Computational Thinking

# Unplugged CT activities for primary and lower secondary school



**CT in STEM Education online webinar**
**March 5th 2026**

Link to slides:
tiny.cc/CTstemBea

Bea (Bjarnheiður Kristinsdóttir)
bjarnhek@hi.is
Faculty of Subject Teacher Education, School of Education and
Faculty of Physical Sciences, School of Sciences and Engineering

UNIVERSITY OF ICELAND













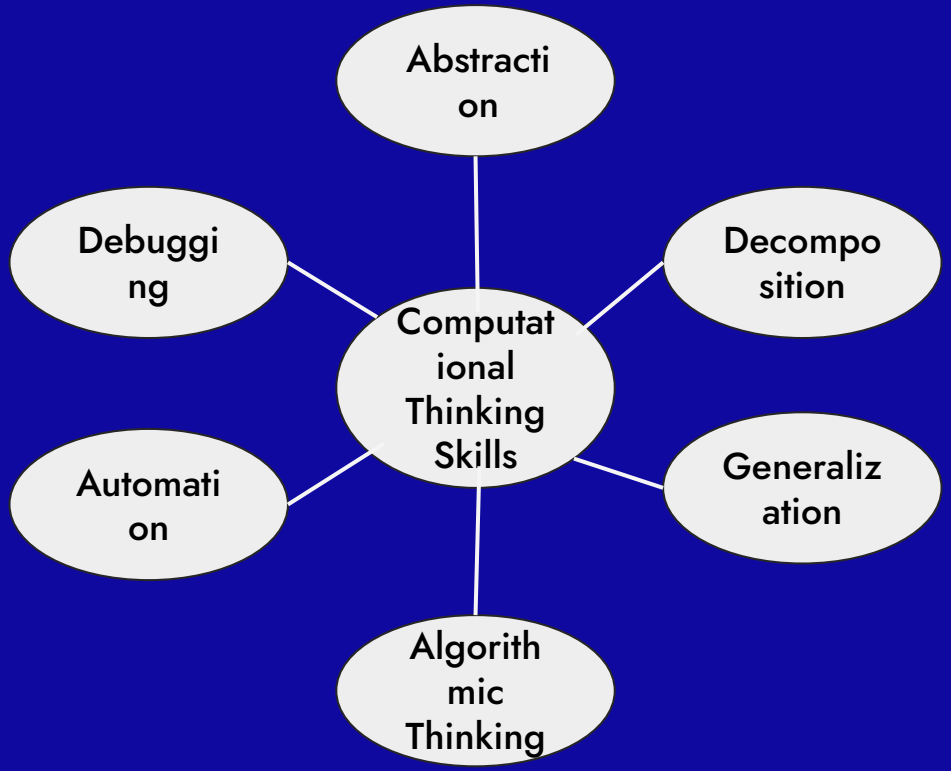
394.530 inhabitants (2025)

103.125 $km^2$

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Computational Thinking Skills

"[Computational Thinking] represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use." (Wing, 2007)



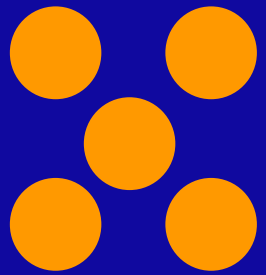

(Bocconi et al., 2022)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Abstraction

- Emphasizing important features
- Reducing complexity by blurring unnecessary details
  - Choosing the right details to hide so that the problem becomes easier, without loosing anything that is important
- Replacing real world objects with models
  - Selecting good representations



