Digit Exp Math Educ (2017) 3:97–114
DOI 10.1007/s40751-016-0026-4

MATHEMATICS AND PROGRAMMING

Assessment in Undergraduate Programming-Based Mathematics Courses

Chantal Buteau[1] · Eric Muller[1]

Published online: 28 November 2016

Abstract Since 2001, undergraduate mathematics majors and future mathematics teachers at Brock University (Canada) learn through a sequence of three courses to use programming to conduct mathematical inquiries or investigate real-world applications. In this paper, we provide a rich description of the assessment process of mathematics implemented in these courses by examining the assessment of two specific programming-based tasks: 1) a first-year assigned project about a discrete dynamical system; and 2) a second-year open-ended project for which students decide their own topics. We also examine the assessment of the overall course sequence, including the two tasks, from the perspective of computational thinking by use of Brennan and Resnick's framework extended for mathematical inquiry. By mainly involving programming-based mathematics projects of increased complexity, the overall assessment of these courses provides a concrete classroom implementation that fits Brennan and Resnick's design scenarios approach to the assessment (in a research setting) of computational thinking.

Keywords Assessment · University mathematics · Digital technology · Computational thinking · Third pillar of scientific inquiry · Constructionism · Programming-based projects · Classroom implementation

Introduction

There is relatively scant literature on assessment of mathematics with mathematically-enabling technology in contrast to the much broader discussion of such technology use

✉ Chantal Buteau
cbuteau@brocku.ca

Eric Muller
emuller@brocku.ca

[1] Brock University, 1812 Sir Isaac Brock Way, St. Catharines L2S3A1 Ontario, Canada

in mathematics education. The post-secondary mathematics classroom reality seems to highlight the difficulty of integrating technology into assessment. For example, Buteau et al. (2014) found in a nation-wide survey ($N = 302$) that among the large proportion (67 %) of Canadian mathematician participants who integrated computer algebra system (CAS) in their teaching of mathematics in university, only 22 % of them permitted their students to use CAS in their final exam. A similar practice was also observed in college mathematics courses in Quebec (Caron and Ben-El-Mechaiekh 2010). In addition, since undergraduate mathematics students seem to directly focus their learning towards the course assessment (Grønbæk et al. 2009) more attention needs to be focused on assessment in university mathematics courses that integrate technology.

In the Third International Handbook of Mathematics Education (Clements et al. 2013), Stacey and Wiliam (2013) review and discuss the use of technology in the assessment of mathematics. Grounding their discussion, they state that: “Using technology calls for new emphases in the learning of mathematics and the goals of the curriculum which, in turn, require different kinds of assessment to probe students’ anticipated new skills and capabilities” (p.721). Their position on the impact of the use of technology on mathematics curriculum aligns with Hoyles and Noss (2008):

> Like Kaput, we noted that the incorporation of technologies into mathematical learning almost inevitably brings to the fore a range of key questions – particularly those concerned with transformation of the what of mathematics rather than merely the how – precisely because digital technologies disrupt many taken-for-granted aspects of what it means to think, explain and prove mathematically and to express relationships in different ways. (p. 87)

By impacting the mathematics curriculum, assessment should be adapted to reflect this change, in particular as Stacey and Wiliam (2013) put it, by “mak[ing] the important measurable rather than making the measurable important” (p.739).

In this paper, we briefly examine the implemented assessment in a sequence of three programming-based undergraduate mathematics courses sustained since 2001 at Brock University. In these courses called *Mathematics Integrated with Computers and Applications (MICA) I-II-III*[1] (Ralph 2001; Ben-El-Mechaiekh et al. 2007), students learn to design, program, and use interactive computer environments, that we have called *exploratory objects* (EOs), in order to systematically investigate mathematics concepts, theorems, conjectures, or real-world situations (Muller et al. 2009). The bulk (~75 %) of the assessment in these courses concerns a total of fourteen programming-based mathematics EO projects. We argue that the high proportion of marks on these EO projects aligns well with the main learning objective of this course sequence. In other words, it aligns with ‘the mathematics that is most important for students to learn’ as decided by our mathematics department when the MICA courses were designed and adopted. Furthermore, the high percentage of the final mark allocated to mathematics projects also underlines the fact that assessment in MICA courses is carried out in a

[1] The MICA I-II-III courses were recently re-numbered and modified. In this paper, we refer to the courses as described in the official 2015–16 Brock calendar (http://www.brocku.ca/webcal/2015/undergrad/math.html), namely MATH 1P40, 2P40, 3P40.

different manner than what is usually found in university mathematics courses. We recently argued (Buteau et al. 2016) that this type of mathematics falls in 'the third pillar of scientific inquiry of complex systems', as identified by the European Mathematical Society, namely: "[t]ogether with theory and experimentation, a third pillar of scientific inquiry of complex systems has emerged in the form of a combination of modeling, simulation, optimization and visualization." (2011, p.2) We also have suggested that one cannot fully engage in this mathematics without programmable technology, and as such, this highlights the impact of technology on the mathematics curriculum of these MICA courses.

The purpose of our study is i) to illustrate an assessment of mathematics in programming-based mathematics tasks, and ii) to investigate an assessment of this type of task in the broader context of computational thinking. In this paper, we examine the assessment in MICA courses that has remained basically unchanged for 15 years. Whereby neither of the authors has been involved in developing the assessment in MICA courses,[2] they both have been involved for many years with these courses in different roles: Muller was Chair of the department when the MICA courses were designed, adopted, and, thereafter, implemented; and Buteau has taught MICA I since 2005 (and used the assessment schemes passed on to her from a colleague), and ever since then has been carrying out, with Muller, reflective and research work about MICA courses. In the next section, we provide a rich description of the assessment process of mathematics used in MICA courses by examining two EO project tasks out of the fourteen, one assigned by the instructor and one on a topic selected by the students. We broaden the context in the following section by bringing in a computational thinking perspective and accordingly examine the overall assessment implemented in the MICA course sequence, including revisiting the assessment of the two EO projects. In order to guide our discussion, we extend Brennan and Resnick (2012)'s framework of computational thinking into the domain of mathematical inquiry. This section also includes comparing one of these authors' three proposed research approaches to assessment to the one used in the MICA classroom. We conclude in the last section by some final remarks. Since these courses were designed and adopted outside an educational research context or purpose, and yet demonstrate satisfying characteristics of a constructionist approach (Buteau et al. 2015a), this paper contributes to the discussion of assessment in a technology-rich (constructionist) mathematics classroom situation. The paper also contributes an extended version of Brennan and Resnick (2012)'s framework of computational thinking into the domain of mathematical inquiry, and an example of assessment of computational thinking in an actual mathematics classroom implementation.

