The GER^{2}I Way

PURDUE UNIVERSITY

NOT YOUR TYPICAL "GIFTED EDUCATION"

Are you ready for the next *giant leap* in your career?

Apply to our Ph.D. or postdoctoral research programs

Purdue's Gifted Education Research & Resource Institute (GER^{2}I) has received two federal grants focused on creating meaningful change in identification and services provided for students with gifts, creativity, and talents from minoritized and marginalized populations, including students with multi-exceptionality and those from low-income families. Graduate assistantships and postdocs available. **Scan the QR Code or contact Dr. Kristen Seward at ksseward@purdue.edu**

USING SCRATCH ASSESSMENT TOOLS TO IDENTIFY STUDENTS WHO ARE GIFTED IN COMPUTER PROGRAMMING

Tugce Karatas, Purdue University

National Association for Gifted Children 69th Annual Convention

SCAN ME

PURDUE UNIVERSITY | College of Education

Closing Excellence and Opportunity Gaps for Students from Traditionally Underserved Populations in Gifted Education: A Multi-Tier Systems of Support Approach

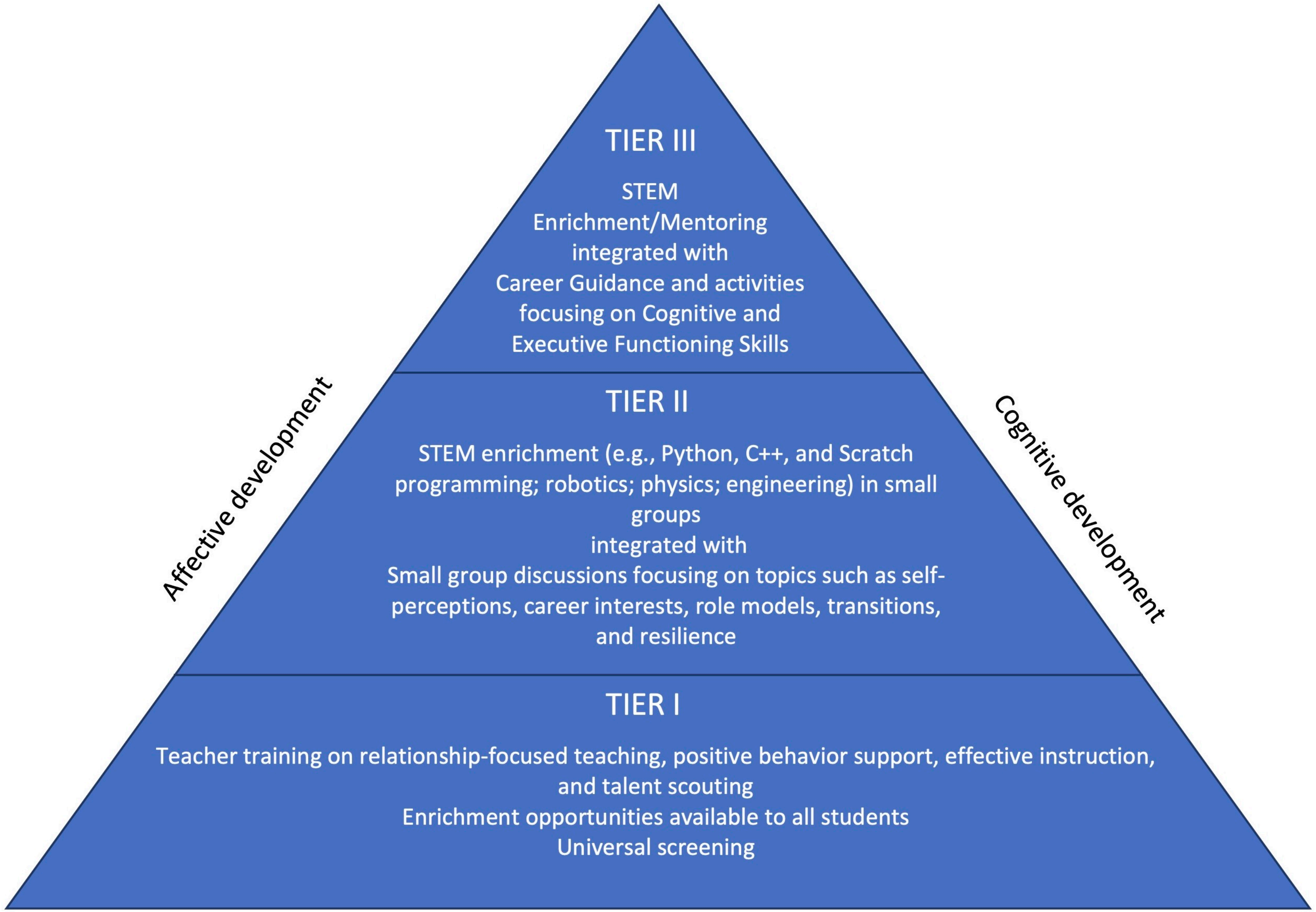
This project is funded by the Jacob K. Javits Gifted and Talented Students Education Program from the U.S. Department of Education

