A Large-scale Evaluation of a Rubric for the Automatic Assessment of Algorithms and Programming Concepts

Nathalia da Cruz Alves, Christiane Gresse von Wangenheim,
Jean Carlo Rossa Hauck, Adriano Ferreti Borgatto

Federal University of Santa Catarina - Florianópolis/Brazil

SIGCSE 2020
P·O·R·T·L·A·N·D

COMPUTAÇÃO NA ESCOLA

GQS
Grupo de Qualidade de Software

FEDERAL UNIVERSITY OF SANTA CATARINA

Context

Home › News & Events › Computer programming and coding in schools

Computer programming and coding in schools — an emerging trend

News | 06.03.2015 | 3 | 6 | 150

Why every child should learn to code

Will every job involve programming? No. But it is crucial we equip future generations to think about the world in a new way

Why schools kids should be taught programming skills

Programming skills are the number one skills needed now among tech skills for a great career in the future. Here are the reasons programming skills should be taught to students right from school.

India Today Web Desk
New Delhi
March 2, 2020 UPDATED: March 2, 2020 18:33 IST

The New York Times

INTERNATIONAL EDUCATION

Adding Coding to the Curriculum

By Beth Gardiner March 23, 2014

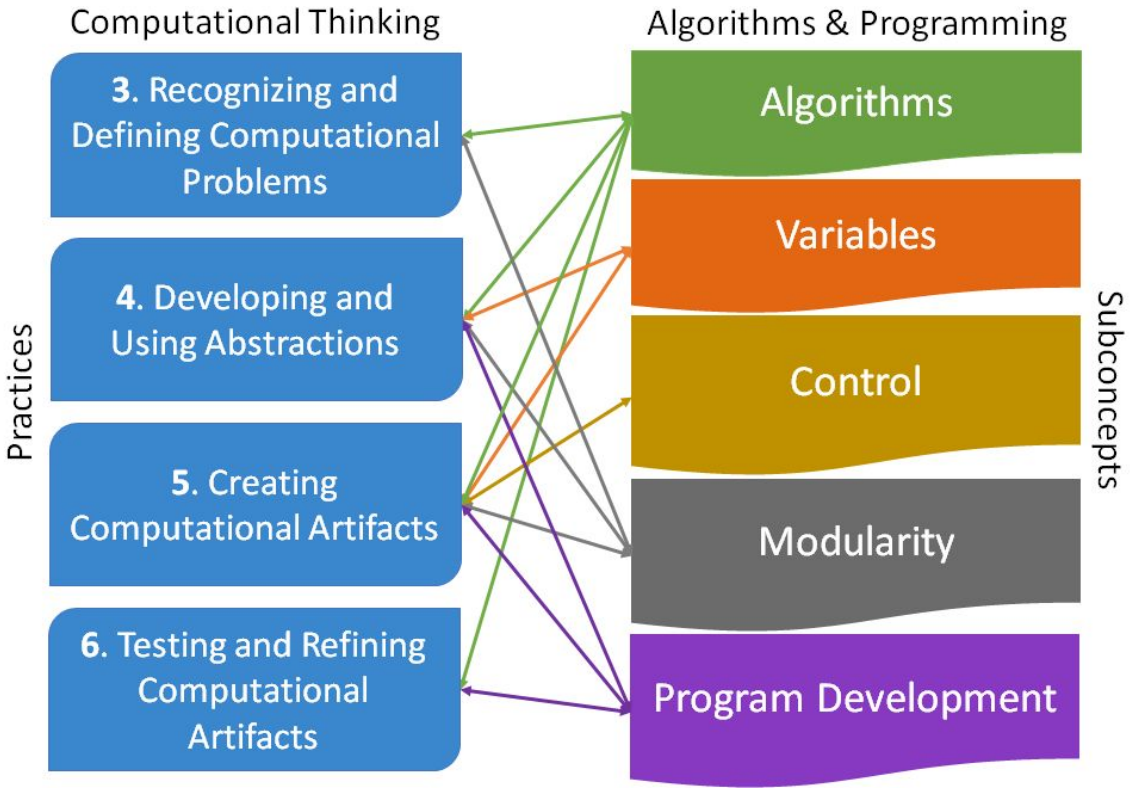
Why Computer Science Education in K-12 Settings Is Becoming Increasingly Essential

