Coding Master

Competency

Student understands and writes code in multiple coding languages.

Key Method

Student uses a range of coding languages to create projects.

Method Components

What is coding?

To put it simply, coding is the language that computers speak. Much like human languages, many different coding languages are used around the world, all of which have different strengths and uses.

To use computers as a tool to shape the world around us, we need to learn these languages. Luckily, coding languages are based on a system of rules. When we are writing code, we are providing a set of instructions for a computer to follow in a way that it understands. Once we understand the fundamental rules that underpin coding as a whole, we can start to apply those rules to all the different coding languages and their unique syntax.

Why should you learn to code?

One of the reasons most often shared for why young people should learn to code is “to prepare them for the jobs of the future.” This is a worthwhile goal; jobs in STEM fields are growing at a rate of close to 8%, compared to just 3.7% for non-STEM jobs. Over 70% of jobs in STEM are actually computing jobs or use computer science in a major way (US Bureau of Labor Statistics, 2021). Learning to code can be a valuable skill in the workforce.

However, you may not end up in a STEM field. Nevertheless, you should still learn how to code!

Coding is not only an enjoyable pursuit; it also helps you to understand technology. Learning to code means that, rather than being a passive user of technology, you better understand how technology is shaping our lives and societies and how you can use technology to make a positive impact in the world.

5 Skills of Great Programmers

Writing code is about more than, well, just writing code! The 5 skills embraced by great coders are:

1. **Understanding Technology Concepts**

Yes, an important part of coding is knowing how to code! Great programmers have a solid understanding of the fundamental concepts of code, including sequence, loops, conditionals, and more. They recognize that these fundamentals are consistent across most programming languages, although the exact way they are written and used may be different.

2. **Critical Thinking and Problem Solving**

Great programmers know that mistakes = learning! They recognize that *something* is likely to go wrong or present a challenge every time they work on a project. They are able to analyze problems, break them down into smaller steps, and work through them while still keeping their cool. They use their previous knowledge and experience to find appropriate solutions and approaches to achieve their goals.

3. **Communication and Collaboration**

Great programmers communicate effectively with others about their learning and work in a variety of contexts. They are able to work independently and with others to create unique projects, share experiences, and build new skills.

4. **Knowledge Constructor**

No single person, course, or video can teach you everything you need to know about coding. Great programmers know that they need to combine their knowledge and skills in new settings to find solutions to new problems and create new projects. They are also able to use a variety of resources and tools to build new knowledge independently.

5. **Digital Citizenship**

With great coding power comes great responsibility! Truly great programmers recognize the power of coding and technology to shape the world around them and use their skills responsibly. They respect the work of others, are kind online, and always keep an eye on cybersecurity to help keep themselves safe.

Supporting Rationale and Research

Canada Learning Code. (n.d.). *Learning for the digital world: A pan-Canadian K-12 computer science education framework*. https://k12csframework.ca/wp-content/uploads/Learning-for-the-Digital-Future_Framework_Final.pdf

Canada Learning Code. (n.d.). *Teachers learning code digital toolbox: Quick start guide*. https://www.canadalearningcode.ca/wp-content/uploads/teacher-quick-start-guide.pdf

Reach. (August, 2017). *Field report on K12 computer science.* https://drive.google.com/file/d/0B2eCjHNmaBGZeGpwTGlRTUJKZlU/view

Computing at School. (n.d.) *Computational thinking: How do we think about computing so that computers can help?* https://community.computingatschool.org.uk/files/8221/original.pdf

International Society for Technology in Education (ISTE), Computer Science Teachers Association (CSTA). (2011). *Computational thinking teacher resources: Second edition.* https://cdn.iste.org/www-root/2020-10/ISTE_CT_Teacher_Resources_2ed.pdf?_ga=2.206381526.472047317.1620655308-1370069785.1620246492

Computing at School. (n.d.). *What is programming?* https://community.computingatschool.org.uk/files/8222/original.pdf

International Society for Technology in Education (ISTE). (n.d.). *Computational thinking competencies*. https://www.iste.org/standards/computational-thinking

Computer Science Teachers Association (CSTA). (2012) *Special issue computer science K-8: Building a strong foundation.* https://csteachers.org/documents/en-us/25b0b43f-0817-4c08-9210-f1a96f2b1c7d/1/

Goode, D., et al. (January 21, 2020). *Computer science for equity: Teacher education, agency, and statewide reform.* https://csteachers.org/documents/en-us/8fe1ba43-45fd-4360-8820-2e0dd33a1252/1/

