An Investigation of Italian Primary School Teachers' View on Coding and Programming

Isabella Corradini, Michael Lodi, Enrico Nardelli

An Investigation of Italian Primary School Teachers' View on Coding and Programming

Isabella Corradini[1], Michael Lodi[2], and Enrico Nardelli[3]

[1] Themis Research Centre, Rome, Italy, isabellacorradini@themiscrime.com
[2] Univ. Bologna & INRIA Focus, Bologna, Italy, michael.lodi@unibo.it
[3] Univ. Roma "Tor Vergata", Rome, Italy, nardelli@mat.uniroma2.it

Abstract. This paper reports the results of an investigation involving almost a thousand primary school teachers in Italy, to explore their views on the terms "coding" and "programming", and how they are related to their ideas on "computational thinking".
When directly asked "if coding is different from writing programs", roughly 2 out of 3 teachers answered "no". Among the teachers who answered "yes", almost 160 tried to motivate the difference: a few of them gave admissible explanations, while the others showed various misunderstandings, which we classify and discuss.
By contrast, when asked about their idea of "what coding is", only 4 out of 10 of the teachers explicitly linked coding to programming, but an additional 2 out of 10 cited an information processing agent executing instructions. The remaining part of the sample did not provide explicit or implicit links between coding and programming.
Our investigation shows that untrained teachers hold misconceptions regarding CS and its related terms. Given the general public and media attention on "coding" in schools, currently taught by existing teachers - mostly not appropriately trained, professional development actions focusing on CS scientific principles and methods are therefore a top priority for the effectiveness of CS education in schools.

Keywords: Coding and programming · Primary school teachers · Informatics education.

1 Introduction

1.1 Context

The word "coding" is becoming more and more a buzzword in Computer Science Education (CSEd), especially in K-12 education. There are a lot of initiatives, like Code.org, CoderDojo, Code Clubs and so on, aiming to teach students to "code". These initiatives are spreading and, since many governments are introducing *computational thinking* (CT) or *computer science* (CS) in school curricula, the term is used in many schools as well, especially referring to introductory programming activities.

Unlike the expression "computational thinking", that may sound abstract and pretentious, and "programming" that seems to recall a boring professional

activity [4], the expression "coding" can capture the interest of students, and "*also provides an element of mystery (there are hints of a secret code), and achievement (cracking the code)*" [10]. There is a tendency in the media to use the term "coding" extensively, as noted - among others - by [1, 14], when talking for example about "coding education". Some media observers have noted that "coding" is often used to denote a "*more playful and non-intimidating description of programming for beginners*" [13].

In popular culture, the term has also come to be used on the one hand as a synonym for the entire software development process, and on the other as a means to speak about what needs to be taught in school. This overlooks both the fact that coding/programming is only a part of the software development process, and that software development is only one of the important areas of computer science [3].

We think that, while in the scientific community it is clear that "coding" and "programming" have a strict relation, and that they are only tools to teach what matters (i.e., CS - or CT, used to denote CS core aspects), the confusion induced by this "coding mania" in the media can be very harmful. In fact, in our culture CS has been plagued almost since its teenage years by a lot of misconceptions [9] and it took decades to eradicate the limiting idea that CS is only programming [1, 16].

The expression "coding" is currently invested with excessive importance [3], and this may lead to the wrong idea that its value is greater than the CS scientific concepts themselves. This is particularly relevant since, in this initial phase of introduction of CS in schools, many teachers self-train themselves and look for ideas and materials in the media, given training materials and professional developments initiatives are scarcely available.

Anecdotally, we spotted these tendencies in Italy too. Moreover, in Italian the term "programming" (translated as *programmazione*) has a very broad meaning (e.g. it is used for "schedules" like "movie show-times") and, in the context of schools, it is used to indicate "didactic planning"[4]. Furthermore, in Italy, there is a trend to incorporate "as they are" foreign terms indicating new concepts, rather than finding a corresponding Italian word. In facts, the term "coding" was explicitly used (untranslated) in a major plan launched in 2015 by the Italian government and aiming at rendering Italian schools more digital (Italian National Plan for Digital Education - *Piano Nazionale Scuola Digitale* [11]), and widely reported in communication actions related to it.

In view of the above, we decided to investigate the theme of relation between *coding* and *programming* among Italian teachers.

1.2 Literature Overview

A few studies analyzed conceptions and misconceptions about computational thinking in school teachers (see [6]), however none of them specifically investigated their ideas about the relationship between *coding* and *programming*.

