A Didactic Study of an Algorithmic and Programming MOOC: Learning Strategies Adopted by Students and their Difficulties

https://doi.org/10.3991/ijet.v17i19.32161

Abdelghani Babori[1,2(✉)]
[1]Hassan First University, Settat, Morocco
[2]University of Lille, Lille, France
abdelghani.babori@gmail.com

Abstract—While researches on programming MOOCs give importance to the pedagogical issues (assessment, using of integrated development environment, analysing performance or dropout in programming MOOCs), studies focus less on didactic issues. Thus, this study aims to understand the learning process within the MOOC of "Algorithms and Programming" delivered on the national platform "Morocco Digital University (MUN)" and presents a description of the learning strategies within programming MOOCs and difficulties encountered by participants drawn from in-depth interviews with 39 participants. The results revealed that to learn within the MOOC, the students used "cognitive strategies" in terms of elaboration strategies such as linking the content with prior programming knowledge and organization strategies such as the use of flowcharts to build step by step the algorithmic knowledge and "technical strategies" in terms of mobilization of videos and quizzes of the MOOC. The students have difficulties in decomposing problems, passing from the analysis phase to the algorithmic development phase and complex treatments composed of several conditions and loops. These results can provide MOOC instructors and researchers with insights into the study and design of programming MOOCs by taking into account the various learning strategies used by students and their experienced difficulties.

Keywords—distance education, didactics, learning strategies, MOOC, difficulties, programming

1 Introduction

Over the last few years, Moroccan universities have experienced strong growth in student numbers. This massification, which makes the quality of learning low, has led Moroccan universities to express their concern about the lack of human resources as well as the conditions of teaching based on courses given in amphitheaters. This traditional model of teaching at universities has proved its inadequacy. Thus, we observe weak supervision of students, a pedagogy not adapted to the student's rhythm, a high dropout rate, a training not adapted to the job market (low employability) but also not accessible to the greatest number of students [1].

MOOCs appear as a solution to these challenges. In particular, several Moroccan universities are already involved in the creation of distance learning platforms hosting free massive open online courses at the universities of Marrakech, Fez, Settat, Beni Mellal and others. In 2016, an agreement was signed between the Ministry of Higher Education, Research and Executive Training in Morocco, the GIP FUN-MOOC and the French Embassy to create the first national-scale platform in Africa "Morocco digital university" (MUN). This agreement aims to enable Moroccan universities to develop MOOCs and SPOCs (private online courses for small groups) as well as to strengthen partnerships with French universities in this new field. Out of a total of 119 submitted projects, 49 MOOC projects were selected to be the first MOOCs present on MUN when it officially launches in June 2019. The MOOC Algorithms: Basic Concepts and Applications is one of the selected projects. This MOOC aims to assist students in the design and development of algorithms. This assistance was set up to overcome the difficulties encountered in the classroom by students developing algorithms including their execution while confronting them with other audiences. Indeed, algorithmic is a theoretical discipline that is a basis for the appropriation of all technical programming languages (C, C++, Java, PHP, etc.). It is considered an interesting step towards the development of software or computer applications. Yet, it is a difficult subject that has often been a source of problems for novice learners. The failure or drop-out rate in introductory programming courses varies from 25% to 80% worldwide. This failure is caused by both the cognitive aspects of the subject matter itself as well as the strategies adopted for teaching [2]. Thus, learners generally find themselves demotivated and unable to transfer the acquired knowledge (loops, variables, conditions, etc.) to face new encountered problems.

1.1 Learning algorithms and programming in traditional contexts

Medeiros et al. [3] found that problem solving: understanding the context of a problem, identifying key data and developing a plan to solve it, is one of the most frequently cited skills in the 100 studies that they analysed. A lack of mastery of these techniques makes it difficult to understand the programming. These authors found that problem solving: understanding the context of a problem, identifying key data and developing a plan to solve it, is one of the most frequently cited skills in the 100 articles that they explored. A lack of mastery of these techniques makes it difficult to understand the programming content. Similarly, Cheah [4], in his literature review about the difficulties in learning programming, stated that learning this subject is very hard at the early stage of education. This author stipulates that programming is characterized by a high failure rate.

In this sense, Gomes and Mendes [5] consider that the abstract nature of programming and its characteristics, as well as programming language syntax, are conceived especially for professionals and not for learners. More specifically, the notion of variable poses difficulties for novices. Indeed, Nijimbere [6] notes that the difficulty of learning a variable by students is due to the absence of this concept in their cognitive past. This author highlighted that if the mathematical variable is initially known among novices, the notion of a variable in computer programming is new. Another aspect of

this difficulty concerns the different states of a variable. In this perspective, Briant [7] highlights that elementary operations pose problems for novices. A novice student cannot distinguish between elementary operations and compound operations. Briant gives the example of the primality test for a natural number: why divide a number and check that the result is an integer are elementary operations? Otherwise, why would not the task of checking if a number is prime be an elementary operation?

1.2 Computer programming MOOCs

Massive open online courses (MOOCs) have received considerable attention from educational institutions and private enterprises. The studies undertaken on MOOCs have experienced rapid growth. This is evidenced by the production of several empirical and review studies since 2008, published in journals specialized in educational technologies [8]. Some studies focusing on programming MOOCs have been published in the last decade; among them, we note the study of Luik and Lepp [9] which emphasizes that mastering programming has been growing in recent years. Indeed, nowadays companies and institutions need increasingly people with programming skills. These authors underline that MOOCs are a possible way to satisfy this demand to teach a huge range of people. According to de Oliveira et al. [10], previous literature on learning algorithmic and programming has attempted to investigate the problems associated with computer programming learning within MOOCs. The findings stated that students are unable to apply algorithmic or programming procedures. Students also lack the motivation to apprehend the content. In a recent study conducted by Babori et al. [2] which examined the various difficulties encountered by students to apprehend the programming MOOC content, the results stated that the most difficulties faced by students are related to programming content such as decomposing problems into sub-problems and treatments composed of several conditions and nested loops.