PURDUE UNIVERSITY | College of Education

Project Goals

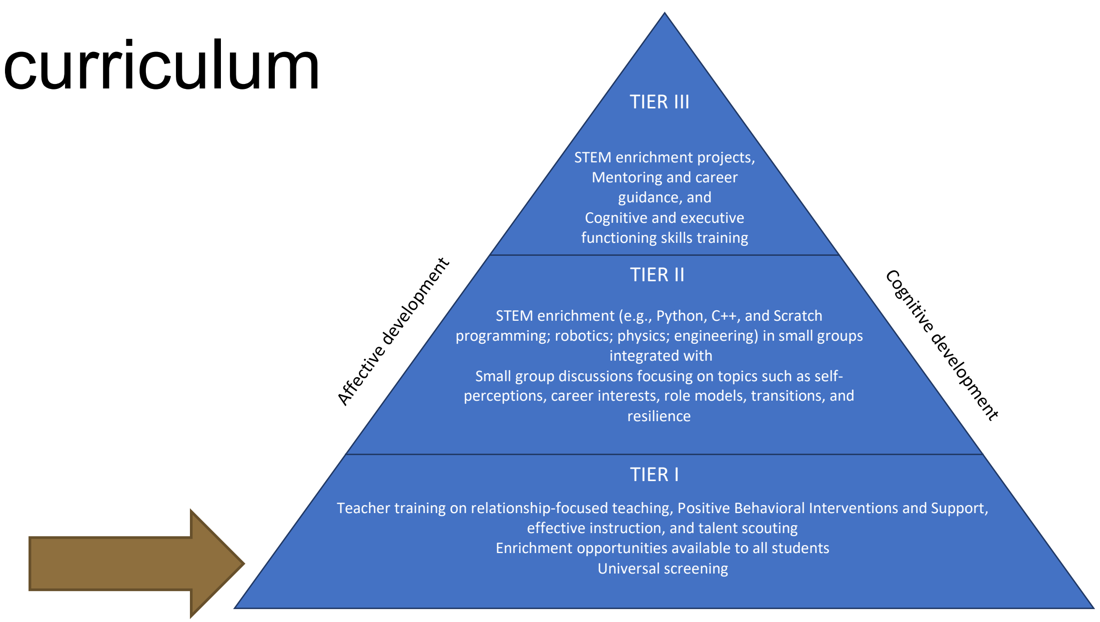
1. To **implement and evaluate the effectiveness of the Integrated STEM Talent Development** (INSTEM).
2. To **improve teacher knowledge, skills, and perceptions** regarding **socioemotional needs** and **support for talent development** for traditionally underserved students.
3. To **improve identification and access to opportunities** for students from traditionally underserved populations.
4. To **increase student achievement, engagement, motivation, wellbeing, and self-efficacy in STEM**.
5. To enable school personnel across the country to implement the INSTEM model (**dissemination of materials and research**).

Our Intervention



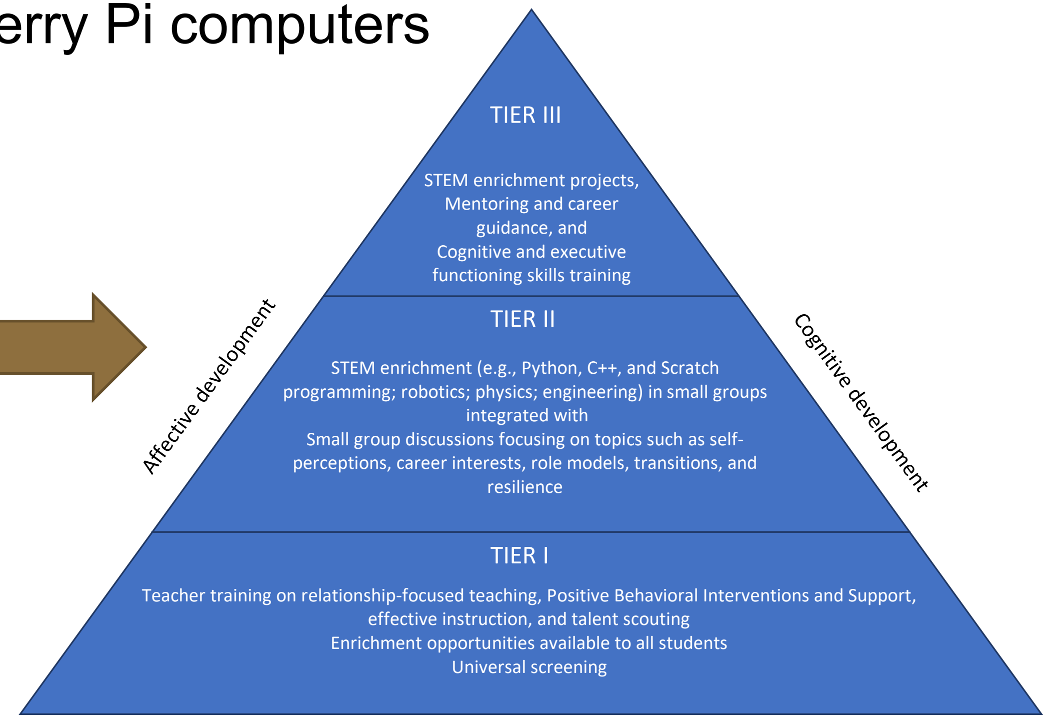
Tier I STEM Enrichment

- **Self-guided online STEM enrichment** modules (Canvas)
- Students select activities in an **area of interest**
- **Scratch-based** modules on STEAM Labs, Game Design, Internet of Things (IoT), and Autonomous Cars
- Universal access to STEM curriculum