Assessing mathematics in programming-based mathematics project tasks: two examples in MICA courses

In university mathematics courses, the correctness of the mathematics in a student's mathematics production (e.g. exams, assignments) is assumed to correspond to a

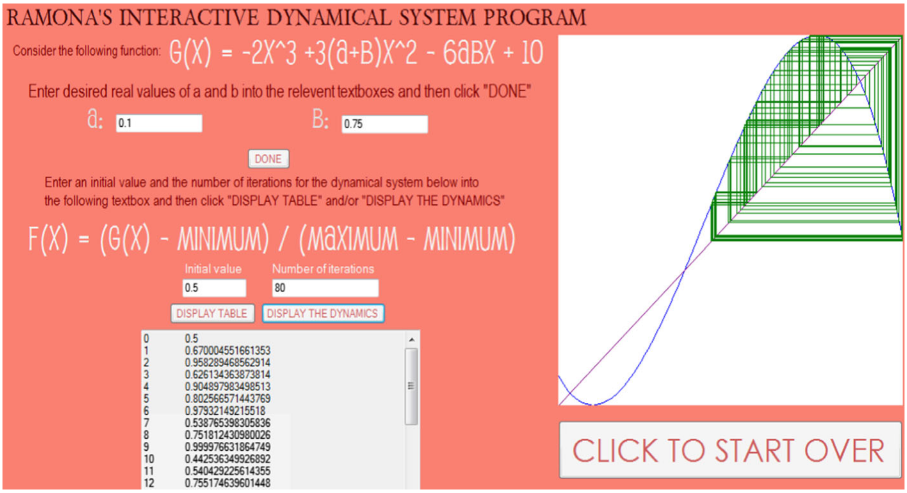
[2] Except for one minor element added by Buteau in 2012 as briefly mentioned in Assessing computational thinking for mathematical inquiry in MICA courses.

measure of the student's understanding of mathematics. In this paper, we illustrate the assessment of mathematics in a student's mathematics EO project by identifying parts of the grading scheme used by the MICA instructor that aim at specifically assessing the student's mathematics. We illustrate some of our arguments by excerpts of two selected students' EO projects collected from former studies and which are both of good quality: i) Ramona's assigned EO project was part of a case study of a student's mathematics learning experience through her 14 MICA EO projects (Buteau et al. 2016); and ii) Adam's original EO project was among a few EO projects initially selected from the hundreds of student projects due to their diverse topics and good quality, then made available online (MICA URL n.d.), and which we later studied (Buteau and Muller 2014).

Each MICA course includes 2 h of lecture and 2 h of computer laboratory session weekly. The lectures mostly provide the mathematical content as background and motivation for the programming-based mathematical tasks initiated, and sometimes completed, during the laboratory sessions. For example, the mathematics content may involve topics such as basic number theory and application to RSA encryption method, discrete and continuous dynamical systems, traffic light synchronization, prey–predator biological models, statistical applications to stock market, stochastic models of bacterial growth, cellular automata, etc. —see (Buteau et al. 2015b) for an overview of the mathematics content summarized, in a table, through the 14 EO projects. All of our students use the same digital technology in their EO projects: in the MICA I-II courses, students have been using Visual Studio (with vb.net programming language), and in the MICA III course, students have either continued using vb.net or have also been required to use Maple, a CAS used in their calculus and linear algebra courses. In this section, we illustrate the assessment of mathematics in two of the EO project tasks.

Assessing students' understanding of mathematics in an assigned EO project

The third EO project corresponds to the last assigned project in the MICA I course prior to the final original project (described in Assessing students' understanding of mathematics in original EO projects). During lectures, students learn about dynamical systems and cobweb diagrams with an emphasis on the logistic function. During laboratory sessions, students are guided to create an EO about the dynamical system based on the logistic function (involving one parameter) and explore its behaviour. When a student creates an EO in Visual Studio, s/he designs a graphical user interface by drag-and-drop manipulations and programs in vb.net language all of the functionalities, including the mathematics. As their third assigned EO project, students are asked to individually modify, extend, and use their code to explore the dynamical system based on a cubic function involving two parameters, and submit their EO and their written report: see Appendix 1 for the student guidelines, including a grading scheme summary. The hand-out given to students of the grading scheme together with the task guidelines aligns well with what Houston (2002) recommends for the assessment of individual undergraduate mathematics work (and particularly individual mathematics projects): "There should be transparent assessment criteria, which should be explained to the students, if possible with examples." (p. 413) Similar guidelines with grading schemes are provided to students for all the other ten assigned EO projects. A screenshot of one student's, Ramona, approach to the third EO assignment is shown in Fig. 1.



RAMONA'S INTERACTIVE DYNAMICAL SYSTEM PROGRAM

Consider the following function: G(X) = -2X^3 +3(a+B)X^2 - 6aBX + 10

Enter desired real values of a and b into the relevent textboxes and then click "DONE"

a: 0.1 B: 0.75

Enter an initial value and the number of iterations for the dynamical system below into the following textbox and then click "DISPLAY TABLE" and/or "DISPLAY THE DYNAMICS"

F(X) = (G(X) - MINIMUM) / (MAXIMUM - MINIMUM)

Initial value: 0.5

Number of iterations: 80

```
0     0.5
1     0.670004551661353
2     0.958289468562914
3     0.626134363873814
4     0.904897983498513
5     0.802566571443769
6     0.97932149215518
7     0.538765398305836
8     0.751812430980026
9     0.999976631864749
10    0.442536349926892
11    0.540429225614355
12    0.755174639601448
```