[4] For example in primary schools teachers meet weekly to do an "hour of programming", namely to agree on the content of lessons of the week

A research in the ACM Digital Library, restricted to the SIGCSE publications, returns 1,186 hits for the search term "coding" compared to the 8,674 hits for "programming". However, the former shows an exponential growth from 1970s, while the latter just a linear growth (Appendix, Fig. 1).

There is no agreement in the CSEd community about the relationship between coding and programming. In fact, some authors use the two terms interchangeably as synonyms or state both (e.g., they write "programming/coding"). On the other side, a few authors do not consider them as equivalent and analyzed their difference. They agree the term "coding" is more and more used in tech business world as a jargon synonym word for programming, understood by other professionals [1] (e.g. asking for "coders" instead of "programmers" in job offers). They also observe that on one hand the term "coding" has a broader meaning in CS (e.g. in cryptography or in information theory), and on the other hand it is often used to indicate the stage of software development when programs are actually written [10, 1, 2]. In other words, "coding" is considered as a narrower concept excluding important phases like analysis, design, testing, debugging [3].

1.3 A related study on CT definition

We analyzed in [6] a large sample of 972 Italian teachers of primary school (students aged from 6 to 10) to investigate their knowledge level of CT. That analysis was based on the descriptions of CT that teachers in the sample provided by answering to the open-ended question "In my view computational thinking is...". Some of these answers did not provide a definition or were completely out of scope (e.g.: they answered "interesting" or "useful"). We assigned each of the remaining 779 *admissible* answers to one or more of 17 categories whose identification was grounded on the descriptions themselves.

Subsequently, we evaluated the level of teachers' understanding of CT with the following model. First, we rated each category with an integer weight in [-1, +2] to denote its relevance for CT definition, on the basis of the existing literature. Then, we computed the level of an answer as the sum of weights of categories it is assigned to. We used a value of at least 8 as the threshold to identify a "good" admissible definition (again on the basis of the literature) and a level of at least 6 to identify "acceptable" admissible definitions (since to reach a level of 6 an answer had to be classified with at least one category with weight +2 and at least some additional categories with positive weights).

We found the following results: 1% (8) provided a good admissible definition of CT, 10% (76) provided an acceptable (but not good) admissible definition, and the vast majority (89%, 695) of the 779 admissible answers did not provide an acceptable definition.

1.4 Purpose of the study

This research has investigated how Italian primary school teachers define *coding* and which relations they see between it and *programming*. More specifically we addressed the following research questions:

RQ1 *how do they define coding?*
RQ2 *how do they perceive the relation between coding and writing programs?*

2 Methods

2.1 Context

The Italian project "Programma il Futuro" disseminates in Italian schools, since school year 2014-15, a better awareness of CS as the scientific basis of digital technologies [7] and carries out periodical surveys among teachers. The questionnaire sent in December 2016 received 3,593 answers from school teachers of all levels, from kindergarten to upper secondary. In [6] we analyzed the 972 answers of primary schools teachers, who participated to the project for the first time in school-year 2016-17, to questions regarding their viewpoint on CT. For the same set of teachers we analyze here their answers to a different set of questions.

2.2 Tools

We focused on teachers' answers to the following questions (in square brackets the actual Italian wording: *coding* is untranslated, as usual for this term in Italy).

The first one asked them to provide their definition of coding by completing:

Q1 *In your view coding is...* [Secondo te fare coding è...]

The second one asked teachers to answer:

Q2 *In your view is there any difference between coding and writing programs?* [Secondo te c'è differenza tra "fare coding" e "scrivere programmi"?]

and to those answering positively it was asked:

Q3 *If you wish, explain why* [Se vuoi, spiega perché:]

2.3 Sample description

In the current study we considered the same sample of 972 teachers analyzed in [6]. The sample is made up for 93.7% (911) by women, hence 6.3% (61) are men, which is almost the double of the national value (3.6%) for Italian primary school teachers. This points in the direction of confirming the stereotype seeing men more attracted to computing than women.

This is the age distribution in the sample: up to 30: 8 (0.8%), 31 to 40: 133 (13.7%), 41 to 50: 415 (42.7%), 51 to 60: 374 (38.5%), 61 and more: 42 (4.3%).

Teaching seniority in the sample is the following: up to 2: 18 (1.9%), 3 to 5: 16 (1.6%), 6 to 10: 104 (10.7%), more than 10: 835 (85.8%).

Both distributions show the sample is made, to a very large extent (>80%), by mature and experienced teachers. This provides a reliable base of subjects for the research, but it is also a sign that they do not have a formal or structured CS training.

