem or process into subcomponents.
b. Recognize characteristics or patterns to determine commonalities.
c. Abstract relevant information to identify essential details.
d. Use pseudocode and/or flowcharts to address complex problems as algorithms.

MSCSE-9.AP.2 The student will use iterative design process to
create a program.

a. Identify the goal and objectives of the program.
b. Plan for the design or prototype of the program.
c. Develop an outline for the program's functionality.
d. Engage with peers to collect feedback on relevant aspects.

MSCSE-9.AP.3 The student will plan and implement algorithms that include loops, variables, user input, compound and nested conditional control structures, and
procedure definitions that accept parameters using block-based or text-based programming.
a. Read and interpret algorithms expressed using plain language, pseudocode, and block-based or text-based programming languages.
b. Create an algorithm using plain language, pseudocode, or diagrams.
c. Implement programs that accept input values, use variables, and produce output.
d. Write and test algorithms using block-based or text-based programming languages.

MSCSE-9.AP.4 The student will use an interactive approach to trace, predict, test, and debug to improve existing programs and solve problems.
a. Trace a program for accuracy.
b. Analyze and describe the results of a program for validity.
c. Revise and improve an algorithm to resolve errors or produce desired outcomes.
d. Document programs to make them easier to trace, test, and debug.

2024 Standards of Learning
Knowledge and Skills (KS)
Essential Knowledge and Skills (EKS)
18 – Week Module

MSCSE-18.AP.1 The student will improve existing solutions to problems to create new programs.
a. Categorize problems as classification, prediction, combinational search, or sequential decision problems.
b. Determine when problems can be solved with programs and automation.
c. Create a variety of programs while considering the needs and preferences of diverse user groups.
d. Utilize existing code, media, and libraries into original programs, and give attribution.

MSCSE-18.AP.2 The student will systematically use multiple test cases to verify the accuracy of a program.
a. Predict and test the outcome or output of multiple test cases.
b. Verify and refine the program based on the outcome of multiple test cases.

MSCSE-18.AP.3 The student will work collaboratively in an iterative design process to solve problems, including peer review and feedback.
a. Collaboratively plan, design, and revise programs.
b. Explain design choices, including constraints, and audiences.

c. Provide constructive feedback through peer review.
d. Reflect on collaborative experiences.

MSCSE-18.AP.4 The student will analyze, solve, and document a problem-solving process for others to duplicate the solution.
a. Analyze and decompose a problem.
b. Use abstraction to determine a solution to a problem.

MSCSE-18.AP.5 The student will investigate different coding languages.
a. Identify characteristics of block-based and text based coding languages.
b. Analyze the advantages and disadvantages of block-based and text-based coding languages.

2024 Standards of Learning
Knowledge and Skills (KS)
Essential Knowledge and Skills (EKS)
36 – Week Module

MSCSE-36.AP.1 The student will develop a solution to a real-world problem using programming.
a. Identify problems that can be solved with a program.
b. Justify a proposed solution to a problem.
c. Use project management tools to support collaboration.
d. Engage in peer review and incorporate evaluative feedback through the design process.
e. Incorporate feedback provided through peer review to refine prototypes.

MSCSE-36.AP.2 The student will decompose problems and subcomponents into parts to facilitate the design, implementation, and review of programs.
a. Decompose problems into subcomponents to facilitate the creation of a program.
b. Use documentation to explain the purpose of a section of code and its relationship to other parts of a program.
c. Evaluate the limitations of models, algorithms, and programs considering multiple perspectives.
d. Systematically test and refine programs.

MSCSE-36.AP.3 The student will design and create algorithms
using a text-based programming language.
a. Read and write programs that combine loops
and conditional control structures.b. Read and write programs using functions and procedures.

2024 CS Foundations Computer Science Standards of Learning

2024 Standards of Learning
Knowledge and Skills (KS), STEM and Engineering

Algorithms and Programming (AP)

CSF.AP.1 The student will apply computational thinking to address a computational problem.
a. Decompose a problem or process into sub-components.
b. Implement abstractions to improve program molarity,
reusability, and readability.
c. Identify computing-based solutions to address a computational problem.

CSF.AP.2 The student will use the iterative design process to create, test, and refine programs using a text-based programming language.
a. Create programs using a text-based programming language.
b. Trace the execution of an algorithm and predict its results.
c. Analyze the outcomes of programs to identify logic and syntax errors.
d. Use multiple test cases to verify and refine the program.
e. Revise and improve an algorithm to resolve errors or produce desired outcomes.
f. Use version control and incorporate user feedback to refine program.

CSF.AP.3 The student will plan and implement algorithms and programs that include loops, variables, and compound and nested control structures using a text-based programming language.
a. Read and interpret algorithms and programs expressed using plain language, pseudocode, and text-based programming languages
b. Create design documents using plain language, pseudocode, or diagrams.
c. Read and write algorithms and programs that accept multiple input values, use variables, and produce output.
d. Read and write algorithms and programs that include predefined functions and procedures with parameters and returns.
e. Compare several implementations of the same algorithm using different control structures.

CSF.AP.4 The student will design programs that use and manipulate data.
a. Determine appropriate data structures to address program specifications.
b. Apply basic computations on numeric and non-numeric data types.
c. Read and write programs that create, store, and manipulate primitive data
d. Read and write programs that create, store, and manipulate linear collections of primitive data types: arrays or lists.
e. Read and write programs that use relational, logical, and arithmetic expressions.
f. Read and write programs that iterate over higher-order data structures.

CSF.AP.5 The student will define and describe neural network learning algorithms.
a. Define and describe neural network learning algorithms.
b. Identify and discuss examples of computing technologies that utilize neural networks.
c. Compare and contrast a decision tree learning algorithm and a neural network learning algorithm.

CSF.AP.6 The student will investigate different coding languages.
a. Identify and describe characteristics of block-based and text-based coding languages.
b. Analyze the advantages and disadvantages of block based and text-based coding languages.
c. Analyze the advantages and disadvantages of various text-based coding languages.

CSF.AP.7 The student will use and search algorithms and sort algorithms.
a. Define the concept and role of a search algorithm.
b. Define the concept and role of a sort algorithm.
c. Compare and contrast bubble sort, quick sort, and merge sort.
d. Compare and contrast linear search and binary search.
e. Evaluate and determine the best search or sort algorithm to use based on intended results.

CSF.AP.8 The student will work collaboratively in an iterative design process to solve problems, including peer review and feedback.
a. Identify project management frameworks and methodologies that emphasize iteration.
b. Discuss the significance of communication and methods of communication when working collaboratively.
c. Distribute roles and responsibilities and adhere to predetermined timeline and/or project scope.
d. Collaboratively plan, design, and revise programs.
e. Provide constructive feedback through peer review.
f. Use project management tools to support collaboration.
g. Justify and explain design choices, including constraints, and audiences.
h. Reflect and discuss collaborative experience with team.