By ACM, the Association for Computing Machinery

Computing education in K-12

K-12 Computer Science Framework - CSTA

- 7 practices
- 5 concepts



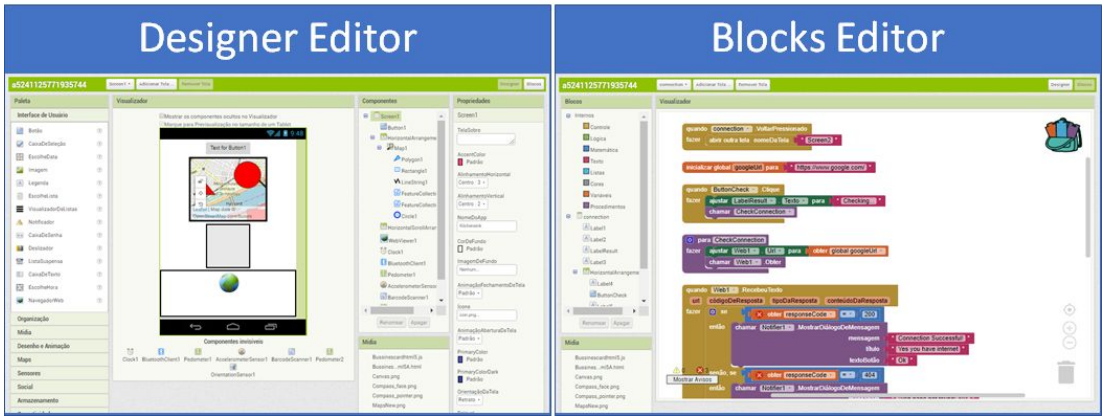
Visual programming environment

Algorithms and programming can be taught through app development with **App Inventor**

App Inventor is a visual programming language that allows to:

- **Create** Android apps
- **Program** by drag-and-drop
- **Test** the application in real-time on any Android smartphone

App Inventor interface



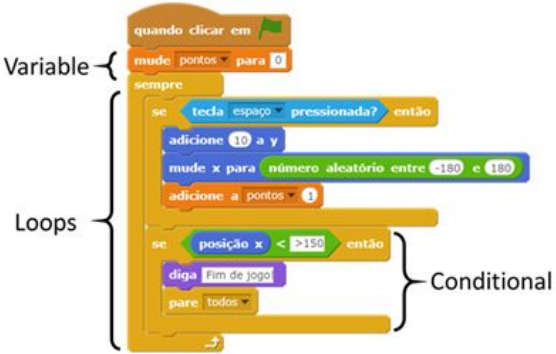
Assessment process via source code

Computational Thinking **assessment** process based on **source code**

The teacher defines a **topic** in an ill-defined activity → The students **programs** an app related to the topic → The **source code** of the app created by the student is **analyzed** → The student's app is **assessed** based on the analysis data

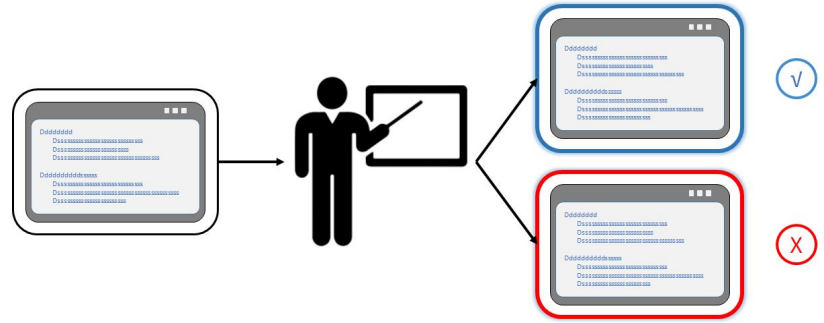
Example:

