Fostering Computational Thinking in Undergraduated Music Conservatory Students

Marcella Mandanici[a],
Music Conservatory "L. Marenzio", p.tta A.B. Michelangeli 1, 25121 Brescia, Italy

Keywords: Computational Thinking, Music Technology, Programming Skills, The "Four C's".

Abstract: This paper presents and discusses the syllabus of a music technology course at undergraduate level in a Music Conservatory. The aim of the course is to introduce the students to computational thinking through the use of Pure Data, a free visual programming environment for music production. Starting from very basic notions in music technology and in the use of the program, the students are guided towards the knowledge of the fundamental steps of computational thinking in music production. Simple musical structures such as pattern repetition, transposition, polyphony and chords are implemented with the use of loops, abstractions, cycles, pattern storage systems and timing controls. While students become acquainted with these processes, also creative group activities are proposed with the aim of exploring the potentialities of the numeric control of sound events and musical form.

1 INTRODUCTION

Enhancing digital skills and competences for the digital transformation is one of the two priority areas of the Digital Education Action Plan 2021-2027 [1]. Promoted by the European Union, the Digital Education Action Plan supports the effort of the educational systems to adapt to the digital age. Digital competences are considered an essential part of the curriculum of the XXI century citizen mainly after the experience of the Covid-19 pandemic, which led teachers and students to adopt as soon as possible digital tools to preserve school activities. For instance in the field of music education the importance of the use of web platforms, music production software and in general of digital music education tools emerged with unprecedented evidence, showing all their strong potentialities (Avanzini et al., 2021). This trend is confirmed by the figures collected in the open public consultation on the Digital Education Action Plan 2021-2027 which show that:

- almost 60% of respondents had not used distance and online learning before the crisis
- 95% consider that the Covid-19 pandemic marks a turning point for how technology is used in education and training
- over 60% felt that they had improved their digital skills during the crisis [2].

Computational thinking is the core of digital skills because it describes a set of attitudes and ways of thinking that allow humans to communicate with machines (Cuny et al., 2010; Shute et al., 2017). According to (Selby and Woollard, 2014), these can be represented by three fundamental ideas:

- the thought process
- the concept of abstraction
- the concept of decomposition.

The though process includes " ... *formulating problems so their solutions can be represented as computational steps and algorithms.*" (Aho, 2012) and implies abilities in problem solving (Wing, 2011), analysis and pattern recognition. Abstraction is the process of deciding what aspects of the problem to be solved are essential and need to be included in the algorithm and what can be neglected (Wing, 2008). It implies the ability to generalize a problem among many other similar cases and to write simpler and more widely reusable code. Finally, decomposition is the ability of breaking down a big problem into smaller parts and it is necessary when managing complex tasks (Selby and Woollard, 2014). Com-

[a] https://orcid.org/0000-0003-1863-4178

[1] https://education.ec.europa.eu/focus-topics/digital/education-action-plan

[2] https://ec.europa.eu/info/law/better-regulation/have-your-say/initiatives/12453-Digital-Education-Action-Plan/public-consultation_en

Mandanici, M.
Fostering Computational Thinking in Undergraduated Music Conservatory Students.
DOI: 10.5220/0011134600003182
In *Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022) - Volume 1*, pages 449-457
ISBN: 978-989-758-562-3; ISSN: 2184-5026

putational thinking in education is also connected to the ”Four C’s” that characterize the ”Learning and Innovation Skills” for the XXI century learning. The ”Four C’s” stand for Critical thinking, Creativity, Collaboration and Communication and embrace a series of social skills such as performance, simulation, appropriation, multitasking, distributed cognition, collective intelligence, judgement, transmedia navigation, networking and others (Jenkins, 2009). Critical thinking refers to the ability of evaluating the reliability of different information sources but also to the ability of checking the effectiveness of different solutions to solve a problem. Creativity is the final stage of the learning process in the revised Bloom’s taxonomy (Krathwohl, 2002), which considers the ability of putting the elements together to produce a new coherent whole as the maximum level of cognitive development. Creativity is also very important, mainly when computational thinking is applied in the field of artistic expression.