In fact, to become a pre-school and/or a primary school teacher in Italy, one currently has to get a 5-year (Combined Bachelor and Master) Degree in

Primary Teacher Education. The course prepares students to become generalist teachers, by giving theoretical, methodological and practical training to teach all subjects included in primary school (Italian, Math, History, Geography, English, Science, Technology, Sports, Music, to name the most important ones). Most of the teachers teach more than one subject in a class, sometimes both literary subjects and scientific/technical ones.

Informatics is not part of the primary school national curriculum (but can be introduced as a local project of a specific school), even though it was associated with the subject "Technology" for a brief period of time some years ago: unfortunately, it was (and still is) mostly taught as learning how to use ICT tools (e.g. using drawing programs or word processors).

Because of the lack of Informatics in the national curriculum, its contents and teaching methods are not part of the Primary Teacher Education degree.

Much worse, that degree is mandatory to teach in primary schools in Italy only since 2002. Before that year, you could become a primary school teacher just with a High School Diploma specializing on Primary Teaching (again, preparing students to teach all the subjects), without even the need of a degree. All teachers that got the Diploma before 2002 (more or less all teachers older than 40) are primary school teachers without a Primary Teacher Education degree. Due to the seniority and age of teachers in our sample, the vast majority of them belongs to this category.

In both cases, apart from isolated professional development courses, all teachers most probably did not receive any formal training in CS and CS teaching methods.

2.4 Procedures

We used a mixed methods approach, including both quantitative and qualitative analysis.

Quantitative analysis We used standard descriptive statistical methods to analyze the frequencies of both actual and relevant answers to our three questions. Moreover, we used the data and the model on CT definition of [6] to analyze how the values provided by that model are distributed when restricted to answers to our questions Q1, Q2 and Q3.

Qualitative analysis We filtered out answers to Q1 that did not provide a definition (e.g. "innovative") and answers to Q3 that did not explained the difference (e.g. "coding is a discovery").

We then proceeded for each of the remaining *relevant* answers to identify, by reading and discussing, the conceptual categories to be used for their classification.

In a first phase each of us independently analyzed the definitions and proposed, for each question, a set of conceptual categories to classify them. We used a mixed approach: some categories were defined "a priori", on the basis of literature overview and related work described in section 1.2, others were grounded on the definitions themselves.

Secondly, we jointly examined, for each question, the proposed sets of categories and through discussion we agreed to a preliminary set.

Subsequently, we manually assigned each answer to one or more categories, if the statement either declared the same nature as the category or stated being relative to or useful for the category. For this process the set of answers for each question was split between us, and we assigned answers in our own subset to one or more categories. During this process proposals for modifications to categories emerged.

Lastly, we jointly examined, for each question, both these proposed modifications and assignments. Through discussion, we came to agree on the final set of categories for each question (described in subsections 4.1 for Q1 and 4.2 for Q3) and the final assignment of each definition to one or more categories.

3 Quantitative results

3.1 Q1 - Coding is...

A definition was present in 88% (854) of all 972 answers. Among the 118 ones which did not provide it, 50.8% (60) did not answer to Q2 either, 24.6% (29) answered negatively with respect to (wrt, from now on) the difference between *coding* and *writing programs* while a same (by chance) 24.6% (29) answered positively (but only 2 answers contained an explanation). Among the 854 provided definitions, 7% (56) of them were not relevant (e.g., "coding is innovative"). The qualitative analysis of the remaining 798 (=854-56) ones is in sub-section 4.1.

3.2 Q2 - Is coding different from writing programs?

An answer was provided to Q2 by 78% (758) of the 972 teachers in the sample and 60% (456) of them answered "no" and 40% (302) answered "yes". Among the 214 ones which did not provide an answer, 28% (60) did not answer to Q1 either.

Relationship with CT definition We used the CT definition evaluation model and the related set of data in [6] to analyze the distributions of values provided by such model when restricted to the two significant subsets of possible answers to our question Q2 (namely, "yes" or "no").

The sample of 972 answers in [6] contained 779 answers with admissible CT definitions. We analyzed the admissible ones and found 396 "no" and 246 "yes" answers to Q2, while 137 did not answer at all. The percentage of *acceptable* CT definitions[5] is slightly higher for the admissible CT answers who also answered "no" to Q2 (12%, 46/396) than for those who answered "yes" (10%, 25/246).