Tier II STEM Enrichment and Affective Curriculum

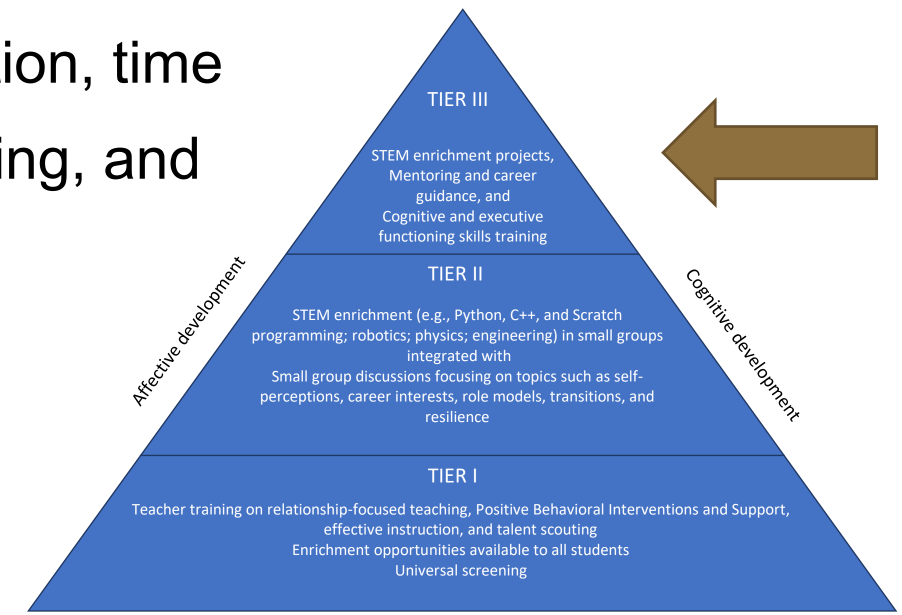
- 8-10 students per group/class, six 1-hour sessions
- STEM curriculum includes modules on Autonomous Cars, Game Design, STEAM Labs, Cybercrime, and Internet of Things, using C++ and Python programming languages with Arduinos, robotic cars, and Raspberry Pi computers
- Affective curriculum focuses on personal strengths and limitations, resilience, stress management, and career planning for STEM fields



PURDUE UNIVERSITY | College of Education

Tier III Mentoring and Affective Curriculum

- Students have **one-on-one mentoring with an expert** in the students' fields of interest
- Students complete a project that addresses a **real-life challenge** in a STEM-related area
- Affective curriculum focuses on developing cognitive and executive functioning skills, such as task initiation, time management, creative problem solving, and project planning



PURDUE UNIVERSITY | College of Education

INSTEM Identification Procedures

- Teacher Input

  - HOPE Teacher Rating Scale-STEM (11 items, asks teachers to rate each student as compared to others of similar background and experience in order to create a specific norm / comparison group.

- Student Input

  - HOPE Student Rating Scale-STEM

- Performance in Tier 1 STEM Activities
- Student achievement (used for inclusion only)

Why start with coding?

- Computer programing is rapidly becoming an essential skill in the present **tech-oriented world**.

- "A career in coding is not for everyone. However, the skills students develop while learning the coding process certainly are" (Siegle, 2017, p.117)

- Gifted and talented students excel at and are drawn to the **thinking strategies used in the coding process** (e.g., problem-solve, sequence tasks, express ideas in creative ways), which makes coding a viable option for gifted and talented students (Resnick & Rusk, 2020; Siegle, 2017).

- Coding may be seen as a type of giftedness (O'Brien et al., 2005).

Why Scratch?