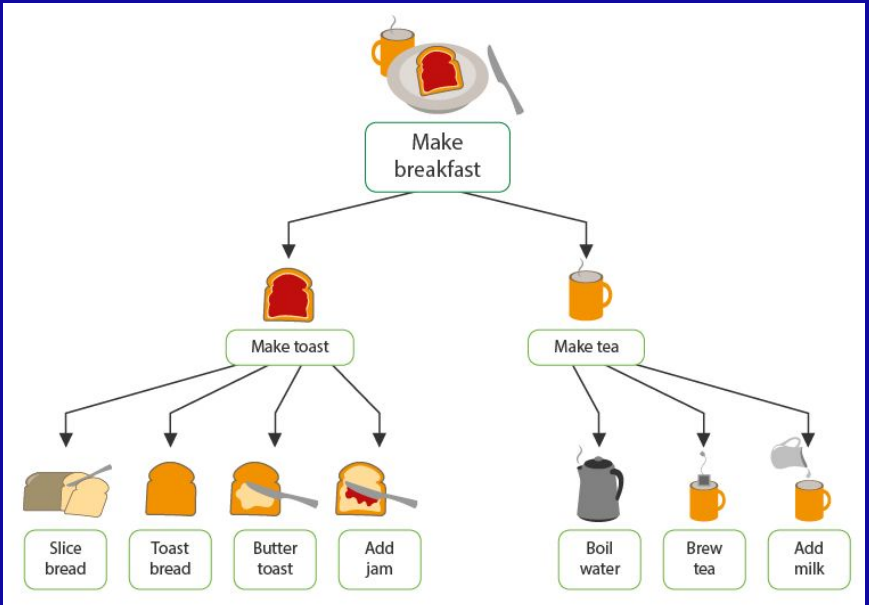


(Bocconi et al., 2016)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Decomposition

- Breaking problems down into (simpler/smaller) sub-problems, tasks or parts

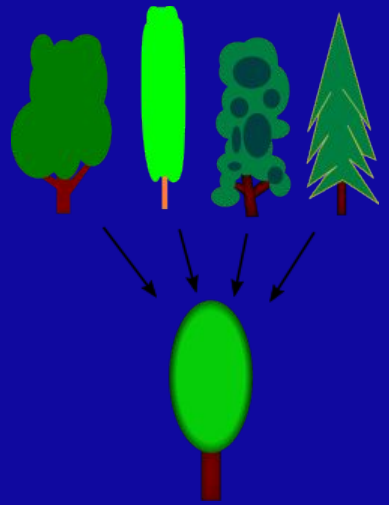



(Bocconi et al., 2016)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Generalization

- Identifying patterns, similarities and connections
- Recognizing the problem as an instance of a more general set of problems
  - Solve a problem based on previous solutions to other problems, building on prior experience
- Noticing recurrent patterns that can be repeated (loops)

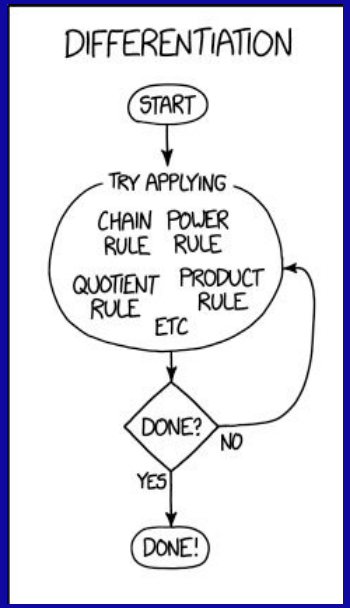


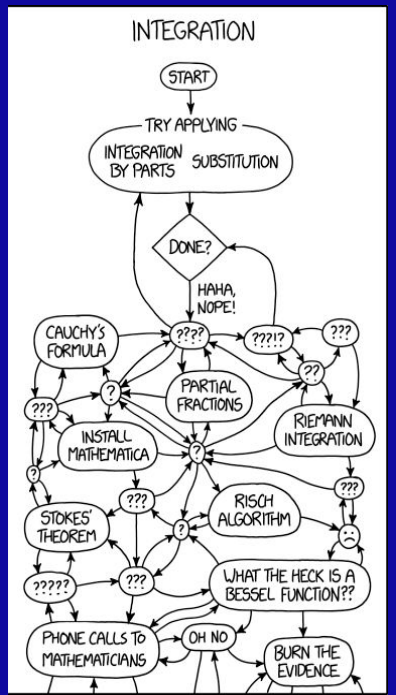
(Bocconi et al., 2016)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Algorithmic Thinking