Fig. 1 Screenshot of MICA I student Ramona's third assigned EO project for the exploration of the dynamical system based on a cubic (Buteau et al. 2016)

The assessment of mathematics in such EO projects entails the examination of both the computer environment and the written report. First, the students' understanding of the mathematics involved in the EO project is measured through their coding of the concepts. Through programming, a student has to clearly articulate processes and relationships among mathematics concepts in order to complete the first part of the task, namely to graphically and numerically represent the dynamical system. Noss and Hoyles (1996) stress that, "it is in this process of articulation that a learner can create mathematics and simultaneously reveal this act of creation to an observer" (p. 54). For example, Ramona needed to program the iterative process to generate the sequences of the specified system and draw the cobweb (Part I. 4 in guidelines). Another example concerns the mathematical subtlety involved in Part I. 2 of the EO guidelines: in this case, Ramona correctly used the closed interval method to find the global extrema by noticing the dependence on the values of the parameters and consequently by separating the different cases within the code, which resulted in displaying the graph of the f function as requested (Part I. 3). The assessment of students' mathematical understanding is complemented through their elaboration of the related mathematics concepts as part of their written reports (Part II. 2). For example, Ramona describes the generation of a sequence through the iterative process (Part II. 2.e), whereby she set a = 2 and b = 1:

The sequence is built [in] the following way:

$$
\begin{aligned}
x_0 &= 0.1\\
x_1 &= f(x_0)\\
&= \mathrm{f}(0.1)\\
&= (g(x_0)\text{–min})/(\text{max–min})\\
&= (g(0.1) – 5/(10–5)\\
&= \left[(\text{-}2{*}0.1^3 + 9{*}0.1^2\text{-}12{*}0.1 + 10)\text{-}5)/5\right]\\
&= 0.7776\\
x_2 &= f(x_1) = 0.0340771741696001\\
&etc.
\end{aligned}
$$

The students' understanding of mathematics is also assessed through their mathematical inquiry as reported in the written report (Part II. 1,3,4). In this part of the EO project, students like Ramona must understand not only how fixed points (Part II. 3a) and orbits (Part II. 3b) are defined, but also what to look for in the graphical and numerical representations of the system in order to identify them. As well, they must provide 'evidence', collected from their EO or through an algebraic argument, about their mathematical work. For example, Ramona included her findings about parameter values for which the system has at least three fixed points (Part II. 3a):

> Using my program, I found values of a and b so that f(x) has three fixed points…. Let a = 0.2, b = 0.8, and $x_0 = 0.2$. The dynamics, in this case, is as follows: [screen shot]… One can clearly see that f(x) intersects the diagonal line y = x three times. In identifying the fixed points, I followed the same approach I used in g). I used MAPLE for all computations… The three fixed points are x = 0.09750776490, x = 0.4999999983 and x = 0.9024922368, as found in MAPLE.

This is mathematics for which (first-year undergraduate) students need technology to engage with, and we expect that the way they use technology in the task supports their learning of the underlying mathematical concepts. For example, Ramona voluntarily writes in the conclusion of her report: "Both creating and working with this program has assisted me to fully grasp the way a dynamical system works by observing the table, the graphs, and the cobweb with countless test values for a, b, and x_0." The students' EO-related mathematics understanding is also tested later in the course through traditional in-class mathematics tests.

Assessing students' understanding of mathematics in original EO projects

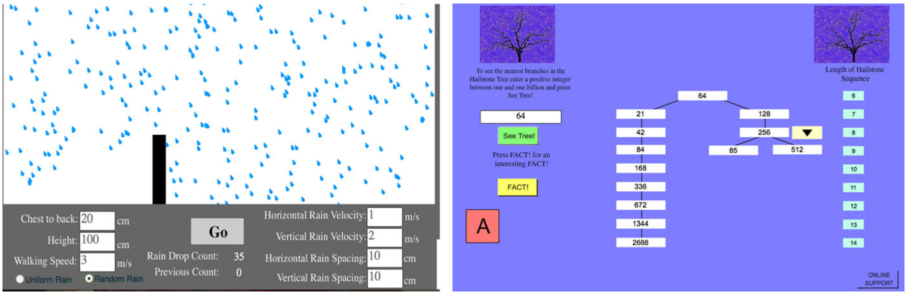
Each of the MICA courses culminates with an original end-of-term EO project (fourth, ninth, and fourteenth EO of the sequence of 14) — see Appendix 2 for an example of the MICA I original project guidelines and grading scheme. For these EO projects, students are encouraged to select a topic of interest to them. They are done in pairs or individually and in the last 2 weeks of the course. Future teachers may decide to create an EO for the step-wise guided learning of a school mathematics concept (Muller et al. 2009). Examples of student original EO projects can be found on MICA URL (n.d.); for example, MICA II students Matthew and Kylie wondered if it is better to walk or run in the rain (Fig. 2a), while MICA I student Colin investigated the structure of hailstone sequences (Fig. 2b).

Similarly to the previous task, the assessment of students' mathematics understanding in original EO projects entails the examination of both the computer environment and the written report. And also similarly to the previous task, the assessment of mathematics is measured through their coding of the mathematics involved (point 2 in the grading scheme), as well as through their mathematical inquiry as reported in the written report (point 4 in the grading scheme). Furthermore, specific to the original EO projects, the assessment of student's mathematics understanding involves the mathematical topic selected by the student(s) and his/her (their) choice of the way(s) to investigate it. This is reflected in points 1) and 3) of the grading scheme, respectively. To illustrate this, we consider MICA II student Adam's pure mathematics EO project about the Mandelbrot set and the bounded area of the iterative complex function defining the Mandelbrot set as the exponent increases. Adam's mathematical understanding was assessed through his choice to investigate the Mandelbrot set in a general form, and his choice of representations to support his investigation: i) the usual graphical representation with zoom-in features (see Fig. 3a); ii) similar graphical representation of related Julia sets (see Fig. 3b); and iii) the graphic representation of the bounded area approximations (see Fig. 3c). In the two first representations, the user (Adam) can choose a point for zooming-in (and visually see the symmetries) by either typing in coordinates or mouse clicking on the graphical representation, which also reflects his understanding of the mathematics involved.