The recent researches focusing on the learning process in programming MOOCs deal with "tools for automatic program assessment", comparing the automatic evaluation of two automatic assessment approaches [11] or "the use of integrated development environment (IDE)" in the MOOC platforms such as Codeboard tool MOOC [12] that allows improving the level of engagement and the behavior of the learners who are learning the programming content. These authors point out that this IDE can be integrated into the pedagogical scenario of the course to increase the interaction of learners in programming MOOCs. In this sense, from a design perspective, Spyropoulou et al. [13] described the process of developing a computer programming MOOC.

By analysing "student performance" in a programming SPOC, in terms of grades obtained in final exams, Psathas et al. [14] pointed out that students taking the SPOC were satisfied with the designed system and achieved higher scores on exams. In this sense, Shen and Lee (2020) examined the evaluation made by students of the programming MOOCs distributed on the Codeacademy platform which proposes to learn programming languages such as PHP, Javascript, Python. By analysing 218 responses (from 62 questions) on Quora, the authors provide implications for the pedagogical scenario of programming MOOCs without taking into account the learning strategies of students or their difficulties. Similarly, Feklistova et al. [15] explored the learner's

performance in a programming MOOC. More specifically, the authors examined how various groups of learners vary in performance.

Studies have also explored the various "predictors of students' retention" in a computer science MOOC such as programming experience, gender and pre-computational thinking skills [16]. In the same perspective, by analysing the dropout in a programming MOOC, Haar and Bell [17] examined the performance of repeat learners: learners who took the same course multiple times. The authors found that these learners complete and engage with a course more than the single-run counterparts.

Other studies examined the dropout rate in computer programming MOOC [18]. To solve this problem, Rõõm et al. [18] suggest that students take course at a level that fit them as well as to encourage learners to use course resources for help.

These same authors, in another study [19], examined the data collected from a questionnaire given to learners enrolled in an introductory computer programming MOOC in Estonia, the order in which learners follow the MOOC activities: learners generally followed the sequence of activities presented in the course environment.

Other studies analysed the forum discussions of programming MOOC [1], [20]. The study of Babori [1] described the learning strategies (cognitive and social strategies) adopted by students by examining discussion forums in a MOOC of algorithms and programming. More particularly, Nelimarkka et al. [21] examined the social help-seeking strategies in a programming MOOC: seeking help from friends and seeking help from alumni and teacher communities. Regarding the study of [20], it investigates how instructor involvement and platform features affect the quantity and quality of discussions in MOOCs.

This review of literature highlights the fact that the majority of these studies focusing on learning programming within MOOCs deal with pedagogical issues (assessment, integration of IDE, analysing performance or dropout in programming MOOCs) that are not related directly to the conveyed content (for example what knowledge is assessed in these programming MOOCs or what are the what are the concepts that pose difficulties to students). As highlighted by a recent literature about MOOCs [3]; while giving importance to the learning process, previous studies focus less on content as a research object. More specifically, a paucity of research focused on issues examining the disciplinary structure of content, the skills and abilities that are required to learn algorithmic content or the difficulties faced by students in learning specific content. This leaves an important gap and reveals the need to examine the learning strategies, the knowledge learned by students or their encountered difficulties.

In this sense, we situate this study in this context of didactic researches on MOOCs by examining the learning strategies and difficulties faced by students to learn algorithmic and programming MOOC within the conceived MOOC: "Algorithm: Basic concepts and applications" which we will describe in the next section. More particularly. This study will attempt to respond to the following research questions: **What are the learning strategies adopted by students? What is the content acquired by students within the MOOC? What are the difficulties experienced by these participants?**

2 Methodology

2.1 Context and methodology

The Algorithm: Basic concepts and applications MOOC offered by MUN (Morocco Digital University) attracted 1052 registrants. The MOOC aimed to learn how to conceive algorithms and programs. The MOOC is organized into weeks. There are six mandatory weeks dedicated to algorithmic and two optional weeks of programming in C language. Each week consists of several units composed of videos, web pages, quizzes and a discussion forum that allows discussing MOOC content. Week zero presents the overall site including an introduction to the MUN platform and a general introduction to "Algorithm MOOC: Basic concepts and applications". The other weeks are organized into themes according to a specified order: from simple to complex. For instance, in the videos or quizzes, we start with easy examples and then gradually we tackle complex concepts. In week one, variables and basic instructions are processed. Week two presents conditions. Week three covers the loops: we proposed a multitude of illustrative examples to better understand the progress of each loop. Weeks four and five deal respectively with one-dimensional and two-dimensional arrays.

2.2 Data collection method

Data were collected between January 2019 and January 2020. All registered participants (n=1052) were asked if they were willing to do interviews. Thirty-nine participants accepted our invitation. We have anonymized all interlocutors and removed any information that would expose their identify. The results of this study were sent to participants for review, comment and approval.

In order understand the learning process of students during their interactions with the MOOC, the theoretical framework mobilizing the concept of learning strategy is the main basis of the interview. This framework allows to identify three categories of learning strategies as determined by Pintrich [22]: cognitive, metacognitive and resource management.

The interview makes it possible to identify the learning process by adapting to our context, interviews carried out in previous studies [6, 23, 24]. The studies of Segantin Teruggi [23] and Miligan and Littlejohn [24] aimed respectively to identify learning strategies in open distance learning in foreign languages and MOOCs and the one of Nijimbere [7] explored how do learners appropriate algorithmic concepts. Thus, these authors examined learning strategies adopted by students in face-to-face classrooms and their difficulties which include content difficulties related to algorithmic concepts and procedures, and technical difficulties related to programming features or tools. Demographic information regarding the registered students was given in the Table 1 below (gender, age, degree of education, prior programming experience). Thirty-nine participants, 61.53% of whom were females were given 60 days to do interviews. The average age was 19 years old. Of the 39 participants, 87.17% have prior programming experience. The majority of participants had only a baccalaureate degree (74.36%). The majority of interviewed students came with previous experience especially with variables and basic instructions (assignment, input and output) (see Table 2).

