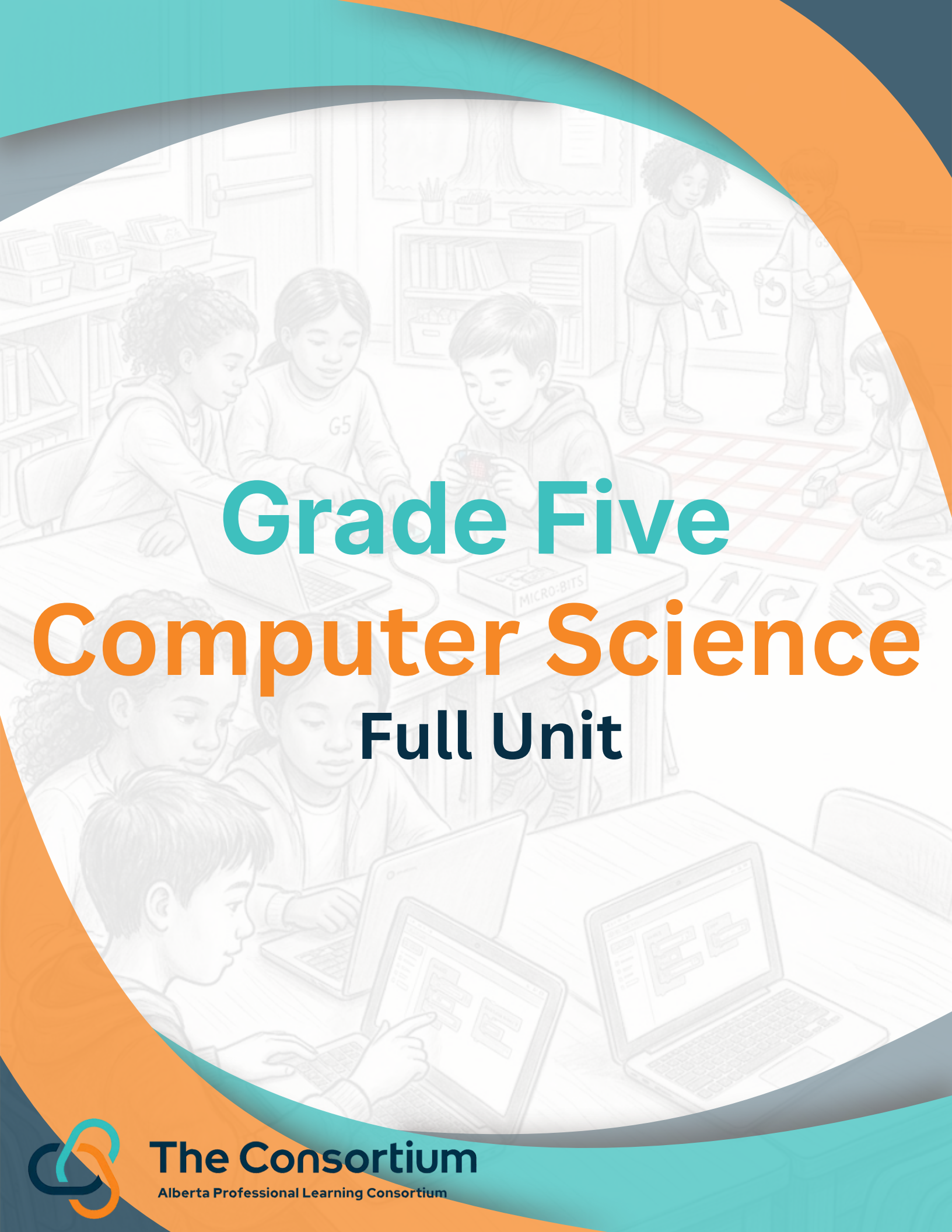


Grade Five
Computer Science
Full Unit

The Consortium
Alberta Professional Learning Consortium

Grade 4 CS Teacher Notes

This unit provides a comprehensive, design-focused curriculum for Grade 4 students. It is built on the core idea that **Design** involves processes that transform ideas into artifacts—both physical and digital—to meet specific needs. Each chapter includes force copy links to student notes, worksheets and full lesson plans.

Unit Overview

The curriculum is divided into an introduction, six core chapters, and a concluding final project. Each section focuses on a different aspect of computer science and design thinking:

Introduction: Defines design, needs, and the basic four-step design process.
Chapter 1: Deep dive into the seven iterative steps of the **Design Thinking Process**.
Chapter 2: Focuses on the power of **Feedback** and the iterative nature of design.
Chapter 3: Explores different types of **Artifacts**, such as models, blueprints, and digital programs.
Chapter 4: Applies design thinking to **Complex World Problems**, featuring case studies on mobile medical clinics and self-driving cars.
Chapter 5: Covers practical **Design Considerations**, specifically the availability and cost of materials.
Chapter 6: Introduces **Algorithms** as precise, step-by-step instructions (the "recipes" for computers).

How to Use This Unit

Each chapter follows a flexible, three-step lesson structure designed to support differentiated instruction and varied classroom technology access.

Step 1: Guided or Self-Guided Learning

Begin each chapter by introducing the core concepts. You can guide the class together or allow students to choose how they consume the information through the following provided resources:

Primary Text: Student notes and chapter content.
Multimedia: Links to videos, podcasts, and infographics.
Check for Understanding: Use the included **Student Worksheets** (available for print or digital use) to assess initial comprehension.