Awareness of the fight against Zika virus



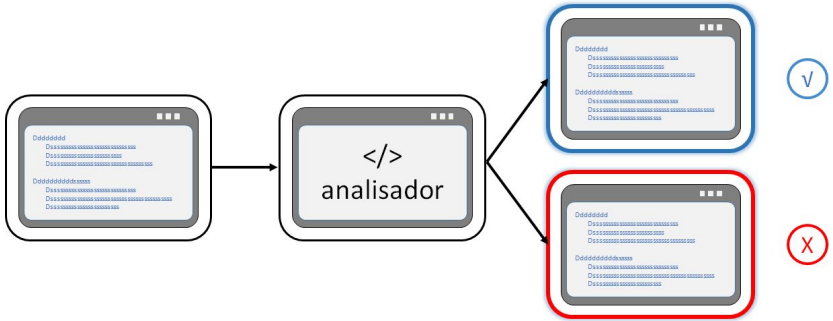
Criterion	Score
Variable	3
Loops	2
...	...
Final grade:	**8**

Assessment of programming activities



Teacher assessment

- Repetitive task
- Different teachers emphasize different characteristics
- Consumes considerable time and effort
- Requires substantial knowledge of computing



Assessment by an automated analyzer

- Consistency
- Speed
- Frees the teacher to assess other aspects and/or help the student

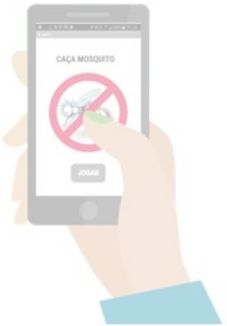
Automated assessment process

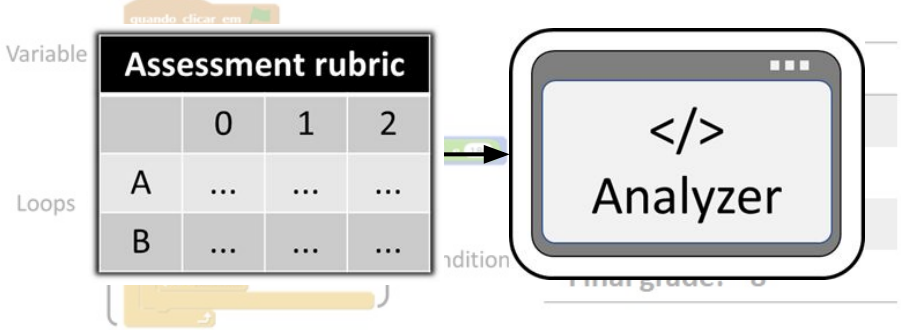
Computational Thinking Assessment

The teacher defines a **topic** in an ill-defined activity → The student **programs** an app on the topic → The **source code** of the app created by the student is **analyzed** → The student's app is **assessed** based on the analysis data

Example:

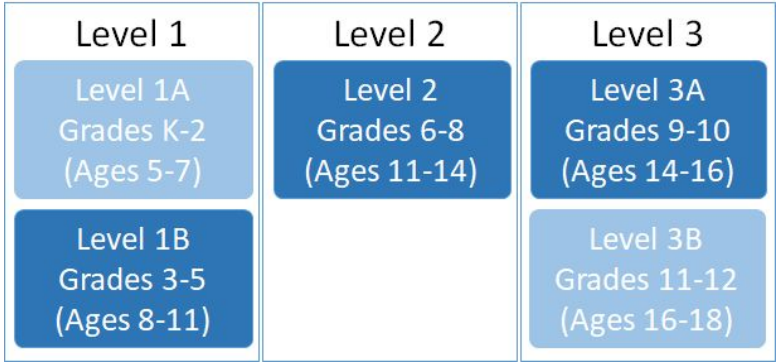
Awareness of the fight against Zika virus





