The "Abstract" Activity Types

Abstraction is a fundamental concept in programming as it simplifies the complexity and development of computational artifacts. For students to be able to form abstractions they must be able to identify patterns and common features of problems. Students must be able to create systems of modules, standalone parts of a program, through subdividing the main program.

Table 4: "Abstract" Activity Types

Activity Type		**Possible Technologies**
Identify/Extract Patterns	Students identify and extract patterns which are opportunities for abstraction	Mindmapping/brainstorming tools (e.g. Popplet, Coggle, MindMup), interactive whiteboard, online whiteboard (e.g. Realtime Board)
Simplify Complex Code	Students substitute parts of a code for a single segment which uses variables to account for any differences	Coding learning environments (e.g. CodeStudio, Codecademy, CodeBender, BlueJ), Mindmapping/ brainstorming tools (e.g. Popplet, Coggle, MindMup), Online diagram tools (e.g. draw.io, Google Drawings)
Assess/Use Existing Functionalities	Students assess and use existing functions, libraries, and application programming interfaces (APIs)	Coding learning environments (e.g. CodeStudio, Codecademy, CodeBender, BlueJ), development platforms/communities (e.g. GitHub, StackOverflow)
Design/Create Modules	Students design and create systems of interacting modules and abstractions	Coding learning environments (e.g. CodeStudio, Codecademy, CodeBender, BlueJ), online diagram tools (e.g. draw.io, Google Drawings)
Model/Simulate	Students represent patterns,processes, or phenomena through models and simulations	Presentation tools (e.g. Prezi, Google**Slides), video creation (e.g. WeVideo, PowToon), Online diagram tools (e.g. draw.io, Google Drawings)**