Step 2: Activity Selection (Differentiated Application)

Choose one (or more) of the following activity paths based on your classroom resources and student interests:

Unplugged: Collaborative, low-tech activities focusing on critical thinking.

Micro:bit: Introduction to hardware and physical computing using block coding.
Scratch: Software development and digital artifact creation through visual coding.

Step 3: Assessment

Conclude each chapter with a digital quiz. The unit culminates in a **Final Project** and a comprehensive test, allowing students to demonstrate their mastery of the design process.
∗Note you will need to use the folder copy link provided above to access quizzes/tests.

Key Resource Repository

Each chapter in the Unit Plan contains direct Google Drive template links to:
Student Notes: Detailed content summaries for student reference.
Lesson Plans: Step-by-step guides for teachers.
Worksheets: Differentiated tasks for self-guided or unplugged learning.

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Full Unit

Copy Grade 5 CS Lesson Materials List (Chapters 1-6)

Copy Grade 5 CS Combined Chapters 1-6 Lesson Plans

Open Form: Grade 5 Computer Science Unit Final Exam 💻📝

Copy Grade 5 Computer Science

Grade 5 CS Unit Materials List (Chapters 1-6)

Digital/Hardware Requirements (All Lessons)

These items are essential for all micro:bit and Scratch lessons, which make up the majority of the unit.

Item	Purpose	Quantity Needed	Notes
Internet-enabled device (Computer/Tablet)	Access to MakeCode (micro:bit) and Scratch environments.	1 per student or 1 per pair	Must be capable of running modern browsers and visual block-based languages.
micro:bit devices	Used in all micro:bit lessons (Chapters 1, 2, 3, 4, 5, 6).	1 per student or 1 per pair	The core hardware component for physical computing.
USB cables	Connecting micro:bit devices to computers for programming/transfer.	1 per micro:bit	Standard micro USB cables.
Headsets with microphones	Recommended for Lesson 4.3: The Remix Project (Scratch) for recording audio artifacts.	Optional, but recommended for group work.	Allows for quality audio recording for Scratch projects.

Unplugged Materials (Physical/Analog)

These materials are necessary for the hands-on, non-digital lessons.

Item	Description	Notes
Printer paper/cardstock	For paper prototyping the app interface and building paper planes/bookmarks.	Various colors may be helpful for Lesson 3.1 iteration.
Pencils/Markers/Colored Pencils	For sketching, writing algorithms, and creating the "Inclusive Playground" blueprint.	
Sandwich ingredients (Bread, Jam, etc.)	Used for the "Robot Chef" demonstration (Teacher acts as robot).	Used to demonstrate the necessity of precise algorithms.
Obstacle Course materials	Desks, chairs, cones, or other classroom items to create a simple course.	Course must be safe for a blindfolded student.
Designer Notebooks	Notebooks or dedicated paper for students to write and debug algorithms.	Mentioned specifically in Lesson 2.1.
Tape and Scissors	For building and iterating on the physical prototypes (paper planes, bookmarks).	
Small objects	Used for testing the prototype bookmarks (e.g., small erasers, coins).	
Large paper/poster board	For creating the group "Inclusive Playground" master blueprint.	Needs to be large enough for collaborative drawing.
"Key" templates for symbol code	Paper templates for students to map actions to symbols for the dance coder activity.	
Timer/Stopwatch	To measure the duration of the "Human Loop" fitness routine.	A digital timer on a device is acceptable.
Space for simple physical activity	Open area in the classroom or outside for the "Human Loop" workout routine.	

What is a Computational Artifact?

A **computational artifact** is a term used to describe *anything* created by a human using a computer. Think of it as a digital creation! 👩‍💻 These artifacts are the results of using computational thinking and technology.

They can be simple or very complex. When you use a computer to make something—whether it's writing a story, drawing a picture, or building a website—you are creating a computational artifact.**What is a Computational Artifact?**

A **computational artifact** is a term used to describe *anything* created by a human using a computer. Think of it as a digital creation! 👩‍💻 These artifacts are the results of using computational thinking and technology.

They can be simple or very complex. When you use a computer to make something—whether it's writing a story, drawing a picture, or building a website—you are creating a computational artifact.

A computational artifact can also be a plan or design created *for* a computer or machine to execute. For example, a detailed **algorithm** (a set of steps to solve a problem) or a **paper prototype** of an app are also considered computational artifacts because they are the results of computational thinking, even before they are coded.

Examples of Computational Artifacts

Computational artifacts come in many shapes and sizes. Here are some of the most common types you see and use every day:

1. Computer Programs and Code

At the heart of every computational artifact is code! A **computer program** is a set of instructions written in a programming language that a computer follows to perform a task. The code itself is an artifact.

Artifact	Description
Program Code	The written instructions that make software work.
Software	Applications like video games, word processors, or web browsers.

2. Images 🖼️

Any picture or graphic you create, edit, or generate using a computer is a computational artifact.

Artifact	Description
Digital Photos	Pictures taken with a phone or camera and stored digitally.
Computer Graphics	Drawings, designs, or artwork created using digital art software.
Edited Images	Photos that have been changed using editing software.

3. Audio 🎶

If sound is recorded, edited, or created with the help of a computer, it becomes a computational artifact.

Artifact	Description
Recorded Music	Songs or soundtracks recorded and stored as digital files (like MP3s).
Sound Effects	Short sounds created digitally for games, videos, or apps.
Podcasts	Recorded spoken word content distributed digitally.