This shows that teachers having correctly identified that there is no difference between *coding* and *writing programs* have performed slightly better, for what regards the definition of CT, than those who think there is a difference. This is confirmed also by comparing the average value for acceptable CT definitions in the two subsets of teacher having answered "no" (avg = 6.33) and "yes" (6.12).

[5] In [6] a value of at least 6 characterizes an "acceptable" CT definition

3.3 Q3 - The difference between coding and writing programs is...

Among the 302 answers to Q2 incorrectly stating *coding* and *writing programs* are different, only 53% (159) explained why by answering Q3: 25 of these were not relevant, while the qualitative analysis of the remaining 134 ones is in subsection 4.2.

Relationship with CT definition Among the 779 admissible CT definitions of [6] there were 123 who also answered to Q3, and none of these definitions has a value greater than 6 according to their model. Also, the percentage of those receiving a value at least 6 (acceptable definition) is lower in this subset of teachers (7%, 9/123) than in the overall set (11%, 84/779) and the percentage of unacceptable (<6) definitions in this subset (93%, 114/123) is higher than in the overall set (89%, 697/779).

The fact that teachers having tried to characterize a difference between *coding* and *writing programs* were not able to provide an acceptable CT definition shows an agreement between this research and the evaluation provided by the model in [6].

4 Qualitative results

4.1 Q1 - Coding is...

Categories The analysis of the 798 relevant answers to Q1 using the procedure described in 2.4 resulted in 10 categories. We grouped them in two classes, according to whether they were somewhat related to *writing programs* or not.

- **Related**: All categories here somehow "speak" about writing programs, either in a full (`PROG`) or simplified (`SIMP`) way, or are concerned with writing algorithms (or lists of instructions) making reference to some information processing agent able to execute them mechanically (`PROC`).
  - `PROC` *Specifying processes*: devising an algorithm to solve a problem; providing a list of instructions to solve a problem; making an information processing agent execute a sequence of elementary steps
  - `PROG` *Writing programs*: using programming languages
  - `SIMP` *Simplified programming*: programming with simplified environments/ languages (e.g.: visually, blockly); learning the basics of programming
- **Unrelated**: Categories in this class are not directly concerned to writing programs in some form.
  - `ACTI` *Being active towards information technology*: creating computational artifacts instead of simply using them; being able to find creative or original solutions to problems
  - `COLE` *Cognition and learning*: reflecting about thinking or learning; program to learn; learning to learn; develop/ improve cognitive abilities; a method/ approach to teaching/learning
  - `DECT` *Developing computational thinking*: a way to teach/ develop/ apply CT

`ENGA` *Engagement*: doing playful/ funny/ attractive/ interesting/ inspiring activities

`LOCR` *Logical/critical thinking*: logical or reasoning or analytical skills; applying/developing critical thinking

`PROB` *Solving problems*: plan(s), design(s), action(s) or process(es) leading to solve a problem, to reach a goal, to face a complex situation (including splitting a complex problem in simpler subproblems to solve it more easily)

`TRAN` *Transversal competence*: e.g. fourth skill, transversal skill, life skill, useful in other fields, of general use

Analysis of category distribution The distribution of categories for the 798 relevant answers to this question is shown in figure 2 (see Appendix).

Category `PROG`, which directly relates *coding* to *programming*, is understandably the most frequent one, but appears in only 4 out of 10 relevant answers (323/798). If only the 456 teachers answering also "no" to Q2 are analyzed, this percentage slightly increases to 43% (194/456), a slightly positive sign that those teachers correctly relating "coding" and "writing programs" (i.e., the "no" answers to Q2) were also better able to describe coding in terms of programming.

On the other side, by aggregating the answers in the *related* class (i.e., `PROG`, `PROC`, and `SIMP` - remember each answer can receive more than one label) we obtain that 59% (469) of answers relevant for Q1 use an expression somewhat related to *programming*. In the light of the characteristics of our sample (see 2.3), the fact that 6 out of 10 teachers were somewhat able to identify a correct relation between *coding* (which for large part of their professional career most probably they never heard about) and *programming* is certainly positive.

Also note that, given each relevant answer was assigned to one or more categories, there is a 29% (232) of answers falling both in the *related* and *unrelated* classes. On the other side, there was a 30% (237) of answers belonging to at least one of the *related* categories and none of the *unrelated* ones and a 41% (329) of answers belonging to one of the *unrelated* ones and none of the *related* ones.