Each original EO project contains mathematics specific to the topic selected by the student(s), and therefore presents a difficult task for the instructor to assess all EO projects in a consistent manner. Indeed, Houston (2002) asserts that it is a challenge 'to assess reliably and validly' such individual mathematics projects. But he further indicates that,

> Experienced project assessors can usually come to an accurate judgement of a student's work fairly quickly and can defend that judgement with their peers. But there still is an element of subjectivity in this and, to remove as much of this as possible and to achieve consistent marking by several assessors, consultation and training are necessary. (p.413)

In MICA courses, the assessment of the original EO projects is normally conducted by the sole instructor of the course. Since MICA courses have been offered every year since 2001, and generally are taught by the same instructors, the latter have become 'experienced EO project assessors'. In some cases, teaching assistants assist the instructor by examining the technical aspects of these original projects. On the other hand, since the assigned EOs, such as the EO project described in Assessing students' understanding of mathematics in an assigned EO project, focus on a specific mathematics, they are usually assessed by the teaching assistants (who have completed all MICA courses with distinction) using grading guidelines from the instructor.



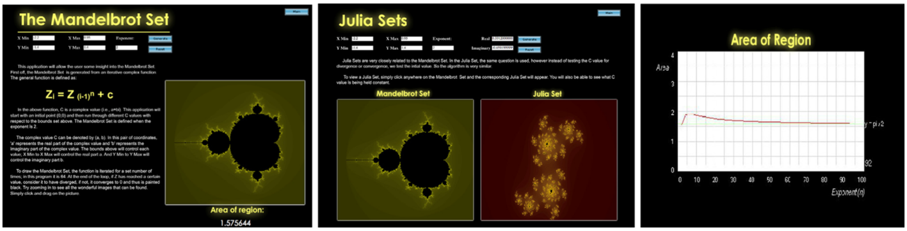
Chest to back: 20 cm
Height: 100 cm
Walking Speed: 3 m/s
○ Uniform Rain ● Random Rain
Rain Drop Count: 35
Previous Count: 0
Horizontal Rain Velocity: 1 m/s
Vertical Rain Velocity: 2 m/s
Horizontal Rain Spacing: 10 cm
Vertical Rain Spacing: 10 cm

To see the nearest branches in the Hailstone Tree enter a positive integer between one and one billion and press See Tree!

64

Press FACT! for an interesting FACT!

Fig. 2 **a**. MICA II students Matthew and Kylie's original EO project of a real-world application: "*Is it better to walk or run in the rain?*"; **b**. MICA I student Colin's original pure mathematics EO project about the investigation of the structure of hailstone sequences. See MICA URL (n.d.) to run these EOs or for other examples of students' original EO projects



This application will allow the user some insight into the Mandelbrot Set. First off, the Mandelbrot Set is generated from an iterative complex function. The general function is defined as:

$$Z_i = Z_{(i-1)}^n + c$$

In the above function, C is a complex value (i.e., a+bi). This application will start with an initial point (0,0) and then run through different C values with respect to the bounds set above. The Mandelbrot Set is defined when the exponent is 2.

The complex value C can be denoted by (a, b). In this pair of coordinates, 'a' represents the real part of the complex value and 'b' represents the imaginary part of the complex value. The bounds above will control each value; X Min to X Max will control the real part a. And Y Min to Y Max will control the imaginary part b.

To draw the Mandelbrot Set, the function is iterated for a set number of times; in this program it is 64. At the end of the loop, if Z has reached a certain value, consider it to have diverged, if not, it converges to 0 and thus is painted black. Try zooming in to see all the wonderful images that can be found. Simply click and drag on the picture.

Julia Sets are very closely related to the Mandelbrot Set. In the Julia Set, the same question is used, however instead of testing the C value for divergence or convergence, we test the intial value. So the algorithm is very similar.

To view a Julia Set, simply click anywhere on the Mandelbrot Set and the corresponding Julia Set will appear. You will also be able to see what C value is being held constant.

Fig. 3 Different representations in the MICA II student Adam's pure mathematics EO project about the Mandelbrot set in a general form (the exponent of the iterative complex function defining the set is a parameter) – Buteau and Muller (2014): **a**. the usual graphical representation with zoom-in features (by typing coordinates or by mouse clicking); **b**. the graphical representation of related Julia sets with the same zoom-in features; and **c**. the graphic representation of the bounded area approximations as the exponent increases

This section has illustrated how, in MICA courses, undergraduate students' understanding of mathematics is assessed through their assigned and original EO projects. However, these projects contain aspects, such as the EO interface design and programming sophistication, which are more programming related and not fully encompassed within the described assessment of mathematics of the tasks. There are also aspects that are specific to programming, such as the incorporation in the code of comments and modules (i.e., blocks of code lines) for future remixing purposes and clarity of the overall code structure. Although the latter may involve mathematical thinking (structure is, after all, mathematics), this may be associated to the mathematics of programming in contrast to the programming of mathematics. In order to encompass mathematics and programming-related aspects assessed in a student's EO project, we propose in the next section to examine the assessment implemented in MICA courses from the broader context of computational thinking, including revisiting the assessment of the two EO project tasks we considered in this section.

Assessing computational thinking for mathematical inquiry in MICA courses

Computational thinking can be described as "taking an approach to solving problems, designing systems and understanding human behaviour by drawing on concepts fundamental to computer science" (Wing 2006, p.33). Positioning programming in computational thinking, Grover and Pea (2013) argue that it "is not only a fundamental skill of [computer science] and a key tool for supporting the cognitive tasks involved in [computational thinking] but a demonstration of computational competencies as well" (p. 40). To guide our discussion about assessing MICA students' computational thinking, we consider Brennan and Resnick's (2012) framework that emerged from their studies of the design-based learning activities of programming interactive media in Scratch (n.d.) by young people. The authors describe Scratch as a "programming environment that enables young people to create their own interactive stories, games, and simulations, and then share those creations in an online community with other young programmers from around the world" (p.1). Their proposed framework of computational thinking involves three dimensions:

> *computational concepts* (the concepts designers engage with as they program, such as iteration, parallelism, etc.), *computational practices* (the practices designers develop as they engage with the concepts, such as debugging projects or remixing others' work), and *computational perspectives* (the perspectives designers form about the world around them and about themselves). (p.1)

