**Computational X**

*Integrating Computational Thinking Across Disciplines*

**Learning Objectives**

The Introduction to Computational Thinking class (CS1014) meets four learning objectives:

1. Explain the application of computational or quantitative thinking across multiple knowledge domains.
2. Apply the foundational principles of computational or quantitative thinking to frame a question and devise a solution in a particular field of study.
3. Construct a model based on computational methods to analyze complex or large-scale phenomenon.
4. Identify the impacts of computing and information technology on humanity.

**Pedagogical Approach**

* Multi-disciplinary 4-student groups
* Active in-class learning
* Flipped classroom
* Project-based data science context
* Scaffolded tools (e.g., blocks-to-text)
* Mastery-based approach
* Set reasonable expectations
* Sell the experience to students

**Discussion Questions**

1. Can you identify data resources that are available in your discipline that could be used for student learning through computational exploration?
2. Are there opportunities for incorporating some form of computational learning experience in your course (e.g., using games, discipline specific tools)?
3. Can you identify challenges for you or your students to the adoption of some computational thinking elements in your course?
4. What support/changes would be needed to overcome these challenges?
5. What prerequisites skills/knowledge would students need to engage in computational exploration in your course?
6. Are there research programs in your department that could contribute data resources for your use and use by others in related fields?
7. If your course/department is related to a destination area are there ways in which computational thinking could be applied in this course?
8. Would you be interested in participating in further discussions?

**Contact Information**

* Austin Cory Bart: acbart@vt.edu
* Dennis Kafura: kafura@vt.edu
* Clifford A. Shaffer: shaffer@vt.edu

**Resources for Instructors**

* How To Think Like a Computer Scientist (https://interactivepython.org)

Free, interactive online book for learning about programming.

* CodeCademy (https://www.codecademy.com/)

Free, web-based coding lessons at your own pace

* Code.org (https://code.org/)

Free, beginner-friendly resources for learning to code.

**Environments for Data Science**

* BlockPy (https://blockpy.com)

Block/text programming environment with guided practice data science problems.

* Google Sheets (https://docs.google.com/spreadsheets/)

Web-based spreadsheet software provided by Google.

* Tableau (https://www.tableau.com/)

Desktop software for exploring and analyzing datasets

**Environments for Web Development**

* Cloud 9 (https://c9.io/)

An online programming environment that includes web development tools

* Thimble (https://thimble.mozilla.org/en-US/)

An online code editor that lets you edit and publish your own web pages

**Environments for App Development**

* App Lab (https://code.org/educate/applab)

Design an app with blocks or JavaScript then share your app in seconds.

* App Inventor (http://appinventor.mit.edu/)

Simple online environment for building phone apps

**Environments for Game Development**

* Game Maker (http://www.yoyogames.com/gamemaker)

Desktop environment for making games with either blocks or text

* Scratch (https://scratch.mit.edu/)

Kid friendly programming environment for making games with blocks

**Environments for Art & Creativity**

* Processing in SketchPad (http://sketchpad.cc/)

Online environment for creating artwork by writing programs

* Pixly (http://outreach.cs.ua.edu/pixly/)

Block-based programming environment to manipulate images and create music

**Environments for Scientific Modelling**

* NetLogo (https://ccl.northwestern.edu/netlogo/)

Interactive environment with a huge gallery of scientific models

* Agent-based Models (http://www.agent-based-models.com/blog/resources/simulators/)

List of agent-based modelling simulators