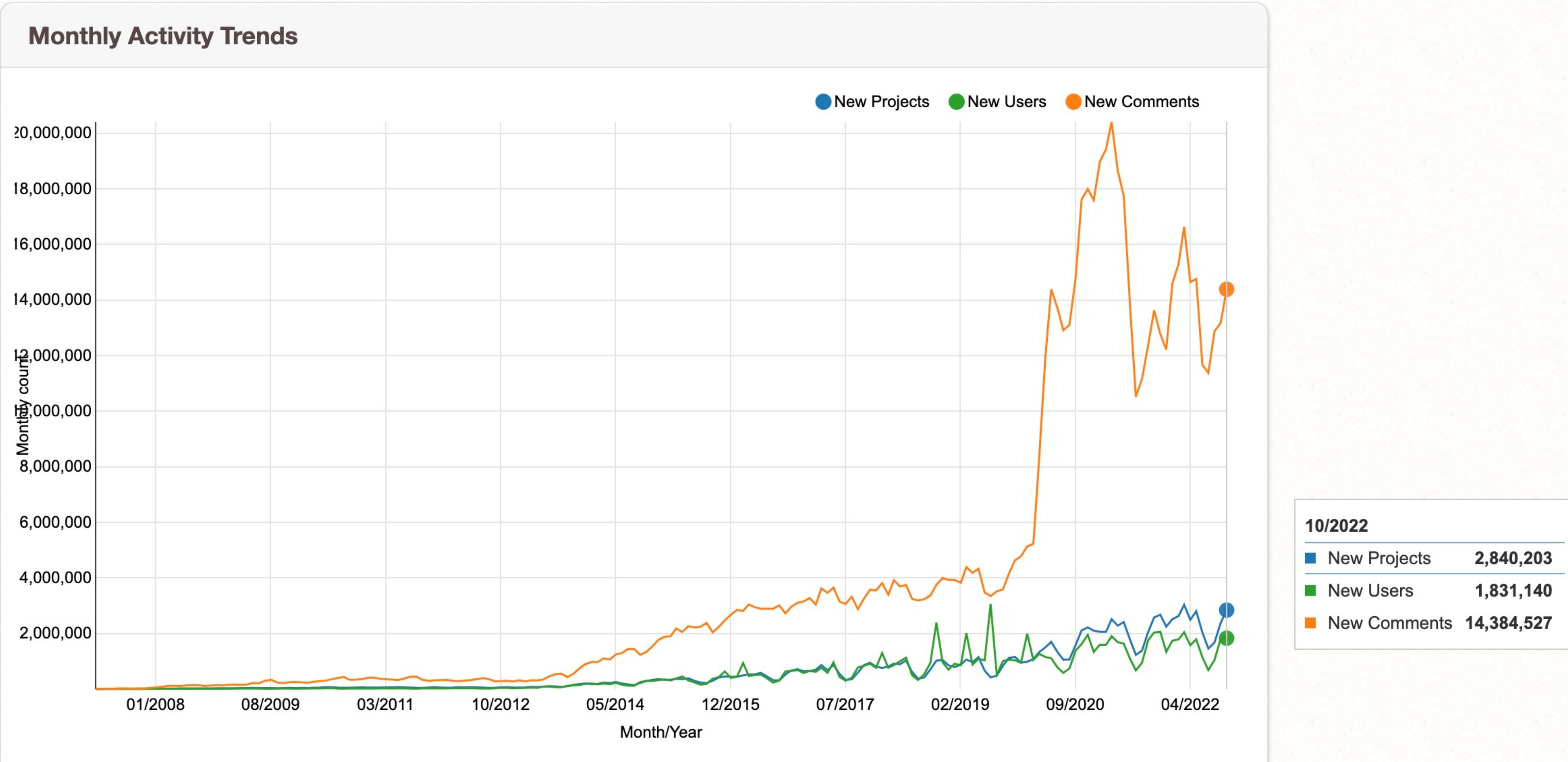
- Scratch is an **easy-to-use visual** block-based programming language.
- It is used in **more than 200 countries** and **70 languages**.
- Scratch can be used either to interest students with **little to no coding experience** or to engage them in more **creative and sophisticated project designs** based on their diverse interests and talents (Hagge, 2017; Siegle, 2017).

Why Scratch?

Four Guiding Principles (Resnick & Rusk, 2020):