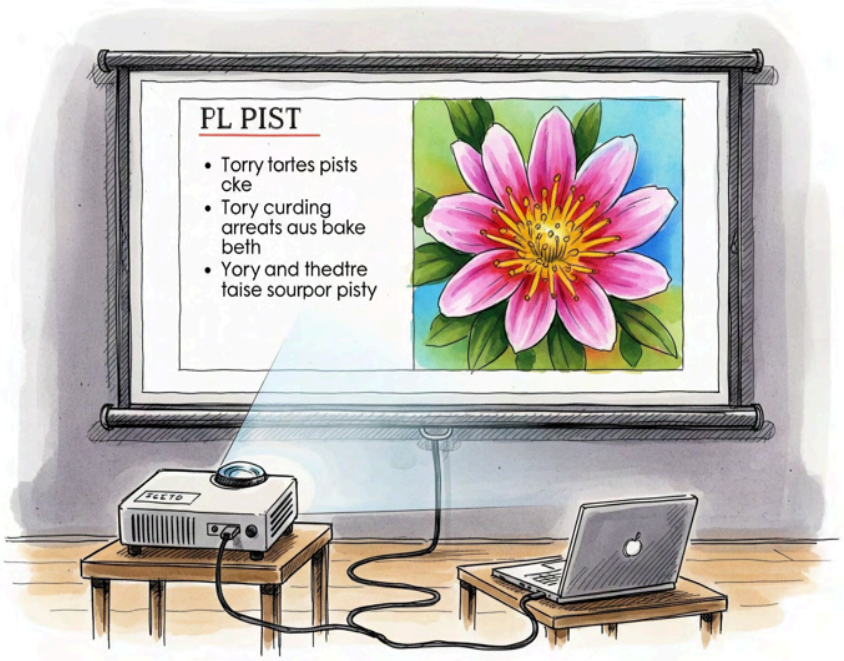
4. Video 🎬

A video is a computational artifact because it requires a computer to record, edit, and play back.

Artifact	Description
Movies and Films	Full-length features that have been digitally edited.
Online Videos	Content uploaded to platforms like YouTube.
Animations	Cartoons and moving graphics created using digital tools.

5. Presentations 📊

Presentations are created using software like Google Slides or PowerPoint. These digital files are a way to organize and share information.



Artifact	Description
Slide Decks	A collection of digital slides used to present information.
Handouts	Digital documents created from the presentation slides.

6. Web Pages 🌐

Web pages and entire websites are some of the most common computational artifacts. They are made up of code, text, images, and videos all put together using a computer.

Artifact	Description
Website	A collection of related web pages.
HTML Files	The code structure of a web page.
Interactive Forms	Digital forms used to collect information on a website.

Why are Computational Artifacts Important?

Computational artifacts are how we communicate, learn, and play in the modern world. They are the building blocks of the digital environment we live in. Everything you interact with on a smartphone, tablet, or computer is an artifact! 👍

Feature	Description
Sharing Ideas	Artifacts like presentations and web pages let people share ideas quickly around the world.
Entertainment	Video games, movies, and music are all computational artifacts that entertain us.
Learning	Digital textbooks, educational apps, and online videos are key learning artifacts.

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 1 Lesson Plans

Copy Grade 5 CS Chapter 1: What is a Computational Artifact

View Grade 5 CS Chapter 1 Infographic

View Grade 5 CS Chapter 1 Podcast

View Grade 5 CS Chapter 1 Video

Copy Grade 5 CS Chapter 1 Slides

Copy Grade 5 CS Chapter 1 Scratch Activity Worksheet

Copy Grade 5 CS Chapter 1 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 1 Unplugged Activity Worksheet

Copy Grade 5 CS Chapter 1 Self Guided Learning Worksheet

Open Form: Grade 5 CS Chapter 1 Quiz

Chapter 2: Designing Our Digital World: Algorithms and Code 🤖💻

Did you know that everything you see on a computer, a phone, or a tablet was created by someone? It's true! People called "designers" and "programmers" work together to bring apps, games, and websites to life. They use a special way of thinking called **design** to make sure the digital world works just right!

Design is like creating a blueprint or a recipe before you start building or cooking. It's the plan that helps you figure out *what* you want to make and *how* to make it.

From Design to Code: The Digital Recipe 👩‍🍳

In the digital world, design is used to create **algorithms**. An algorithm is just a step-by-step set of instructions used to solve a problem or complete a task. Think of it like this:

Task	The Algorithm (The Steps)
Brushing Your Teeth	1. Pick up toothbrush. 2. Wet the toothbrush. 3. Put toothpaste on the brush. 4. Brush for two minutes. 5. Rinse your mouth.
Making a Sandwich	1. Get two slices of bread. 2. Spread butter or mayonnaise on one slice. 3. Add cheese and ham. 4. Place the second slice of bread on top. 5. Cut the sandwich in half.

The Connection to Computers:

Designers and programmers use design to plan out the algorithms for apps and websites. Once the algorithm is designed (the steps are clear!), they **translate** those steps into **code**.

Code is the special language that computers understand. It’s like translating the cooking recipe from English into Robot Language! The computer follows the code (the translated steps) exactly, and that's how it performs the tasks you want it to, like opening an app or making a video play.

What Makes a Design Great? 🌟

When designers are planning a new app or piece of software, they don't just think about what it looks like. They have to think about many important things to make sure the final product is successful and helpful for people. These things are called **design factors**.