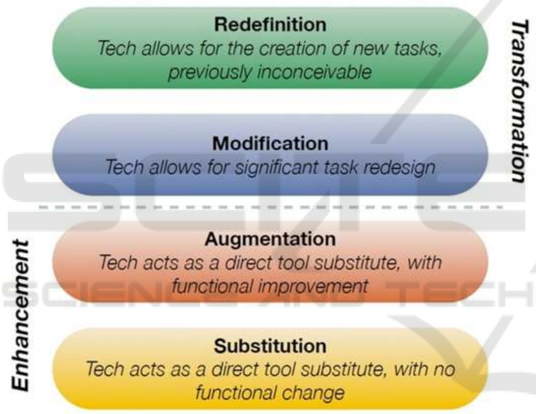

Figure 1: The SAMR (Substitution, Augmentation, Modification and Redefinition) model with its fundamental steps (Enhancement and Transformation).

The SAMR model establishes four levels of technology integration in education (Hamilton et al., 2016). Starting from the lower level (simple tool substitution), the model drives the attention towards the higher one (content redefinition), stressing the cognitive value of discovering tasks and performance possibilities impossible without the use of technology (see Figure 1). Here is where computational thinking in artistic production hits its final point, realizing artifacts and ways of expression previously inconceivable. Giving a look at the history of electronic and computer music (Copeland and Long, 2017; Collins et al., 2013), hundreds of examples of music ”redefinition” and ”transformation” can be found in the evolution and in the artistic use of sound synthesis techniques and algorithmic composition, providing thus the best proof of the intimate connection between computational thinking and creativity. Collaboration is the ability of working together to reach a common goal. It is particularly important in the process of technology integration because it helps in overcoming difficulties and knowledge gaps (consider for instance the role of blogs and online communities of practitioners where to share information and solutions to specific problems). Finally, communication is a fundamental ability when working with others. It is also the core of the artistic work and of its propagation through customized platforms, social media and video production.

In spite of the long and important tradition of electronic and computer music - which is intimately connected to computational thinking - really very few examples of application of this kind of knowledge can be found in music education (Israel-Fishelson and Hershkovitz, 2022). In Section 2 current approaches, software and tools are presented with the aim of providing a short overview of computational thinking in music education. Particularly the work of V.J. Manzo (Manzo, 2016) has inspired the design of the course presented in this paper, which will be examined in detail in Section 3. The characteristics of computational thinking and the peculiarities of the ”Four C’s” above discussed will be referenced in the presentation of the course contents, with a particular focus on creativity that should be regarded as one of the most important goals of the course. Assessment methods and tools will be discussed in Section 4, and conclusions drawn in Section 5.

