# An Exploratory Study on the Availability of Open Educational Resources to Support the Teaching and Learning of Programming

William Simão de Deus and Ellen Francine Barbosa
*Institute of Mathematics and Computer Science (ICMC)*
*University of São Paulo (USP)*
São Carlos, Brazil
williamsimao@usp.br, francine@icmc.usp.br

***Abstract*—*Context:* This Research Full Paper presents an exploratory study on the availability of open educational resources (OERs) to support the teaching and learning of programming. Currently, programming is considered a fundamental course in many undergraduate programs. Moreover, many countries are teaching children and young people how to program, considering this as an essential skill in the future. However, students report many difficulties when they are learning to program due to limitations of abstraction/logic, syntax, structure, among others. Similarly, teachers also argue that teaching programming is an arduous task due to the preparation of presentations, correction of exercises and creation of projects. *Objective:* Given this scenario, our objective was analyze the availability of OERs - open materials that present different contents and formats distributed by internet - and their support for programming students and teachers. *Method:* Aiming to reach the aforementioned objective, we conducted an exploratory study among 3,990 OERs, using computational processing and descriptive statistic. *Results:* We analyzed the OER descriptions and compared them with the current programming challenges. Also, we investigated aspects of the availability of these materials. *Conclusion:* There are several OER available that can support teaching and learning of programming. However, to enhancing their dissemination, we must facilitate its discovery, reuse and sharing.**

***Index Terms*—Teaching and Learning of Programming, Open Educational Resources, Exploratory Study**

## I. Introduction

In the last years, programming has been considered a fundamental course for many undergraduate programs as a way to support student development [1], especially in STEM degrees (Science, Technology, Engineering, and Mathematics) [2]. In addition, many countries have suggested and proposed initiatives to disseminate programming education to children and young people, realizing that knowing how to program will be an essential skill in the near future. For instance, United Kingdom and Australia included introductory programming to primary and secondary curricula [3]. Also, more recently, the Brazilian Computer Society released a set of guidelines for teaching computing in basic education in Brazil [4].

Traditionally, teaching a student how to program involves different stages of preparation and use of different supporting materials [5]. Throughout the teaching process, the teacher is involved in activities such as preparing presentations, correcting exercises and creating projects, which requires a lot of concentration and effort, due to its complexity [6]. At the same time, students also face many challenges. Medeiros *et al.* [2], for instance, reported difficulties regarding abstract/logical limitations, programming language syntax, structure language (control, data and code), among others. Together, these factors evidence that although the teaching and learning of programming is necessary, it is still a complex task for both teachers and students [7].

In this perspective, OERs emerge as a potential solution [8]. In short, OERs are free and open teaching materials made available on the internet [9]. These resources are licensed under regulations that encourage their use, reuse and sharing among people for educational purposes. Additionally, OERs are known to have different types and forms, such as textbooks, courses, exercises, software, etc [10]. Because of these characteristics, OERs have a huge potential to support teachers during programming classes, offering, for instance, presentations that can be personalized, exercises, code snippets, texts and videos, among others [11]. Motivated by this scenario, many initiatives have been developed based on OER with programming content. As an example, we highlight the OER repository with programming exercises presented by Staubitz *et al.* [12]. Another case was presented by Dol and Indi [13], in which OERs have been adopted to teach C programming. In both cases, the authors reported good results with the use of OERs among students.

However, despite these positive indicators, the use of OERs to support programming education still has many challenges. First, many teachers are not familiar with OERs reality yet. As commented by Marcolino and Barbosa [14], commonly the instructors adopt outdated materials, such as lists of programming exercises and text-based programming environments, disregarding the use of OERs. Second, it is difficult to gather relevant programming resources. According to Piedra *et al.* [15], this is due to the growth of unorganized OER collections. Thus, there is no pattern between title, abstract and keywords. As a result, a search of "teaching programming" could return irrelevant results, such as *linear programming* or *genetic programming*. Finally, it is difficult to exactly estimate how many resources exist and which ones are available for use, reuse and sharing. This occurs because there is little empirical evidence on the types of adoption and use of OERs in the repositories [16].

Considering this context, in this exploratory study we investigated the availability of OERs and their support the teaching and learning of programming. Initially, we searched in different digital sources for resources on programming. So next, we extracted information about the resources using web crawlers[1]. Thus, through the use of computational processing and descriptive statistics we identified the main programming languages, types of resources, licenses and restrictions of use, adapt and change.

The remainder of this paper is organized as follow: Section II provides a summary of programming and OERs. Section III presents a comparison of related works. The extraction process is described in Section IV. Section V is analyzed the current challenges for the teaching and learning of programming and their support by OERs. Next, Section VI shown the availability, licences and languages of OERs for programming. Discussions are presented in Section VII. Finally, the conclusion and future works are presented in Section VIII.

## II. Background

### A. Teaching and Learning of Programming

According to Lokar and Pretnar [17], there are different models and techniques for the teaching and learning of programming. However, recent researches show that most teachers only adopt classic forms, such as problems exposure and lists of exercises [14]. Although this is the most common way of teaching, teachers, and students of programming end up overwhelmed.

Initially, from the teacher's point of view, teaching programming is a complex task. Borges *et al.* [18], for example, noted that the work is intensified due to the need of reviewing the code developed by students, and as each one has its own programming logic, it is essential to reflect and analyze several possibilities. Oliveira *et al.* [19] also noted that the high number of students by classroom and their distinct degrees of code complexity also end up increasing the workload.

Regarding the student's view, we highlight that programming learning is also complicated. According to Yildiz [20], many students have difficulties to learn the coding process, producing low-quality code. Moreover, for many students, it is difficult to assimilate abstract aspects that the learning of programming involves [21]. Thus, such factors end up contributing to demotivation, making the programming courses achieve the highest failure rates in degree programs [18].