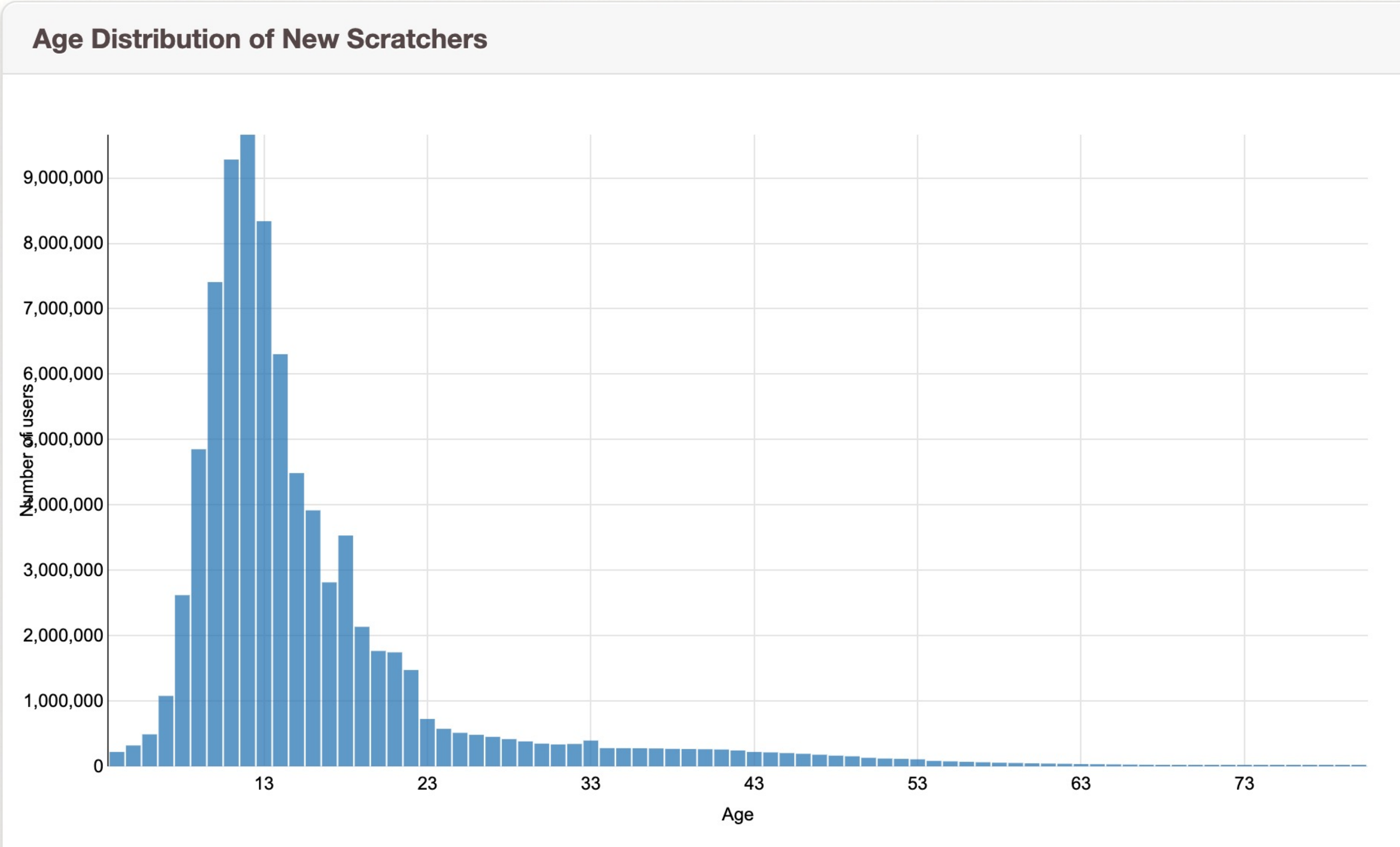
- **Projects:** students experience the process of turning an initial idea into a creation.
- **Passion:** students work on projects based on their interests.
- **Peers:** students collaborate, share and learn by remixing the work of others.
- **Play:** students try new things and experience playful experimentation and tinkering.

Why Scratch? Some statistics…



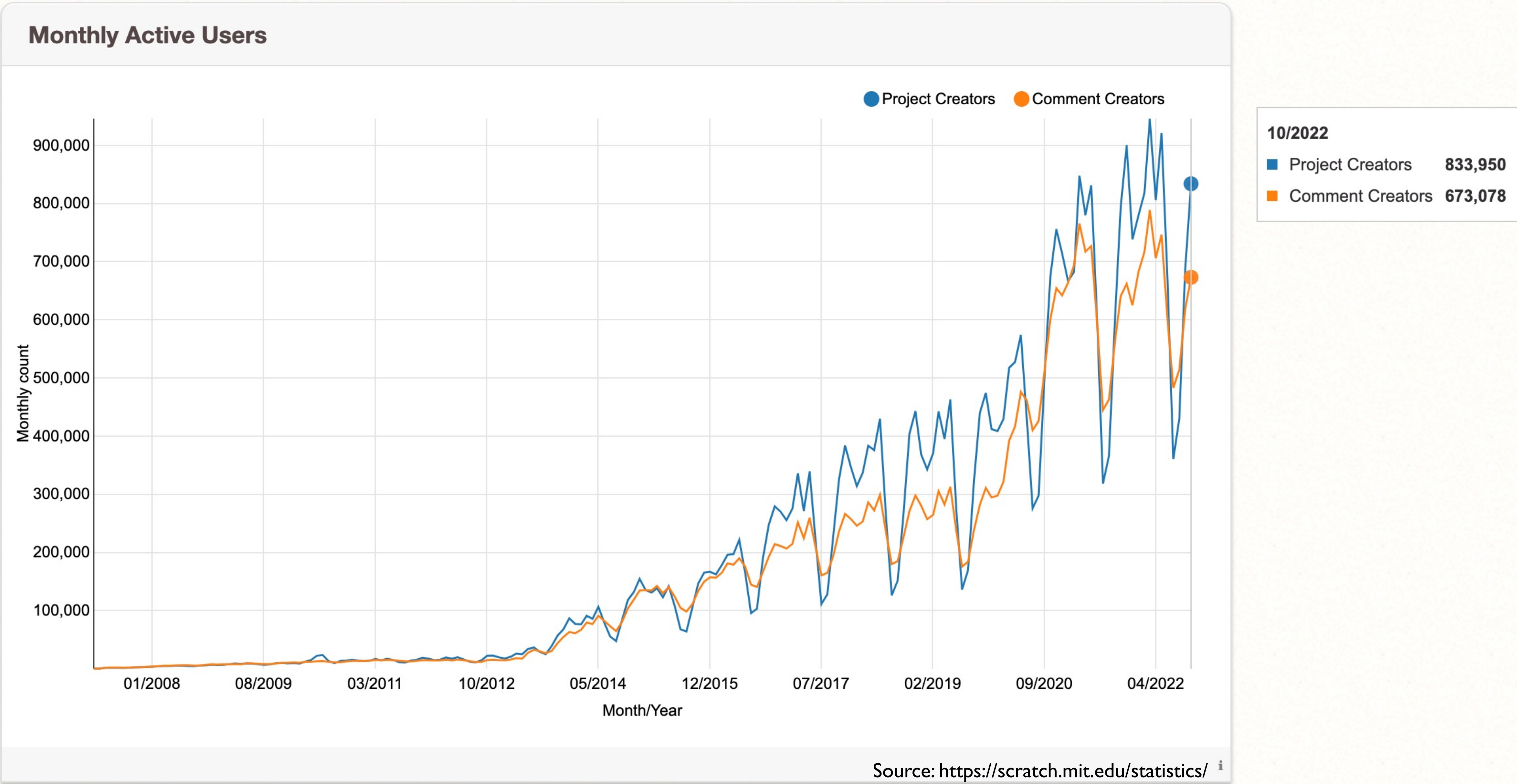
Source: https://scratch.mit.edu/statistics/