2 COMPUTATIONAL THINKING IN MUSIC EDUCATION

The few experiences of computational thinking in music education found in literature refer to a limited set of approaches and software tools. Some of them address the problem of introducing computational thinking from the very early stages of school curriculum through various strategies. (Bell and Bell, 2018) suggest an interesting embodied experience of a physical parallel sorting network, a game which results in obtaining elements in ascending order. Extension of this game beyond simple numerical values can involve various domains such as music (putting sounds or notes in ascending order) or language (putting names in alphabetical order, or story elements in temporal order), and so on. Beyond demonstrating the power of abstraction in a very simple and effective way, this game also recalls the concept of algomotricity, which is an approach based on kinesthetic learning activities aimed at highlighting specific computer science topics (Bellettini et al., 2014). Another activity proposed by (Bell and Bell, 2018) is the use of little robots programmed to reach specific positions on a physical stave as to produce sound events (melodies or chords). This idea derives directly from the famous Logo turtle by Seymour Papert, the father of computer-based environments aimed at teaching computation to children (Resnick et al., 1988). The same authors also employ Scratch, a very popular visual programming language, used both at elementary and university level (Maloney et al., 2010). As Greher and Heines affirm in their book (Greher and Heines, 2014), although Scratch is designed for children it offers important tools for teaching computational thinking such as code blocks, global and local variables, lists, loops, conditionals, etc. Another popular similar resource is Blockly, a visual block language development kit that allows the building of programs aimed at various educational goals (Trower and Gray, 2015). Blockly has been used by (Baratè et al., 2017) to build a platform for melody composition and manipulation, or by (Gorson et al., 2017) to produce a sound composition tablet application. Another trend in the development of computational thinking is the use of command-line languages for music production. These are Sonic Pi (Aaron et al., 2016) and Ear Sketch (Engelman et al., 2017). While Sonic Pi is more oriented towards live performance improvisations, Ear Sketch aims at engaging students to learn programming through sound. Both programs address high school and university students. Visual programming is very popular among musicians, as the main software in use belong to this family of programs (MAX [3] and Pure Data [4]). The Pure Data free programming environment has been used by (Hancock, 2014) to develop a play-based learning course to introduce Conservatory students to music programming. A step by step approach is followed by (Manzo, 2016) who - reflecting the traditional environment division of Max, MSP (audio) and Jitter (video) - presents basic programming activities in three main domains: algorithmic composition and music training (MIDI); audio playback, processing and effects (audio); live and recorded video. Particularly in the first section of the book the author provides the very basic notions of programming in the MAX environment to lead the students to the production of simple musical structures (scales, intervals and chords). This is an interesting approach for Conservatory students for at least two reasons: firstly very basic music features may unveil their algorithmic side, showing how computational thinking is deeply rooted in music analysis; secondly the step-by-step method followed by the author very well shows the process of abstraction from particular cases towards more tidy, less redundant and general ways of formulating the same algorithm. These ideas inspired the methodology and contents of the music technology course that is the focus of this paper.

[3] https://cycling74.com/products/max

[4] https://puredata.info/

3 COURSE DESIGN

In the current programs for undergraduate instrumental studies in Italian Conservatories the technological courses (Acoustics and Music Technology) weight respectively 2 and 3 ECTS (less than the 3% of the overall 180 ECTS). Thus very often they are underestimated by teachers and students as they result completely disconnected from the other subjects of the curriculum. Yet, what little technological integration is entrusted only to these subjects. This is why the design of the course of Music Technology is so important and should be regarded as the only occasion to meet computational thinking in the whole undergraduate Conservatory curriculum.

3.1 Course Objectives

The course activities aim at the following goals:

1. introducing the students to computational thinking by enhancing their abilities in music analysis and abstraction of musical concepts
2. fostering the potentialities of music creativity through the manipulation and numeric control of simple music production algorithms
3. offering them an experience related to the ”Four C’s” (critical thinking, creativity, collaboration and communication).

3.2 Material

The course is developed in Pure Data, which is a free always available software, while MAX requires to buy a license. MAX is very rich from the graphical point of view and can rely on a lot of supportive tools. In spite of this, also if Cycling’74 offers many facilities (academic licenses, institutional discounts, monthly payments, 30 days free demos), the advantages of a free software are greater for non professional users. Thus, some of the basic algorithms presented by (Manzo, 2016) have been ported in Pure

Table 1: Course organization and content.

Week	Lesson	Workshop	Home work
1	Introduction Computational thinking The ”Four C’s”	Introduction to Pure Data Controlling numerical flow (`+random+`,`+moses+`,`+clip+`)	Pre-course questionnaires lab1_assignment (logic)
2	Audio and MIDI MIDI events	MIDI production (noteout, makenote) MIDI controls (modulation, volume, pan and sustain)	lab2_assignment (exploration)
3	Counter (recursivity, reset, loops)	The `+counter+`abstraction. A pattern of a musical form.	lab3_assignment (musical form)
4	Scales (midi notes and arrays)	Major and minor scales. Scale transposition	lab4_assignment _scale fragments (creativity)
5	Musical structures (patterns, polyphony, chords)	Pattern repetition (cycles), polyphony	lab5_assignment (chords)
6	Melodies	Augmentation, musical streams	lab6_assignment (creativity)
7	Patch modules	Work group. Graphical scores	Final group assignment
8	Group assignment discussion	Group assignment discussion	Delivery of the final project Post-course questionnaires

Data and further processed to comply the course program.