As this framework is not discipline specific, recently Weintrop et al. (2016) proposed a taxonomy of computational thinking in the mathematics and science classroom that include four main categories: i) data practices; ii) modelling and simulation practices; iii) computational problem solving practices; and iv) systems thinking practices. For example, modelling abstract mathematical concepts into concrete code, designing and coding a mathematical simulation, and engaging systematically in a computer-assisted mathematical inquiry are considered as *computational practices* in a mathematics classroom. Furthermore, Weintrop et al. (2016) envision their proposed taxonomy "being used to bring current educational efforts in line with the increasingly computational nature of modern science and mathematics." (p. 127) Thereby the perspectives the designers form about mathematics as a discipline may be interpreted as part of the *computational perspectives* dimension in Brennan and Resnick's (2012) framework when focused on a mathematics context. In the following we examine the implemented assessment in MICA courses according to this three-dimensional framework for mathematical inquiry.

Furthermore, Brennan and Resnick (2012) propose three (research) approaches to assessing the development of computational thinking: the *project analysis* approach involving an analysis of the coding found in a project or collection of projects; the *artefact-based interviews* approach involving a posteriori interview(s) with a learner about his/her coding of selected projects; and the *design scenarios* approach involving real-time observation of a learner needing to 'fix' or extend in certain ways a collection of projects, of increasing complexity, created from a virtual Scratcher (i.e. the scenarios). We propose that the overall assessment in MICA courses aligns mainly with a design scenarios approach, and discuss it further in relation to Brennan and Resnick's (2012) insights.

Assessing computational thinking for mathematical inquiry in MICA courses

The MICA I course is designed for students to learn the basics of programming, while they are introduced to designing, programming, and using interactive computer environments (i.e., EOs) for the systematic investigation of mathematics concepts, conjectures, and applications. The second-year MICA II and third-year MICA III courses aim at students learning to create sophisticated EOs for the exploration of more advanced mathematics and applications. Central to the course sequence design are the 11 assigned EOs that we have argued can be viewed as stepping stones to guide the student's learning of programming as a tool within a context of increasingly complex mathematical ideas identified in the third pillar of scientific inquiry (Buteau et al. 2016). These 11 assigned EOs are mathematics scenarios given to students in the form of

Table 1 Overall mark distribution according to the assessment activities in the MICA I-II-III courses, and related dimensions of computational thinking for mathematical inquiry

Assessment type	MICA assessment activity	Assessment weight		Dimensions of computational thinking for mathematical inquiry
		MICA I	MICA II-III	
Formative	Coding quizzes (3)	9 %		concepts
	Assigned EOs	42 % (3)	45 % (4)	concepts and practices
Summative	Original EO (1)	28 %	35 %	concepts, practices, and perspectives
	Math tests (2)	21 %	20 %	related to practices and perspectives

guidelines. In the following we briefly discuss the assessment implemented in MICA courses of students' computational thinking for mathematical inquiry– see Table 1 for an overview of the quantitative assessment.

The assessment of students' development of computational concepts involves two types of MICA assessment activities: i) the coding quizzes[3] in the first MICA course strictly aiming at assessing students' understanding of basic programming concepts (variables, loops, arrays, graphics, modules, etc.) as needed for programming-based mathematical inquiries; and ii) all of the assigned and original EOs in which students' fluency of the different programming concepts is assessed indirectly through their coding. Since most students enrol in the MICA course sequence with no prior programming background, the introductory MICA I course contains a significant learning component of programming basics put in a mathematical context. For example, MICA I students learn in the third laboratory session to use loops and decision structures by creating an EO to check whether an integer is prime or not. These (non-graded) programming-based mathematical tasks in MICA I lab sessions, together with the EO assignments, have been carefully designed to support students' learning of programming for mathematics (Buteau and Muller 2014).

In the MICA courses, the assessment of students' development of computational practices for mathematical inquiry accounts for 70 to 80 % of the total grade through the students' EO projects. In the two examples provided in Assessing mathematics in programming-based mathematics project tasks: two examples in MICA courses, this dimension is assessed throughout, namely adding up to 100 % of their respective grading scheme. See Table 2 for a break down according to selected practices, which we elaborate in the following. First, the engagement in a mathematical inquiry accounts in the third EO project for 45 % whereas in the original EO it accounts for 65 %. In both cases, it includes a written report summarizing the mathematical inquiry and results. This mathematical inquiry is facilitated by the interactivity (for parameter change of values) and dynamic visualization (e.g. graphs) affordances of the student's EO interface (Buteau and Muller 2010). Another practice, only found in the original EO, is the design of a mathematical simulation (40 % in original EO grading scheme) whereby

[3] Unlike the other assessment components, the coding quizzes are a recent addition in MICA courses. They were added by Buteau in response to a seemingly newer trend of students completing the MICA I course without sufficient proficiency with programming basics.

Table 2 Parts and weights of the assessment according to selected computational practices in computational thinking for mathematical inquiry of the i) third assigned EO project, and ii) original EO project

Computational practices	Third EO project		Original EO project	
	Assessment scheme parts	Weight of assessment	Assessment scheme parts	Weight of assessment
Engagement in mathematical inquiry	Part 0 and Part II	45 %	points 1), 3) and part of 4)	65 %
Designing a simulation	N/A	N/A	point 3)	40 %
Modeling abstract mathematical concepts into concrete code	Part I. 3),4)	45 %	points 2), 3)	60 %
Others (debugging, remixing, etc.)	Part I	55 %	points 2), 3) and part of 4)	75 %

> the potentiality of interactivity encourages the student to make explicit the parameters that could play a role in the investigation of his/her conjecture or real-world situation in such a way that they are accessible from the interface. The potentiality of visualization urges the student to decide on the representations to be displayed in his/her interface so as to best support his/her investigation. (Buteau and Muller 2010, p.1118)

This simulation design practice cannot be fully separated from the engagement in a mathematical inquiry practice since the student's design of a simulation often evolves as s/he gets more insight into the simulated situation by using his/her EO. Indeed, when a student uses his/her EO interface to engage in a mathematical inquiry (15 % in original EO grading scheme),