Why Scratch? Some statistics…



Source: https://scratch.mit.edu/statistics/

Why Scratch? Some statistics…



Source: https://scratch.mit.edu/statistics/

Why Scratch? Some research…

- The Scratch learning environment provides opportunities to nurture students' **potential talents** (Kafai & Burke, 2014).

- The integration of Scratch programming language into the curriculum enhances students’

  - academic performance (Moreno-León et al., 2016):

  - creativity (e.g., Hagge, 2017),

  - social skills, critical thinking, mathematical problem-solving, and self-management (Popat & Starkey, 2019).

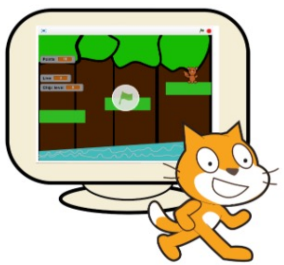
Challenges to coding (in general)

- Students are introduced to computer science with the terminology **without enough practice**.
- Students are introduced to coding by copying the exact same - code **without experimenting**.
- There is often **limited time** for learning to code in schools.
- Researchers and educators are adopting **automated assessment tools** that analyze the codes in students' projects **without considering the project details**.

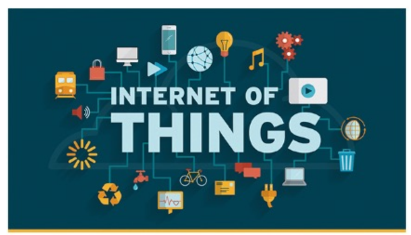
(Resnick & Rusk, 2020)

INSTEM Tier I Modules

Introduction to Scratch



Game Design



Internet of Things

STEAM Labs

Autonomous Cars

Tier I Modules: Game Design

1.Guess The Number

In this activity, you will pick a secret number between 1 and 5, then invite someone to guess your number.

2.Guess The Random Number Game

In this activity, the computer will pick a random number between 1 and 100 and the player will guess the number correctly within 5 tries

3.Load Up Your Truck

In this activity, you will design a Scratch game that asks the player to load up a truck by collecting apples falling from the top.

Tier I Modules: Internet of Things

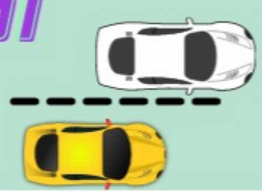
1. DETECT MY COLOR

In this activity, you will use a camera as a sensor to detect color and interact with your sprite through your camera by using colored objects.

3. POP THE BALLOONS

In this activity, you will design a Scratch project in which you will use motion and color detection together. You will pop the blue balloons by motion, and avoid touching the yellow balloons.

2. BE FAST TO BE FIRST



In this activity, you will design a drag race project, and interact with your project through a camera. You will control one of the car sprites by using motion detection. Be fast to win the race!

4. KEEP BATS AWAY

In this activity, you will design a Scratch project by using color detection. You will use a colored object to fight against the bats. You will have just 30 seconds on a countdown timer, and a bat counter will keep track of how many bats you clear on the screen.

Tier I Modules: STEAM Labs

1- INTRODUCTION TO RUBE GOLDBERG MACHINES

In this activity, you will be introduced to the concept of the Rube Goldberg Machines and learn more about its components.

2- MY FIRST RUBE GOLDBERG MACHINE

In this activity, you will complete a simple project in which you will create a ball gliding over at least two different tracks. When the ball hits the other objects, it will stop and transfer its motion to the objects.