The third most frequent category is `PROB`, that with a 24% (190) is very close to the 25% (197) of the second one, `PROC`, confirming the trend emerged in [6] for CT, that in Italian schools CS education is often considered as a general instrument for problem solving. The strict relation between CT and programming is confirmed by the 17% (138) seeing coding as a way to teach/ develop CT (`DECT`).

It is interesting to note that 17% (138) of teachers highlights the *engagement* value of coding (`ENGA`) and 15% (117) sees it as an aid for teaching or developing cognitive abilities (`COLE`). We think these are important elements to ensure a diffusion of computing education in schools, although one has to pay attention they do not overshadow its core elements. The same reasoning applies to `LOCR` (11%, 87) and `TRAN` (4%, 31).

Only a 7% (54) of teachers has remarked the importance of coding to become active towards Information Technologies (`ACTI`), which is anyway positive given the question was not investigating the role/ purpose of *coding*.

Relationship with CT definition Among the 798 relevant answers to Q1 there were 743 who also provided an admissible CT definition.

We show in Table 1 (see Appendix) how these 743 definitions are distributed according to two subsets: one made up by all 458 answers to Q1 using terms somewhat *related* to "writing programs" and the other one made up by the 285 remaining answers. The average value of the CT definition evaluation model for all *related* teachers is 3.37, while for all the remaining ones is 2.62, showing a positive correlation between understanding CT and being able to properly define "coding".

4.2 Q3 - The difference between coding and writing programs is...

Categories The analysis of the 134 relevant answers to Q3 using the procedure described in 2.4 resulted in the 11 categories described below. We grouped them in three classes according to how they described the difference between "coding" (C, in the following description) and "writing programs" (P). Some descriptions are tolerable while others are unacceptable. A few are completely out-of-scope.

- **Tolerable**: in this class we have categories expressing admissible relations, given the wide variety of ways in which the two terms are used in both literature and profession.
  `COMP` - C is a part of P
  `EASY` - C is a simplified P
  `PROP` - C is preparatory to P
- **Unacceptable**: categories in this class refer to wrong ways of describing relations between C and P.
  `CONC` - C is the conceptual part of P
  `GENA` - C is more general/ abstract than P
  `LUPR` - C is for playing/ learning, P is for working
- **Out of scope**: here we have categories which do not really address the difference but simply refer to characteristics of C.
  `DEVC` - C is a means to develop computational thinking
  `GECO` - C is a general competence
  `LOTH` - C is a means to learn other subjects
  `SOCI` - C has a social value
  `SOLV` - C is problem solving

Analysis of category distribution The distribution of categories is shown in figure 3 (see Appendix).

In 43% (58) of relevant answers there were elements characterizing *coding* as simpler than (`EASY`, occurring in 35 answers, 26%) or preparatory to (`PROP`, 23, 17%) or part of (`COMP`, 5, 4%) *programming*: we collectively denote these answers as *tolerable*. In 54% (72) of cases there were elements expressing (at least one) actual (and *unacceptable*) misunderstanding of the relation between *coding* and *programming*. There were 17 answers with both *tolerable* terms and *unacceptable* ones.

We classified these misunderstandings in three *unacceptable* categories: both `CONC` (characterizing 13 relevant answers, 10%) and `GENA` (25, 19%) reverse the position, expressed in the literature (see 1.1), that considers *coding* as a narrower concept than *programming* in the software production process. The former by assigning it a conceptual role CT is concerned with, the latter by ascribing to *coding* a scope wider than the mere act of writing programs.

Answers classified as `LUPR` (28%, 38) have elements characterizing *coding* as just a "toy" activity distinct from "the real thing", that is programming in a professional context. Clearly, this misunderstanding goes in the opposite direction as the two previous ones, and none of the 134 relevant answers was self-contradictory by expressing both `LUPR` and (`CONC` or `GENA`).

Reconsidering the relation between *coding* and *writing programs* Figure 4 (see Appendix) shows the Venn diagram of the classification of answers to Q3 according to the 3 classes grouping the 11 categories. A minority of the 58 relevant answers classified in the *tolerable* class were also classified in the *unacceptable* one, but there were 41 whose classification did not have *unacceptable* categories.

These are therefore teachers considering *coding* as distinct from *writing programs*, but whose answers can be aggregated with the 456 negative answers to Q2 and removed from the positive ones. We thus obtain a 66% (497) of the 758 teachers who answered Q2 having an acceptably correct view of the relation between *coding* and *writing programs*.