The design process is influenced by many factors. Here are some of the most important ones:

Factor	Definition	Example in the Real World
Functionality 🛠️	The quality of being useful to do the job for which something was designed.	A calculator app must be able to add, subtract, multiply, and divide. If it can't, it has poor functionality.
Usability 🤔	The degree of ease with which something can be used to achieve an outcome.	A remote control that is easy to hold and has big buttons that are clearly labeled has good usability. If you need a long instruction manual to use it, it has poor usability.
Safety 🛡️	How secure and protected the user and their information are.	An online game must have a safety feature to stop strangers from talking to younger players. A password protects your online accounts.
Reliability ⚙️	The ability of the software or product to work correctly every time it is used.	If a video game crashes (stops working) all the time, it is not reliable. A reliable navigation app should always give you the correct directions.
Efficiency ⚡	How well the product works without wasting time or energy.	An efficient delivery app finds the shortest route for the driver so the food arrives quickly. A search engine is efficient if it finds the answer you need in less than a second.
Aesthetics ✨	How attractive and good-looking the product is.	A website with bright colors, clear pictures, and a cool font has pleasing aesthetics. Aesthetics help make the product more enjoyable to use.

Deep Dive into Functionality 🛠️

Functionality is all about what a product *does*. It's the purpose it was built for. If something is functional, it means it *works*.

Real-World Examples of Functionality:

- **A Door:** Its function is to open and close to allow passage and keep things secure. If the handle falls off, it loses its functionality.
- **A Washing Machine:** Its function is to clean clothes. If you put dirty clothes in and they come out clean, it has good functionality.
- **A Mapping App:** Its main function is to show you how to get from one place to another. If it can’t find the place you typed in, its core functionality is broken.

A designer must first make sure the product is 100% functional before worrying about anything else!

Deep Dive into Usability 🤔

Usability is all about *how* easy a product is to use. Can a person figure out how to use the product without getting confused or frustrated?

Real-World Examples of Usability:

- **A Water Fountain:** A good design has the button right in front, and you press it to get water. This is very usable. A poorly designed fountain might have a hidden button or be too high for a Grade 5 student to reach easily.
- **A Tablet App:** If a designer wants a user to open a picture, they might put a big, clear button that says "Open Photo." This is good usability. If they hide the button in a menu with lots of confusing words, it has poor usability.
- **An Elevator:** The buttons are usually organized clearly: up buttons together, down buttons together, and the floor numbers in order. This excellent usability means almost anyone can use an elevator easily.

Good usability makes people happy to use the product, and bad usability can make people give up on it!

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 2 Lesson Plans

Copy Grade 5 CS Chapter 2: Designing Our Digital World: Algorithms and Code

View Grade 5 CS Chapter 2 Video

View Grade 5 CS Chapter 2 Podcast

View Grade 5 CS Chapter 2 Infographic

Copy Grade 5 CS Chapter 2 Slides

Copy Grade 5 CS Chapter 2 Unplugged Activity Worksheet

Copy Grade 5 CS Chapter 2 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 2 Scratch Activity Worksheet

Copy Grade 5 CS Chapter 2 Self Guided Learning Worksheet

Open Form: Digital World Design Quiz: Algorithms and Code 

Chapter 3: Design Iteration: Making Better Designs! ✨

Welcome to a fascinating topic: how we make things better through design!

Have you ever tried to draw a perfect picture or build a perfect tower on your first try? It's usually tricky, right? The best designs, whether they are toys, buildings, or apps, are rarely made perfectly the first time. They go through a process of making, testing, and improving. This is called **iteration**.

What is Iteration? 🤔

Iteration simply means doing something over and over again, but each time with small improvements. Think of it like taking a spiral staircase 🔄. You keep walking in a circle, but you are always moving higher and getting closer to the top!

The goal of iteration is to make a design that better meets a specific **need**.

Design can better meet needs through the development of multiple iterations.

When a designer or engineer develops **multiple iterations**, they are creating several different versions of their product.

Version	Description	Goal
First Iteration (Prototype 1)	The very first, rough version.	To see if the basic idea works.
Second Iteration (Prototype 2)	The version after fixing the biggest problems from Prototype 1.	To improve function and appearance.
Third Iteration (Prototype 3)	The final version after making small refinements.	To create the best possible product before launch.

Design Processes That Support Multiple Iterations

Designers use specific steps to help them make these versions better and better. The main two ways to improve a design from one iteration to the next are **enhancing** and **refining**.

1. Enhancing (Making it Better/Stronger) 💪

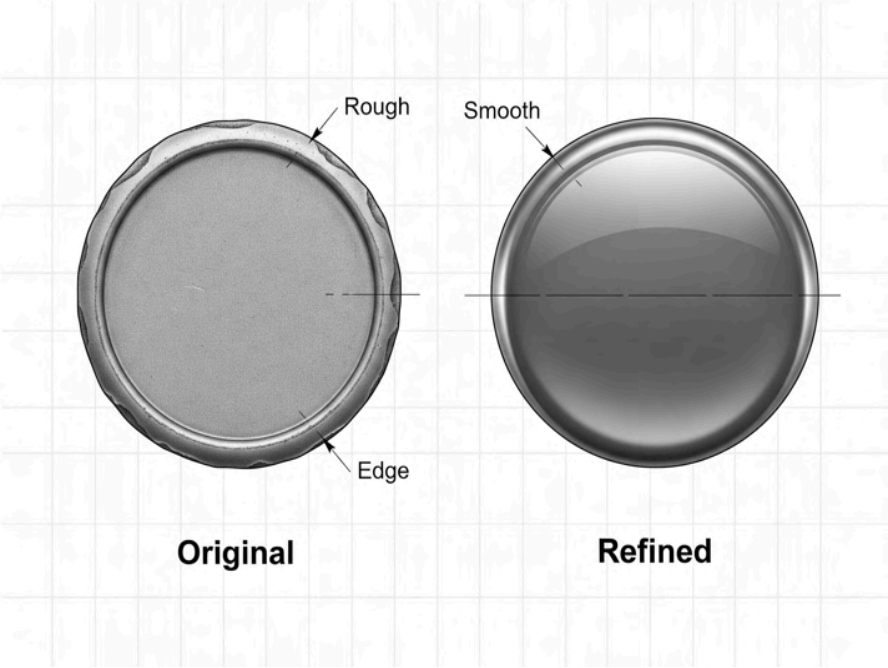
Enhancing means adding new features, making something stronger, or increasing its abilities. It's about taking the existing design and giving it a significant upgrade to solve problems or add value.