Proposed model

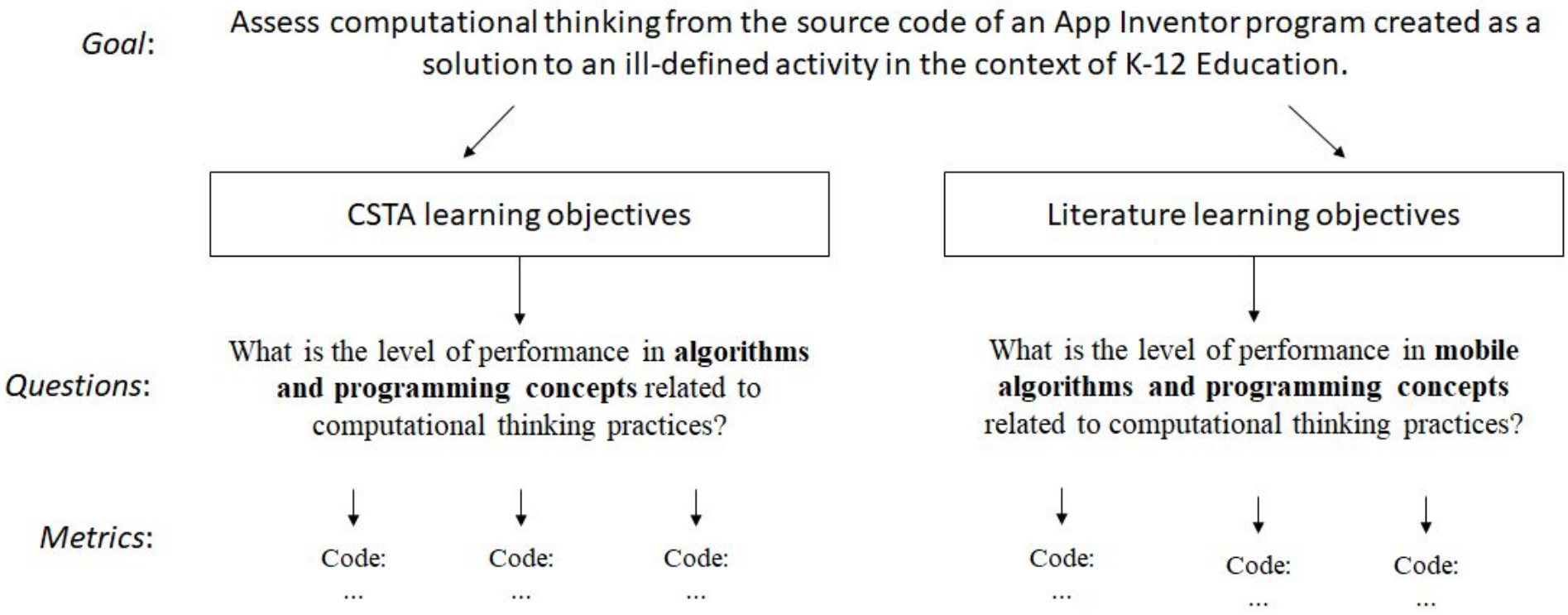
- Based on learning objectives of levels 1B, 2, and 3A from the K-12 Computer Science Framework proposed by the CSTA
- **Algorithms and programming** learning objectives related to **computational thinking practices** (P3-6)



Example of learning objectives for the concept **algorithms and programming**:

Subconcept	**Level 1B (Ages 8-11)** ***By the end of Grade 5, students will be able to...***	**Level 2 (Ages 11-14)** ***By the end of Grade 8, students will be able to...***	**Level 3A (Ages 14-16)** ***By the end of Grade 10, students will be able to...***
Variables	1B-AP-09 Create programs that use variables to store and modify data. (P5.2)	2-AP-11 Create clearly named variables that represent different data types and perform operations on their values. (P5.1, P5.2)	3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables. (P4.1)

