INTELLECTUAL CORE

PROGRAMMING - LEADER KEY

BADGE REQUIREMENTS CHECKLIST (BRC)
VERSION: 2026.1

SAINTS GLOBAL MEMBER

NAME: | BATTALION: | TROOP:

SKILL BADGE ADVISOR

NAME: | EMAIL: | PHONE:

LEADER KEY | WITH EVALUATION HINTS

**Gold boxes contain leader hints for evaluating each requirement*

STEP 1 | DISCOVER

INITIALS

a) Complete a digital safety briefing approved by your parent/guardian and leader (e.g., a Digital Safety video) and explain two rules you will follow to protect yourself and others online.

> **LEADER KEY**
> Look for practical safety decisions, not vague statements.

b) Discuss prevention and first aid for programming-related injuries (repetitive strain, eyestrain) and demonstrate an ergonomic workstation setup.

> **LEADER KEY**
> Ensure the Saint can explain what to do if symptoms begin.

c) Explain the history of programming and language evolution by describing at least three major milestones and why each mattered.

> **LEADER KEY**
> Listen for cause-and-effect understanding, not trivia.

d) List five popular programming languages and describe industries where each is commonly used, then name three programmed devices you rely on daily.

> **LEADER KEY**
> Look for correct, practical associations between language and use.

STEP 2 | PLAN

INITIALS

a) Explain four types of intellectual property (copyright, patent, trademark, trade secret) as they relate to software, and explain licensing vs owning software (including freeware, open source, and commercial terms). _____

> **LEADER KEY**
> Ensure understanding includes practical implications, not just definitions.

b) Select three programming languages and development environments you will use for this badge and define a simple input→decision→output project for each. _____

> **LEADER KEY**
> Plans must be feasible and measurable by demonstration.

b.1) Name each language and environment (IDE/editor/runtime) _____

b.2) Define the input type(s) and the expected output(s) _____

b.3) Identify at least one decision point (if/else, match, branching logic) in each program _____

b.4) Describe how you will test and debug each program _____

c) Define what 'agentic programming' means and plan an agent workflow that includes tools, constraints, and human approval checkpoints. _____

> **LEADER KEY**
> Look for conservative safety boundaries and clear approvals.

c.1) Define the agent's goal and what it is NOT allowed to do _____

c.2) Identify at least two tools the agent may use (e.g., calculator, file search, web requests in a safe sandbox, API calls where permitted) _____

c.3) Add an approval step before any external action (send, buy, post, delete, publish, or run code on a real system) _____

c.4) List three failure modes (bad data, hallucinated facts, prompt injection) and the guardrails you will use _____

STEP 3 | ACT

INITIALS

a) Build, debug, and demonstrate Program 1 in Language/Environment #1 that takes user input, makes at least one decision, and produces computed output. _____

> **LEADER KEY**
> Verify the Saint understands what the code is doing line-by-line at key points.

a.1) Demonstrate the program running with at least three test inputs _____

a.2) Explain the decision logic and what conditions trigger each path _____

a.3) Identify one bug you encountered (or intentionally introduce one), then show how you debugged it _____

a.4) Explain how you would improve readability (naming, comments, structure) _____

b) Build, debug, and demonstrate Program 2 in Language/Environment #2 with different input and decision logic than Program 1. _____

> **LEADER KEY**
> Look for meaningful contrast between the two builds.

b.1) Use a different decision structure than Program 1 (e.g., loops + branching, pattern match, or data lookup) _____

b.2) Demonstrate the program with at least three test inputs _____

b.3) Explain how this language/environment changes development (tooling, syntax, runtime) _____

b.4) Explain one tradeoff of this language for this problem _____

c) Build, debug, and demonstrate Program 3 in Language/Environment #3 with a new type of input/output (e.g., file, JSON, simple UI, or command-line arguments). _____

> **LEADER KEY**
> Ensure the Saint can explain the data flow end-to-end.

c.1) Accept input from a different channel than Programs 1–2 _____

c.2) Produce an output that is saved or formatted (e.g., file output, JSON, report) _____

c.3) Demonstrate at least two test scenarios _____

c.4) Explain how you verified correctness _____

d) Implement and demonstrate an agentic program that performs a multi-step task with constraints, tool use, and a human approval checkpoint. _____

> **LEADER KEY**
> This is the integrity core—ensure strong guardrails and human-in-the-loop control.

d.1) Agent receives a goal and breaks it into steps (plan or task list) _____

d.2) Agent uses at least one tool (e.g., calculator, local data lookup, file read/write in a safe sandbox) _____

d.3) Agent asks for explicit human approval before the final action (e.g., generating the final output file, sending a message draft, or applying changes) _____

d.4) Agent logs decisions and includes a 'stop rule' when uncertain _____

d.5) Explain how you defended against prompt injection or bad instructions _____

STEP 4 | REFLECT

INITIALS

a) Explain what you learned about debugging, discipline, and responsibility when software affects other people, and describe one habit you will keep as you continue programming. _____

> **LEADER KEY**
> Look for specific examples and realistic habits.

b) Identify three programming-related career pathways and describe the education/training for one you might explore.

LEADER KEY

Ensure the pathway discussion is realistic and actionable.

END OF REQUIREMENTS

BY SIGNING BELOW, I CERTIFY TO THE BEST OF MY KNOWLEDGE THAT ALL REQUIREMENTS WERE MET AT OR ABOVE THE REQUIRED STANDARDS AS OUTLINED IN THE BADGE REQUIREMENTS CHECKLIST.

SKILL BADGE ADVISOR _____ DATE *(YYYY-MM-DD)* _____