- **Example 1:** A remote-control car's wheels keep falling off.
    - **Enhancement:** The designer changes the plastic axle to a metal axle to make the wheels stronger and prevent them from breaking.
- **Example 2:** A backpack is designed for hiking, but it has no place to hold a water bottle.
    - **Enhancement:** The designer adds mesh pockets on the sides of the backpack specifically for water bottles.

2. Refining (Making it Polished/Precise) 💎

Refining means making small, precise adjustments to the design. It's about smoothing out rough edges, fixing tiny mistakes, or making the product more comfortable or efficient. It doesn't usually involve adding a big new feature, but making the existing parts work perfectly.

- **Example 1:** The buttons on a video game controller are a little too far apart for a child's hands.
    - **Refinement:** The designer moves the buttons 2 millimeters closer together so the controller is more comfortable to hold and use.
- **Example 2:** The door handle on a new toy house has a sharp edge that might scratch someone.
    - **Refinement:** The designer changes the handle's shape to have a rounded, smooth edge.



Both enhancing and refining are vital parts of the iteration process. By going through multiple iterations of **enhancing** and **refining**, designers can be sure that their final product will perfectly meet the needs of the people who will use it!

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 3 Lesson Plans

Copy Grade 5 CS Chapter 3: Design Iteration

View Grade 5 CS Chapter 3 Infographic

View Grade 5 CS Chapter 3 Podcast

View Grade 5 CS Chapter 3 Video

Copy Grade 5 CS Chapter 3 Slides

Copy Grade 5 CS Chapter 3 Scratch Activity Worksheet

Copy Grade 5 CS Chapter 3 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 3 Self Guided Learning Worksheet

Copy Grade 5 CS Chapter 3 Unplugged Activity Worksheet

Open Form: Grade 5 CS Chapter 3 Quiz

Chapter 4: Improving Design Through Collaboration 🤝

Collaboration means working together with others to create or achieve something. When it comes to design, collaboration is a superpower! Imagine trying to build a complex robot 🤖 all by yourself. It would be much harder and take much longer than if you worked with a team.

What is Design?

Design is the process of planning and creating something. This could be anything from a new video game, a piece of clothing, a bridge, or even the layout of a school playground. Good design solves a problem or makes something better.

For example, if you design a backpack 🎒, you are trying to solve the problem of carrying books and supplies. A good design might include padded straps, multiple pockets, and a durable material.

The Power of Multiple Minds

When a single person works on a design, they use only their own ideas, experiences, and skills. They might miss a problem or not think of a really great solution.

Design Approach	Source of Ideas	Potential Weakness
Individual Design	One person's ideas	Limited perspective
Collaborative Design	Many people's ideas	Clearer path for improvement

When a team works together, they bring a variety of:

1. **Skills:** One person might be great at drawing, another at building, and a third at planning.
2. **Experiences:** Someone who rides a bike every day will have different ideas for a bike rack design than someone who doesn't.
3. **Perspectives:** Different people will see different problems and suggest different solutions. This is the most important part!

How Collaboration Improves Design

Collaboration is essential because it helps to identify and fix flaws in a design early on.

1. Finding Flaws 🔎

When you are the only person looking at a design, your brain might automatically fill in the gaps or overlook mistakes. A fresh pair of eyes from a collaborator will spot things like:

- A part that is too expensive to make.
- A feature that is confusing to use.
- A color scheme that is hard to see.

If you design a poster 🖼️ for a school event, a collaborator might point out that the date is too small to read from far away. You didn't notice because you already knew the date!

2. Brainstorming Better Solutions 💡

When a problem comes up, a team is much more likely to find a creative and effective solution.

Imagine a team is designing a new water bottle 💧. The first idea is a plain cylinder. One team member says, "It's hard to carry when my hands are full."

- **Collaborator A** suggests adding a loop on the cap.
- **Collaborator B** suggests making the body of the bottle square so it doesn't roll away.
- **Collaborator C** suggests a slightly curved shape that fits better in the hand.

The final design might use the best parts of all three ideas, resulting in a **much better** water bottle!

3. Sharing the Workload 🏋️

Design projects can be huge! By sharing the workload, the team can complete the project faster and each part will get the focus it needs.

For example, when designing a school play set:

- **Person 1:** Focuses on the safety features and materials.
- **Person 2:** Designs the look and theme of the play set.
- **Person 3:** Creates the instruction manual for building it.