Currently, it is argued that these problems occur due to the overload of teachers and to the lack of basic foundations, such as mathematics, abstraction thinking, and problem-solving in the student formation [19]. With this in mind, next, we discuss how OERs can reduce the incidence of these problems supporting teachers and students.

### B. Open Educational Resources

To understand how OER can support teaching and learning programming, it is necessary to verify its origins. The definition of OER was first introduced by UNESCO [9] as *"the open provision of educational resources, enabled by information and communication technologies, for consultation, use, and adaptation by a community of users for non-commercial purposes"*. Thus, OER has two main characteristics: first, it refers to information and communication technologies. Second, it is about the use, reuse, and adaptation.

Due to information and communication technologies, OERs can support different types of resources [10]. Thus, it is very common to find resources such as videos, audios, courses, software, exercises, etc. This feature can already support teachers and students, considering that OERs can offer materials to support the preparation of classes and also maximize student learning. Furthermore, when referring to use, reuse, and adaptation, OERs are subject to their own licenses that make it possible to share teaching and learning materials with freedom of remix [22]. Thus, to prevent plagiarism and/or piracy problems, the resources are generally completely free or licensed under specific permissions [23]. Therefore, the materials can be adopted as an attempt to suppress the basic difficulties of the students at the same time that they can be found in different forms, also avoiding demotivation of the classic forms of teaching.

## III. Related Work

The intersection between OERs and the teaching and learning of programming is being investigated by different authors. Initially, we highlight the investigation conducted by Staubitz *et al.* [12]. The authors presented a collection of OERs for open auto-gradable programming exercises. In parallel, Maksimenkova and Podbelskiy [24] provided OERs to develop tasks and assessments for programming students. However, both approaches have the same limitations: the OER collections had few resources and have type restrictions. Also, resources did not cover different programming domains.

More specifically, Dol and Indi [13] presented how OERs can assist the teaching of C language programming. Nonetheless, the proposed solution was planned considering specific platforms, requiring integration among different tools to work. Due to the complexity of this operation, the solution was not practical.

Considering the use of OERs in the general aspects, Tovar *et al.* [25] presented how the Merlot repository[2] started communities dedicated to the use, reuse and sharing of resources in Computer Science (CS) and Information Systems (IT). Nevertheless, the approach has two main limitations: (i) the authors used only a single source of resources (Merlot repository) discarding the others; and (ii) only data about the

[1]Tool that visits a web page and extracts preformatted information.

[2]https://www.merlot.org/merlot/

top 100 CS and IT materials introduced was analyzed. In Table I we summarize the limitations of related work.

TABLE I
SYNTHESIS OF LIMITATIONS IN RELATED WORK

| Similar Work | Limitations |
|---|---|
| Staubitz *et al.* [12], Maksimenkova and Podbelskiy [24] | Few resources, few programming domains and resources type restrictions |
| Dol and Indi [13] | External dependency and does not support different resources type |
| Tovar *et al.* [25] | Unique source and small sample (top 100) |

## IV. PLANNING OF EXPLORATORY STUDY

### A. Research Questions

As mentioned before, our goal is to analyze the availability of OERs and their support and availability for programming students and teachers. For this, we elaborated the following research questions (RQs):

- **$RQ_1$:** What challenges of teaching and learning programming are being supported by OERs?
  - **$RQ_{1.1}$:** What are the programming languages?
  - **$RQ_{1.2}$:** What types of resources?
- **$RQ_2$:** How are OERs being made available?
  - **$RQ_{2.1}$:** What are the licenses?
  - **$RQ_{2.2}$:** What are the languages?

We elaborated two main RQs: in $RQ_1$ we investigated the current programming challenges. To deepen this analysis, we checked the programming languages ($RQ_{1.1}$) and the type of resources ($RQ_{1.2}$) adopted. Meanwhile, in $RQ_2$ we examined how the resources are built and made available. For this, we expanded this question with an analysis of permissions ($RQ_{2.1}$) and the languages (*idiom*) ($RQ_{2.2}$).

### B. Data collection

Next, we defined how to collect data to solve the RQs. For this we searched for digital sources in the OER World Map[3] according following criteria:

- *Automatic search:* support automatic search to allow the web crawler to browse the collection;
- *Programming resources:* have collection of resources related to programming; and,
- *Languages:* must be in a language in which the researchers have domain and understanding.

The first criterion has been set to remove sources that support only manual search in their collections. That is, the user can only browse certain topics and cannot search for terms. The second criterion was defined to prevent sources without programming resources collections from being analyzed. Meanwhile in the third criterion, we defined that only sources in English or Portuguese should be selected, considering the languages that the authors could understand the structure of the sources and consult supporting materials, such as documentation.

After applying these criteria, we elected 11 different digital sources. The list of selected sources are shown in Table II.

TABLE II
LIST OF SELECTED SOURCES

| Number | Name | Link |
|---|---|---|
| 01 | Curriki | https://www.curriki.org/ |
| 02 | EduCAPES | https://educapes.capes.gov.br/ |
| 03 | Merlot | https://www.merlot.org/merlot/ |
| 04 | OER Commons | https://www.oercommons.org/ |
| 05 | OpenStax | https://openstax.org/ |
| 06 | Open UCT | https://open.uct.ac.za/ |
| 07 | Skills Commons | https://www.skillscommons.org/ |
| 08 | Temoa | http://temoa.tec.mx/node |
| 09 | TIB | https://www.tib.eu/en/ |
| 10 | Xpert | https://www.nottingham.ac.uk/xpert/ |
| 11 | Zenodo | https://zenodo.org/ |

### C. Data Extraction

Finally, after the preparation was started the extraction process. For this, we adopted computational processing. Thereby, for each source a web crawler was created. Initially, the crawlers searched for the term *"programming"* in selected sources. Soon after, the crawler checked the total of retrieved results and started the extraction process. Basically, this process involved reading the resource HTML, downloading its properties and converting the content to textual format ($.csv$). At the end of the process, the crawler adds all the content to a unique spreadsheet.