3.3 Organization and Participants

The course is addressed to Music Conservatory students at undergraduate level. To access this level a high school diploma is required, also if exceptions are allowed in case of particularly gifted students. The course takes 24 hours, subdivided into 8 weeks starting from February 8, 2022. It is held online through the institutional platform of the Conservatory[5]. All the activities are recorded and made available for the students until about two weeks after the end of the course, expected for March 29, 2022. The credits for each lesson are assigned only after delivery of the assignments contained in the videos. Currently take part in the class 55 students (24 females), aged $Mean = 23.43$ years, $StandardDeviation = 7.75$.

3.4 Contents

The program of the course is reported in detail in Table 1[6]. Course activities are subdivided into lessons - where each topic is presented in detail by the teacher -, and workshop activities - where each topic is put in practice through the realization of exercises with the participation of the whole class -. Each lesson has its own home assignment, where the students are required to develop autonomously the topics experienced during the lesson. The course activities are inspired by the ”Four C’s” abilities, whose finalization is described in the following Sections 3.4.1, 3.4.2, and 3.4.3.

3.4.1 Problem Solving and Abstraction

The ability of solving a problem is linked to critical thinking and to out of the box reasoning. Here the focus is not only on simple problem solving, but rather on the ability of formulating the solution in a general way, such to be reused in different conditions (abstraction). To show an example of this process, let’s look at the activities of week 4, which focus on the building of a musical scale. The first step shows an algorithm for the production of a C major scale, starting from MIDI pitch 60 (see Figure 2).

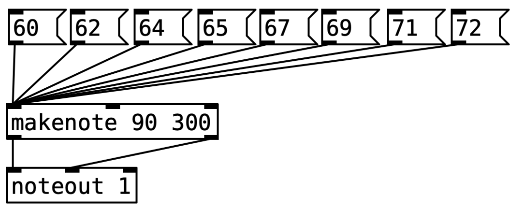

Figure 2: The basic algorithm for the production of a C major scale.

To allow the building of the scale on different fundamental pitches, the algorithm may be changed as in Figure 3, where the concept of the use of a variable (the fundamental pitch of the scale) is introduced.

[5] https://en.wikipedia.org/wiki/Google_Classroom

[6] The patches used in the course are available at https://github.com/marmanda/CompT_Pd

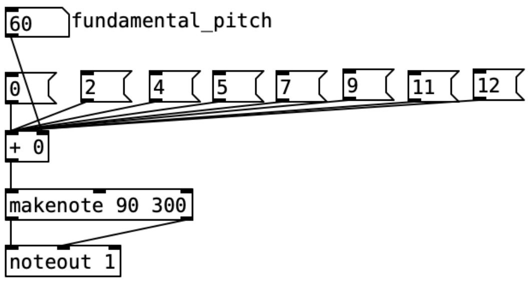

Figure 3: The algorithm for the production of a C major scale with the use of the fundamental pitch variable.

The final solution proposed is depicted in Figure 4 where the use of an +array+containing the interval information related to each scale provides a good degree of flexibility and generalization. The contents of the +array+may be changed with a message; the intervals to be summed to the fundamental of the scale are available as the output of the +tabread+object after it receives the index number.

The algorithm of Figure 5 produces a scale followed by its transpositions one fourth and one fifth up. The patch is composed by different modules aimed respectively at:

- music information management
- time control
- musical form management
- transposition
- sound production

The music information module is depicted in Figure 4. It is put into action by the time control module (+metro+and counter). The +counter+produces the sequence of indices to be sent at timed intervals to the +tabread+object. The +module+(+%+)object keeps the count inside the size limits of the +array+, producing thus a continuous cycle of scales. The module that controls the musical form keeps track of the output of the counter object and through a selector (the sel object) provides the transposition of the scales. This is realized by adding to the fundamental the quantity of semitones of the transposition interval. The same mechanism is used to stop the performance. At least the sound production module receives the MIDI data (makenote) and outputs them to an external virtual synthesizer (noteout).