When people collaborate, the final design is almost always stronger, safer, and more successful than one created by a single person. Teamwork makes the dream work! ✨

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 4: Improving Design Through Collaboration

Copy Grade 5 CS Chapter 4 Lesson Plans

View Grade 5 CS Chapter 4 Video

View Grade 5 CS Chapter 4 Podcast

View Grade 5 CS Chapter 4 Infographic

Copy Grade 5 CS Chapter 4 Slides

Copy Grade 5 CS Chapter 4 Scratch Activity Worksheet

Copy Grade 5 CS Chapter 4 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 4 Unplugged Activity Worksheet

Copy Grade 5 CS Chapter 4 Self Guided Learning Worksheet

Open Form: Grade 5 CS Chapter 4 Quiz

Chapter 5: Introduction to Design and Code 💻

Welcome to the exciting world of design and code! These two ideas work together to make computers do amazing things.

Design and Algorithms

Have you ever followed a recipe or a set of directions? That's kind of like an **algorithm**! An algorithm is just a list of steps or rules to follow to solve a problem or complete a task.

Design is how we plan out these steps. We use design to create the algorithm, and then we can translate that algorithm into code that a computer can understand. Think of it like a blueprint for a building before construction begins!

What is Code?

Code is a language that a computer can understand and run. It's the set of instructions that tells the computer exactly what to do. Without code, a computer is just a box of parts.

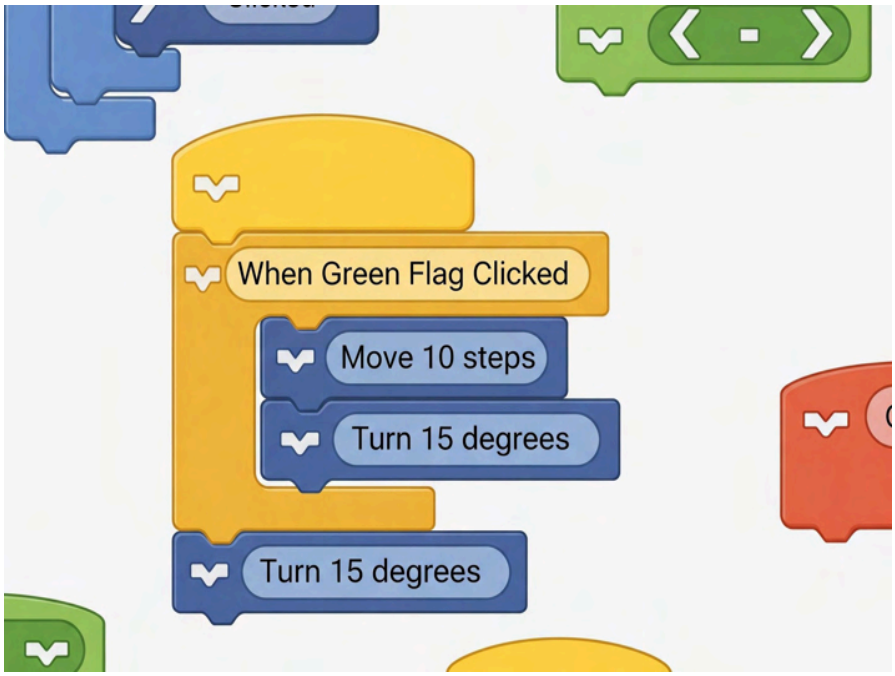
There are many different languages a computer can "speak." Some are text-based, looking like words and symbols, and others are visual.

Type of Code	Description
Text-Based	Uses words, numbers, and symbols to write instructions.
Visual Block-Based	Uses graphical blocks that snap together.

Visual Block-Based Languages 🧩

Visual block-based languages are a super fun and easy way to start coding!

Imagine you have a box of colorful puzzle pieces. Each piece has an instruction on it, like "Move Forward 10 Steps" or "Say Hello."



In a visual block-based language, these prepared chunks of instructions are in **drag-and-drop blocks** that fit together like puzzle pieces. You drag the blocks onto your screen and snap them into place to design a program. This makes it easy to see the instructions and understand how they will run.

Why Does Code Matter? 🤖

A computer is a powerful tool, but it cannot think for itself!

It must rely completely on **code** for everything it does. If you want a game to play, an app to run, or a website to load, a human had to design an algorithm and write the code for it first. The computer simply follows the code's instructions, step by step!

Remember: The computer is the loyal follower, and your code is the boss! 👍

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 5 Lesson Plans

Copy Grade 5 CS Chapter 5: Introduction to Design and Code

View Grade 5 CS Chapter 5 Video

View Grade 5 CS Chapter 5 Podcast

View Grade 5 CS Chapter 5 Infographic

Copy Grade 5 CS Chapter 5 Slides

Copy Grade 5 CS Chapter 5 Self Guided Learning Worksheet

Copy Grade 5 CS Chapter 5 Unplugged Activity Worksheet

Copy Grade 5 CS Chapter 5 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 5 Scratch Activity Worksheet

Open Form: Grade 5 CS Chapter 5 Quiz

Chapter 6: What is a Loop? 🔁

Have you ever had to do the same thing over and over again? Maybe you had to clap your hands many times during a song, or maybe you had to take many steps to walk across the playground. When you repeat an action, you are doing something similar to what a **loop** does in a computer program!

A loop is a set of instructions that a computer repeats a number of times. It's a super important idea in computer science because it lets us tell the computer to do a lot of work with only a little bit of code.