Finally, we have also analyzed the subset of teachers whose answers to Q1 featured both a classification as (`PROG or SIMP`) and as `PROC`: there are 71 of them. In this subset of *strongly related* answers (9% of the 798 relevant answers for Q1) there are 51 (72% of the subset) who were able to correctly relate *coding* to *programming* (in the enlarged sense described in 4.2) while only 1 of them described an *unacceptable* difference between *coding* and *writing programs*.

Relationship with CT definition Let us now define as *tolerable-only* answers the 41 ones classified with at least one *tolerable* category and none *unacceptable*, and as *unacceptable-only* the 55 ones with at least one *unacceptable* and none *tolerable* (see again figure 4).

Table 2 (see Appendix) shows the values of CT definitions provided from the model in [6] for these two subsets. Remember that no teacher who answered Q3 received a value greater than 6 in the CT definition evaluation. Note that there are fewer answers in *tolerable-only* subset than in *unacceptable-only* (39 vs 48), and teachers in the *tolerable-only* subset have a slightly higher average value (3.49 vs 3.38) for the CT definition evaluation model.

4.3 Joint distribution of Q1 and Q3

All the 134 teachers answering Q3 answered also to Q1. We present in Table 3 (see Appendix) the joint distribution of all answers to Q1 and those answers to Q3 classified as *tolerable-only* or *unacceptable-only*.

In Tables 4 and 5 we present the two respective marginal distributions.

Table 4 (see Appendix) shows that both a majority (68%) of relevant answers classified as *tolerable-only* wrt Q3 belong to *related* wrt Q1 and a majority (60%) of relevant answers classified as *unacceptable-only* wrt Q3 belong to *remaining*[6] answers in Q1. This indicates a positive correlation between the capability of describing what coding is (Q1) and the capability of providing a description of the difference between coding and programming (Q3).

This is confirmed also by Table 5 (see Appendix), showing that both a majority (56%) of relevant answers classified as *related* wrt Q1 belong to *tolerable-only* wrt Q3 and that a majority (72%) of the *remaining* relevant answers to Q1 belong to *unacceptable-only* wrt Q3.

5 Conclusions and further work

We analyzed Italian primary school teachers' ideas about coding and its relationship with programming.

Regarding RQ1, we found that only 4 answers out of 10 directly mentioned *programming* when defining *coding*. On the other hand, if we consider also answers mentioning simplified programming environments/languages or the act of designing algorithms or giving instructions to an executing agent in the definition of *coding*, the number of good answers grows to a more comforting 6 out of 10. Answers highlighted also side aspects of coding, often overlapping with elements more rightly belonging to CT.

For what concerns RQ2, when directly asked if there is a difference between *coding* and *write programs*, 60.2% of them answered *no*. Another 5.4%, even if answered *yes*, gave a completely tolerable explanation in the light of the variety of ways the term *coding* is used in different contexts, resulting in an overall 2 out of 3 teachers expressing an acceptably correct relation between coding and programming. The others giving an explanation (half of those answering *yes*) listed some characteristics of *coding* without really explaining the difference or used wrong (and conflicting) motivations: coding is the conceptual part of programming, or more general and abstract than programming, or just a toy while programming only for professionals.

We also compared our findings with ideas our sample had about CT [6]. We found that teachers having acceptable ideas about it performed slightly better in: (i) describing coding with programming-related terms, (ii) correctly identifying *coding* with *writing programs*.

Finally, when comparing *coding* definitions and explanations of differences between *coding* and *writing programs*, we found that 68% of those who provided a programming-related definition managed to provide also a completely tolerable explanation of the difference. Dually, the vast majority of those who failed to relate coding to programming activities in its definition also provided unacceptable motivations for the difference.

[6] This is the subset of Q1 answers that has not been classified in any of the *related* categories.

Despite being limited to Italy, our study - showing that teachers have not a clear picture of the relations between coding and programming - can be representative of similar situations in K-12 education of many developed countries. It would be interesting to learn more about the situation in other countries from the local replication of a similar investigation.

The most probable cause for the misconceptions revealed by this study is the fact that teachers have not been appropriately trained in CS and its teaching methods. The importance of teacher training has already been identified in other reports (e.g. [15, 5]) as a key factor for a successful uptake of CS education in schools.

These results support worries about the fact that focusing only on a specific activity/ tool of CS (i.e., on programming) can be harmful, especially if referring to it with a "buzzword" like *coding*, which takes on conflicting meanings. In fact, our results show that such misconceptions have a high correlation with the presence of reductive or wrong ideas about CS. On the other hand, having an appropriate understanding of what coding and programming are is an important requirement for teachers to be able to provide good CS education in schools.