To discard irrelevant resources, we performed a new search based on the spreadsheet previously generated. In this new search, we only selected the resources that had some programming language name. To validate this, we adopted a list that contains 752 names of currently known programming languages[4]. As a final result, we retrieved 3,990 OERs. The detailed summary of each sources is showed in Table III.

TABLE III
DETAILS OF EXTRACTION

| Source | Resources | % |
|---|---|---|
| Curriki | 143 | 3.58% |
| EduCAPES | 88 | 2.21% |
| Merlot | 223 | 5.59% |
| OER Commons | 126 | 3.16% |
| OpenStax | 602 | 15.09% |
| Open UCT | 21 | 0.53% |
| SkillsCommons | 102 | 2.56% |
| Temoa | 2,147 | 53.81% |
| TIB | 51 | 1.28% |
| Xpert | 32 | 0.80% |
| Zenodo | 455 | 11.40% |
| **TOTAL** | **3,990** | **100%** |

[3]OER World Map is a project that lists OERs initiatives around the world, including repositories, events, publications, etc. For more information access: https://oerworldmap.org/resource/.

[4]http://bit.ly/programming-languages-list

## V. Programming Challenges Addressed by OERs

To address $RQ_1$, we summarized each challenge and its solution by resources, starting from the students' view. According to Medeiros *et al.* [2], from this perspective there are four groups of challenges:

1) **Problem formulation:** This group brings together the problems solving (difficulties to solve programming problems); abstraction (problems related to the programming concepts); and algorithm (construct and understand algorithms). We found 535 resources that directly cite such problems or have a synonym in their description. The algorithm problem was the most noticed, reaching 385 resources.
2) **Solution expression:** Medeiros *et al.* [2] organized this group into five different problems with generic and very similar challenges. So, we considered only two challenges: syntax (problems related to writing a code) and structure (data use, control structure and code). 138 resources were found quoting these term, 134 mentioned structural problems.
3) **Execution/Evaluation:** This group deals with the challenges of debugging (debugging code) and tracing (error tracking). We did not find any resource that supported the tracking problem, mainly because it is a very specific term that is not widely adopted in resources. However, 71 resources were quoted about debugging.
4) **Behavior:** This group is concerned with problems of motivation/engagement (when students are not motivated in the course), time management (schedule organization), study skills (self-teaching) and confidence (asking for help when needed). The original classification had another category called "perception of programming as a complex discipline", but we discarded this category due to its specific name. This group was mentioned in 232 resources, with motivation/engagement having 165 citations.

Considering the teachers' perspective, Medeiros *et al.* [2] originally mapped seven challenges. However, two were very similar to the student's ones, so we omitted them. The final list of challenges for teachers was:

- **Tools for teaching:** Problems related to the lack of tools and methods to teach programming. In this category, we found 538 resources that mentioned the term or synonyms.
- **Scale problems:** Teachers reported problems when escalating the exercises, such as many students and few resources for teaching. Due to the particularities of the term, we did not find relevant resources.
- **Feedback:** Common challenges regarding the lack of feedback between teachers and students. In this category, we found 240 resources that cited the term.
- **Curriculum:** Difficulties in organizing the students curriculum. We found 88 resources with this characteristic in description.
- **Mathematics background:** Gaps in mathematical knowledge. 200 resources cited this feature.

A synthesis among the problems and resource descriptions is presented in the Table IV. In parallel, we show some the OER examples in Table V. As can be seen, many of the challenges reported in the literature have the terms or synonyms in OER description. Nevertheless, the solutions are different. Most are related to the programming language, while the other challenges refer to the use of more effective techniques or tools. Because of this, we analyze in the following sections we separately analyze how resources can support students and teachers.

TABLE IV
OERs and the Challenges of Programming

| | Challenge | Resources | % |
|---|---|---|---|
| **Student** | **1) Problem formulation** | **535** | **13.41%** |
| | 1.1) Problem solving | 73 | 1.83% |
| | 1.2) Abstraction | 77 | 1.93% |
| | 1.3) Algorithm | 385 | 9.65% |
| | **2) Solution expression** | **138** | **3.46%** |
| | 2.1) Syntax | 4 | 0.10% |
| | 2.2) Structure (data, control, code) | 134 | 3.36% |
| | **3) Execution/Evaluation** | **71** | **1.78%** |
| | 3.1) Debugging | 71 | 1.78% |
| | 3.2) Tracing | 0 | 0.00% |
| | **4) Behaviour** | **232** | **5.81%** |
| | 4.1) Motivation/Engagement | 165 | 4.14% |
| | 4.2) Time Management | 17 | 0.43% |
| | 4.3) Study Skills | 15 | 0.38% |
| | 4.4) Confidence | 35 | 0.88% |
| **Teacher** | **5) Tools for teaching** | **538** | **13.48%** |
| | **6) Scale problems** | **0** | **0%** |
| | **7) Feedback** | **240** | **6.02%** |
| | **8) Curriculum** | **88** | **2.21%** |
| | **9) Mathematics background** | **200** | **5.01%** |

**Note:** We analyzed the description of the resources via computational processing, checking for terms or synonyms in the metadata. We do not verify the content of OERs.

### A. The Programming Languages ($RQ_{1.1}$)

Many students find it difficult to learn how to program, especially when learning a specific language. So, it is important to check which programming languages are being used in the elaboration of OER. Thus, we also solve the $RQ_{1.1}$. For this, we analyzed the three initial groups of challenges related to the students' view, presented in Table IV, crossing the occurrences with the 10 most common programming languages found in the data set.