Items creation process



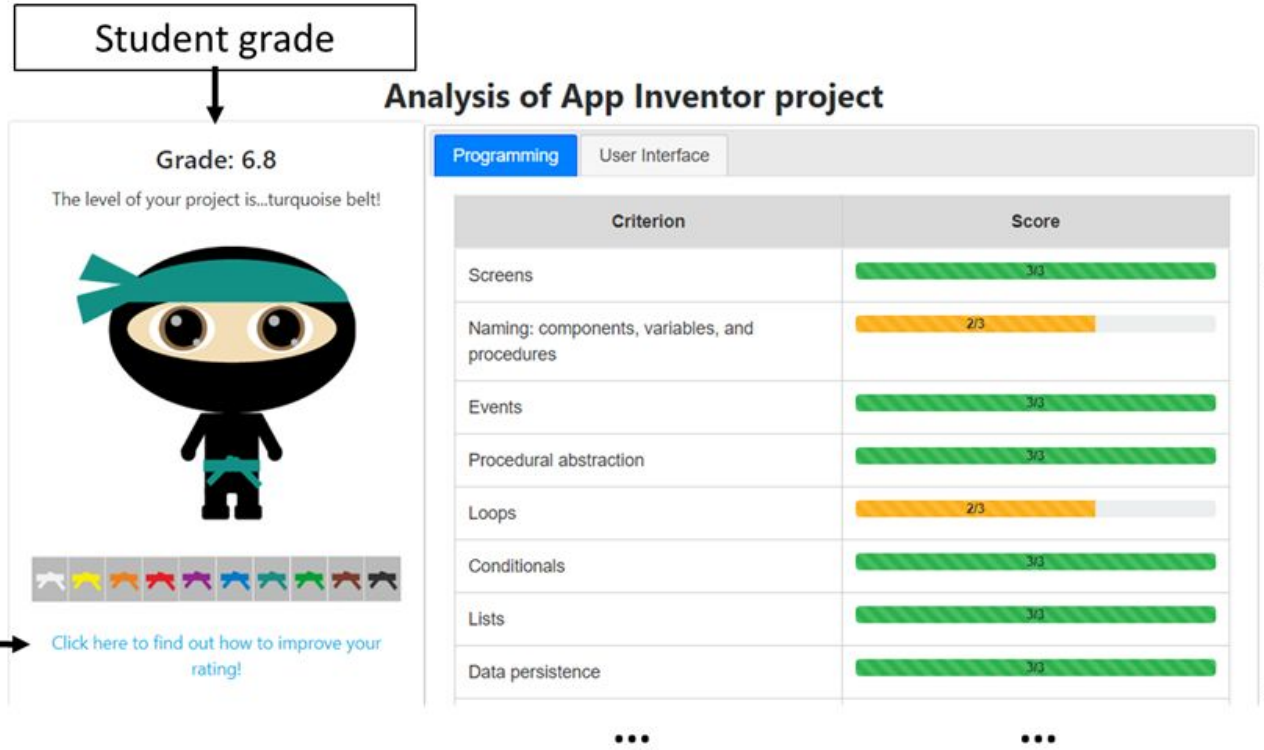
CodeMaster rubric (excerpt)

CT Sub-dimension	Criterion	Performance Level			
		0	1	2	3
Algorithms and Programming concepts	**1. Operators**	No operator blocks are used.	Arithmetic operator blocks are used.	Relational operator blocks are used.	Boolean operator blocks are used.
	2. Variables	No use of variables.	Modification or use of predefined variables.	Creation and operation with variables.	-
	3. Strings	No use of strings.	Use of creating string block to change design elements texts.	Creation and operation with strings.	-
	4. Naming	Few or no names are changed from their defaults.	10 to 25% of the names are changed from their defaults.	26 to 75% of the names are changed from their defaults.	More than 75% of the names are changed from their defaults.
	...	...	...	...	...
Mobile concepts	**12. Sensors**	No use of sensors.	One type of sensor is used.	Two types of sensors are used.	More than two types of sensors are used.
	13. Drawing and Animation	No use of drawing and animation components.	Uses canvas component.	Uses ball component.	Uses image sprite component.
	14. Maps	No use of maps.	Use of a map block	Use of map markers blocks.	-
	...	...	...	...	...

Proposed model (available online)

CodeMaster online

SCAN ME



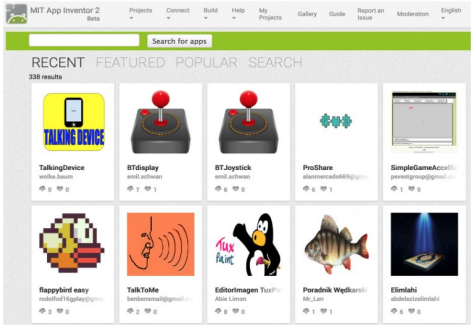
Available at http://apps.computacaonaescola.ufsc.br:8080/

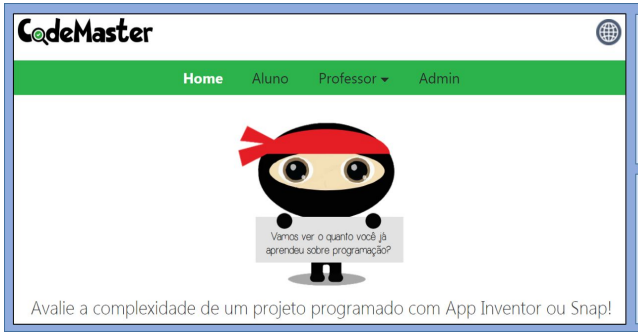
Evaluation of the CodeMaster rubric

Can we assess App Inventor apps with CodeMaster in a reliable and valid manner?