This research has shed some light on the fact that lack of proper training joined with confusion in terminology spread by media originated dangerous misconceptions, which may harm effectiveness of CS education actions.

We therefore recommend, when speaking about CS education, to stress the importance of CS scientific principles and methods. It has to be clearly stated that CS (and not CT or coding) is the scientific discipline to be taught at all school levels [12], both because it is the science underpinning the development of the current digital society and because it provides conceptual methods contributing to a better understanding of other disciplines [8]. This has to be done at a communication level when presenting and discussing CS school education initiatives, at the organizational level of CS school curricula specification and in the context of teacher training in CS.

We plan to extend our analysis to teachers of other school levels and to compare these results with those of teachers with more experience in CS teaching. It would also be interesting to carry out surveys in other countries to obtain a wider picture of the relations between misconceptions regarding CS related terms and teacher training.

Acknowledgements

We greatly thank teachers and students involved in Programma il Futuro project (coordinated by EN) and Code.org for their cooperation.

We acknowledge the financial support of Engineering, TIM; CA Technologies, De Agostini Scuola, SeeWeb. Other companies have financially supported the project for two school-years only: Samsung Italia; Microsoft Italia; Cisco, Hewlett-Packard, Oracle, Facebook.

Rai Cultura, the culture department of Italian national public broadcasting company, is a media partner of the project since February 2017.

References

1. Armoni, M.: Computing in schools: Computer science, computational thinking, programming, coding: The anomalies of transitivity in k-12 computer science education. ACM Inroads **7**(4), 24–27 (Nov 2016)
2. Barendsen, E., Mannila, L., Demo, B., Grgurina, N., Izu, C., Mirolo, C., Sentance, S., Settle, A., Stupurienė, G.: Concepts in k-9 computer science education. In: ITICSE-WGR '15. pp. 85–116. ACM (2015)
3. Bell, T.: What's all the fuss about coding? In: ACER Research Conference 2016. Australian Council for Educational Research, Melbourne, Australia (2016)
4. Ben-Ari, M.: In defense of programming. In: ITiCSE'15. ACM (2015)
5. Caspersen, M.E., Gal-Ezer, J., Nardelli, E., Vahrenhold, J., Westermeier, M.: The cece report: Creating a map of informatics in european schools. In: SIGCSE '18
6. Corradini, I., Lodi, M., Nardelli, E.: Conceptions and misconceptions about computational thinking among italian primary school teachers. In: ICER'17
7. Corradini, I., Lodi, M., Nardelli, E.: Computational Thinking in Italian Schools: Quantitative Data and Teachers' Sentiment Analysis after Two Years of "Programma il Futuro" Project. In: ITiCSE '17. ACM, New York, NY, USA (2017)
8. Denning, P.J., Rosenbloom, P.S.: The profession of IT: Computing: The fourth great domain of science. Commun. ACM **52**(9), 27–29 (Sep 2009)
9. Denning, P.J., Tedre, M., Yongpradit, P.: Misconceptions about computer science. Commun. ACM **60**(3), 31–33 (Feb 2017)
10. Duncan, C., Bell, T., Tanimoto, S.: Should your 8-year-old learn coding? In: Proceedings WiPSCE '14. pp. 60–69. ACM (2014)
11. Italian Ministry of Education, University and Research: National plan for digital education (2016)
12. Nardelli, E.: Do we really need computational thinking? Commun. ACM (To be published) (2018)
13. Prottsman, K.: Coding vs. programming - battle of the terms! (2015), http://www.huffingtonpost.com/kiki-prottsman/coding-vs-programming-bat_b_7042816.html
14. Sentance, S.: Introducing why teach computer science in school. In: Sentance, S., Barendsen, E., Schulte, C. (eds.) Computer Science Education. Perspectives on Teaching and Learning in School, chap. 1, pp. 3–4. Bloomsbury Academic, London (2018)
15. The Royal Society: After the reboot: computing education in UK schools. The Royal Society, London (2017)
16. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (Mar 2006)