3- FROM ONE STAGE TO ANOTHER

In this mini activity, you are going to create the animation of a sketch in Scratch. Specifically, you will focus on how the change a backdrop while animation in Scratch continues.

4-CREATE YOUR MACHINE

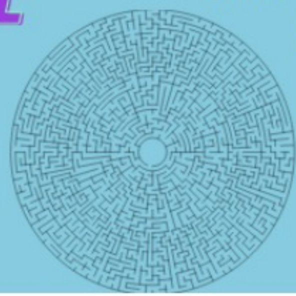
In this activity, you will create your own Rube Goldberg Machine. Find your task, plan, and create it.

Tier I Modules: Autonomous Cars

1. SIMPLE MAZE

AC2: Line Following Car

activity, you will work on your first scratch project in which an arrow will complete a simple maze that you created.



2. LINE FOLLOWING CAR

In this activity, you will first learn what a sensor is and how it works. Then, on Scratch, you will create your sensors to build a line following the car.

3. AUTONOMOUS CAR IN THE RACECOURSE

In this activity, you will draw a racetrack and create your autonomous car in order to complete the track.

4.MY SEMI-AUTONOMOUS CAR

In the activity, you will focus on the concept of the semi-autonomous car. This time, you will create a semi-smart car that is controlled by the keyboard but not going out of the racecourse even if you want to drive it out.

Computational thinking assessment

- CT assessment research is still limited, and none of the existing assessment covers all age groups and all CT concepts (Bocconi et al., 2016; Cutumisu et al., 2019).

- Three categories of CT assessment (Poulakis & Politis, 2021):
    1. **Using specific programming environments**
    2. **Using CT assessment criteria and/or psychometric tools**
    3. **Using multiple forms of assessment**

Computational thinking assessment (cont'd)

1. Using specific programming environments

-Several programming environments are used, but **Scratch is dominant** (Poulakis & Politis, 2021).

-CT concepts are assessed through **automated assessment tools**.

-In this session, we will focus on **Dr. Scratch**.

Computational thinking assessment (cont'd)

2. Using CT assessment criteria and/or psychometric tools (e.g., perceptions-attitudes scales)

3. Using multiple forms of assessment

-Project portfolios

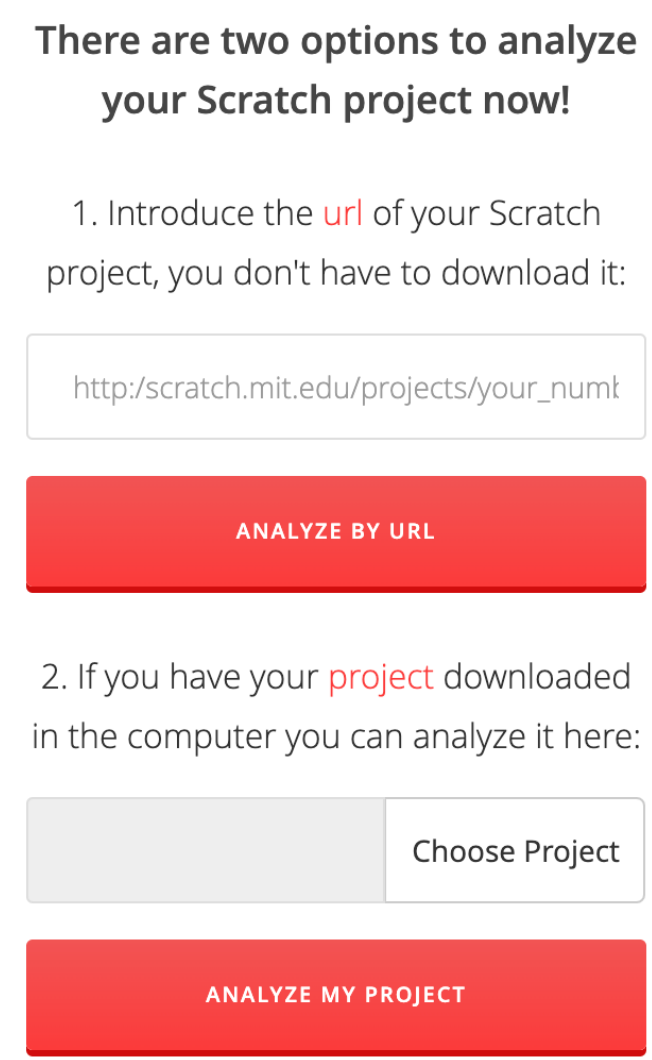
-Participant observation

-Artifact-based student interviews

Dr. Scratch

- Free and open-source automated assessment tool that analyzes Scratch projects (Moreno-Leon et al., 2015; Moreno-León et al. 2016; Moreno-León et al. 2017a, b).