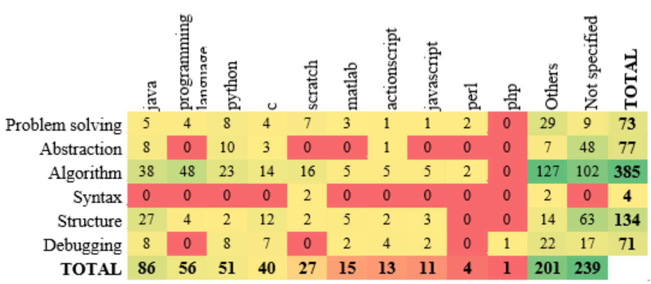
In general, the *Java* language was the most used in resources, totaling 86 occurrences. The resources identified as *"programming language"* obtained the second most cited occurrence. Although it is not, in fact a programming language the term was used to present generic resources. We noticed that 56 resources adopted this designation. The *Python* and *C* languages were, respectively, the third and fourth most common occurrences, obtaining 51 and 40 occurrences each. The *Scratch* language appeared in the fifth position, with 27 occurrences. *Matlab* had 15 occurrences and reached the sixth position. Meanwhile, *Actionscript* reached 13 occurrences and

TABLE V
EXAMPLES OF OER DESCRIPTIONS

| Challenge | Resource description |
|---|---|
| 1.1) Problem Solving | *"introduction to computers and engineering problem solving"* |
| 1.2) Abstraction | *"introduction to computer science: programming abstractions"* |
| 1.3) Algorithm | *"algorithms"* |
| 2.1) Syntax | *"the lesson targets those students who know the syntax of programming in any language"* |
| 2.2) Structure | *"pseudocode examples for control structures"* |
| 3.1) Debugging | *"[...] you'll learn more about debugging including what it is, why we use it, and what it looks like in action [...]"* |
| 4.1) Motivation/Engagement | *"this is a presentation held during the third r-ladies belgrade meetup about motivation for learning r programming [...]"* |
| 4.2) Time Management | *"self-regulated learning in higher education : strategies adopted by computer programming students"* |
| 4.3) Study Skills | *"promoting different kinds of learners towards active learning in the web-based environment"* |
| 4.4) Confidence | *"student perception of their online learning experience"* |
| 5) Tools for teaching | *"[...] a textook that teaches java programming in a novel and fun way using visuals puzzles and mysteries"* |
| 7) Feedback | *"interactive tutorial on basic programming concepts [...] there is a lot of feedback on how well they do"* |
| 8) Curriculum | *"the purpose of this curriculum is to provide an additional technology option"* |
| 9) Mathematics background | *"examine two sample programs and a sample game-programming math library [...]"* |

**Note:** challenges that did not have associated resources were omitted. We used sentences found in resource metadata such as title and descriptions, to provide an overview of the content.

was ranked in seventh. Closing the 10 most used languages, *Javascript* with 11 occurrences was in eighth, *Perl* with 4 occurrences was in ninth and *PHP* with only 1 occurrence was tenth. In addition to these languages, others were also used; however, due to their lower occurrence, they were suppressed and grouped in the "other" group. In the end, there were 201 occurrences. Finally, it was noted that 239 resources had no language specified. Observe in Figure 1 a heat map that shows the occurrence of each programming language in relation to the mapped challenges.

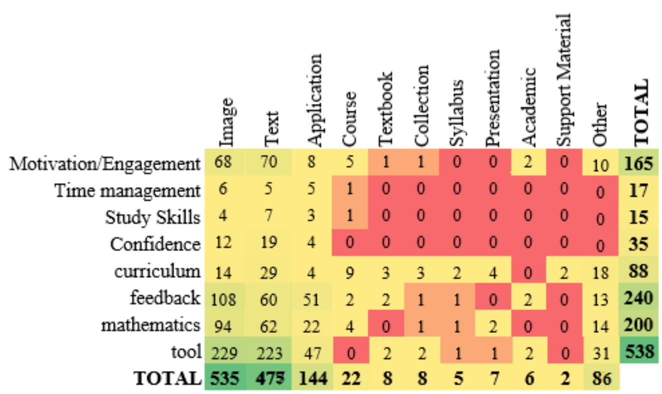


| | java | programming language | python | c | scratch | matlab | actionscript | javascript | perl | php | Others | Not specified | TOTAL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Problem solving | 5 | 4 | 8 | 4 | 7 | 3 | 1 | 1 | 2 | 0 | 29 | 9 | **73** |
| Abstraction | 8 | 0 | 10 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 7 | 48 | **77** |
| Algorithm | 38 | 48 | 23 | 14 | 16 | 5 | 5 | 5 | 2 | 0 | 127 | 102 | **385** |
| Syntax | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | **4** |
| Structure | 27 | 4 | 2 | 12 | 2 | 5 | 2 | 3 | 0 | 0 | 14 | 63 | **134** |
| Debugging | 8 | 0 | 8 | 7 | 0 | 2 | 4 | 2 | 0 | 1 | 22 | 17 | **71** |
| **TOTAL** | **86** | **56** | **51** | **40** | **27** | **15** | **13** | **11** | **4** | **1** | **201** | **239** | |

Fig. 1. Heat Map of Students Challenges *versus* Programming Languages

## B. The Resource Types ($RQ_{1.2}$)

We find many types of resources to support the challenges of teaching and learning programming, solving the $RQ_{1.2}$. The most common occurrence was the use of *Images*, with total of 535 resources. Then, resources with type of *Text* presented 475 occurrences. In this case, the resources were in plain text format or more technical, such as HTML or CSS files. Soon after, came resources of *Application* with 144 occurrences, containing software and executable programs. The *Course* resources obtained 22 occurrences. Both *Textbook* and *Collection* presented 8 occurrences, being a large part open textbooks and online courses. From this, the individual other types of resources had a frequency equal to or less than 7. This was the case of resources of the type *Presentations* that had 7 occurrences, *Academic* (such as thesis, research papers) with 6 occurrences, *Syllabus* with 5 occurrences, and *Support Material* (such as references) with only 2 occurrences. The other types were grouped in the *Other*, with 86 occurrences. Figure 2 shows the synthesis of challenges and resources types.