- **Data**: 88.864 apps from the App Inventor Gallery
- **Items**: assessment of 15 criteria of the CodeMaster rubric
- **Grade**: 4-point scale

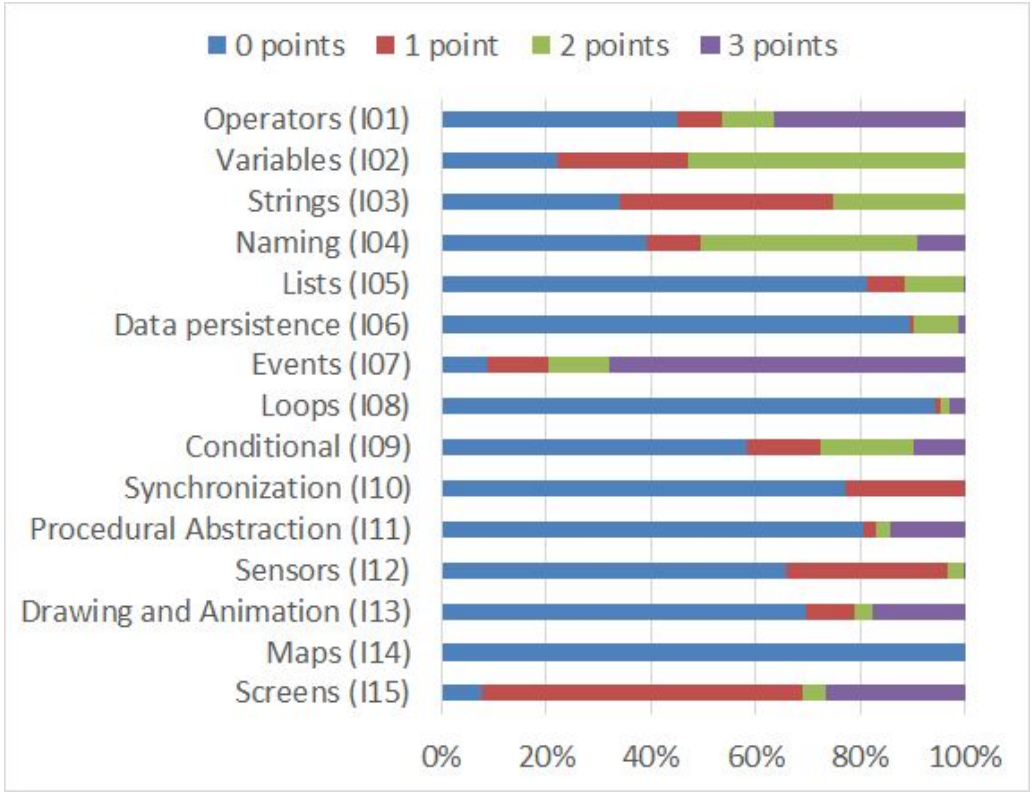
note: all items considered mandatory





App	I01	I02	...	I16
1	3	2	...	0
2	1	0	...	2
...				
88.812	0	1	2	0

CodeMaster evaluation - X-ray data assessment



- Of the 88,864 projects, 88,812 were successfully analyzed
- Item 7 (Events) and 15 (Screens) have more than 80% programs with scores above 0
- Item 14 (Maps) has only 72 programs with scores above 0

Confiability analysis of CodeMaster rubric

Cronbach alpha values:
$0.7 < \alpha \leq 0.8 \rightarrow$ acceptable
$0.8 < \alpha \leq 0.9 \rightarrow$ good
$\alpha \geq 0.9 \rightarrow$ excellent