Table 1. Demographic information and prior programming experience for the 39 students

Gender	Males (38.47%), Females (61.53%)
Age	18–19 (51.28%), 20–22 (25.64%), 23–25 (12.82%), 26–30 (10.26%)
Degree of education	Baccalaureate level (74.36%), BAC+3[1] (12.82%), BAC+4 (5.13%), Master's degree (5.13%), PhD degree (2.56%)
Prior programming experience	Yes (87.17%), No (12.83%)

Table 2. Prior experience with programming concepts

Prior Experience with Programming Concepts	%[2]
Variables	87.17 (n=34)
Basic instructions	87.17 (n=34)
Conditions	51.28 (n=20)
Repetitive instructions	51.28 (n=20)
Arrays	25.64 (n=10)

We conducted separate 20 min interviews with participants: face to face interviews with 20 students in the study room of the Faculty of Sciences and Techniques of Settat and skype interviews with 19 participants from the Moroccan universities (Hassan II University, University of Moulay Slimane, Ibn Zohr University, Cadi Ayyad University, Mohammed V University). Then, we recorded interviews with Audacity software and transcribed verbatims. The interviews were conducted as semi-structured based on an interview guide and including four sub-axes including 13 open-ended questions (see Appendix): knowledge acquired, interactions with videos and interactions with MOOC activities, and the difficulties faced by participants. The sub axis "knowledge acquired" of the interview concerns contains questions that identify acquired elements to design algorithms.

The sub-axis "interactions with MOOC videos" deals with student's actions during and after viewing MOOC videos. This sub-axis contains questions that specify how students interact with the videos, such as remembering video content, returning to the videos, taking notes, using graphics or screenshots, and so on. The sub-axis "interactions with MOOC activities" includes questions that allow identifying actions during interaction with activities such as consulting a resource or a quiz. These actions include, for example, the learner's behavior regarding the quizzes (feedback provided, repetition of what students did not understand, the order in which the MOOC sequences are followed and so on.).

The sub-axis "difficulties encountered by students" deals with difficulties faced by participants during the task of developing an algorithm. We asked participants, questions that focused on the difficulties of understanding the concepts of algorithms and using the platform's features.

[1] A three year degree after the baccalaureate (Licence degree)

[2] This is a multiple-answer question. Participants were asked to select all that. % = n/39

2.3 Data analysis techniques

We used a thematic content analysis technique. The data obtained from the interviews were analysed using qualitative Nvivo software. This software assists in the organization of the corpus into themes and sub-themes. Indeed, the excerpts of interviews were collected and read thoroughly to suggest thematic categories. These excerpts were divided into units of meaning (shorter segments of text that can be associated with a category). For instance, for this transcript 'I understand the process of a loop but once I find myself faced with a for loop inside another for loop, I block even worse if I don't know that I must use a nested loop if the problem requires it' (Participant 2), was assigned to the category 'difficulties related to content whereas this excerpt 'I have difficulties in the handling of Codecast which took time to me. I do not know how this tool works' (Participant 6) was assigned to "technical difficulties". Moreover, this transcript 'I tried to make flowcharts to understand the logic of the algorithm' (Participant 7), was assigned to the category "Organization cognitive strategy" whereas this excerpt 'For a week I watched the videos and MOOC resources' (Participant 9) was assigned to "technical strategy".

The data was first coded and the themes were created by the author. Randomly selected segments related to themes were given to two researchers. An inter-rater agreement allowed checking that each category was associated with the proper thematic types. A percentage agreement report was 86%. We note that the categories must be explicit and mutually exclusive (each unit of meaning must only fall under one category) and they must make sense in terms of research in the field.

3 Results

3.1 Learning strategies and acquired content

Cognitive learning strategies. The interviews show that students adopted cognitive learning strategies through the use of elaboration and organization strategies. The elaboration strategy "make the connection between what is being learned and previous knowledge" was adopted by students to better retain new knowledge and relate it to prior knowledge. Students make these connections by comparing new knowledge to existing schemes (representations). Thus, for example, to assimilate the notion of algorithms, two students link with algorithms issued from situations of daily life (Participant 4) or video games (Participant 7).

> *"uh …. once I read the definition of an algorithm, I gave myself some concrete examples to assimilate this notion such as the description of the steps to follow to find a place with Google Map" (Participant 4)*
>
> *"I automatically thought of the games that allow me to program people so that they can do things such as climb a staircase, score a goal and so on." (Participant 7).*

The analysis of interviews shows that to begin reading the resources, the students asked themselves questions related to the content. These questions are mobilized to make connections with previous knowledge (Participant 4, Participant 14,

Participant 15) or experienced situations (Participant 9, Participant 10, Participant 11, Participant 15, Participant 16). The Table 3 below presents examples of excerpts that describe questions asked by students to understand algorithmic concepts.

Table 3. Example of questions asked by students during the discovery of concepts

Questions Asked so that Prior Knowledge is Activated	Questions Asked so that Prior Experience is Activated
"What does the concept discussed makes me think of?" (Participant 4)	*"Have I seen a similar situation?" (Participant 16)*
"What is the usefulness of one instruction compared to another?" (Participant 4)	*"Can I compare the problem being addressed to one I've seen before? If yes, what are the steps to be described? input and output data? And son on" (Participant 10)*
"Do I know the syntax of the algorithms used?" (Participant 14)	*"What links can I make between the problem and another problem already seen" (Participant 15)*
"What types of data and instructions are used in the algorithms presented?" (Participant 15)	*"Does this knowledge fall under the scope of experiences from everyday life or from other subjects?" (Participant 9)*
"What are the input and output data?" (Participant 15)	*"If I take a concrete example, in this case how can I execute the algorithm?" (Participant 11)*

Students also mobilize organizational strategies to perceive and capture data, to organize information and build links between it. More specifically, the students used flow charts in order to understand the sequence of algorithms using conditions and loops. Thus, for example, these flowcharts are mainly used by students, before embarking on the development of algorithms so as to represent basic instructions, complex control structures (conditions and loops) and execute algorithms by looking for all the errors that could occur as illustrated by these excerpts.