3.4.2 Creativity

During the workshop the students become familiar with all the modules contained in the patch and then receive an assignment where the same patch structure is employed to create something that looks and sounds very different. The assignment is shown in Figure 6 in the form of a possible desired output. Consider the possibility of assigning a random fundamental pitch (range 30-80) each time the scale cycle begins and also to assign a random value (range 1-8) to the argument of the module object. This produces scale fragments of different length starting always from different fundamental pitches. The aim of this assignment is to stress the creative possibilities offered by a simple scale when its production algorithm reaches a level of abstraction that allows numerical manipulation. It also evidences that the traditional musical scale is only a particular case of a much wider musical universe, which can be explored with the use of these very simple programming tools. Further musical development can be obtained by programming musical streams (note bands with common timbre constrained inside height and duration ranges), impulsive events (clusters or chords), and polyphony (i.e. piping events), and so on. Such creative activities may also be inspired by the use of graphical scores, where visual elements can be freely interpreted by the students.

3.4.3 Collaboration and Communication

Weeks 7 and 8 are devoted to group activities. Each group is formed by 5-6 people. These activities aim at preparing a creative patch responding to a precise composition plan (descending from a graphical score or other organization criteria). The patch must contain the instructions for the performance of the composition, a general description of the composition plan as well as the comments explaining the functions of the objects employed. These activities involve group organization, communication among the members, creativity and the ability of negotiating with others about the best solutions to be chosen.

4 ASSESSMENT

Greher and Heines (Greher and Heines, 2014) have prepared a pre and post-course questionnaire for the assessment of course outcomes. It is divided into 4 sections:

1. reasons for taking the course
2. expectations for the course
3. self perception and
4. opinions about computing and music

The aim of this questionnaire is to take a picture as accurate as possible of the ideas of the students and

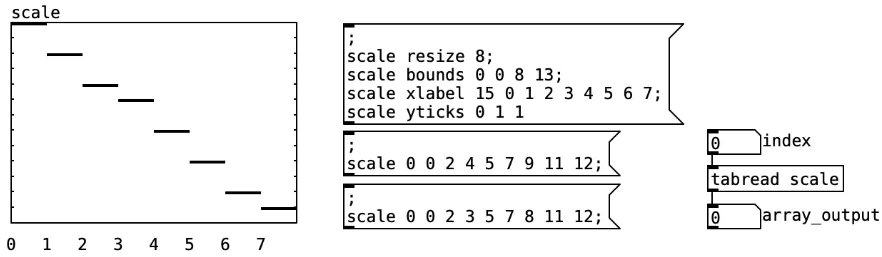

Figure 4: The music information module.

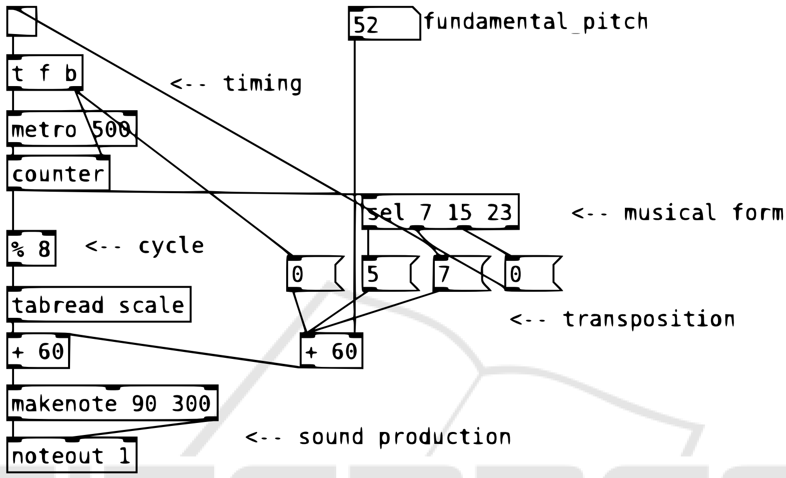

Figure 5: The complete algorithm for the production a major scale with transpositions: the fundamental pitch variable, the timing and cycle module, the control of musical form and transpositions and the sound production module.