- Creating a series of clearly defined steps (an algorithm) that humans and machines can follow



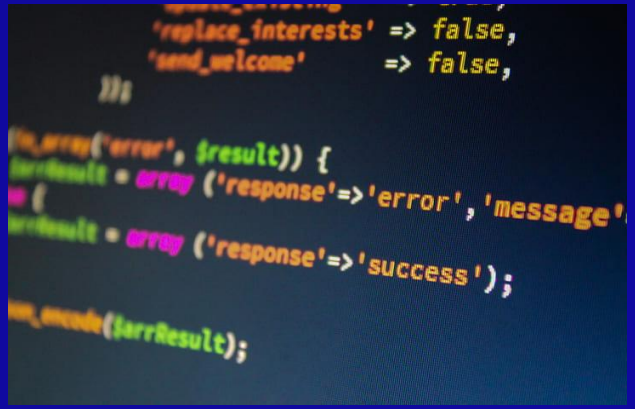

(Bocconi et al., 2016)

Illustrations: xkcd.com

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Automation

- Labour saving process
- Instructing a computer to execute a set of repetitive tasks quickly and efficiently



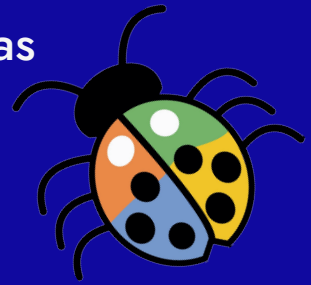

(Bocconi et al., 2016)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Debugging

- Finding mistakes and fixing them
- Systematic application of analysis and evaluation using skills such as
  - Testing
  - Tracing
  - Logical thinking

  to predict and verify outcomes
- Going through an algorithm step-by-step and understanding what it does



(Bocconi et al., 2016)

CC BY-NC-ND Bjarnheiður Kristinsdóttir



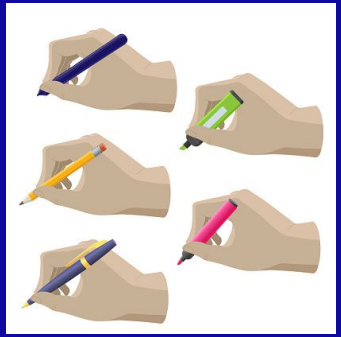
UNIVERSITY OF ICELAND

"Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction." (Wing, 2007)

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Normally, we would need...

Something to draw with, e.g. pencil, pen, marker, digital stylus



Something to draw on, e.g. paper, whiteboard, tablet



Illustrations: artpaseka and genAI

CC BY-NC-ND Bjarnheiður Kristinsdóttir

# For simplicity in this webinar session, we can use GeoGebra

UNIVERSITY OF ICELAND

Students,

Join the lesson at www.geogebra.org/classroom with the code:

**THN2 ZXBH**

Or you can also share the following link with your students:

