GLITCH

ENGINEERING AN ESCAPE ROOM

Educational Standards Alignment

Tried&True

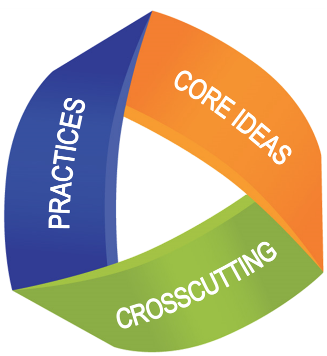
GLITCH

ENGINEERING AN ESCAPE ROOM

Educational Standards Alignment

NEXT GENERATION SCIENCE STANDARDS

There are three distinct and equally important dimensions to learning science. These dimensions are combined to form each standard – or **performance expectation** – and each dimension works with the other two to help students build a cohesive understanding of science over time.



Disciplinary Core Ideas are the key ideas in science that have broad importance within or across multiple science or engineering disciplines. These core ideas build on each other as students progress through grade levels and are grouped into four domains.

Science and Engineering Practices describes what scientists do to investigate the natural world and what engineers do to design and build systems. The practices better explain and extend what is meant by “inquiry” in science and the range of cognitive, social, and physical practices that it requires. Students engage in practices to build, deepen, and apply their knowledge of core ideas and crosscutting concepts.

Crosscutting Concepts help students explore connections across the four domains of science, including Physical Science, Life Science, Earth and Space Science, and Engineering Design. When these concepts are made explicit, they can help students develop a coherent and scientifically-based view of the world around them.

DISCIPLINARY CORE IDEAS

Summary of all Disciplinary Core Ideas	
Physical Science **PS1** Matter and its Interactions **PS2** Motion and Stability: Forces and Interactions **PS3** Energy **PS4** Waves and their Applications in Technologies for Information Transfer	**Life Science** **LS1** From Molecules to Organisms: Structure and Processes **LS2** Ecosystems: Interactions, Energy, and Dynamics **LS3** Heredity: Inheritance and Variation of Traits **LS4** Biological Evolution: Unity and Diversity
Earth and Space Science **ESS1** Earth’s Place in the Universe **ESS2** Earth’s Systems **ESS3** Earth and Human Activity	**Engineering, Technology, and Applications of Science** **ETS1** Engineering Design

Specific Disciplinary Core Ideas in this Curriculum		
DCI	**Topic**	**Description**
PS1.A	Structure and Properties of Matter	Each atom has a charged substructure consisting of a nucleus, which is made of protons and neutrons, surrounded by electrons.
PS2.B	Types of Interactions	Newton's law of universal gravitation and Coulomb's law provide the mathematical models to describe and predict the effects of gravitational and electrostatic forces between distant objects.
PS3.A	Definitions of Energy	"Electrical energy" may mean energy stored in a battery or energy transmitted by electric currents. At the macroscopic scale, energy manifests itself in multiple ways, such as in motion, sound, light, and thermal energy.
PS4.A	Wave Properties	Information can be digitized (e.g., a picture stored as the values of an array of pixels); in this form, it can be stored reliably in computer memory and sent over long distances as a series of wave pulses.
PS4.B	Electromagnetic Radiation	Photovoltaic materials emit electrons when they absorb light of a high enough frequency.
PS4.C	Information Technologies and Instrumentation	Multiple technologies based on the understanding of waves and their interactions with matter are part of everyday experiences in the modern world (e.g, medical imaging, communications, scanners) and in scientific research. They are essential tools for producing, transmitting, and capturing signals and for storing and interpreting the information contained in them.
ESS3.A	Natural Resources	Resource availability has guided the development of human society. All forms of energy production and other resource extraction have associated economic, social, environmental, and geopolitical costs and risks as well as benefits. New technologies and social regulations can change the balance of these factors.
ETS1.A	Defining and Delimiting and Engineering Problem	Criteria and constraints also include satisfying any requirements set by society, such as taking issues of risk mitigation into account, and they should be quantified to the extent possible and stated in such a way that one can tell if a given design meets them.
EST1.B	Developing Possible Solutions	When evaluating solutions it is important to take into account a range of constraints, including cost, safety, reliability and aesthetics and to consider social, cultural and environmental impacts. Both physical models and computers can be used in various ways to aid in the engineering design process. Computers are useful for a variety of purposes, such as running simulations to test different ways of solving a problem or to see which one is most efficient or economical; and in making a persuasive presentation to a client about how a given design will meet his or her needs.
ETS1.C	Optimizing the Design Solution	Criteria may need to be broken down into simpler ones that can be approached systematically, and decisions about the priority of certain criteria over others (trade offs) may be needed.