| | Image | Text | Application | Course | Textbook | Collection | Syllabus | Presentation | Academic | Support Material | Other | TOTAL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Motivation/Engagement | 68 | 70 | 8 | 5 | 1 | 1 | 0 | 0 | 2 | 0 | 10 | **165** |
| Time management | 6 | 5 | 5 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **17** |
| Study Skills | 4 | 7 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **15** |
| Confidence | 12 | 19 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **35** |
| curriculum | 14 | 29 | 4 | 9 | 3 | 3 | 2 | 4 | 0 | 2 | 18 | **88** |
| feedback | 108 | 60 | 51 | 2 | 2 | 1 | 1 | 0 | 2 | 0 | 13 | **240** |
| mathematics | 94 | 62 | 22 | 4 | 0 | 1 | 1 | 2 | 0 | 0 | 14 | **200** |
| tool | 229 | 223 | 47 | 0 | 2 | 2 | 1 | 1 | 2 | 0 | 31 | **538** |
| **TOTAL** | **535** | **475** | **144** | **22** | **8** | **8** | **5** | **7** | **6** | **2** | **86** | |

Fig. 2. Heat Map of Students Challenges *versus* Resources Type

In general, we noted that the three most common types of resources (*Text, Image* and *Application*) support virtually all challenges, except the Scale problem. Although the other types of resources do not have such broad coverage, they are still able to provide at least one type of resource for each challenge.

# VI. AVAILABILITY OF OPEN EDUCATIONAL RESOURCES TO PROGRAMMING

In this section we analyze $RQ_2$ about the availability of resources. Initially, to resolve this question, we extracted information about localization of OER publishers, verifying whether the OER production is a concentrated effort in some regions or if it is dispersed in initiatives around the world. However, only 273 resources (6.84% of data set) showed this information. Nevertheless, it was possible to map that the production of OERs for programming follows a global trend.

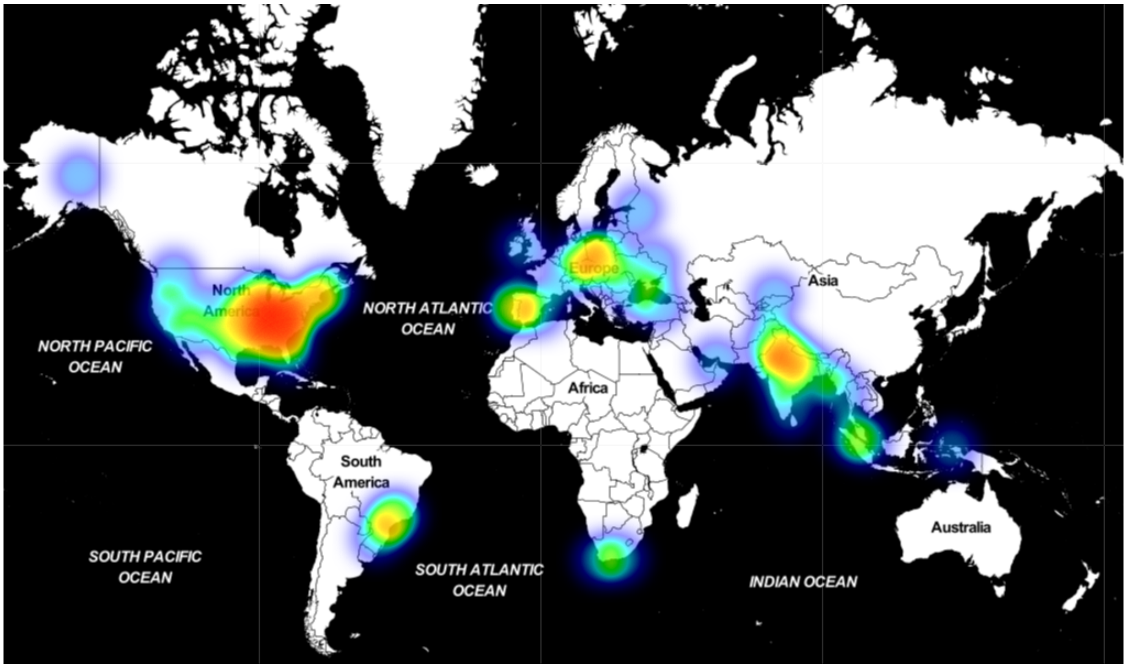



Fig. 3. Initiatives of OERs on the Teaching and Learning of Programming around the World

We found that 21 different countries are producing OER with programming content, are they: Brazil, United States, South Africa, Portugal, Spain, Ireland, Switzerland, Germany, Czech Republic, Poland, Romania, Ukraine, Russia, Turkey, Kyrgyzstan, Pakistan, United Arab Emirates, India, Myanmar, Malaysia, and Indonesia. Among these countries, United States stood out the most in terms of resources production. Practically all states near the East Cost have resources with focus on teaching programming. Another country that also had a significant presence of OERs for programming was India, with a wide dispersion of initiatives across the country.

We also noticed a significant difference between the resources produced by both countries (United States and India) related to the source. While American resources largely come from universities and colleges, Indian resources come from scientific sources, such as open access journals.

Other highlights are the countries of Brazil, Portugal and South Africa. Brazilian initiatives are disseminated in five different states via public universities. We found three initiatives in Portugal that have produced OERs with content related to the teaching and learning of programming. Finally, South Africa has two initiatives, one of which is located in Cape Town, the birthplace of the OER declaration [26]. All initiatives from different countries found in the data set are shown in Figure 3.

## A. Licenses ($RQ_{2.1}$)

We also analyzed the availability of OERs through their licenses. Initially, our investigation would be based on the identification of restrictive licenses, which could reduce the use and reuse of OERs. However, another problem was revealed. We noticed that 2,095 (52.50%) of the resources did not have any type of license. That is, the authors did not present what types of permissions users have rights over the resource (reuse, adaptation, etc.). In addition, 143 (3.58%) resources had the value *"not specified"* in the license field and 99 (2.48%) presented an invalid value, such as numbers, invalid links or business names. In the other resources, 1,451 (36.36%) had some permission derived from Creative Commons (CC) and 202 (5.06%) had other types of permissions, which we investigate next. In Figure 4 we show the dispersion between resources and licenses.