Figure 6: A possible desired output example for the home assignment of lesson 4.

of how these may be changed after course activities. Many questions concern the ”Four C’s” abilities, and try to provoke a reflection in the students about their relationship with computers, music composition and creativity. For the present course the original questionnaire has been slightly adapted and integrated with further questions concerning basic music and music technology knowledge. The main challenge in the preparation of this part of the questionnaire has been how to measure the logical, abstraction and analytical abilities of the students before the beginning of the teaching activities. As the great majority of the students (about the 85%) has never used any music programming language before, the assessment cannot be made through language specific questions, but rather involve more general concepts. These are:

- music analysis
- patch logic and
- flow diagram.

In the pre-course questionnaire there are 2 questions for music analysis and one question for each of the remaining items. Similar questions will be prepared for the post-course questionnaire.

4.1 Music Analysis

For testing the music analysis abilities a melody is presented to the students with some examples of pattern repetition and pattern transposition (see Figure 7). In the first question the students are required to identify how many patterns they could find in the melody. In the second question they must identify the pattern repetitions and transpositions in the same melodic excerpt.

4.2 Patch Logic

This test is to check what kind of functions students consider as the most important for the automatic performance of a melody. The test is presented in the

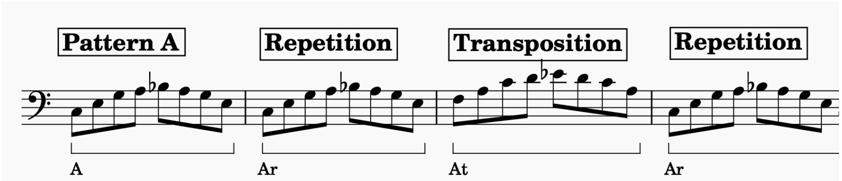

Figure 7: An example of pattern repetition and transposition.

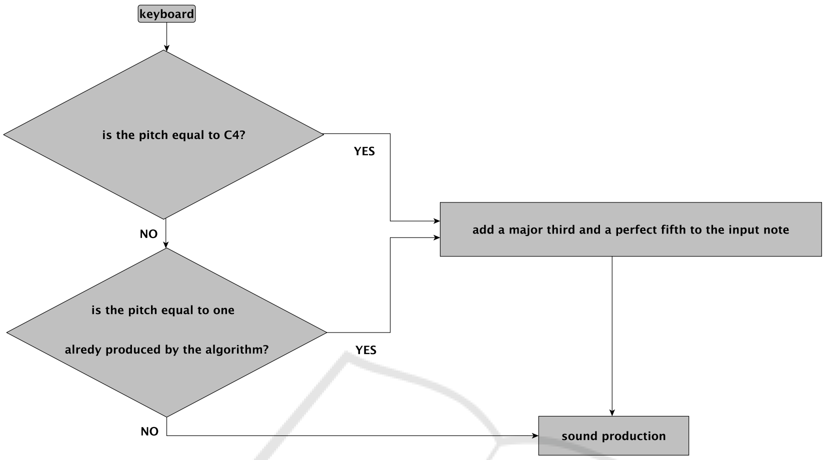

Figure 8: A flow diagram of a music production algorithm.

form of a five point Likert scale matrix (1 not important – 5 very important) with the following 15 items:

1. A sound production system
2. Start and stop controls
3. An information storage system
4. A way for controlling the number of voices for the performance
5. A control for pattern repetition
6. A list of possible note durations
7. A time control mechanism
8. A list of available note timbres
9. A control for pattern transposition
10. An external electronic keyboard
11. A way to express the pitches of the notes
12. A counter
13. A way to introduce a variable
14. A list of possible keys
15. A way to control the speed of the performance

This list has been filled in with modules actually used in the patches as well as with other musical elements that are the output of the same modules. At the end of the course students will be presented with the same matrix to verify if they can identify such dependencies.

4.3 Flow Diagram

The students are presented with the flow diagram depicted in Figure 8. They are asked to identify what can be the output if a short sequence of notes is used as input. This test checks accurately if the students understand how the algorithm works and how the system output changes after the input has been processed. A similar question will be inserted in the post-course questionnaire to see if some improvement has been achieved.