SCIENCE & ENGINEERING PRACTICES

Summary of all Science and Engineering Practices	
Asking Questions & Defining Problems	A practice of science is to ask and refine questions that lead to descriptions and explanations of how the natural and designed world works and which can be empirically tested.
Developing & Using Models	A practice of both science and engineering is to use and construct models as helpful tools for representing ideas and explanations. These tools include diagrams, drawings, physical replicas, mathematical representations, analogies, and computer simulations
Planning & Carrying Out Investigations	Scientists and engineers plan and carry out investigations in the field or laboratory, working collaboratively as well as individually. Their investigations are systematic and require clarifying what counts as data and identifying variables or parameters.
Analyzing & Interpreting Data	Scientific investigations produce data that must be analyzed in order to derive meaning. Because data patterns and trends are not always obvious, scientists use a range of tools – including tabulation, graphical interpretation, visualization, and statistical analysis – to identify the significant features and patterns in the data. Scientists identify sources of error in the investigations and calculate the degree of certainty in the results.
Using Mathematical & Computational Thinking	In both science and engineering, mathematics and computation are fundamental tools for representing physical variables and their relationships. They are used for a range of tasks such as constructing simulations; solving equations exactly or approximately; and recognizing, expressing, and applying quantitative relationships.
Constructing Explanations & Designing Solutions	The end-products of science are explanations and the end-products of engineering are solutions. The goal of science is the construction of theories that provide explanatory accounts of the world. A theory becomes accepted when it has multiple lines of empirical evidence and greater explanatory power of phenomena than previous theories.
Engaging in Argument from Evidence	Argumentation is the process by which evidence-based conclusions and solutions are reached. In science and engineering, reasoning and argument based on evidence are essential to identifying the best explanation for a natural phenomenon or the best solution to a design problem.
Obtaining, Evaluating, & Communicating Information	Scientists and engineers must be able to communicate clearly and persuasively the ideas and methods they generate. Critiquing and communicating ideas individually and in groups in a critical professional activity.

Summary of all Science and Engineering Practices

Practice	Chapter																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Asking questions and defining problems			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Developing and using models	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Planning and carrying out investigations	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Analyzing and			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

interpreting data																		
Using mathematical and computational thinking	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Constructing explanations and designing solutions	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Engaging in argument from evidence	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Obtaining, evaluating, and communicating information	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CROSSCUTTING CONCEPTS

Summary of all Crosscutting Concepts	
Patterns	Observed patterns in nature guide organization and classification and prompt questions about relationships and causes underlying them.
Cause & Effect	Events have causes, sometimes simple, sometimes multifaceted. Deciphering causal relationships, and the mechanisms by which they are mediated, is a major activity of science and engineering.
Scale, Proportion, & Quantity	In considering phenomena, it is critical to recognize what is relevant at different size, time, and energy scales, and to recognize proportional relationships between different quantities as scales change.
Systems & System Models	A system is an organized group of related objects or components; models can be used for understanding and predicting the behavior of systems.
Energy & Matter	Tracking energy and matter flows, into, out of, and within systems helps one understand their system's behavior.
Structure & Function	The way an object is shaped or structured determines many of its properties and functions.
Stability & Change	For both designed and natural systems, conditions that affect stability and factors that control rates of change are critical elements to consider and understand.

Specific Crosscutting Concepts in this Curriculum

Concept	Chapter																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Patterns	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Cause and effect	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Scale, proportion, and quantity			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Systems and system models	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Energy and matter	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Structure and function	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Stability and change				✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

NGSS PERFORMANCE EXPECTATIONS

Each performance expectation incorporates all three dimensions - disciplinary core ideas, science and engineering practices, and crosscutting concepts. Performance expectations are the assessable statements of what students should know and be able to do.

Specific Performance Expectations in this Curriculum		
PE	**Topic**	**Description**
HS-PS3-3	Energy	Design, build, and refine a device that works within given constraints to convert one form of energy into another form of energy.
PS-PS3-5	Energy	Develop and use a model of two objects interactive through electric or magnetic fields to illustrate the forces between objects and the changes in energy of the objects due to the interaction.
HS-PS4-2	Waves and Electromagnetic Radiation	Evaluate questions about the advantages of using digital transmission and storage of information.
HS-ESS3-4	Human Sustainability	Evaluate or refine a technological solution that reduces impacts of human activities on natural systems.
HS-ETS1-1	Engineering Design	Analyze a major global challenge to specify qualitative and quantitative criteria and constraints for solutions that account for societal needs and wants.
HS-ETS1-2	Engineering Design	Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.
HS-ETS1-3	Engineering Design	Evaluate a solution to a complex real-world problem based on prioritized criteria and trade-offs that account for a range of constraints, including cost, safety, reliability, and aesthetics, as well as possible social, cultural, and environmental impacts.