www.geogebra.org/classroom/thn2zxbh

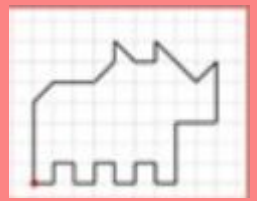
CC BY-NC-ND Bjarnheiður Kristinsdóttir

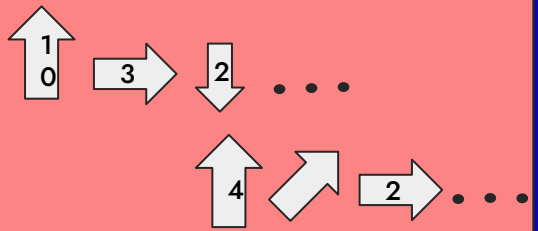
# Drawing Activity

UNIVERSITY OF ICELAND

**Receive an image.**








**Create drawing instructions.**

**Give your instructions to a partner.**



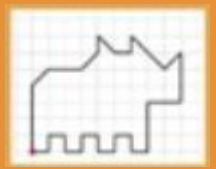


**They will draw accordingly.**

**Do the drawings match?**







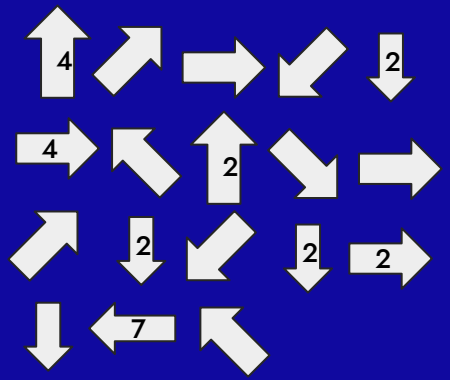


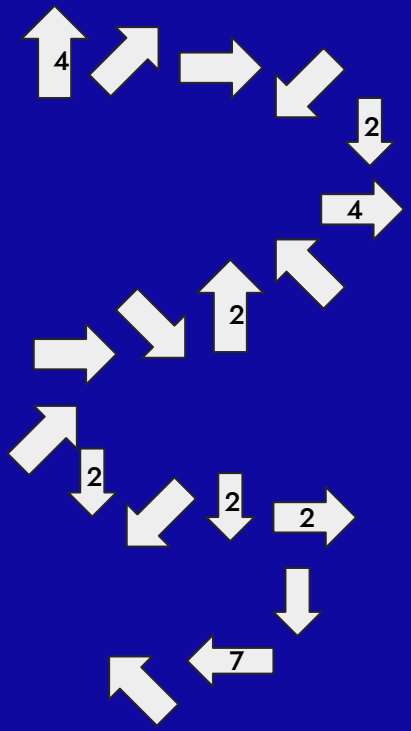
**Where / Why did the error occur?**

CC BY-NC-ND Bjarnheiður Kristinsdóttir

Adapted from the ongoing Digital first project inspired by Bebras(R) computing challenge

UNIVERSITY OF ICELAND



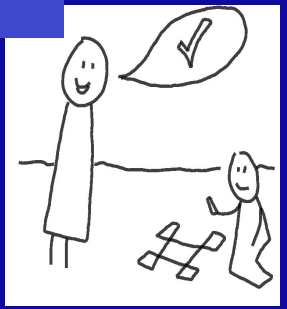




CC BY-NC-ND Bjarnheiður Kristinsdóttir

UNIVERSITY OF ICELAND

Original version

Outline version






CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Drawing Activity: Which CT skills?

UNIVERSITY OF ICELAND

**Abstraction:** emphasizing important features, reducing complexity by blurring/ignoring unnecessary details.

**Decomposition:** breaking problems into smaller manageable sub-problem parts.

**Generalization:** solving the problem by drawing on previous experiences, involves pattern recognition, noticing similarities, connections, and processes that can be repeated.

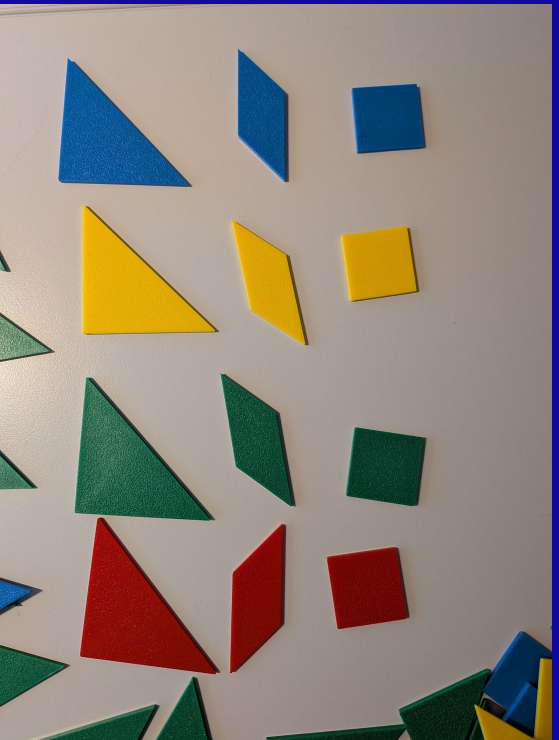
**Algorithmic Thinking:** developing an algorithm, i.e. creating a series of clearly defined steps that humans and machines can follow.