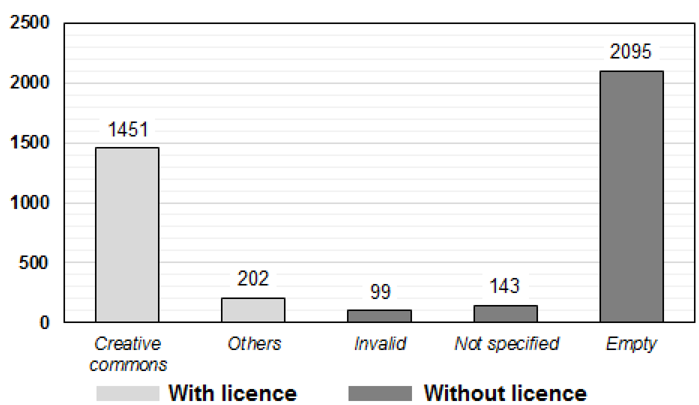



Fig. 4. The Occurrence of Licenses

Additionally, we analyze the permission level for each license. For this, we investigate three perspectives of an OER license:

- **Reuse:** Allows a particular resource to be used and reused by other users. It is the most basic concept of an OER.

- **Adapt:** Ensures that a resource can be adapted or remixed. It is a more elaborate permission, allowing users to add, remove or unify portions of the resources.
- **Change:** Grant a user the ability to change the permission when the resource is adapted. Thus, the user can adapt the OER and define a license with a lower or higher level of restriction.

In general, we noticed that the resources licensed by CC have a great support for use and adaptation. The coverage rate is 87.78% by use and 85.24% by adapt. This means that more than 85% of the data set resources that are under a CC license can be used and adapted freely by users. While this, 73.26% of CC licensed resources can be changed in your permissions in case of adaptation. We also observed that the license *CC BY* was the most common occurrence, while the *CC BY-ND*, was the least recurring.

Relating others licenses, we found 9 different types. Initially, we highlight *Custom* licenses. In most cases, the resource has the value of *"Custom License"*, and it is not possible to verify whether users can adapt and change these resources. Next, we realized the licenses derived from *Apache* and *GNU*. Both have characteristics inspired by the open software model, and for that reason they have permission to use, adapt and change. Similarly, two institutions that inspired and contributed to the dissemination of the open education movement have licensed resources. They are *MIT* and *Open University*, even for their openness characteristics, all resources licensed under these regulations have the option to reuse, adapt and change. Licenses named *Web Perl* and *BSD* also reported occurrences. However, in both cases it was not possible to verify whether they allow a license change in derived resources. The *Flicker* web site also supports the dissemination of OERs and therefore has the option for the user to license their images with permissive regulations. Finally, resources licensed as *Public Domain* were also found. In these cases, all permissions are enabled. The summary of the permissions granted by each type of license is showed in Table VI.

TABLE VI
OER PROGRAMMING LICENSES

| license | Use | Adapt | Change |
|---|---|---|---|
| **Creative Commons** | **87.78%** | **85.24%** | **73.26%** |
| CC BY | 62.01% | 62.01% | 62.01% |
| CC BY-NC | 8.71% | 8.71% | 8.71% |
| CC BY-NC-ND | 1.94% | 0.00% | 1.94% |
| CC BY-NC-SA | 13.25% | 13.25% | 0.00% |
| CC BY-ND | 0.60% | 0.00% | 0.60% |
| CC BY-SA | 1.27% | 1.27% | 0.00% |
| **Others** | **12.22%** | **10.71%** | **10.47%** |
| Custom | 1.51% | 0.00% | 0.00% |
| Apache | 0.06% | 0.06% | 0.06% |
| BSD license | 0.18% | 0.18% | 0.00% |
| Flicker | 0.06% | 0.06% | 0.06% |
| GNU | 0.48% | 0.48% | 0.48% |
| MIT license | 0.48% | 0.48% | 0.48% |
| Open University | 0.67% | 0.67% | 0.67% |
| Public Domain | 8.71% | 8.71% | 8.71% |
| Web Perl | 0.06% | 0.06% | 0.00% |
| **TOTAL** | **100%** | **95.95%** | **83.73%** |

### B. Languages ($RQ_{2.2}$)

Finally, we also analyze the occurrence of languages among the resources in the data set. In total, we found 14 different languages. However, predominantly English was the most used language among the analyzed resources, being present in 97.95% of the resources. Soon after came the resources in Portuguese (1.38%), German (0.18%), Spanish (0.15%) and Chinese (0.06%) languages. All other languages concentrated on 0.03% of occurrence each, as shown in Figure 5.

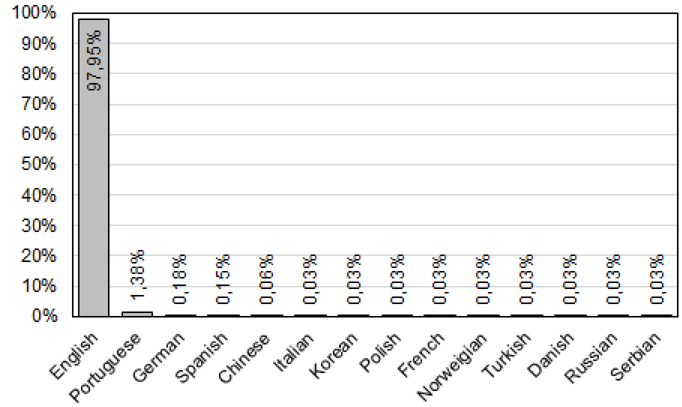



Fig. 5. Occurrence of Languages

## VII. DISCUSSION

It is possible that many current challenges in the teaching and learning of programming can be reduced or overcome with the support of OERs. As previously discussed, many problems for teachers and students already have resources that vary in format and programming language.