- Go to: http://drscratch.org



Dr. Scratch (Cont'd)

- **Convergent validity** has been reported, meaning a strong correlation between the assessment by Dr. Scratch vs. the assessment by human experts (*r* >.70; Moreno-Leon et al. 2017).
- Debugging, design, originality, and creativity are **not** taken into consideration in Dr. Scratch.
- Dr. Scratch should be used as a **supporting tool,** not as a replacement (Moreno-Leon et al. 2017).

Dr. Scratch (Cont'd)

- Scratch projects are analyzed in seven dimensions of CT competence in grades 5-10 (Moreno-Leon et al., 2015):

1. Abstraction and problem decomposition
2. Logical thinking
3. Synchronization
4. Parallelism
5. Algorithmic notions of flow control
6. User interactivity
7. Data representation

Competence level for each CT concept (Moreno-Leon, et al., 2015)

CT Concept	Competence level			
	Null (0)	Basic (1 point)	Developing (2 points)	Proficiency (3 points)
Abstraction	-	More than one script and more than one sprite	Definition of blocks	Use of clones
Parallelism	-	Two scripts on green flag	Two scripts on key pressed, two scripts on sprite clicked on the same sprite	Two scripts on when I receive message, create clone, two scripts when %s is > %s, two scripts on when backdrop change to
Logic	-	If	If else	Logic operations
Synchronization	-	Wait	Broadcast, when I receive message, stop all, stop program, stop programs sprite	Wait until when backdrop change to, broadcast and wait
Flow control	-	Sequence of blocks	Repeat, forever	Repeat until
User interactivity	-	Green flag	Key pressed, sprite clicked, ask and wait, mouse blocks	When %s is >%s, video, audio
Data representation	-	Modifiers of sprites properties	Operations on variables	Operations on lists

Dr. Scratch (Cont'd)

Score: 18/21 Tweet

The level of your project is...

MASTER!

Youre the master of the universe!!!

Come back to your Scratch project.

Bad habits

- 2 duplicated scripts.
- 0 sprite naming.
- 1 backdrop naming.
- 6 dead code.

Project certificate

https://scratch.mit.edu/projects/626180038/

Download

Level up	Level
Flow control	2/3
Data representation	2/3
Abstraction	3/3
User interactivity	2/3
Synchronization	3/3
Parallelism	3/3
Logic	3/3

Rubric

- "Artifact analysis shows that a student built something—not that they understood something" (Salac & Franklin, 2020, p. 478).
- It is important to **assess a collection of projects** over time rather than one project.
- Rubrics should focus on the **complexity of coding concepts** rather than the number of blocks used (Basu, 2019).

Rubric (cont'd)

- Multi-dimensional rubrics for analyzing free-choice Scratch programming projects (Basu, 2019)
- Middle school students' free-choice block-based programming projects (e.g., Scratch, App Inventor)

  - Overall proficiency
  - User experience
  - Coding and CS constructs

Hands-on Activity

Let’s assess the following Scratch projects

through Dr. Scratch and a rubric!

SCAN ME

Conclusion

- There is a lot of **potential** in using evaluation assessments to identify students who are gifted/talented in computer programming.
- Students' artifacts could be considered as **alternative input** into the identification process.
- CT is a process and should not be evaluated as an end product.
- Multiple methods of assessment should be used (e.g., using interviews, and think-aloud protocols).

Selected References

Cheah, C.S.(2020). Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemp. Educ. Technol. 12.*

Hagge, J. (2017). Scratching beyond the surface of literacy: Programming for early adolescent gifted students. *Gifted Child Today*, *40*(3), 154-162.

Kafai, Y. B., Burke, Q. (2014). *Connected code: Why children need to learn programming*. Cambridge, MA: MIT Press.

Moreno-León, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it belong in the K-12 curriculum? *Journal of Information Technology Education: Research, 15,* 283-303. Retrieved from http://www.informingscience.org/Pub

O'Brien, B., Friedman-Nimz, R., Locey, J., Denson, D. (2005). From bits and bytes to C++ and web sites: What is computer talent made of? *Gifted Child Today, 28*(3), 56-64.

Ota, G., Morimoto, Y., & Kato, H. (2016). Ninja code village for scratch: Function samples/function analyser and automatic assessment of computational thinking concepts. In A. Blackwell, G. Stapleton, & B. Plimmer (Eds.), *2016 IEEE symposium on visual languages and human-centric computing* (pp. 238–239). New Jersey: IEEE.

Poulakis, E., Politis, P. (2021). Computational Thinking Assessment: Literature Review. In: Tsiatsos, T., Demetriadis, S., Mikropoulos, A., Dagdilelis, V. (eds) Research on E-Learning and ICT in Education. Springer, Cham. https://doi.org/10.1007/978-3-030-64363-8_7

Popat, S., & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education*, *128*, 365-376.

Siegle, D. (2017). Technology: Encouraging creativity and problem-solving through coding. *Gifted Child Today*, *40*(2), 117-123.

THANK YOU

Tugce Karatas, tkaratas@purdue.edu

For more information about the INSTEM project, please see our website:

http://instem.education.purdue.edu

Or contact the Principal Investigator:
Nielsen Pereira, npereira@purdue.edu