> [Interactivity] can be seen as a dialogue between the student and the computer, though the discussion is fully controlled by the student… the student sets a question by fixing values to parameters (interactivity), the computer answers the question (visualization), and the dialogue continues in that way unless the student concludes that the answers are not satisfactory to meet his/her goal and decides to refine the [EO] (Buteau and Muller 2010, p.1118);

which could involve refining the initial problem (10 % in original EO grading scheme). This practice exemplifies how the mathematical and computational aspects of an EO project can be fully intertwined. Another computational practice that is assessed in students' EO projects is modeling abstract mathematical concepts into concrete code. Through students' coding of mathematics concepts, this accounts for 45 % in the third EO project whereas in the original EO it accounts for 60 %. Finally considering other computational practices of more general type such as debugging or remixing, the grading scheme accounts for 55 % in the third EO and for 60 % in the original EO which is related to the coding of mathematics (that, for example, unavoidably involves debugging at some point) as well as using 'good programming style' such as the incorporation of comments and modules in the code. Specific to the original EO, is the last component of the grading scheme point 4 (worth 15 %), namely, the list of five 'awesome features' in the student's EO project. We suggest that this encourages students' self-assessment, and that it also contributes to the other computational practices for a total of 75 %.

Whereas the computational perspectives dimension does not arise in the assessment of assigned EO projects it is assessed in the original EO where we view it as part of the student's selection and statement of their project (Part I of grading scheme). For example, Adam's original project topic (see Fig. 3), a generalized form of Mandelbrot Set, reflects in part his perspective of a kind of mathematics that can be explored and more deeply understood by use of an interactive computational environment.

In MICA courses there is one more assessment activity that we have not mentioned, which is the summative paper and pencil mathematics tests that aim at assessing the students' understanding of the assigned EO-related mathematics. As highlighted in Assessing mathematics in programming-based mathematics project tasks: two examples in MICA courses, it is mathematics that grounds the inquiry (i.e. the programmed mathematics) as well as mathematics learned from the inquiry (i.e., mathematics made accessible to students through the use of their EOs) which is related to a student's computational practices and perspectives.

Discussion on the grounding design scenarios approach in MICA to assess computational thinking

According to Brennan and Resnick (2012), the *design scenarios* approach in a research context provides three major strengths, two of which we identify and interpret in the implemented assessment in the MICA courses: 1) each assigned EO project provides an opportunity to involve specific computational concepts, mathematical concepts, and their applications, and the assessment of an EO somewhat measures the students' fluency of these computational concepts and practices; 2) the sequence of the 11 assigned EOs over the three MICA courses highlights a developmental and formative approach. The third strength mentioned by the authors is that the approach "emphasize[s] process-in-action, rather than process-via-memory" as one observes learners during the whole scenario activity. In MICA courses, students complete their assigned EOs mostly outside classroom time, and as such, this third strength does not apply to the implemented formal assessment in MICA courses. However, since course sections are capped at 35 students, their process-in-action is observed by the instructor during weekly laboratory sessions thereby providing opportunities for real-time informal (i.e., not graded) assessment of their computational thinking development.

The authors also point out limitations of the design scenarios approach which are not reflected in the MICA courses partly due to our experience over 15 years of implementation: 1) the adequate assistance given to students to complete a project; and 2) the issue of amount of time allocation for a scenario activity (which, we stress again, that in MICA courses, EO projects are completed mostly outside classroom time). The third issue raised is the issue of "externally-selected projects [that] may not connect to personal interests and the learner's sense of intrinsic motivation" (p.21). This is addressed in MICA courses through part of the summative assessment, that is, the original EO projects for which students select their topic of interest.

The authors conclude by providing six suggestions for (research) assessing computational thinking via programming. We examine each of these suggestions in the context of MICA courses.

1. *Supporting further learning.* As discussed in Assessing computational thinking for mathematical inquiry in MICA courses, we argue that the formative assessment in MICA courses, mainly designed around the 11 assigned EO projects, supports a student's development of computational thinking for mathematical inquiry.
2. *Incorporating artefacts.* The assessment in MICA courses contains 70 to 80 % of artefact examination.
3. *Illuminating processes.* The authors suggest that the assessment should include a measurement of a student's development process when engaged in a programming activity. This is not reflected in the MICA course assessment and points to a constraint of the university mathematics classroom reality whereby continuous individual observation is not possible.
4. *Checking in at multiple waypoints.* The sequence of 11 assigned EO projects spreads over three terms provides checking at multiple waypoints.
5. *Valuing multiple ways of knowing.* The assessment in MICA contains different activities (see Table 1). For example, the computational concepts dimension is assessed through coding quizzes and EO projects.
6. *Including multiple viewpoints.* Here the authors include the viewpoint of self, peer, parent, online community, etc. As is the case in the third suggestion, this is not reflected in the MICA course assessment and points to another constraint of the university mathematics classroom reality.

Concluding Remarks

In a study on inquiry science programs, Liu et al. (2010) note that "Many science education researchers have implemented inquiry science teaching programs to improve the current situation of science learning and teaching by placing more emphasis on fostering students' deep scientific understanding and less emphasis on memorizing science facts" (p.70). They also observe that "a common challenge these new curricula and/or programs face is the selection of assessments used to measure student learning outcomes" (p.70). At the conclusion of their study, they report that their results "point to the need for multifaceted assessment in evaluating student learning of technology-enhanced inquiry science." (p.84) In the terminology of Liu et al. (2010), the three MICA courses make up an inquiry mathematics program. And, as we have detailed in this paper, the selection of both formative and summative assessments used to measure student learning is multifaceted through tests in mathematics and programming, prescribed and original programming-based projects and their written reports.