**Automation:** using a computer for repetitive tasks quickly and efficiently.

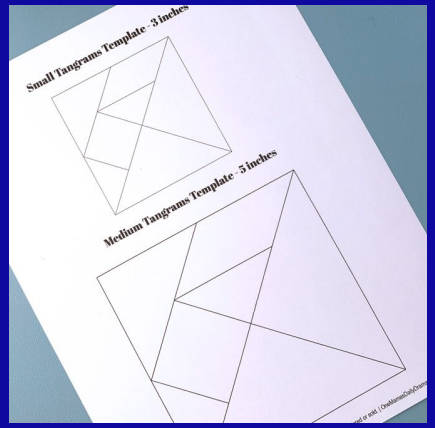
**Debugging:** finding mistakes and fixing them (testing, tracing, logical thinking).

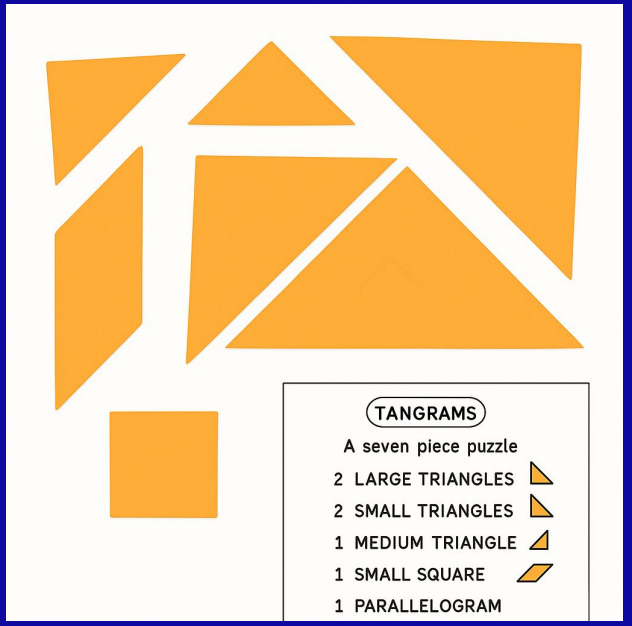
CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Normally, we would use



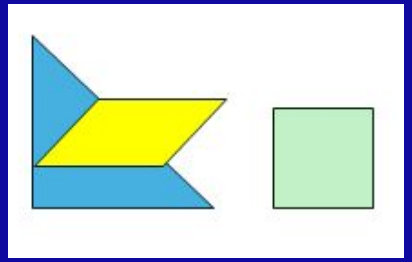
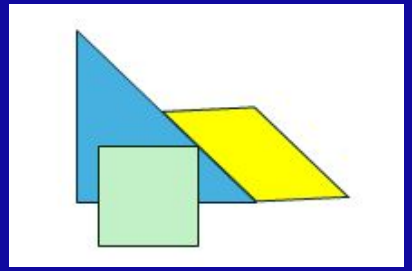
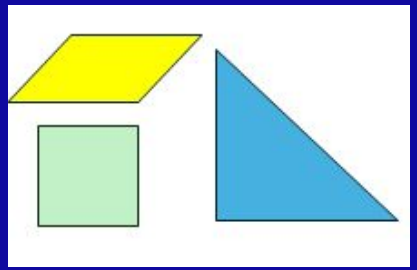
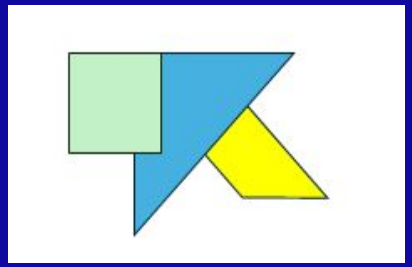
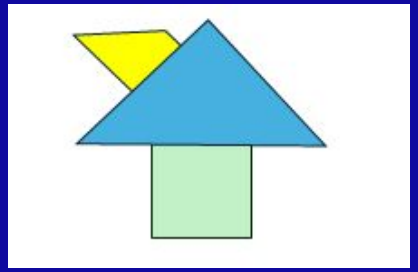
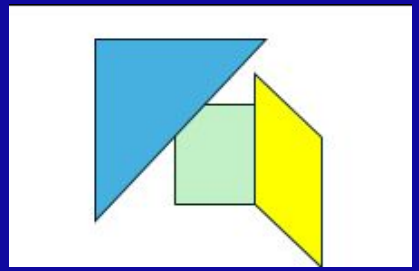
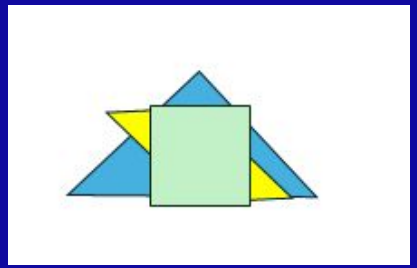
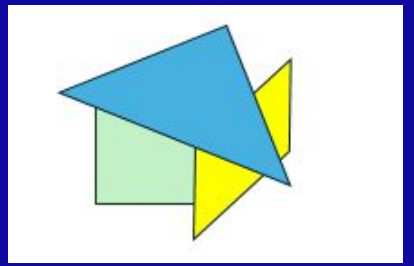