> *"I watched the videos of week one dealing with the basic instructions so as to see how this MOOC is organized… at the same time […] to execute in the form of flowcharts the algorithms because the teacher explain briefly" (Participant 5)*
>
> *"Yes, I tried to make flowcharts in order to understand the logic of the algorithms studied" (Participant 7)*
>
> *"I use flowcharts to run the algorithms because imagine an algorithm where you have to use several conditions and loops at the same time. That is to say a nesting of several if and/or loops…, uh…, in this case we have to make the task a little easier by organizing the steps of the algorithm, that is to say, what will be first in order so as to understand the sequence of actions before proceeding to write the algorithm and then… the program" (Participant 9)*

Technical strategies. The analysis of the interviews shows that students used two pedagogical resources: "videos" and "quizzes" to carry out tasks. Thus, several respondents stated that they followed the MOOC by watching videos (reading, pausing, downloading and so on.) or by answering automatic exercises (answering a quiz, gap-filling text) as illustrated in the following excerpts:

"I watched the two videos of the week and tried to download the videos first and save them on my computer so I could play them without worrying about the internet connection" (Participant 7).

"For a week I watched the videos and tried to understand what is said in the videos" (Participant 9)

"I watched the entire videos to see what they [treat] as information and then I came back to note the algorithms" (Participant 10).

Some students use other MOOC resources by viewing a web page, posting or answering questions on the discussion forums:

"For me, for the serious games proposed in week one, … uh, …, at first I didn't understand how to solve the problems but after trying to understand and rereading the examples, I understand [provided]" (Participant 2)

"No, I didn't watch the videos … I used the discussion forums instead of asking my questions directly" (Participant 14)

Acquired content. The algorithmic acquired content varies from one participant to another: they range from the notion of a variable to repetitive instructions and basic instructions used to solve problems. More particularly, content is presented in two forms: conceptual knowledge and procedural knowledge (Table 4).

Table 4. Algorithmic content learned by students within the MOOC

Category	Acquired Knowledge
Conceptual knowledge	✓ Loops ✓ Conditions ✓ Link between variable and basic instruction
Procedural knowledge	✓ The procedure to follow to achieve the result ✓ Create an organization chart (flowchart) ✓ Develop an algorithm ✓ Break down the problem into sub-problems that are easier to solve ✓ Understand the problem statement ✓ Analysing requirements ✓ Determine the data required to solve a problem ✓ Determine the resulting data

The first category of knowledge "conceptual knowledge" includes basic theoretical knowledge in algorithms such as loops and conditions. The second category of knowledge "procedural knowledge" refers to the procedures to be followed to develop algorithms. The interviews showed that participants acquired conceptual and procedural knowledge. When asked to report the elements they learned during the MOOC, participants placed more emphasis on conceptual and procedural knowledge. Regarding conceptual knowledge, eight participants mentioned loops as concepts learned during the six weeks, seven mentioned the notion of condition. One participant mentioned linking the two concepts: basic instructions and variables. In terms of procedural knowledge, 21 participants highlighted the process to be followed to achieve results, one of them explicitly specified this procedure which consists of 1) understanding the problem statement, 2) dividing the problem into sub-problems that are simpler to solve,

3) associating each sub-problem with a necessary data and resulting data, 4) starting from a data set to achieve the result. Eight highlighted the creation of an organization chart and its transformation into an algorithm. Seven specified procedures for developing algorithms.

3.2 Encountered difficulties

Technical difficulties. The analysis of interviews shows that some participants have encountered technical problems related more to the use of programming tools than to the MOOC's functionalities. Indeed, we note that almost none of the interviewees took time to visualize the "Getting Started with the MOOC" web page, which describes the basics of navigation through the MUN platform. Participant 1 justifies this by the fact that the use of the platform does not require specific technical skills:

"I managed all alone to navigate the MOOC on my own; I find it easy to move from one activity to another" (Participant 1).

Regarding the difficulties related to the code development tools, we note that some participants expressed their difficulties in using the tools integrated into the platform. Thus, participants expressed their views on the difficulty of using and understanding the serious game "Blockly games" and the code editor "Codecast". We give here some examples from their statements:

"For me, for this tool: serious games offered in week 1, at first I didn't understand how this serious game works" (Participant 2)

"I had difficulties in using the Codecast tool which took me a long time" (Participant 2).

"I have difficulties in the handling of Codecast who took time to me. I do not know how this tool work" (Participant 6).

Content difficulties. The analysis of interviews makes it possible to deepen and enrich the results obtained previously. The interviews allowed us to characterize the difficulties related to algorithmic content. Thus, we identified difficulties related to variables, assignments, complex treatments and problem-solving skills. We note that some participants have considered while loops to be easy to learn. Difficulties related to the nesting of these loops remain quite common for participants as illustrated by these statements:

"I understand the process of a loop but once I am faced with nested loops, I block even worse if I don't know that I have to use interlocking loops if the problem requires it" (Participant 2).

In this sense, two participants expressed difficulties related to the complex treatment (several instructions):

'I found difficulties in developing complex algorithms that use compose complex treatments' (Participant 18).

'I can solve the simple problems such as sum, difference, product, etc. but once I find myself with complex structures and complex treatments, I get stuck in the resolution process' (Participant 4).

More specifically, among these loops, the "While" loop appears to be the most difficult for interviewees to understand than the "For" loop. However, "Repeat…Until" is considered the easiest to understand and the most affordable instruction to develop algorithms.

'especially the While loop as long as it was a problem for me in the course than For and repeat until loops' (Participant 2).

Another particular difficulty highlighted by participant 11 is the notion of a variable and its assignment, which seems to create confusion, among participants, with mathematical knowledge:

'At first, I confuse the equality relationship with the assignment instruction. So, I tried to make it simple by taking examples of algorithms and converting them into Pascal programs to see the evolution of the values' (Participant 11).

In terms of acquired resolution approaches, some participants try to solve problems without fully understanding them. Sometimes this happens because participants have difficulties with interpreting problem statements, data and actions. As a result, they do not correctly interpret the statement when describing the given problem as illustrated by participant 1:

'Among the difficulties I have encountered in this MOOC is the problem of automatic payment application in fact I did not understand the cases presented to pay the sum of money, there are many cases and I do not know how to formulate them in the algorithm' (Participant 1).