4.4 Expected Results

The main expected results for this music technology course may be grouped into two main areas: improvement of computer literacy/programming skills; enhancement of musical creativity.

4.4.1 Improvement of Computer and Programming Skills

The 6 pre-course questions about computers and programming are outlined in Table 2. The answers were collected as ratings on a Likert scale ranging from 1 (false) to 5 (true). Results are reported in Figure 9.

Although a good number of students seem to be interested in learning something more about computers and programming (Q1 and Q2), a smaller number

Table 2: The 6 pre-course questions about computers and programming.

Q1	I want to learn something more about computers
Q2	I want to learn something more about computers and programming
Q3	I enjoy working with computers
Q4	I am confident using a computer language to accomplish complex tasks
Q5	Computer programming is fun
Q6	Programming a computer is difficult

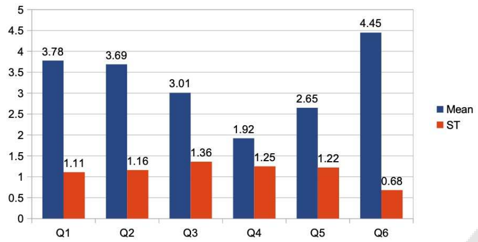

Figure 9: Results of the 6 pre-course questions about computers and programming, with answers mean and standard deviation.

enjoys working with computers (Q3) and a very small number is confident in dealing with programming languages (Q4). Moreover computer programming is not considered to be a very funny activity (Q5), while it is for sure very difficult for the majority of the students (Q6). These results depict a not very promising starting point for the course activities, also if Q1 and Q2 seem to show some room for improvement. A great result should be a higher mean in Q3, 4 and 5, and a decrease in Q6, showing that actually some students have earned a major level of computer literacy and have become more confident in the use of computers.

4.4.2 Enhancement of Musical Creativity

In the pre-course questionnaire there are 6 questions addressing the topic of musical creativity, particularly

Table 3: The 6 pre-course questions about musical creativity and composition.

Q1	I want to learn more about the integration of computers and music
Q2	I want to learn more about music and composing
Q3	I want to learn to be more creative
Q4	I enjoy creating my own music
Q5	I am confident in my ability to express myself through music
Q6	Computers can be used to create cool music

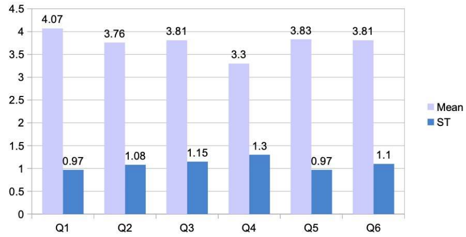

Figure 10: Results of the 6 pre-course questions about musical creativity, with answers mean and standard deviation.

music composition. The questions are outlined in Table 3 and results reported in Figure 10. Here the starting situation seems to be more promising, showing that students feel much more comfortable with creativity than with computer programming. They seem to be particularly curious about the relationship between music and computers (Q1) and show a positive attitude towards music created with computers (Q5), also if a lower number enjoys creating music (Q4). The link between computational thinking and creativity is one of the main objectives of the course (see Section 3.1). Hence an increase of the mean of Q4 and Q6 could indicate a positive evaluation of the computer–mediated creative activities proposed during the course.

5 CONCLUSION AND FURTHER WORK

In this paper a music technology course program has been presented. The contents of the course have been discussed in the light of the enhancement of the ”Four C’s” abilities (critical thinking, creativity, collaboration and communication). A road from a basic, stiff and not reusable algorithm (Figure 2) to a more general and flexible production module (Figure 4) has been shown, and the possibilities of creative use of the same algorithm have been suggested. The aim of this work is to demonstrate the connection between computational thinking and creativity as one of the most powerful expressions of contemporary art. Finally, some ideas for the assessment of computational thinking abilities in pre and post-course questionnaires have been developed and presented. At the conclusion of the course many important data will be available for analysis:

- quantitative data from pre and post-course questionnaires
- the final course assignments where students will produce and comment an original music creation algorithm

- the lesson-by-lesson assignments, which will show the road that each student has followed to reach the course objectives.