Houston (2002) in his contribution to assessment for the ICMI study on *The Teaching and Learning of Mathematics at University Level* notes that "[Students] need to know what performance standards are required and they need to be able to recognise within themselves whether they have achieved these standards or not. This ties in with self-assessment and ways of promoting self-assessment." (p.409) The assessment in MICA courses involves 14 EO projects through which, from our experience, students come to know what standards are required and come to develop a recognition, in the words of Houston (2002), "within themselves whether they have achieved these standards or not." (p. 409)

The implemented assessment in MICA courses also meet what Stacey and Wiliam (2013) have suggested as the three principles "for the assessment of mathematics that are relevant at the personal, class, and system level" (p.739): i) the *mathematics principle* requiring the assessment to concern the mathematics that is most important for students to learn is reflected in the bulk (~75 %) of the assessment in MICA courses involving the 14 EO projects aligned with the third pillar of scientific inquiry of complex systems (ESM 2011); ii) the *learning principle* requiring assessment to "enhance mathematics learning and support good instructional practice" (p.739) which is met through the careful sequencing of 11 EO project assignments increasing mathematical and programming complexity (i.e., a design scenario approach); and iii) the *equity principle* whereby all of our students use the same digital technology in their EO projects available on all computers on our campus and with a free home licence for the students. Stacey and Wiliam (2013) indicate that, the "three principles are statements of values, rather than the more familiar principles of educational measurement, they do, in effect, subsume traditional concerns such as validity" (p.739).

This paper provided a rich description of the assessment process used in constructionist programming-based university mathematics MICA courses. It also proposed an extended version of Brennan and Resnick's (2012) computational thinking framework into the domain of mathematical inquiry. One of the dimensions of Brennan & Resnick's framework is *computational perspectives*. In their applications of the three different approaches to assessment, this dimension is, however, not directly assessed which is also the case in MICA courses, except for the students' original projects. Indeed, our experience in mentoring students for their original EO projects, and assessing the latter, suggests that these may provide an insight into the students' perspectives they form about mathematics as a discipline which we have associated with the *computational perspectives* dimension of computational thinking for mathematical inquiry. The European Mathematical Society's (2011) statement positioning the third pillar of scientific inquiry of complex systems provides a view of the mathematics discipline that has been impacted by digital technology. We view the MICA course sequence as a suitably constrained implementation of this mathematics in undergraduate mathematics classroom (Buteau et al. 2016). In this paper we have argued, indirectly, that the implemented assessment in MICA may support students to develop fluency in this mathematics and thereby may come to more closely align their perspectives on mathematics to that of the European Mathematical Society.

Appendix 1. Third EO Assignment Guidelines and Assessment Scheme (Winter Term 2014)

Your goal is to write a program that allows a user to first input a particular cubic equation and then explore its dynamics on the interval [0,1].

Part 0: Participation in the lecture [about the interactive exploration of the dynamical system of the logistic function] with a *working* program**: (5 marks)**

Part I: **Your interactive dynamical system exploration program** on a CD (or USB key)

1) The user should be able to enter the parameters a and b for the function $g(x) = -2x^3 + 3(a + b)x^2 - 6abx + 8$
2) Your program should (internally) find the exact maximum M and minimum m of the function g(x) on the interval [0,1]. *(Hint: Use the closed interval method)*
3) Set f(x) = (g(x)-m)/(M-m). Note that the range of f(x) is exactly [0,1]. At the click of a button, we see the graphs of y = f(x) and y = x appear. (These graphs should just touch the bottom and top of your picture box.) **(20 marks)**
4) The user should be able to enter an initial value for the dynamical system determined by f and at the click of another button see a table of values appear and the dynamics (cobweb) drawn in the picture box as in parts 6) to 11) of lab#9. **(25 marks)**

   NOTE: Your program should have an attractive user-friendly interface and good programming style: it should use comments, functions and sub procedures, and should be efficient. **(10 marks)**

Part II: The exploration and hand-written (or typed) report. Your hard-copy report will consist of four parts under the following headings:

1. INTRODUCTION. Write a short paragraph introducing your project. If you use resources (internet, book, article, etc.), give the reference(s) — up to 8 lines **(2 marks)**
2. MATHEMATICS BACKGROUND USING AN EXAMPLE — up to 2 pages **(8 marks)**

   a. Use the two last digits, d_1 and d_2, of your student number and set the values $a = d_1/10$ and $b = d_2/10$; this defines a specific function g. Use it in the following.
   b. Using any technology (e.g. Maple), draw the graph of g with domain [0,1].
   c. Find the maximum and minimum of g, and define f as in step 3 (Part I).
   d. Using any technology (e.g. Maple), draw the graph of f, and write a sentence or two to explain its relation to the graph of g.
   e. Select an initial values x0, use your program to compute the first 10 terms of the sequence of the iterative function system based on f, and use the data to explain how the sequence is built. Identify the convergence or divergence of the sequence.
   f. Draw manually the corresponding cob-web (in the graph of f), and describe how the convergence or divergence of the sequence is visualized.
   g. Show how to find (algebraically) the fixed points of f (you may use Maple for computations), and plot them in the graph of f. Using your program, classify them (attracting, repelling or neither) and describe in your own words what each classification means.
3. DATA COLLECTION ABOUT INTERESTING CASES— up to half a page

   a. Use your program to find values of a and b so that f(x) has three fixed points. Use any method (including Maple) to prove that they are fixed points.

Classify each point as attracting or repelling or neither and give written evidence for your claims. **(20 marks)**

b. Find 3 different pairs of values of a and b and a starting value so that subsequent values oscillate closer and closer to a finite number (between 3 and 100) of values. Describe what happens. **(7 marks)**

4. DISCUSSION/CONCLUSION. Write a short paragraph concluding your exploratory work (e.g., discuss further about 3a) or 3b); about dynamical systems; about the use of cobwebs, etc.). If you use resources (internet, book, article, etc.), give the reference(s) — up to half a page **(3 marks)**

Appendix 2. Student guidelines and grading scheme for their MICA I final project

The project consists of an interactive computer environment (in VB.net) together with a written report. It can be one of three types, all of which being considered *equally important*:

a) an **INVESTIGATION** of a mathematical conjecture/problem/theorem
b) a **LEARNING OBJECT** designed to teach/test mathematical concepts at a specific grade level (Grades 4–12)
c) a "real world" **APPLICATION** of mathematics

The project should demonstrate sophisticated knowledge of vb.net tools.

The interface must be very friendly, self-explanatory and a pleasure to work with.