In this sense, OERs can serve in different perspectives. Initially, we noticed the diversity of programming languages in the resources. These languages support different paradigms, such as structured and object-oriented programming; and diverse domains, for instance web and console. Thus, OERs provide materials for various contexts of students and teachers. Similarly, we found many types of resources, such as texts, images, applications, courses, etc. This diversity of resources types can also facilitate the teaching and learning of programming, adapted according to different needs.

Furthermore, it was possible to verify a dispersion among the main producers of OER in the global scenario. In short, we revealed 21 different countries that produce OERs with content of teaching and learning of programming. This set includes countries from different continents, cultures and levels of application in programming. Thus, OERs production also supports the argument that different countries are turning their attention to developing the ability to program ( [4], [3]). Moreover, there is also the distribution of resources to assist this.

A central aspect perceived was the gap among the OERs analyzed and the behavioral challenges. Although there are resources to support such challenges, the occurrence in the data set was small. Maybe, for being generic aspects faced by many students who are learning some subject, the resources capable of supporting these challenges may be related in other areas/domains.

In parallel, two aspects called attention: first, the concentration of resources in the English language; and second, the lack of allocation of licenses on resources. The restriction of the English language is an expected trend, and many of the available materials adopt this language as a standard. However, the concentration of resources in a single language allied to the lack of licenses can reduce the potential of OERs, creating barriers where challenges should be overcome. It should also be commented the extra layers that OERs have. One of these layers was presented when we analyzed the challenges and their most common programming languages in OER (Figure 1). Most of the resources did not specify the programming language or used generic description (*"programming language"*). Another layer is the lack of attribution of licenses by users (Figure 4). These combined factors end up reducing and making the discovery, reuse and adaptation of resources more complex.

This all serves as an important factor for the wider adoption of OERs in the teaching and learning of programming. Understanding the difficulties and proposing solutions can unify an intersection between teaching/learning programming and OERs. As revealed by this investigation, many of the current problems already have support from OERs. That remains is to make the discovery of resources a more practical process.

### A. Threats to Validity

Initially, a potential threat in this investigation is related to the difficulty of generalizing the results. To avoid this, we analyzed 11 different sources of OERs and analyzing resources with a list that has 752 names of currently known programming languages. Another threat is derived from observations of the challenges of teaching programming. We restricted our analysis considering the problems cited by Medeiros *et al.* [2]. However, probably there are problems that the authors did not related. Thus, our investigation may have been affected. To avoid this, we searched for resources with close terms or synonyms present in their metadata. In this sense, it should be noted that the contents of the OER were not analyzed. The analysis was based only on the description of the metadata using computational processing.

Finally, the main limitation of our investigation refers to the difficulty of detecting resources. As there is still no effective way to identify OERs, during the collection and extraction data and their analysis, it may be that relevant resources have been discarded (or yet, that irrelevant resources have been added). This is a real challenge for any OER investigation. Currently, there are no effective search and classification criteria for resources. To mitigate this problem, we take two initiatives: initially, we created the data set only with resources that had the name of a programming language in their description; after, we gather random resources in the data set to ensure that the materials could support the teaching and learning of programming.

## VIII. Conclusion and Future Work

We conducted an exploratory study on the availability of OERs and their support for teaching and learning of programming. For this, we collected 3,990 resources with different formats and characteristics from 11 distinct sources. More specifically, we observed the main challenges faced by teachers and students, revealed the main programming languages and the most common material types presented in OERs. Also, we also checked the availability of OERs, analyzing their dissemination worldwide and their level of permission presented by the licenses.

The observed results showed us that there are many resources available. Such materials have different formats and have permissions that can facilitate their reuse and adaptation by other users. On the one hand, OERs have great potential to reduce the effort employed by students and teachers of programming. On the other hand, despite this scenario, the real potential of OERs for programming has not been fully explored yet. The problems are inherited, mainly, due to the difficulty of finding relevant resources for each user due to the wide range of options.

Considering this scenario, we believe that in the future new mechanisms must be developed capable of supporting the teaching and learning of programming. In this sense, we initially suggest the construction of a guideline to programming OER, highlighting aspects such as categories, metadata, and objectives. In addition, we suggest develop models to classify OER according to their characteristics.

## Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and the Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP 2018/26636-2).

## References

[1] A. Luxton-Reilly, Simon, I. Albluwi, B. A. Becker, M. Giannakos, A. N. Kumar, L. Ott, J. Paterson, M. J. Scott, J. Sheard, and et al., "Introductory programming: A systematic literature review," in *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, ser. ITiCSE 2018 Companion. New York, NY, USA: Association for Computing Machinery, 2018, p. 55–106. [Online]. Available: https://doi.org/10.1145/3293881.3295779

[2] R. P. Medeiros, G. L. Ramalho, and T. P. Falcão, "A systematic literature review on teaching and learning introductory programming in higher education," *IEEE Transactions on Education*, vol. 62, no. 2, pp. 77–90, May 2019.

[3] C. Duncan, T. Bell, and S. Tanimoto, "Should your 8-year-old learn coding?" in *Proceedings of the 9th Workshop in Primary and Secondary Computing Education*, ser. WiPSCE '14. New York, NY, USA: ACM, 2014, pp. 60–69. [Online]. Available: http://doi.acm.org/10.1145/2670757.2670774

[4] S. B. de Compuração, “Introductory programming: A systematic literature review,” in *Diretrizes para ensino de Computação na Educação Básica*. Porto Alegre, Rio Grande do Sul, Brasil: SBC, 2019, pp. 1–25. [Online]. Available: https://www.sbc.org.br/documentos-da-sbc/send/203-educacao-basica/1220-bncc-em-itinerario-informativo-computacao-2

[5] A. Ferchichi, “Teaching programming via specification, execution and modification of reusable components: An integrated approach,” in *Shifting Paradigms in Software Engineering*, R. Mittermeir, Ed. Vienna: Springer Vienna, 1992, pp. 238–249. [Online]. Available: https://doi.org/10.1007/978-3-7091-9258-0_27

[6] A. Wakatani and T. Maeda, “Automatic generation of programming exercises for learning programming language,” in *2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS)*, June 2015, pp. 461–465. [Online]. Available: https://doi.org/10.1109/ICIS.2015.7166637

[7] R. Minelli, A. M. and, and M. Lanza, “I know what you did last summer: An investigation of how developers spend their time,” in *Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension*, ser. ICPC ’15. IEEE Press, 2015, p. 25–35.