Also if some results have been speculated in Section 4.4, only a global analysis of all these data will provide significant elements for the evaluation of the course design, if it is successful in reaching the important goals listed in Section 3.1 and, mainly, if it is useful in contributing to the progress of the students.

ACKNOWLEDGEMENTS

The present research has been funded by Fondazione ASM[7], Brescia (Italy).

REFERENCES

Aaron, S., Blackwell, A. F., and Burnard, P. (2016). The development of sonic pi and its use in educational partnerships: Co-creating pedagogies for learning computer programming. *Journal of Music, Technology & Education*, 9(1):75–94.

Aho, A. V. (2012). Computation and computational thinking. *The computer journal*, 55(7):832–835.

Avanzini, F., Barate, A., Ludovico, L. A., and Marcella, M. (2021). Songs in music education: Design and early experimentation of a web tool for the recognition of harmonic changes. In *International Conference on Computer Supported Education*, pages 709–720. SCITEPRESS.

Baratè, A., Ludovico, L. A., and Malchiodi, D. (2017). Fostering computational thinking in primary school through a lego®-based music notation. *Procedia computer science*, 112:1334–1344.

Bell, J. and Bell, T. (2018). Integrating computational thinking with a music education context. *Informatics in Education*, 17(2):151–166.

Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A., Torelli, M., and Zecca, L. (2014). Extracurricular activities for improving the perception of informatics in secondary schools. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, pages 161–172. Springer.

Collins, N., Collins, N., Schedel, M., and Wilson, S. (2013). *Electronic music*. Cambridge University Press.

Copeland, B. J. and Long, J. (2017). Turing and the history of computer music. In *Philosophical Explorations of the Legacy of Alan Turing*, pages 189–218. Springer.

Cuny, J., Snyder, L., and Jeannette, M. (2010). Wing. *Demystifying Computational Thinking for Non-Computer Scientists, work in progress*.

Engelman, S., Magerko, B., McKlin, T., Miller, M., Edwards, D., and Freeman, J. (2017). Creativity in authentic steam education with earsketch. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 183–188.

Gorson, J., Patel, N., Beheshti, E., Magerko, B., and Horn, M. (2017). Tunepad: Computational thinking through sound composition. In *Proceedings of the 2017 Conference on Interaction Design and Children*, pages 484–489.

Greher, G. R. and Heines, J. M. (2014). *Computational thinking in sound: Teaching the art and science of music and technology*. Oxford University Press.

Hamilton, E. R., Rosenberg, J. M., and Akcaoglu, M. (2016). The substitution augmentation modification redefinition (samr) model: A critical review and suggestions for its use. *TechTrends*, 60(5):433–441.

Hancock, O. (2014). Play-based, constructionist learning of pure data: A case study. *Journal of Music, Technology & Education*, 7(1):93–112.

Israel-Fishelson, R. and Hershkovitz, A. (2022). Studying interrelations of computational thinking and creativity: A scoping review (2011–2020). *Computers & Education*, 176:104353.

Jenkins, H. (2009). *Confronting the challenges of participatory culture: Media education for the 21st century*. The MIT Press.

Krathwohl, D. R. (2002). A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, 10(4):1–15.

Manzo, V. J. (2016). *Max/MSP/Jitter for music: A practical guide to developing interactive music systems for education and more*. Oxford University Press.

Resnick, M., Ocko, S., and Papert, S. (1988). Lego, logo, and design. *Children's Environments Quarterly*, pages 14–18.

Selby, C. C. and Woollard, J. (2014). Refining an understanding of computational thinking.

Shute, V. J., Sun, C., and Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22:142–158.

Trower, J. and Gray, J. (2015). Blockly language creation and applications: Visual programming for media computation and bluetooth robotics control. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 5–5.

Wing, J. (2011). Research notebook: Computational thinking—what and why. *The link magazine*, 6:20–23.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881):3717–3725.

[7]https://fondasm.it/