CC BY-NC-ND Bjarnheiður Kristinsdóttir

# Relative Positioning Activity

Students received a stack of cards with various arrangements of the pieces.

Adapted from the ongoing Digital first project inspired by Bebras(R) computing challenge

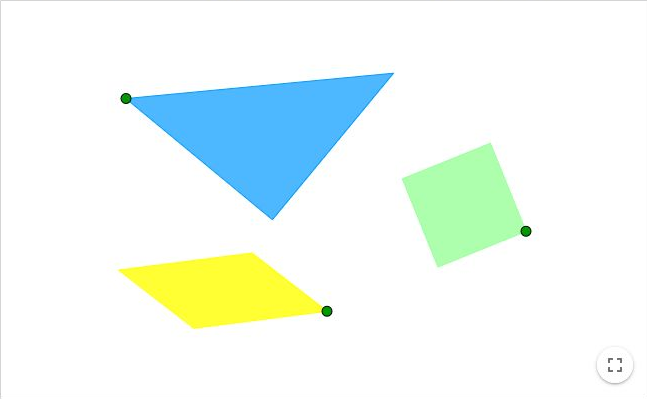
UNIVERSITY OF ICELAND

Relative positioning activity

In the first applet you will arrange the three tangram pieces as you like.
In the second applet you will follow instructions from a peer (based on their arrangement of the pieces). Then you can swap (you describe your original arrangement and they follow your instructions).

Task 3 Student progress: 7 out of 31 DETAILS

Task 3: Arrange the pieces. You can turn and stack them if you like. Describe your arrangement to your peer and they will use the next applet window to follow your instructions. Then you swap.



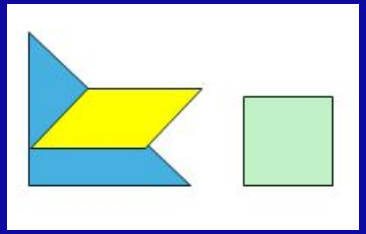
Task 4 Student progress: 3 out of 31 DETAILS

Task 4: Follow instructions from your peer regarding how to arrange the pieces.



We use digital manipulatives today instead.

(same link as before, just scroll down for the next activity)



# Activity: Which CT skills?

UNIVERSITY OF ICELAND

**Abstraction:** emphasizing important features, reducing complexity by blurring/ignoring unnecessary details.

**Decomposition:** breaking problems into smaller manageable sub-problem parts.

**Generalization:** solving the problem by drawing on previous experiences, involves pattern recognition, noticing similarities, connections, and processes that can be repeated.

**Algorithmic Thinking:** developing an algorithm, i.e. creating a series of clearly defined steps that humans and machines can follow.