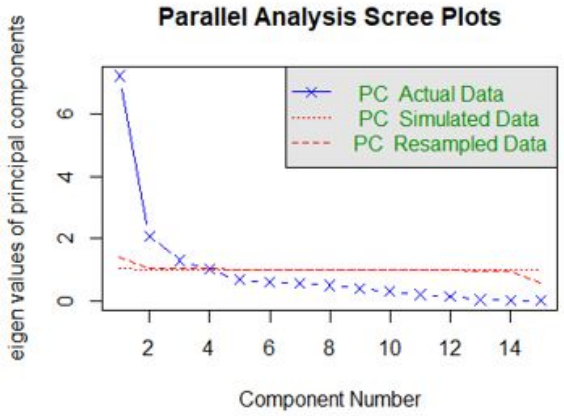
We obtained a Cronbach alpha of 0,84

Criterion (or Item)	Item-total correlation	Cronbach alpha if item dropped
1. Operators	0.694	0.82
2. Variables	0.686	0.82
3. Strings	0.583	0.83
4. Naming	0.585	0.82
5. Lists	0.364	0.84
6. Data persistence	0.325	0.84
7. Events	0.596	0.82
8. Loops	0.286	0.84
9. Conditional	0.618	0.82
10. Synchronization	0.562	0.83
11. Procedural Abstraction	0.548	0.83
12. Sensors	0.448	0.84
13. Drawing and Animation	0.376	0.84
14. Maps	**0.015**	**0.85**
15. Screens	0.324	0.84

Validity analysis of CodeMaster rubric

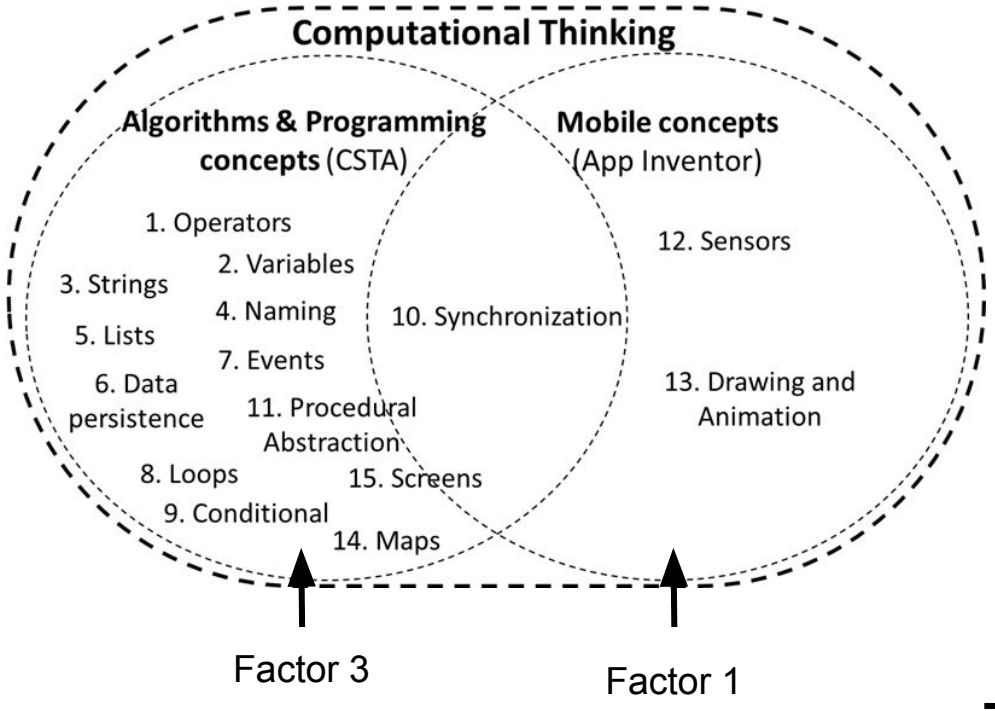
- Verifying sampling adequacy with KMO index
  - Values near 1.0 support a factor analysis, anything less than 0.5 is not likely suitable for useful factor analysis
  - We obtained a KMO index of 0.83 =)

- What is the number of factors that should be retained?
  - We used parallel analysis
  - Scree plot suggested 3 factors
  - So, we run a factor analysis...