Moreover, the ability of decomposing problems and passing from problem analysis to an algorithmic solution seem to be more difficult for many participants. We give here some examples from their statements:

'There is a lot of logic and instructions for example the example of the numerical sequence where there is the factorial and the sum of the numbers, I don't know how I will use what I all know about these two problems of factorial and sum of the numbers already seen to write the algorithm. I do not know where to begin and how I can do it' (Participant 15).

'I found difficulties in formulating algorithms because even if I know how to analyze a problem in terms of input and output data. I can't combine all this to formulate an algorithm' (Participant 17).

'I had difficulties in the problem of robot in the algorithms of sublime, perfect and dilated numbers, there is more calculation I didn't know could I mobilize the loops, in fact I know I have to use it but how that is my problem' (Participant 20).

Participant 19 expressed a lack of knowledge transfer justifying this lack by a poor link between the problems already worked on and the new problem:

'In fact, in other subjects such as physics and life and earth sciences, we can understand the concepts by always using analogies, for example, to understand the electric circuit we use the water circuit, but for the algorithmic, it is something else' (Participant 19).

The majority of participants appreciated the MOOC, which enable participants to manipulate the notions of algorithms; however, a minority of participants (2/20) remain less interested in algorithms:

'*I don't like algorithms because it's different from mathematics, in mathematics we solve equations, there are calculations but in algorithms, I don't see any calculation' (Participant 6).*

'Algorithmic is difficult, how would I say well, I can't think about how to run programs I don't have the imagination to think about that' (Participant 12).

4 Discussion

In the following, the results corresponding to each research question are briefly discussed.

4.1 What are the learning strategies adopted by MOOC participants?

To our knowledge, no study has explicitly identified learning strategies adopted by students in an algorithmic and/or programming MOOC. The results of our study reveal that students used cognitive strategies of elaboration (asking questions, linking content to prior knowledge) and organization (using flowchart) to appropriate algorithmic content.

If the characterization of the appropriation of the content is not one of the main objectives of previous researches, it is nevertheless implicitly mentioned in several studies [25, 26, 27, 28]. Thus, in a study focusing on the examination of interactions between participants in a cMOOC of Javascript programming, Andersen and Ponti [25] stated, implicitly, that, through the analysis of 160 discussion posts from the MOOC, participants used two cognitive strategies: "problem identification" which is visualized as being mainly centered on the identification of needs by a user: students discuss their problems and tasks in the MOOC and "co-creation of tasks" which consists of a collaboration between MOOC organizers and users who expand their proximal areas of development (by interacting with more experienced users).

Unlike the study of Cohen and Magen-Nagar [26] which reports that MOOC participants weakly used cognitive learning strategies, our study shows that students adopted more cognitive strategies such as "make a connection between what is being learned and previous knowledge" and the organization strategy "using flowcharts" to appropriate algorithmic content (essentially loops and conditions). This result can be explained

by the fact that students are aware of the strategies which are adapted to learn the abstract concepts of algorithmics and that a simple repetition or a learning by heart of the concepts do not allow to assimilate the foundations of this subject [29].

The technical strategies adopted by students consist of using pedagogical resources. In particular, students used videos to build essentially the two concepts: loops and conditions. The students mobilized learning strategies during and after viewing videos such as taking notes, taking screenshots, working with sample algorithms and returning to the videos. These results are in line with Chen and Chen [30] who stated that participants in a Coursera MOOC mentioned using technical learning strategies. Indeed, these participants reported downloading and reading video captions before viewing the videos in order to understand the course content. Other participants stated that they moved from one quiz to another and watch the videos to quickly solve quizzes.

Despite the fact that the platform of our MOOC "Algorithmic: Basic Concepts and Applications" does not incorporate a note-taking feature, it is found that students reported taking notes during or after watching the videos. This result corresponds to the findings of Milligan and Littlejohn [24] who reported that learning in MOOCs is no longer simply confined to viewing videos, but it involves taking notes and doing quizzes. Seen as a way to strengthen their understanding of video content, these authors point out that students have used note-taking by deploying strategies. These results are explained by students' perceived usefulness of the note-taking strategy for understanding video content.

4.2 What are the content learned by participants within the MOOC?

The designed MOOC aimed to help participants develop, step-by-step, design of algorithms. First, with this objective, participants have acquired conceptual and procedural knowledge. This result corresponds with those of other learning contexts (other than MOOCs). For instance, in face-to-face learning, the study of Nijimbere [6] shows that participants learned the concepts of loops and variables (the fundamentals in learning algorithms). Nevertheless, we can think that even if the participants mention, in our research, acquiring the two concepts: loops and conditions, they implicitly acquired other concepts related to conditions and loops such as variables and basic instructions (assignment, reading and writing). Indeed, as Nijimbere points out, the concepts of algorithms are interrelated even if some are more important than others. Thus, for example, we cannot learn the concepts of loops without apprehending variables and basic instructions (assignment, input and output).

Second, the results show that participants have acquired two types of procedural knowledge: analysing problems such as determine the input and output elements and running algorithms. Moreover, we pointed out that although problem-solving skill is considered as essential for developing algorithms and programs [2, 5], participants have weakly acquired this procedural knowledge. As Medeiros et al. [3] highlighted, the problem-solving skill (understanding the context of a problem, identifying key data and developing a plan to solve it) is one of the most frequently cited skills in the 100 research that they analysed. A lack of mastery of these techniques makes it difficult to understand the programming.

4.3 What are the difficulties experienced by MOOC participants?

As described in the introduction, all the studies we have presented could shed light on the fact that participants experienced conceptual and procedural difficulties related to the content of algorithms. However, they do not allow a full understanding of the different difficulties of learning within the MOOC. The interviews revealed that participants also encountered difficulties related to code development tools. These difficulties mainly concern the use of blocky games (serious games) and the Codecast tool integrated into the MOOC platform. In this sense, Liyanagunawardena et al. [31] noted that most participants of the MOOC "Begin programming: build your first mobile game" which focused on programming mobile applications (Android), had technical difficulties related not only to run the game because of system requirements but also to Internet connectivity and download problems. Watson et al. [32] reported that MOOC participants faced technical challenges related to platform functionality: participants felt that the number of allowed attempts (in the quizzes) was too restrictive (2 attempts). This led the pedagogical team to redesign the MOOC quizzes to allow more attempts.