Google. (May 26, 2014). *Women who choose computer science: What really matters.* https://csteachers.org/documents/en-us/c0015462-38a7-41e2-b9fb-b48107953e28/1/

Sawyer, E. (2016, January 5). *Why STEM's future rests in the hands of 12-year-old girls*. TechCrunch. https://techcrunch.com/2016/01/05/why-stems-future-rests-in-the-hands-of-12-year-old-girls/

Smith, L., et al. (September 2017). *Cracking the code: The prevalence and nature of computer science depictions in media.* https://csteachers.org/documents/en-us/db6c4cf7-62d6-4d0f-921c-658a540cc10f/1/

Zilberman, A., & Ice, L. "Why computer occupations are behind strong STEM employment growth in the 2019–29 decade." *Beyond the numbers: Employment and unemployment*, vol. 10, no. 1 (U.S. Bureau of Labor Statistics, January 2021). https://www.bls.gov/opub/btn/volume-10/why-computer-occupations-are-behind-strong-stem-employment-growth.htm

Resources

Terms and Concepts

- Algorithm – a set of step-by-step instructions that tells a computer how to perform a task.
- Program – a set of step-by-step instructions that tells a computer how to perform a task. A program can be simple (includes just one algorithm) or complex (includes multiple algorithms).
- Sequence – the order in which your computer will run your code. A computer will run your code in the exact order in which it is written, so the order in which you write your code matters.
- Event – an event is something a computer is always looking for (e.g. the user pressing a button). Events are often used to trigger other actions within a program (e.g. when the user presses the A key (event), the symbol A will appear on the screen (action)).
- Loop – a repeating section of code. A loop will repeat until a certain condition is met (e.g. repeat # of times, repeat until something else happens, repeat infinitely, etc.)
- Conditional – an "if...then…" statement. The computer will choose between a set of options based on whether a condition is true or false.
- Variable – can store data. It is associated with a symbol or name that can be referred to throughout the program.
- Function – a sub-program within your code that performs a specific action. Often, a function is used to store a series of actions that will be required multiple times throughout the program, but not necessarily one after the other. By assigning a name to the function, it can be "called" at any point in the program to run the series of actions.

Submission Guidelines & Evaluation Criteria

To earn the micro-credential, you must receive a passing evaluation for Parts 1 and 3 and a "Yes" for Part 2.

Part 1. Overview Questions

1. How old are you?
2. How would you rate your confidence when it comes to STEM? (out of 5 stars)
3. How would you rate your ability when it comes to STEM?
    a. I'm a total beginner; I've never done this before.
    b. I'm pretty new to this; I have only a little bit of experience.
    c. I've got some experience and am looking to take my learning to the next level.
    d. I've got lots of experience, and I'm ready for more advanced stuff!
4. How would you rate your interest in exploring a STEM career when you get older? (out of 5 stars)
5. What are you most hoping to get out of your STEAM Hub course? Why?

Passing: The participant has responded to the survey answering all of the question prompts.

Part 2. Work Examples/Artifacts/Evidence

To earn this micro-credential, submit the following artifacts:

Artifact 1: Badges

1. Badge for THREE of the following STEAM Hub courses:
    a. Coding with Scratch
    b. Coding with Python
    c. Coding with Processing
    d. Coding with Arduino
    e. Coding with Micro:Bit
    f. App Development
    g. Wb Development
    h. Coding with Java
    i. Coding with C++

Artifact 2: STEAM Hub Course Final Project

For the STEAM Hub course you selected above, please submit a copy of your final project. It must include:

- the full project file (please do not submit screenshots)
- any relevant share settings appropriately set to allow anyone to view the project

Part 2. Scoring Guide

Artifact	"Yes"	"Almost"	"Not Yet"
Artifact 1	The course badges were provided.	N/A	The course badges were not provided.
Artifact 2	The project provided meets the expectations as outlined in the project rubric within the STEAM Hub course at a level of 80% or higher.	The project provided meets the expectations as outlined in the project rubric within the STEAM Hub course at a level of less than 80%.	The project was not provided.

Part 3. Reflection

Please write your responses below (500 words maximum).

1. What was the most challenging part of creating your project? How did you deal with these challenges?
2. What part of your project are you most proud of? Why?
3. Discuss what you have learned about coding. Could you see yourself pursuing a career in this industry? Why or why not?

Passing: Response provides reasonable and accurate information that outlines their experience with learning to code. Student demonstrates a genuine attempt to reflect on their learning process and how their learning will influence their future.