[8] M. M. Arimoto, L. Barroca, and E. F. Barbosa, “Am-oer: An agile method for the development of open educational resources,” *Informatics in Education*, vol. 15, no. 2, pp. 205–233, 2016.

[9] Unesco, “Forum on the impact of open courseware for higher education in developing countries: Final report,” Unesco, Tech. Rep., jul 2002, acesso em 08 de outubro de 2019. [Online]. Available: https://unesdoc.unesco.org/ark:/48223/pf0000128515

[10] P. Boytchev and S. Boytcheva, “Innovative elearning technologies in the open education era,” in *Proceedings of the 20th International Conference on Computer Systems and Technologies*, ser. CompSysTech ’19. New York, NY, USA: ACM, 2019, pp. 324–331. [Online]. Available: http://doi.acm.org/10.1145/3345252.3345300

[11] C. Dichev and D. Dicheva, “Open educational resources in computer science teaching,” in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, ser. SIGCSE ’12. New York, NY, USA: ACM, 2012, pp. 619–624. [Online]. Available: http://doi.acm.org/10.1145/2157136.2157314

[12] T. Staubitz, R. Teusner, and C. Meinel, “Towards a repository for open auto-gradable programming exercises,” in *2017 IEEE 6th International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, Dec 2017, pp. 66–73. [Online]. Available: https://doi.org/10.1109/TALE.2017.8252306

[13] S. M. Dol and T. S. Indi, “Open educational resource (OER): C-programming handout based lab session using moodle for engineering education,” *Journal of Engineering Education Transformations*, vol. 0, no. 0, 2017. [Online]. Available: http://www.journaleet.org/index.php/jeet/article/view/111816

[14] A. S. Marcolino and E. Barbosa, “A survey on problems related to the teaching of programming in brazilian educational institutions,” in *2017 IEEE Frontiers in Education Conference (FIE)*, Oct 2017, pp. 1–9.

[15] N. Piedra, J. Chicaiza, J. López, O. Martínez, and E. T. Caro, “An approach for description of open educational resources based on semantic technologies,” in *IEEE EDUCON 2010 Conference*, April 2010, pp. 1111–1119. [Online]. Available: https://doi.org/10.1109/EDUCON.2010.5492453

[16] V. Rodés-Paragarino, A. Gewerc-Barujel, and M. Llamas-Nistal, “Use of repositories of digital educational resources: State-of-the-art review,” *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 11, no. 2, pp. 73–78, May 2016. [Online]. Available: https://doi.org/10.1109/RITA.2016.2554000

[17] M. Lokar and M. Pretnar, “A low overhead automated service for teaching programming,” in *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, ser. Koli Calling ’15. New York, NY, USA: ACM, 2015, pp. 132–136. [Online]. Available: http://doi.org/10.1145/2828959.2828964

[18] R. P. Borges, P. R. F. Oliveira, R. G. da R. Lima, and R. W. de Lima, “A systematic review of literature on methodologies, practices, and tools for programming teaching,” *IEEE Latin America Transactions*, vol. 16, no. 5, pp. 1468–1475, May 2018. [Online]. Available: https://doi.org/10.1109/TLA.2018.8408443

[19] T. de Oliveira, D. Stringhini, and D. G. M. Corrêa, “Strategies focused on the teaching of programming logic: A systematic review of brazilian literature,” in *2018 XIII Latin American Conference on Learning Technologies (LACLO)*, Oct 2018, pp. 292–298. [Online]. Available: https://doi.org/10.1109/LACLO.2018.00059

[20] H. Yildiz Durak, “The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving,” *Technology, Knowledge and Learning*, Aug 2018. [Online]. Available: https://doi.org/10.1007/s10758-018-9391-y

[21] J. Hidalgo-Céspedes, G. Marín-Raventós, and V. Lara-Villagrán, “Playing with metaphors: a methodology to design video games for learning abstract programming concepts,” 06 2014. [Online]. Available: http://dx.doi.org/10.1145/2591708.2602661

[22] S. Imberman and A. Fiddler, “Share and share alike:: Using creative commons licenses to create OER,” *ACM Inroads*, vol. 10, no. 2, pp. 16–21, Apr. 2019. [Online]. Available: http://doi.acm.org/10.1145/3324887

[23] D. Hanna, A. Abhari, and A. Ferworn, “Comparing quantitative and comment-based ratings for recommending open educational resources,” in *Proceedings of the 20th Communications & Networking Symposium*, ser. CNS ’17. San Diego, CA, USA: Society for Computer Simulation International, 2017, pp. 4:1–4:10. [Online]. Available: http://dl.acm.org/citation.cfm?id=3107979.3107983

[24] O. Maksimenkova and V. Podbelskiy, “On practice of using open data in construction of training and assessment tasks for programming courses,” in *2015 10th International Conference on Computer Science Education (ICCSE)*, July 2015, pp. 233–236. [Online]. Available: https://doi.org/10.1109/ICCSE.2015.7250248

[25] E. Tovar, H. Chan, and S. Reisman, “Promoting merlot communities based on OERs in computer science and information systems,” in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, July 2017, pp. 700–706. [Online]. Available: https://doi.org/10.1109/COMPSAC.2017.290

[26] S. of declaration, “Cape town open education declaration: Unlocking the promise of open educational resources,” in *Cape Town Open Education Declaration*, Cape Town, South Africa, 2007. [Online]. Available: https://www.capetowndeclaration.org/