Regarding content, the results of the interviews showed that participants expressed difficulties with some elements of algorithms: variables and nested loops. This result can be explained by the fact that it is difficult to make analogies: these elements are abstract concepts and thus cognitively complex to understand without a similar phenomenon in daily life for comparison [33]. This result corresponds with outcomes of other learning contexts (other than MOOCs). For instance, the results of the study of Mhashi and Alakeel [34] which investigated and analysed the problems faced by computer programming participants at the University of Tabuk, revealed that loop structures, recursion, pointers are the three major difficult concepts reported by participants. Likewise, Derus and Ali [35] pointed out that variables, multidimensional arrays, loops and functions are notions that learners find difficult to understand. More particularly, participants considered the "While" loop more difficult to understand than the "For" loop. This result can be explained by the structure of the "For" loop where the number of iterations is known in advance as opposed to the "While" loop [6]. However, the biggest problem of novice programmers does not seem to be the understanding of basic concepts but problem-solving skills. The interviews revealed that decomposing problems and transition from the analysis phase to the algorithmic development phase were considered difficult by most participants. This reflects the findings of other previous studies [3, 33, 36]. Indeed, Jenkins [36] pointed out that the most difficult part of programming is translating the specification into the algorithm. According to this author, by mastering this process, the other processes (such as the translation of the algorithm into any programming language) are essentially mechanical. Piteira and Costa [37] and Lahtinen et al. [33] emphasized that the most difficult issue in programming is understanding how to design a program to solve problems. From a teacher perspective, Adu-ManuSarpong et al. [38] reported that this lack of skills (by participants) in solving problems is due to the techniques adopted by programming lecturers. Medeiros et al. [3], in their literature review on teaching and learning introductory programming in higher education, proposed a categorization of introductory programming challenges. These authors pointed out that problem solving, motivation and engagement,

and difficulties in learning the syntax of programming languages are the most learning challenges faced by novice programming participants.

A minority of previous research about MOOCs also addressed difficulties in learning programming. Indeed, in addition to these elements, which were reported by the majority of participants as difficult, other concepts and procedures were highlighted in the study conducted by Andersen and Ponti [25]. These two authors identified the three concepts of function, scope and closure as well as the execution procedure that pose problems for MOOC participants. By co-creating content, these participants suggest a course based on several practical examples of JavaScript programs that mobilize these elements.

4.4 Conclusion and implications for future work

This study seeks to examine the learning strategies adopted by students and their experienced difficulties within the Algorithm and Programming MOOC. The first research question sought to identify the learning strategies used by MOOC participants. The results show that the majority of interactions between students and the MOOC focused more on algorithmic content. More specifically, the learning strategies adopted by students are declined in terms of elaboration strategies by linking the content with the knowledge already acquired and organization strategies by using flowcharts but also in terms of technical strategies such as the mobilization of MOOC videos. However, students rarely mentioned using planning, concentration and time management strategies. Indeed, the lack of effective use of these strategies can result in a disorganized course. Thus, in addition to the content to be taught, teaching to MOOC participants, the planning strategies would be of great use to registrants for a better organization of participation in the MOOC. Also, the integration of tools in the MOOC such as electronic agenda, daily or weekly schedules and checklists are useful to help participants to be motivated and well organized in the MOOC.

The second research question examined the learned content in terms of conceptual and procedural knowledge that participants acquired to design algorithms. The findings reveal that participants acquired two types of knowledge: “conceptual knowledge”: conditions and loops and “procedural knowledge”: analysing problems and running algorithms.

The third research question examined the difficulties experienced by participants during the learning process in the MOOC. The designed MOOC allowed participants, as shown in the results, to acquire basic programming concepts: “conditions and loops”. However, decomposing problems into sub-problems and complex treatments remain quite difficult to master for most participants. Thus, it would be necessary to proceed to remediation that explicitly describes, step by step, executions of algorithms that mobilize treatments composed of several conditions and loops. These executions should also present how to solve problems while helping participants to develop procedural strategies [29]. Interactive exercises might also be useful by providing direct assistance in the MOOC and by monitoring participant progress so that MOOC designers or teachers can provide supports for low-performing participants.

The article presented here reveals several issues related to learning in a MOOC of algorithmic and programming and offers a possible path for future research. However, it has several limitations. Indeed, only one MOOC was studied. Without repeating the study with other content, it would not be possible to generalize the obtained results. As a future work, we propose other explorations on other subjects such as mechanics, databases, management, analysis, algebra which pose, in particular, problems for students.

This synthesis reveals several implications for future research. Indeed, helping participants to develop problem-solving strategies, such as decomposing problems into sub-problems. More specifically, decomposing complex tasks into a succession of simple tasks, moving gradually from an abstract description of the solution to an algorithm solving the problem [2] may be an avenue to explore for future research on learning programming. The obtained results can provide not only the pedagogical designers of MOOCs with elements on which they can base algorithmic content design, but also research avenues to be explored by researchers. Several possible questions emerge, related to this research orientation: What factors influence the adoption of a learning strategy? What is the link between the learning strategies used by students and academic success? What are the causes of the difficulties experienced by participants within the MOOC? What is the link between the background in programming and these difficulties?