COMPUTER SCIENCE STANDARDS FOR CALIFORNIA PUBLIC SCHOOLS

Computer literacy, educational technology, digital citizenship, and information technology focus on use. Computer science requires students to not merely use technology as passive consumers. Computer science also calls for students to understand why and how computing technologies work, and then build upon that conceptual knowledge by creating computational artifacts.

The standards include five core concept areas, coupled with seven core practices that demonstrate ways in which students actively engage in computer science learning experiences that build conceptual knowledge.

The computer science **core concepts** include:

- Computing Systems (CS)
- Networks and the Internet (NI)
- Data and Analysis (DA)
- Algorithms and Programming (AP)
- Impacts of Computing (IC)

The computer science **core practices** include:

- Fostering an Inclusive Computing Culture
- Collaborating Around Computing
- Recognizing and Defining Computational Problems
- Developing and Using Abstractions
- Creating Computational Artifacts
- Testing and Refining Computational Artifacts
- Communicating About Computing

Specific Performance Expectations in This Curriculum	
Code	Performance Indicator
3-5.CS.1	Describe how computing devices connect to other components to form a system
9-12S.CS.1	Illustrate ways computing systems implement logic through hardware components
K-2.CS.2	Explain the functions of common hardware and software components of computing systems
6-8.CS.2	Design a project that combines hardware and software components to collect and exchange data
K-2.CS.3	Describe basic hardware and software problems using accurate terminology
3-5.CS.3	Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies
6-8.CS.3	Systematically apply troubleshooting strategies to identify and resolve hardware and software problems in computing systems
K-2.DA.7	Store, copy, search, retrieve, modify, and delete information using a computing device, and define the information stored as data
6-8.DA.7	Represent data in multiple ways

9-12.DA.8	Translate between different representations of data abstractions of real-world phenomena, such as characters, numbers, and images
9-12.DA.9	Describe tradeoffs associated with how data elements are organized and stored
9-12.DA.10	Create data visualizations to help others better understand real-world phenomena
6-8.DA.9	Test and analyze the effects of changing variables while using computational models
9-12.DA.11	Refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process
6-8.AP.10	Use flowcharts and/or pseudocode to design and illustrate algorithms that solve complex problems
9-12.AP.12	Design algorithms to solve computational problems using a combination of original and existing algorithms
6-8.AP.11	Create clearly named variables that store data, and perform operations on their contents
9-12.AP.13	Create more generalized computational solutions using collections instead of repeatedly using simple variables
K-12.AP.12	Create programs with sequences of commands and simple loops, to express ideas or address a problem
3-5.AP.12	Create programs that include events ,loops, and conditionals
6-8.AP.12	Design and iteratively develop programs that combine control structures and use compound conditions
9-12.AP.14	Justify the selection of specific control structures by identifying tradeoffs associated with implementation, readability, and performance
9-12S.AP.15	Demonstrate the flow of execution of a recursive algorithm
9-12.AP.16	Decompose problems into smaller subproblems through systematic analysis, using constructs such as procedures, modules, and/or classes
9-12.AP.17	Create computational artifacts using modular design
9-12.AP.18	Systematically design programs for broad audiences by incorporating feedback from users
9-12.AP.20	iteratively evaluate and refine a computational artifact to enhance its performance, reliability, usability, and accessibility
9-12.AP.21	Design and develop computational artifacts working in team roles using collaborative tools
9-12.AP.22	Document decisions made during the design process using text, graphics, presentations, and/or demonstrations in the development of complex programs
6-8.IC.22	Collaborate with many contributors when creating a computational artifact

WISCONSIN STANDARDS FOR COMPUTER SCIENCE

Wisconsin defines computer science (CS) as an academic discipline that encompasses the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, networks, and their impact on society. The standards outlined in this document provide an important foundation to prepare students for post-secondary education and careers.

Summary of Standards in This Curriculum	
Content Area	Standard
Algorithms and Programming	Students will recognize and define computational problems using algorithms and programming
Algorithms and Programming	Students will create computational artifacts using algorithms and programming
Algorithms and Programming	Students will be able to communicate about computing ideas
Algorithms and Programming	Students will develop and use abstractions
Algorithms and Programming	Students will be able to collaborate with diverse teams
Algorithms and Programming	Students will be able to test and refine computational solutions
Computing Systems	Students will communicate about computing systems
Computing Systems	Students will test and refine computing systems
Computing Systems	Students will develop and use abstractions in computing systems
Computing Systems	Students will create and modify computing systems
Data and Analysis	Students will create computational artifacts using data and analysis
Data and Analysis	Students will recognize and define data in computational problems
Data and Analysis	Students will communicate about data in computing
Data and Analysis	Students will develop and use data abstractions
Impact of Computing	Students will experience learning within a collaborative, inclusive computing culture and explain the steps needed to ensure that all people have access to computing.