Appendix

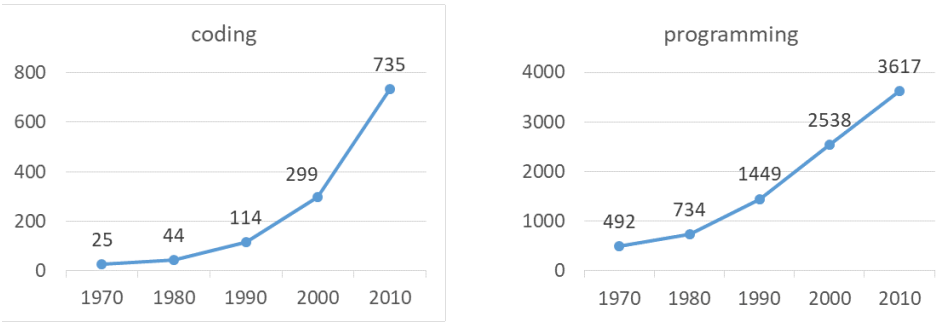

Fig. 1: Growth of search hits for terms *coding* and *programming* in ACM SIGCSE publications. Source: ACM Digital Library search results (Aug. 7th, 2018).

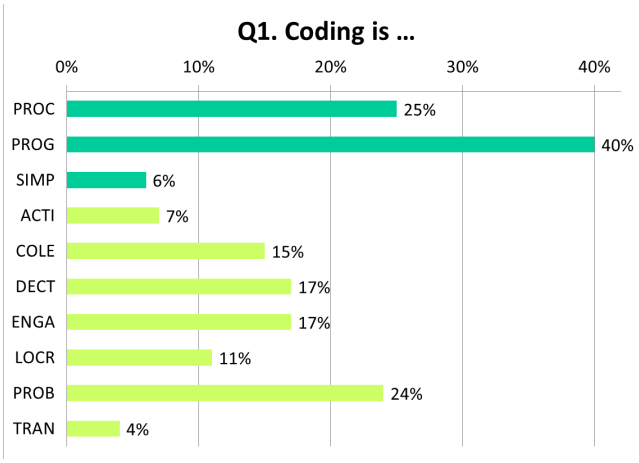

Fig. 2: Frequency of each category in Q1.

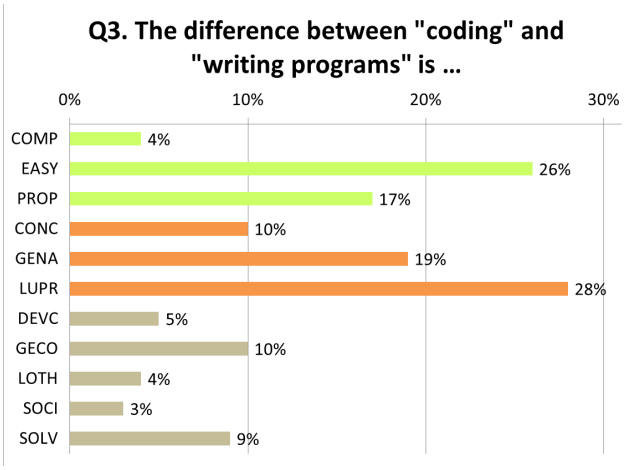

Fig. 3: Frequency of each category in Q3.

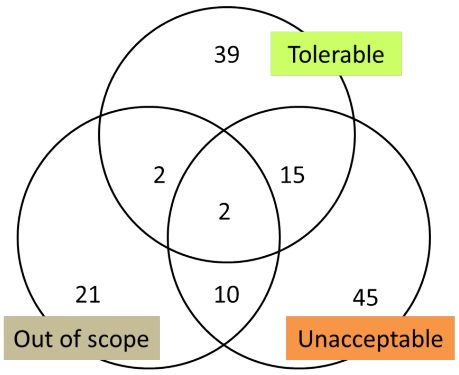

Fig. 4: Distribution of Q3 answers among classes.

Table 1: Distribution of CT values of relevant Q1 answers.

value	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
Related	1	16	40	105	57	149	32	47	6	3	1	0	1	0
Remaining	5	27	65	58	28	60	16	21	2	2	0	0	0	1

Table 2: Distribution of CT values for a subset of Q3 answers.

value	-1	0	1	2	3	4	5	6
Tolerable-only	0	1	2	8	7	12	5	4
Unacceptable-only	0	3	5	7	4	18	7	4

Table 3: Joint distribution of answers to Q1 and Q3.

	Tolerable-only	Unacceptable-only	SUM
Related	28	22	50
Remaining	13	33	46
SUM	41	55	96

Table 4: Marginal distribution of Q3 answers wrt to Q1.

	Tolerable-only	Unacceptable-only
Related	**68%**	40%
Remaining	32%	**60%**
SUM	100%	100%

Table 5: Marginal distribution of Q1 answers wrt to Q3.

	Tolerable-only	Unacceptable-only	SUM
Related	**56%**	44%	100%
Remaining	28%	**72%**	100%