5 References

[1] A. Babori, "Analysis of Discussion Forums of a Programming MOOC," *TEM Journal*, vol. 10, no. 3, pp. 1442–1446, 2021. https://doi.org/10.18421/TEM103-56

[2] A. Babori, H. F. Fassi, A. Hariri, M. Bideq, & A. Zaid, "Using Problem Based Learning Environment to Enhance Algorithmic Problem Solving Skill," *In Global Summit on Computer & Information Technology (GSCIT)*, Sousse, Tunisia, 2016. https://doi.org/10.1109/GSCIT.2016.10

[3] R. P. Medeiros, G. L. Ramalho, and T. P. Falcao, "A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education," *IEEE Trans. Educ*, vol. 62, no. 2, pp. 77–90, 2018. https://doi.org/10.1109/TE.2018.2864133

[4] C. S. Cheah, "Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming: A Literature Review," *Contemp. Educ. Technol*, vol. 12, no. 2, pp. 272, 2020. https://doi.org/10.30935/cedtech/8247

[5] A. Gomes and A. Mendes, "Teacher's View about Introductory Programming Teaching and Learning: Difficulties, Strategies and Motivations," In *Proc. FIE*, Madrid, Spain, 2014. https://doi.org/10.1109/FIE.2014.7044086

[6] C. Nijimbere, "L'enseignement de Savoirs Informatiques Pour Débutants, Du Second Cycle de La Scolarité Secondaire Scien-tifique à l'université En France: Une Étude Comparative," Ph.D. dissertation, Sorbonne University. Paris, France, 2015.

[7] N. Briant, "Etude Didactique de La Reprise de l'algèbre Par l'introduction de l'algorithmique Au Niveau de La Classe de Seconde Du Lycée Français," Ph.D. dissertation. Montpellier II-Sciences et Techniques University, Montpellier, France, 2013.

[8] A. Babori, H. F. Fassi, and A. Zaid, "Research on MOOCs: Current Trends and Taking into Account of Content," *In Proceedings of the 2nd International Conference on Networking, Information Systems & Security,* Rabat, Morocco, 2019. https://doi.org/10.1145/3320326.3320349

[9] P. Luik, M. Lepp, "Are Highly Motivated Learners More Likely to Complete a Computer Programming MOOC?," *Int. Rev. Res. Open Distrib. Learn*, vol. 22, no. 1, pp. 41–58, 2021. https://doi.org/10.19173/irrodl.v22i1.4978

[10] M. M. de Oliveira, L. Natan Paschoal, P. M. Mozzaquatro Chicon, and E. Francine Barbosa, "Towards an Open Educational Resource Sensitive to Student's Context to Support Introductory Programming Courses," In *Proc. FIE*, Uppsala, Sweden, 2020. https://doi.org/10.1109/FIE44824.2020.9273945

[11] A. Bey, P. Jermann, P. A. Dillenbourg, "Comparison between Two Automatic Assessment Approaches for Programming: An Empirical Study on MOOCs," *J. Educ. Techno. Soc*, vol. 21, pp. 259–272, 2018.

[12] J. M. Gallego-Romero, C. Alario-Hoyos, I. Estévez-Ayres, and C. Delgado Kloos, "Analysing Learners' Engagement and Behavior in MOOCs on Programming with the Codeboard IDE," *Educ. Technol. Res. Dev*, vol. 68, no. 5, pp. 2505–2528, 2020. https://doi.org/10.1007/s11423-020-09773-6

[13] N. Spyropoulou, G. Demopoulou, C. Pierrakeas, I. Koutsonikos, and A. Kameas, "Developing a Computer Programming MOOC," *Procedia Computer Science*, vol. 65, pp. 182–191, 2015. https://doi.org/10.1016/j.procs.2015.09.107

[14] G. Psathas, C. Katsanos, T. Tsiatsos, S. Tegos, and S. A. Demetriadis, "A Study on the Learning Effectiveness, Quality and Usability of a SPOC on Introduction to Programming," In *Proc. 24th PCI*, Nicosia, Cyprus, 2020, pp. 46–49. https://doi.org/10.1145/3437120.3437272

[15] L. Feklistova, M. Lepp, P. Luik, "Learners' Performance in a MOOC on Programming. Education Sciences," vol. 11, no. 9, 2021. https://doi.org/10.3390/educsci11090521

[16] C. Chen, G. Sonnert, P. M. Sadler, and D. J. Malan, "Computational Thinking and Assignment Resubmission Predict Persistence in a Computer Science MOOC," *J. Comput. Assist. Learn*, vol. 36, no. 5, pp. 581–594, 2020. https://doi.org/10.1111/jcal.12427

[17] C. C. Haar and A. Bell, "Analysis of Repeat Learners in Computer Science MOOCs," In *Proc. LWMOOCS*, Antigua Guatemala, Guatemala, 2020. https://doi.org/10.1109/LWMOOCS50143.2020.9234342

[18] M. Rõõm, M. Lepp, and P. Luik, "Dropout Time and Learners' Performance in Computer Programming MOOCs," *Education Sciences*, vol. 11, no. 10, 2021. https://doi.org/10.3390/educsci11100643

[19] M. Rööm, P. Luik, and M. Lepp, "Learners' Sequence of the Course Activities During Computer Programming MOOC" *In European Conference on e-Learning?* Berlin, Germany, 2020.

[20] D. R. Waller, K. A. Douglas, and G. Nanda, "A Case Study of Discussion Forums in Two Programming MOOCs on Different Platform," *In ASEE Annual Conference & Exposition*, 2019.

[21] M. Nelimarkka, A. Hellas, "Social Help-Seeking Strategies in a Programming MOOC," *In Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018. https://doi.org/10.1145/3159450.3159495

[22] P. R. Pintrich, "A Motivational Science Perspective on the Role of Student Motivation in Learning and Teaching Contexts," *Journal of Educational Psychology*, vol. 95, pp. 667–686, 2003. https://doi.org/10.1037/0022-0663.95.4.667

[23] A. Segantin Teruggi, "Les Stratégies d'apprentissage En Formation Ouverte à Distance En Langues Étrangères: Le Cas de Trois Apprenants de FLE En Autoapprentissage," Master. dissertation. University of Geneva, Geneva, Switzerland, 2017.