Specific Performance Expectations in This Curriculum	
Code	Performance Indicator
AP1.a.10.h	Provide examples of computationally solvable problems and difficult-to solve problems.
AP1.a.11.h	Decompose a large-scale computational problem by identifying generalizable patterns and applying them in a solution.
AP1.a.12.h	Illustrate the flow of execution of a recursive algorithm.
AP1.a.14.h	Develop and use a series of test cases to verify that a program performs according to its design specifications.
AP1.a.8.h	Analyze a problem and design and implement an algorithmic solution using sequence, selection, and iteration.
AP2.a.11.h	Integrate grade-level appropriate mathematical techniques, concepts and processes in the creation of computing artifacts.
AP2.a.12.h	Design, develop, and implement a computing artifact that responds to an event (e.g., robot that responds to a sensor, mobile app that responds to a text message, sprite that responds to a broadcast).
AP2.a.13.h	Decompose a computational problem by creating new data types, functions or classes

AP2.a.16.h	Demonstrate code reuse by creating programming solutions using libraries and APIs. (e.g., graphics libraries, maps API).
AP3.b.10.h	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
AP3.b.8.h	Evaluate and analyze how algorithms have impacted our society and discuss the benefits and harmful impacts of a variety of technological innovations.
AP3.c.3.h	Interpret the flow of execution of algorithms and predict their outcomes.
AP3.c.4.h	Write appropriate documentation for programs.
AP4.a.12.h	Identify programming language features that can be used to define or specify an abstraction.
AP4.a.13.h	Identify abstractions used in a solution (program or software artifact) and reuse those abstractions to solve a different problem.
AP4.a.4.h	Demonstrate the value of abstraction for managing problem complexity (e.g., using a list instead of discrete variables).
AP4.a.5.h	Understand the notion of hierarchy and abstraction in high-level languages, translation, instruction sets, and logic circuits.
AP4.a.6.h	Deconstruct a complex problem into simpler parts using predefined constructs (e.g., functions and parameters and/or classes).
AP4.a.7.h	Compare and contrast fundamental data structures and their uses (e.g., lists, maps, arrays, stacks, queues, trees, graphs).
AP5.a.6.h	Design and develop a software artifact working in a team.
AP5.a.7.h	Demonstrate how diverse collaborating impacts the design and development of software products (e.g., discussing real-world examples of products that have been improved through having a diverse design team or reflecting on their own team's development experience).
AP5.b.3.h	Create design teams taking into account the strengths and perspectives of potential team members.
AP6.a.4.h	Use a systematic approach and debugging tools to independently debug a program (e.g., setting breakpoints, inspecting variables with a debugger).
CS1.a.6.h	Develop and apply criteria (e.g., power consumption, processing speed, storage space, battery life, cost, operating system) for evaluating a computer system for a given purpose (e.g., system specification needed to run a game, web browsing, graphic design, or video editing).
CS1.a.7.h	Identify the functionality of various categories of hardware components and communication between them (e.g., physical layers, logic gates, chips, input and output devices).
CS1.b.3.h	Explain the role of operating systems (e.g., how programs are stored in memory, how data is organized and retrieved, how processes are managed and multi-tasked).
CS2.a.4.h	Devise a systematic process to identify the source of a problem within individual and connected devices (e.g., research, investigate, problem solve).
CS3.a.2.h	Demonstrate the role and interaction of a computer embedded within a physical system, such as a consumer electronic, biological system, or vehicle, by creating a diagram, model, simulation, or prototype.
CS3.a.3.h	Describe the steps necessary for a computer to execute high-level source code (e.g., compilation to machine language, interpretation, fetch decode-execute cycle).
CS4.a.2.h	Create, extend, or modify existing programs to add new features and behaviors using different forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds).
CS4.a.3.h	Create a new artifact that uses a variety of forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds).
DA1.a.4.h	Convert between binary, decimal, and hexadecimal representations of data (e.g., convert hexadecimal color codes to decimal percentages, ASCII/ Unicode representation).

DA1.a.5.h	Analyze the representation tradeoffs among various forms of digital information (e.g., lossy vs. lossless compression, encrypted vs. unencrypted, various image representations).
DA1.a.6.h	Discuss how data sequences (e.g., binary, hexadecimal, octal) can be interpreted in a variety of forms (e.g., instructions, numbers, text, sound, image).
DA2.a.4.h	Discuss techniques used to store, process, and retrieve different amounts of information (e.g., files, databases, data warehouses).
DA3.a.6.h	Use computational tools to collect, transform, and organize data about a problem to explain to others.
IC1.a.9.h	Describe how computation shares features with art and music by translating human intention into an artifact