Imagine you want a robot to walk 10 steps. You could tell the robot this:

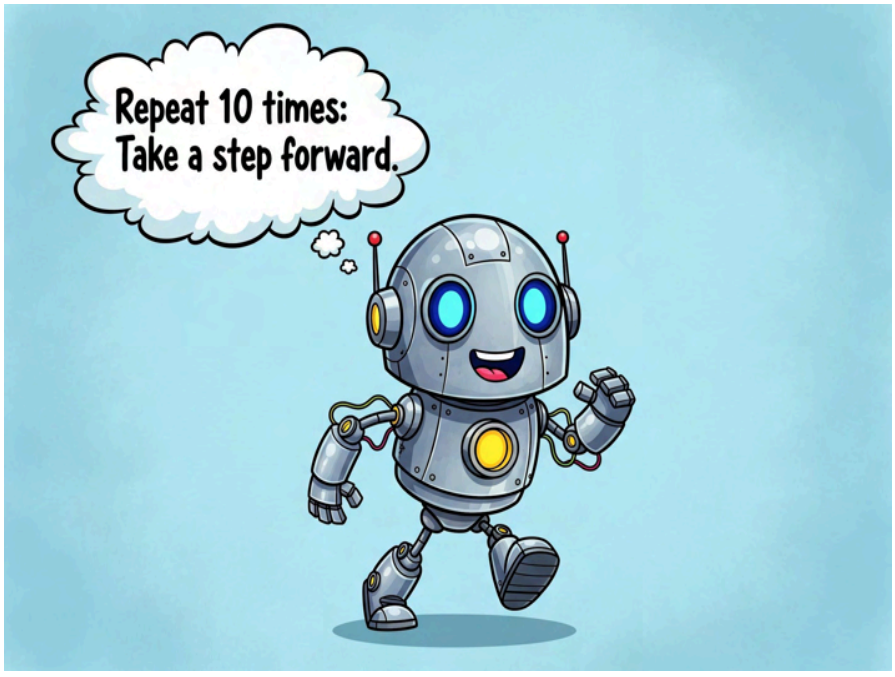
- Take a step forward.
- Take a step forward.
- Take a step forward.
- ... (and so on, 10 times!)

That is a lot of writing! 😩

A loop lets you say it much more simply:

- **Repeat 10 times:** Take a step forward.

See how much shorter that is? Loops help programmers save time and keep their instructions neat and tidy.



How Do Loops Work? 🔄

Loops follow a simple idea: repeat a set of steps until a certain goal is reached or a condition is met. Think of it like following directions!

Here are the basic parts of a loop:

1. The Instructions 📝

This is the code or action that needs to be repeated. In our robot example, the instruction is "Take a step forward." This part is also sometimes called the **body** of the loop.

2. The Repetition Rule 🛑

This is how the computer knows *how many times* or *how long* to repeat the instructions. This is the **condition** that controls the loop.

The Repetition Rule can be one of two main types:

A. Repeat a Set Number of Times (Counted Loop)

This is the simplest kind. The computer is told exactly how many repetitions to make.

Example:

Repeat 5 times: Play the drum sound. 🥁

Step	Action
Start	Set count to 0
Repeat	Play drum sound
Update	Count increases by 1
Stop	Count reaches 5

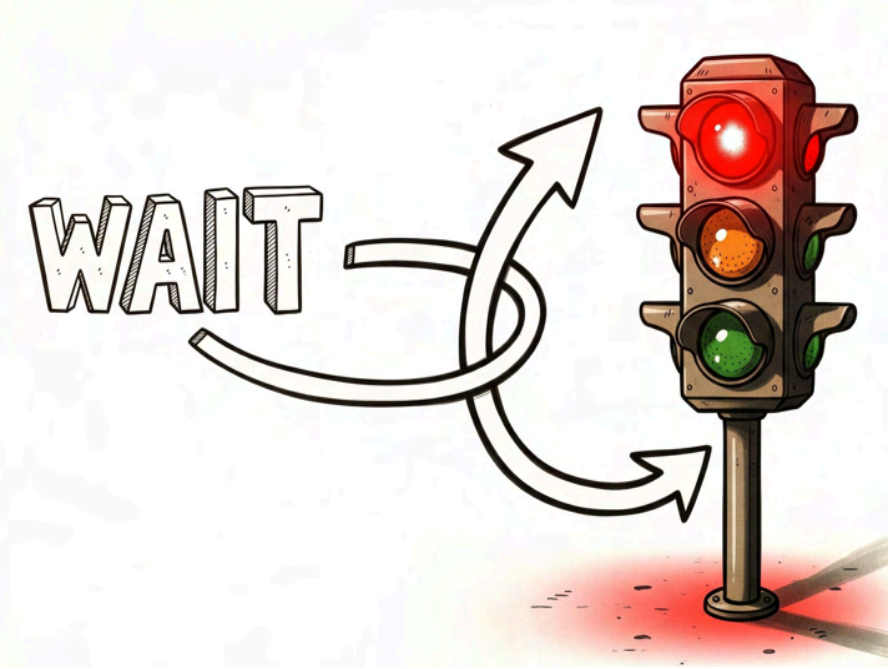
B. Repeat Until Something is True (Conditional Loop)

The computer keeps repeating the instructions until a specific condition changes. The computer doesn't know exactly how many times it will repeat when it starts.

Example:

- **Repeat while** the light is still red: Wait. 🚦

- **Repeat until** you reach the last page: Read the next page. 📖



Visualizing a Loop Flow

We can use a diagram, called a flowchart, to show exactly how a loop works in an algorithm.