[24] C. Milligan, A. Littlejohn, "How Health Professionals Regulate Their Learning in Massive Open Online Courses," *Internet High. Educ*. vol. 31, pp. 113–121, 2016. https://doi.org/10.1016/j.iheduc.2016.07.005

[25] R. Andersen and M. Ponti, "Participatory Pedagogy in an Open Educational Course: Challenges and Opportunities," *Distance Educ*. vol. 35, no. 2, pp. 234–249, 2014. https://doi.org/10.1080/01587919.2014.917703

[26] L. Cohen and N. Magen-Nagar, "Self-Regulated Learning and a Sense of Achievement in MOOCs among High School Science and Technology Students," *Am. J. Distance Educ*, vol. 30, no. 2, pp. 68–79, 2016. https://doi.org/10.1080/08923647.2016.1155905
[27] G. Veletsianos, A. Collier, E. Schneider, "Digging Deeper into Learners' Experiences in MOOCs: Participation in Social Networks Outside of MOOCs, Notetaking and Contexts Surrounding Content Consumption: Digging Deeper into Learners' Experiences in MOOCs," *Br. J. Educ. Technol*, vol. 46, no. 3, pp. 570–587, 2015. https://doi.org/10.1111/bjet.12297
[28] W. R. Watson, W. Kim, and S. L. Watson, "Learning Outcomes of a MOOC Designed for Attitudinal Change: A Case Study of an Animal Behavior and Welfare MOOC," *Comput. Educ*, vol. 96, pp. 83–93, 2016. https://doi.org/10.1016/j.compedu.2016.01.013
[29] A. Bey and T. Bensebaa, "Towards an E-Assessment Approach of Algorithmic Problem-Solving Skills Using Plan-Based Program Understanding Approach," In *Proc.* International Conference on Education and e-Learning Innovations, Sousse, Tunisia, 2012. https://doi.org/10.1109/ICEELI.2012.6360666
[30] Y. H. Chen and P. J. Chen, "MOOC Study Group: Facilitation Strategies, Influential Factors, and Student Perceived Gains," *Comput. Educ*, vol. 86, pp. 55–70, 2015. https://doi.org/10.1016/j.compedu.2015.03.008
[31] T. R. Liyanagunawardena, K. Ø. Lundqvist, and S. A. Williams, "Who Are with Us: MOOC Learners on a Future Learn Course: Who Are with Us: MOOC Learners," *Br. J. Educ. Technol*, vol. 46, no. 3, pp. 557–569, 2015. https://doi.org/10.1111/bjet.12261
[32] S. L. Watson, J. Loizzo, W. R. Watson, C. Mueller, J. Lim, P. A. Ertmer, "Instructional Design, Facilitation, and Perceived Learning Outcomes: An Exploratory Case Study of a Human Trafficking MOOC for Attitudinal Change," *Educ. Technol. Res. Dev*, vol. 64, no. 6, pp. 1273–1300, 2016. https://doi.org/10.1007/s11423-016-9457-2
[33] E. Lahtinen, K. Ala-Mutka, H. M. Järvinen, "A Study of the Difficulties of Novice Programmers," *SIGCSE Bull*, vol. 37, no. 3, pp. 14–18, 2005. https://doi.org/10.1145/1151954.1067453
[34] M. M. Mhashi and A. L. I. M. Alakeel, "Difficulties Facing Students in Learning Computer Programming Skills at Tabuk University," *Recent Advances in Modern Educational Technologies*, pp. 15–24.
[35] S. R. M. Derus and A. Z. M. Ali, "Difficulties in Learning Programming: Views of Students," In *Proc.* ICCIE, Perth, Australia, 2012.
[36] T. Jenkins, "On the Difficulty of Learning to Program," In *Proc* LTSN, Leeds, UK, 2002.
[37] M. Piteira and C. Costa, "Computer Programming and Novice Programmers," In *Proc.* ISDOC, New York, USA, 2012. https://doi.org/10.1145/2311917.2311927
[38] K. Adu-Manu Sarpong, J. Kingsley Arthur, and P. Yaw Owusu Amoako, "Causes of Failure of Students in Computer Programming Courses: The Teacher Learner Perspective," *Int. J. Comput. Appl*., vol. 77, no. 12, pp. 27–32, 2013. https://doi.org/10.5120/13448-1311

6 Appendix

Constructed knowledge

1) What do you think algorithms are used for?
2) What is the most important thing you learned about algorithms in the MOOC
3) What notions do you think are important to develop an algorithm?
   Relaunch: What are the notions or concepts that are essential for developing an algorithm?
   Relaunch: What is a variable for you? What is his position concerning the structure of an algorithm? The same for:

- basic instructions;
- conditions;
- loops;
- arrays.

Interactions with MOOC videos

1) How did you interact with the videos?
2) Did you watch the entire videos once and then pick up on some pieces?
   - Did you take breaks right away?
   - Did you watch the whole video as the last thing? Why? (for all of the above)
3) After viewing a video or resource, did you record in any way what you watched and/or listened to (graphs, charts, screenshots)? And after completing an entire sequence, did you return to the video? Or to other activities?
   - Did you return to the concepts of algorithms?
   - Did you take notes on what you did, learned, failed? How did you do it?

Interactions with MOOC activities

1) Regarding the activities proposed in each sequence of the MOOC, did you?
   - Paid attention to the feedback provided in the quizzes?
   - Repeated what you did not understand well?
   - Repeated certain activities (e.g. the most difficult one for you)?
   - Regarding the sequence proposed in the MOOC, did you do all the activities proposed in the MOOC?
2) Did you follow each sequence in the proposed order in the MOOC?/do the activities take place in a different order than the one proposed?
3) Did you go back to the video/did you ever go back to the video by taking quizzes?

Encountered difficulties

1) Did you encounter any difficulties with the content of the MOOC? Are there some points where you feel you need help from other people (teacher, other learners, etc.)?
2) You learned how to build an algorithm. What difficulties did you encounter in this task?
   Relaunch:
3) Which of the algorithmic concepts you learned did you pose problems? At what level was the difficulty for each concept?
4) Which features (or tools) of the MOOC you used seemed to pose more problems in this task?

7 Author

Abdelghani Babori, Hassan First University, Settat, Morocco; University of Lille, Lille, France. E-mail: abdelghani.babori@gmail.com

Article submitted 2022-05-02. Resubmitted 2022-07-27. Final acceptance 2022-07-27. Final version published as submitted by the authors.