Written Report. Your hard-copy typed[4] report will have the following information under the indicated headings:

A) PROJECT TITLE & TYPE

Provide the name of your project and state whether your program is an Investigation, a Learning Object or an Application.

A-2) TARGET AUDIENCE (for Learning Objects only)

State your programs target audience.

B) PURPOSE AND BACKGROUND

State the overall mathematical or pedagogical purpose of your project and provide the necessary background.

C) SUMMARY OF OBSERVATIONS

In the case of an Investigation or an Application, give a coherent summary of what you observed. For a Learning Object, describe the experience of at least one student, at the appropriate grade level, who worked with your program.

D) DISCUSSION

E) CHECKLIST OF FEATURES

Provide a checklist of 5 special features of your program for which you should be given credit

[4] Mathematical formula may be added by hand to the write-up.

Grading Scheme

1) The mathematical problem or the application is interesting and has a good difficulty level (10 %)

    There is a clear statement of the investigation; it shows originality and depth

2) The project demonstrates sophisticated knowledge of vb.net tools (20 %)

    the project contains (some of the following): great visualization of data, non-trivial programming of mathematics concepts, sophisticated interface features.

3) The interface is very friendly, self-explanatory and a pleasure to work with. It is well designed in order to investigate the problem. (40 %)

    easy access to all parameters needed for the investigation, representation of outcomes meaningful to the investigation, the interface is interactive & attractive, communication of navigation is clear

4) Written document: (15 + 15 % special features)

    – PROJECT TITLE & TYPE
    – PURPOSE & BACKGROUND: clear statement of the mathematical investigation. Describe in details the mathematics
    – SUMMARY OF OBSERVATIONS: provide some data used for your investigation, and state your observations
    – DISCUSSION: discuss your observations and/or interpret your observations in the context of the real-world situation
    – 5 AWESOME FEATURES OF YOUR PROJECT (15 %): list 5 attributes of your project that makes it stunning

References

Ben-El-Mechaiekh, H., Buteau, C., & Ralph, W. (2007). MICA: a novel direction in undergraduate mathematics teaching. *Canadian Mathematics Society Notes, 39*(6), 9–11.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.*

Buteau, C., Jarvis, D. H., & Lavicza, Z. (2014). On the integration of Computer Algebra Systems (CAS) by Canadian mathematicians: Results of a national survey. *Canadian Journal of Science, Mathematics, and Technology Education, 14*(1), 35–57.

Buteau, C., & Muller, E. (2010). *Student development process of designing and implementing exploratory and learning objects* (pp. 1111–1120). Lyon: In Proceedings of the Sixth Conference of European Research in Mathematics Education.

Buteau, C., & Muller, E. (2014). Teaching roles in a technology intensive core undergraduate mathematics course. In A. Clark-Wilson, O. Robutti, & N. Sinclair (Eds.), *The mathematics teacher in the digital era* (pp. 163–185). Dordrecht: Springer.

Buteau, C., Muller, E., Marshall, N., Sacristán, A. I., Mgombelo, J. (2016). Undergraduate mathematics students appropriating programming as a tool for modelling, simulation, and visualization: A case study. *Digital Experiences in Mathematics Education*, 1–25. doi:10.1007/s40751-016-0017-5

Buteau, C., Muller, E., & Marshall, N. (2015a). When a university mathematics department adopted core mathematics courses of an unintentionally constructionist nature: really? *Digital Experiences in Mathematics Education, 1*(2–3), 133–155.

Buteau, C., Muller, E., & Ralph, B. (2015b). Integration of Programming in the undergraduate mathematics program at Brock University. In Online Proceedings of the *Math + Coding Symposium,* London (Canada), June 2015.

Caron, F., & Ben-El-Mechaiekh, H. (2010). *Assessment in mathematics courses integrating technology: Technical/pedagogical challenges and curricular considerations*. Discussion panel at the Workshop on Technology Integration in Teaching Undergraduate Mathematics Students, Toronto, ON, Canada.

Clements, M. K., Bishop, A., Keitel-Kreidt, C., Kilpatrick, J., Leung, F. K. S. (Eds.). (2013). *Third international handbook of mathematics education* (Vol. 27). Springer Science & Business Media.

European Mathematical Society (2011). *Position paper of the European mathematical society on the european commission's contributions to European research* [online]. Available: http://www.euro-math-soc.eu/files/EMSPosPaper13_03_2011_NP.pdf.

Grønbæk, N., Misfeldt, M., Winsløw, C. (2009). Assessment and contract-like relationships in undergraduate mathematics education. In *University science and mathematics education in transition* (pp. 85–105). Springer US.

Grover, S., & Pea, R. (2013). Computational thinking in K–12 a review of the state of the field. *Educational Researcher, 42*(1), 38–43.

Houston, K. (2002). Assessing undergraduate mathematics students. In *The Teaching and Learning of Mathematics at University Level* (pp. 407–422). Springer Netherlands.

Hoyles, C., & Noss, R. (2008). Next steps in implementing Kaput's research programme. *Educational Studies in Mathematics, 68*(2), 85–97.

Liu, O. L., Lee, H.-S., & Linn, M. C. (2010). Multifaceted assessment of inquiry-based science learning. *Educational Assessment, 15*(2), 69–86.

MICA URL (n.d.). *Exploratory and Learning Objects created by Brock students in mathematics courses* [Online]. Available: www.brocku.ca/mathematics/studentprojects.

Muller, E., Buteau, C., Ralph, W., & Mgombelo, J. (2009). Learning mathematics through the design and implementation of Exploratory and Learning Objects. *International Journal for Technology in Mathematics Education, 16*(2), 63–74.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning cultures and computers* (Vol. 17). Dordrecht: Kluwer.

Ralph, W. (2001). Mathematics takes an exciting new direction with MICA program. *Brock Teaching, 1*(1), 1. Available: http://www.brocku.ca/webfm_send/18483.

Scratch (n.d.). https://scratch.mit.edu.

Stacey, K., & Wiliam, D. (2013). Technology and assessment in mathematics. In *Third international handbook of mathematics education* (pp. 721–751). Springer New York.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Wing, J. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–36.