Validity analysis of CodeMaster rubric

Factor analysis result: 2 factors as originally proposed



Criterion (or item)	Factor 1	Factor 2	Factor 3
1. Operators	0,325	0,074	**0,795**
2. Variables	0,453	0,337	**0,763**
3. Strings	-0,028	-0,100	**0,801**
4. Naming	0,198	0,174	**0,659**
5. Lists	-0,070	0,123	**0,690**
6. Data persistence	-0,093	-0,156	**0,786**
7. Events	0,178	-0,157	**0,868**
8. Loops	-0,004	0,209	**0,768**
9. Conditional	0,150	0,048	**0,807**
10. Synchronization	**0,710**	-0,336	**0,616**
11. Procedural Abstraction	0,406	0,241	**0,779**
12. Sensors	**0,713**	-0,432	0,453
13. Drawing and Animation	**0,752**	0,298	0,351
14. Maps	-0,123	-0,389	0,403
15. Screens	-0,158	-0,339	**0,702**

Conclusion

- CodeMaster rubric
  - Provides automated support for assessing programs created by students in the context of K-12 education as a result of ill-defined activities
  - Aligned with the CSTA curriculum guide
  - Evaluation results show evidence that CodeMaster is valid and reliable

- Future work
  - Evaluate with a database containing recent programs, new components such as maps
  - Assess other important 21st skills such as creativity
  - Create a scale using Item Response Theory with pedagogical interpretation for the context of K-12 education

Thank you for your attention. Any questions?

A Large-scale Evaluation of a Rubric for the Automatic Assessment of Algorithms and Programming Concepts

Nathalia da Cruz Alves

nathalia.alves@posgrad.ufsc.br

CodeMaster online

SCAN ME

SIGCSE 2020 PORTLAND

Extra slide - CodeMaster full rubric

http://apps.computacaonaescola.ufsc.br:8080/rubrica_appinventor.jsp

Criteria	Level of Performance			
	0	1	2	3
Screens	Single screen with visual components that do not programmatically change state.	Single screen with visualcomponents that do programmatically change state.	Two screens with visual components and one screen with visual components that do programmatically change state.	Two or more screens with visual components and two or more screens with visual components that do programmatically change state.
User Interface	Uses one visual component without arrangement.	Uses two or more visualcomponents without arrangement.	Uses five or more visual components with one type of arrangement.	Uses five or more visual components with two or more types of arrangement.
Naming: Components, Variables, Procedures	Few or no names were changed from their defaults.	10 to 25% of the names were changed from their defaults.	26 to 75% of the names were changed from their defaults.	More than 75% of the names were changed from their defaults.
Events	No use of any type of event handlers.	Uses one type of event handlers.	Uses two types of event handlers.	Uses more than two types of event handlers.
Procedural Abstraction	No use of procedures.	There is exactly one procedure, and it is called.	More than one procedure is used.	There are procedures for code organization and reuse (with more procedure calls than procedures).
Loops	No use of loops.	Uses simple loops ("while").	Uses "for each" loops with simple variables.	Uses "for each" loops with list items.
Conditional	No use of conditionals.	Uses "if".	Uses one "if then else".	Uses more than one "if then else".
Operators	No use of any operators blocks.	Uses one type of operator blocks.	Uses two types of operator blocks.	Uses more than two types of operator blocks.
Lists	No use of lists.	Uses one single-dimensional list.	Use more than one single-dimensional list.	Uses lists of tuples.
Data persistence	Data are only stored in variables or UI component properties, and do not persist when app is closed.	Data is stored in files (File or FusionTables).	Uses local databases (TinyDB).	Uses web databases (TinyWebDB or Firebase).
Sensors	No use of sensors.	Uses one type of sensor.	Uses two types of sensors.	Uses more than two types of sensors.
Media	No use of media components.	Uses one type of media components.	Uses two types of media components.	Uses more than two types of media components.
Social	No use of social components.	Uses one type of social components.	Uses two types of social components.	Uses more than two types of social components.
Connectivity	No use of connectivity components.	Uses activity starter.	Uses bluetooth connection.	Uses low level web connection.
Drawing and Animation	No use of drawing and animation components.	Uses canvas component.	Uses ball component.	Uses image sprite component.