Imagine we are creating a loop to draw a fencepost 4 times:

[A flowchart demonstrating a loop. It starts with an oval labeled "Start." An arrow points to a diamond shape labeled "Has the fencepost been drawn 4 times?" An arrow labeled "No" points from the diamond to a rectangle labeled "Action: Draw one fencepost." An arrow points from the rectangle back to the diamond, showing the repetition. An arrow labeled "Yes" points from the diamond to an oval labeled "End."]

What happens in the flowchart:

1. **Start:** The algorithm begins.
2. **Check Condition:** The computer asks, "Have I drawn the fencepost 4 times yet?"
3. **If NO:** The computer performs the **Action** (draws one fencepost). Then, it goes right back up to Step 2 to check the condition again. This is the **loop!**
4. **If YES:** The condition is met, the loop is broken, and the algorithm moves to the **End**.

Loops are a powerful way to make algorithms efficient and easy to write! 🚀

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Chapter 6 Lesson PlansView Grade 5 CS Chapter 6 Podcast

Copy Grade 5 CS Chapter 6: Understanding Loops in Algorithms

View Grade 5 CS Chapter 6 Video

View Grade 5 CS Chapter 6 Infographic

Copy Grade 5 CS Chapter 6 Slides

Copy Grade 5 CS Chapter 6 Self Guided Learning Worksheet

Copy Grade 5 CS Chapter 6 Scratch Activity Worksheet

Copy Grade 5 CS Chapter 6 micro:bit Activity Worksheet

Copy Grade 5 CS Chapter 6 Unplugged Activity Worksheet

Open Form: Grade 5 CS Chapter 6 Quiz

Unit Culminating Project: The "Innovation for Good" Challenge

Big Idea: Designers use the full design process—empathy, algorithmic planning, coding, and iteration—to create a functional artifact that solves a real-world problem.

🎯 Learning Objectives

- **Design:** Students will apply the design process to create a computational artifact (using micro:bit or Scratch) that addresses a specific need.
- **Code:** Students will translate a planned algorithm into a functional program including at least one **loop**.
- **Collaborate & Iterate:** Students will use peer feedback to **refine** and **enhance** their artifact through at least two iterations.
- **Evaluate:** Students will explain their artifact's **functionality**, **usability**, and **aesthetics**.

⏱️ Timeline: 3–4 Class Periods (180–240 Minutes)

Phase 1: Empathy & The Blueprint (Chapter 1, 2 & 5)

1. **Identify the Need:** Students choose a "Client" (a classmate, a teacher, or a younger student). They interview them to find a problem:
   - *Example:* "I always forget to water my plant." or "I need a way to keep track of my score in a game."
2. **The Algorithm:** Before touching a computer, students must draw a **flowchart** of their solution.
   - They must identify where a **loop** will be used (e.g., "Repeat until soil is wet" or "Forever check if Button A is pressed").

Phase 2: Building Prototype 1 (Chapter 5 & 6)

1. **Development:** Students choose their tool:
   - **micro:bit:** For physical solutions (e.g., a moisture sensor, a digital fitness tracker, or a classroom noise meter).
   - **Scratch:** For digital solutions (e.g., an educational game, an interactive story for a younger buddy, or a translation tool).
2. **Coding:** Students translate their flowchart into a visual block-based language. They must ensure the **functionality** works—does it actually solve the problem?

Phase 3: The "Fresh Eyes" Peer Review (Chapter 4)

1. **Collaboration:** Students swap projects with a partner.
2. **Feedback Loop:** Using a "Designer Notebook," the partner provides feedback on three specific areas:
   - **Usability:** Was it easy to figure out how to start the program?
   - **Aesthetics:** Does it look/sound appealing?
   - **Reliability:** Did it crash or do anything unexpected?

Phase 4: Iteration & Refinement (Chapter 3)

1. **Refining:** Students fix one "bug" or "usability" issue found by their partner.
2. **Enhancing:** Students add one "bonus feature" to add more value to the artifact (e.g., adding a sound effect when a goal is reached).

🎤 Final Presentation: The "Designer's Pitch"

Students present their final artifact to the class. Their pitch must answer:

- **Need:** What problem does this artifact solve?
- **The Loop:** Where is the loop in your code and why is it efficient?
- **Iteration:** What is one change you made after getting feedback from your collaborator?

🛠️ Materials & Tools

- **Hardware:** micro:bit V2 units, USB cables, battery packs.
- **Software:** MakeCode for micro:bit and/or Scratch.
- **Documentation:** A "Designer Notebook" (can be a digital doc or a physical folder) to store the interview notes, initial flowchart, and peer feedback.

📝 Evaluation Criteria (Based on Curriculum Outcomes)

- **Functionality:** Does the artifact perform the job it was designed for?
- **Algorithm Translation:** Is the code logically organized and does it match the plan?
- **Use of Loops:** Is a loop used correctly to create efficiency?
- **Iteration:** Is there clear evidence that the project improved from Version 1 to Version 2?

Google Drive Resources:

1. Docs, Sheets and Slides will open a Template for you to use to make your own copy.
2. mp3's (audio) and mp4's (video) will give you view only access. Make a copy!
3. Google Forms will open the responder page. You can print this page to PDF.
   Tip! Google Gemini in Forms will automatically turn this PDF into a proper Form.

Copy Grade 5 CS Final Project Rubric

Copy Grade 5 CS Final Project Lesson Plan

Copy Grade 5 CS Chapter 6 Final Project Worksheet