CLIXO
Classroom

Conditional Logic with Clixo

Activity Guide

Written by

Laurie Guyon

Lead Coordinator for Instructional Technology
Gansevoort, NY

Online Activity Page

Conditional Logic with Clixo

Activity Guide

CLIXO Classroom

Overview

Collaborative project for learners to construct a Clixo chain using conditional IF–THEN logic.

Objective

Practice conditional reasoning and iteration by following IF–THEN rules to guide Clixo building decisions.

Materials for Each Student

- A large pile of Clixo pieces (mixed colors and shapes, including Tridots)
- Whiteboard or chart paper to post the rules
- Flat, open workspace for chain–building
- Optional: markers for labeling flowchart sections

Skills

- Conditional reasoning (IF–THEN logic)
- Sequential thinking and turn–taking
- Pattern recognition
- Collaborative problem–solving
- Debugging and iterative refinement
- Visualizing algorithms and flowcharts

Ages

Grades 3 and up

Duration

One class period (45 minutes)

Group Size

Pairs or small groups (3–6 students)



Introduce The IF–THEN Building Chain (10 minutes)

- Present the scenario: computer programs follow rules that check conditions and decide what happens next.
- The computer checks a rule: "IF this is true, do this. If not, do something else."
  - **Look:** What is happening right now?
  - **Ask:** "Is this rule true?" (Yes or no.)
  - **Choose:** If the answer is **yes**, it follows one instruction. If the answer is **no**, it follows a different instruction.
  - **Do:** Follow the rule and act.
- Example
  - Imagine a robot deciding whether to bring an umbrella:
  - **IF it is raining → bring an umbrella ELSE → leave the umbrella at home**
  - The robot doesn't guess. It **looks** outside, **asks** "is it raining?", **chooses** yes if it is raining, and **does** take the umbrella only if the answer is yes.



Build the Chain 20–25 minutes

- The first student starts a simple Clixo chain with one piece.
- Each following student:
  - Looks at the last piece placed
  - Checks it against the three rules **(the Conditional Statements)**
  - Identifies the rule that is **TRUE**
  - Performs the required **THEN** action and adds their piece
- Pass the chain around the group, repeating the check–and–build process.
- If the creation doesn't hold up structurally (e.g., it collapses when asked to "stand up"), the group must **debug**: go back a step and rebuild before continuing.



Share and Reflect 10 minutes

- The group presents the finished chain, tracing it piece by piece.
- Students explain:
  - Which rule applied at each step
  - Where they had to debug and why
  - How the loop section felt different from the linear sections
- Facilitate a brief reflection discussion or written response:
  - How is this like how a computer program runs?
  - What would happen if two rules were true at the same time?



Optional Extension Activities

MATH EXTENSION

Count how many times each rule was triggered; graph the frequency.

COMPUTER SCIENCE EXTENSION

Have students write their three rules as pseudocode (IF/THEN statements) before or after building.

ELA EXTENSION

Write a short "program log" narrating the chain's build in sequence, using conditional language ("if... then...").

Tips

- Post the rules where everyone can see them throughout the build.
- Photograph the finished chain and flowchart stages side by side for documentation.
- Works well as a rotating-turn activity in classrooms, libraries, or makerspaces.
- Encourage students to say the rule out loud before acting — this reinforces the logic, not just the building.

Educational Standards

CSTA K–12 COMPUTER SCIENCE STANDARDS

1B–AP–10
Create programs that include sequences, events, loops, and conditionals

1B–AP–15
Test and debug a program

CCSS MATHEMATICS

CCSS.MATH.CONTENT.3.OA.D.8
Solve problems using reasoning and logical steps

CCSS ENGLISH LANGUAGE ARTS STANDARDS

CCSS.ELA–LITERACY.SL.4.4
Report on a topic with facts and relevant details

CCSS.ELA–LITERACY.W.4.2
Write informative texts to examine a topic and convey ideas

ISTE STANDARDS FOR STUDENTS

1.5.c Computational Thinker
Students break problems into component parts and use algorithmic thinking

1.4.d Innovative Designer
Students demonstrate perseverance and iterate on solutions