**Automation:** using a computer for repetitive tasks quickly and efficiently.

**Debugging:** finding mistakes and fixing them (testing, tracing, logical thinking).

CC BY-NC-ND Bjarnheiður Kristinsdóttir

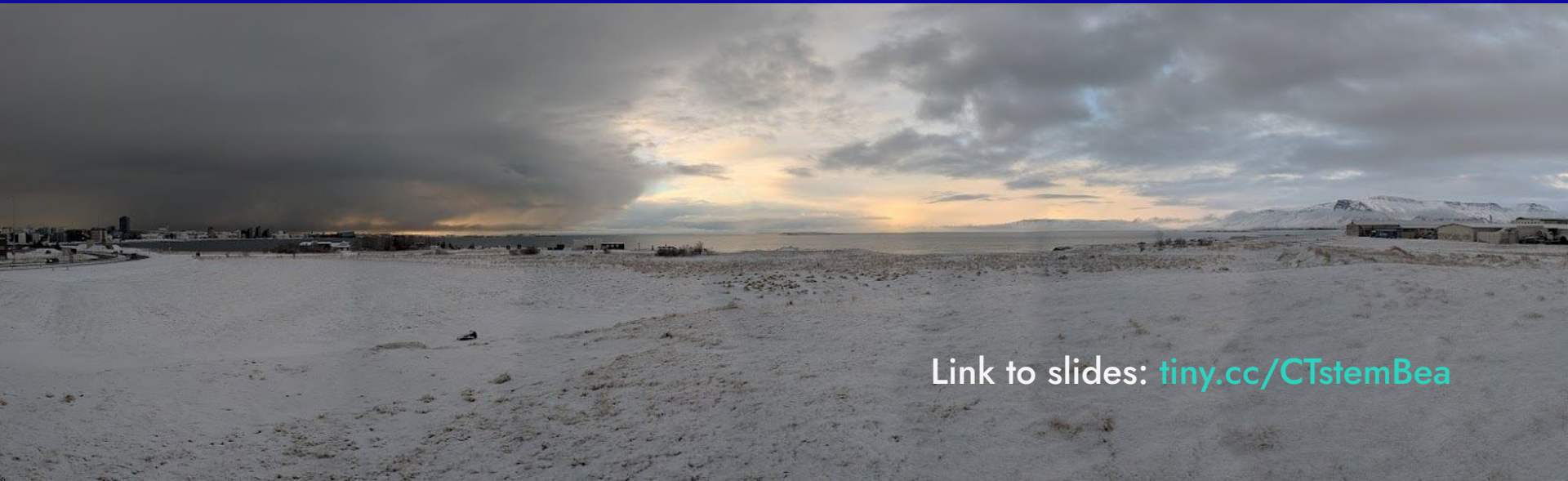
# Thank you!

Questions and comments are welcome



UNIVERSITY OF ICELAND

✉ bjarnhek@hi.is



Link to slides: tiny.cc/CTstemBea

# References


Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing Computational Thinking in Compulsory Education - Implications for policy and practice*. Publications Office of the European Commission. https://doi.org/10.2791/792158

Bocconi, S., Inamorato dos Santos, A., Cachia, R., Chioccariello, A., Giannoutsou, N., Kampylis, P., Dagienė, V., Punie, Y., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M., Jasutė, E., Malagoli, C., Masiulionytė-Dagienė, V., Stupurienė, G. (2022). *Reviewing computational thinking in compulsory education: State of play and practices from computing education*. Publications Office of the European Union. https://data.europa.eu/doi/10.2760/126955

Building computational thinking without screens: unplugged activities for students (2025, June 23). Digital First Network. https://digitalfirstnetwork.eu/building-computational-thinking-without-screens-unplugged-activities-for-students/

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

My GeoGebra page: geogebra.org/u/bjarnheidur.kristinsdottir

# Bonus Activity ggbm.org/m/dp4txrmw

In your mind: Create a rule

Drag objects that fulfill the rule to the upper part (above the horizontal line).

Other objects stay.

Can your peers guess what your rule was, based on what you dragged to the upper part?

