

Project: Building Smarter Solutions with Block Programming

Textbook

Project: Building Smarter Solutions with Block Programming



Project Overview

In this project, students will use block programming to design a **solution to a real-world problem**. They'll integrate multiple file types (e.g., images, text, or spreadsheets), handle different types of programming errors, and practice computational thinking throughout the process. The final product will be a **block-based program** that reflects purposeful algorithm design and data usage.

Learning Objectives

You will be able to:

- Use block programming to design a program with loops, conditionals, variables, and functions.
- Identify and correct syntax, logic, and runtime errors in their code.
- Integrate information from at least one external file type (text, image, CSV, etc.) into their program.

- Apply mathematical and computational thinking to plan, build, and improve their algorithms.
- Reflect on and compare the efficiency of different programming strategies.

Materials Needed

- A block-based programming environment (e.g., [Scratch](#), MakeCode, or [Tynker](#))
- Access to sample data files (e.g., .csv, .txt, image files) provided by the teacher or created by students
- A printed or digital version of the **Smart Solutions Planning Template** (provided below)
- Access to devices (laptops, tablets, or desktops)
- Optional: printed error cards with examples of syntax, logic, and runtime bugs for debugging practice

Project Steps

Step 1: Define the Problem

Brainstorm and choose a real-world problem you would like to address using code. Some ideas include:

- A reminder system for a daily task
- A quiz that helps others learn a topic
- A tracker for healthy habits or reading goals
- A visual display of fun facts pulled from a text or spreadsheet

Deliverable: Write 2–3 sentences explaining their problem and what kind of solution you will build.

Step 2: Plan the Algorithm

Using the **Smart Solutions Planning Template**, outline your program's steps using plain language or pseudocode. They must include:

- One **loop**
- One **conditional**
- One **variable**
- One **function**
- One type of file input (image, CSV, sound, etc.)

Deliverable: Completed planning sheet, approved by the teacher before coding begins.

Step 3: Build the Program

Use your chosen block programming platform to:

- Create a working program that solves the chosen problem
- Integrate at least one external file (image, sound, spreadsheet, or text)
- Use all required programming elements
- Keep track of any **errors** they encounter along the way

Extension Tip: You should be encouraged to make intentional "mistakes" at some point (e.g., using the wrong operator or leaving out a key block) and practice debugging.

Step 4: Identify and Debug Errors

After initial testing, reflect on:

- At least one **syntax error** you encountered or created on purpose
- One **logic error** that made the program behave incorrectly
- Any **runtime errors** that crashed the program or caused unexpected behavior

Now explain how you recognized each type of error and what you did to fix it.

Deliverable: "Bug Buster Journal" section in their project packet, describing the three error types and how they were resolved.

Step 5: Evaluate and Improve

Test your project with a peer and record feedback on:

- How effective their algorithm was
- Whether the data integration worked smoothly
- What could be improved about the efficiency or organization of their code

Then revise at least one part of your program to make it better.

Deliverable: Reflection page in the project packet answering: What worked well in your program? What feedback helped you improve it? What would you do differently if you built it again?

Step 6: Present Your Smart Solution

Prepare a short demo (2–3 minutes) to present your project to the class. In your presentation, you explain:

- The problem you chose and why
- How your algorithm solves the problem
- What types of input/data were used
- One error you faced and how you fixed it
- One way you used computational thinking

Deliverable: Live or recorded presentation with optional slides or screenshots.

Smart Solutions Planning Template

Section	Prompts
Problem Description	What problem are you trying to solve? Who might use your solution?
Algorithm Plan	Write the main steps your code will follow (use bullets or numbers).
Programming Elements	What variable will you use? What loop? What condition? What function?
File Integration	What type of file will you use (image, spreadsheet, etc.) and how?
Error Journal	Keep track of at least one syntax, one logic, and one runtime error. How did you fix them?
Peer Feedback	What did your partner say about your project? What did you change based on their advice?
Reflection	What did you learn from building this project? What part was most challenging or fun?

Questions (1)

1. How many steps were included in this project planning?

MULTIPLE CHOICE

Choose the correct answer:

- A. 5
- B. 7
- C. 6
- D. 3

Answer Keys & Solutions

Questions

1. How many steps were included in this project planning?

MULTIPLE CHOICE

Correct Answer:

- | | |
|------|-------------|
| A. 5 | ✗ Incorrect |
| B. 7 | ✗ Incorrect |
| C. 6 | ✓ Correct |
| D. 3 | ✗ Incorrect |