ENJOYABLE LEARNING OF PROGRAMMING VIA MUSIC

Ilana Lavy

Yezreel Valley College (ISRAEL)

Abstract

This paper describes an ongoing research in which a group of seventh and eighth graders are engaged in exploration activities through which they learn programming principles and concepts via experiencing music programming of common melodies. The students participated in ten sessions of weekly activity. Via the programming of known melodies, using the environment of Scratch 2.0, they learned basic concepts of programming such as methods, variables, repetition and control commands, parallel processes, and more. The study was conducted in the form of an action research in which each lesson is designed according to the students’ feedback and insights gained from previous lessons. The study units were designed following the spiral learning principles, in which the learning concepts are revisited several times while their level of complexity raises. Analysis of the data collected until now reveals that the students were enthusiastic during the learning sessions, they were curious to learn and use advanced concepts right from the first lesson. The music programming provided them with situations that created a need for gaining mastery over concepts that would simplify the music programming. They took responsibility of their learning process sending their improved products to the instructor without deliberately asked to do so.

Keywords: Music programming Scratch environment, programming concepts, spiral learning, self-efficacy.

1 INTRODUCTION

In the recent years, there is a growing consent that in order to prepare future generations to utilise technological developments, we should precede the teaching of programming to low grades of elementary school and even to preschool learning [1]. It is particularly important for the generation of ‘digital natives’ that has to live in a digitized world in which technology is evolving rapidly, creating new fields of study, new forms of employment, requiring new skills and abilities [2]. The earlier learners are exposed to programming, it is easier for them to learn it and to function more efficiently in a world where technology is gaining momentum and controlling almost everything in it. Among the benefits of learning programming structures and principles is that it enhances problem solving abilities, analytic thinking abilities, creative thinking and computer literacy ([3, 4, 1, 5]. Developing problem solving skills involves developing cognitive abilities and high-order thinking that help a person to face thinking challenges. Analytic thinking and creativity are needed during the process of looking for the algorithm that solves the given problem [3]. In “Mindstorms: Children, Computers, and Powerful Ideas”, Seymour Papert [6] drew his vision of how children use a computer: "In many schools today, the phrase "computer-aided instruction" means making the computer teach the child. One might say the computer is being used to program the child. In my vision, the child programs the computer and, in doing so, both acquires a sense of mastery over a piece of the most modern and powerful technology and establishes an intimate contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building" (p.5). Papert believed that the learning of programming should be done through playful exploration in which children take control of their own learning. In fact, he was the first researcher that paved the way to the idea that children will learn to code.

Various teaching methods were developed over the years; among them is the spiral method. Bruner [7] proposed the spiral curriculum, a teaching approach in which each topic or skill area is revisited at intervals, at a more complex level each time. First, the topic is taught in a basic level, followed by several rounds that in each more complexity of the topic is added, reinforcing principles that were previously discussed. This teaching method enables the establishment and solidification of the learning topic in a profound manner. Spiral learning provides a gradual learning process from simplistic ideas to complicated ones [8]. In light of what is written above, I found that the spiral learning method suitable for teaching programming principles to young students. To enable the students invest their efforts in understanding the programming principles and the underlying logical ideas, I chose a simple programming language. The young students will be acquainted with programming principle via the Scratch environment. To avoid unnecessary syntax mistakes, Scratch commands are designed as
blocks that can be dragged and executed. Scratch has been translated into many languages so that students can acquire programming principles in their native language. The Scratch environment is taught in some elementary schools in Israel using the Hebrew language. Despite the difficulty of programming in a language that is not one's native language, I believe it is important to teach programming in English because it is the language of all the programming languages and students have to get used to it right from the beginning. To overcome the language barrier, I added a small dictionary at the end of each learning unit that includes the translation of the new English words in it. To 'soften' the rigid character of the programming commands and to raise the students' curiosity as regards to their functionality I decided to teach programming via music. Via the composing of familiar melodies, the learner will be acquainted with programming concepts such as variables, repetitive structures, conditional commands, methods and so forth. Moreover, the teaching/learning process will follow the constructivist theory in a way that the students will be engaged in personal exploration tasks. At the last fifteen minutes of each session, a class discussion will take place in which the new concepts will be discussed and the students will share with the whole class the insights they gained during their explorations.

The aim of the study is to explore the effect of learning programming via music on the students' perceptions as regards to programming studies.

The research questions derived from this aim are:

1. What were the students' expectations from the programming activity? Were these expectations fulfilled?
2. In what ways the learning of programming via music influenced the students?
3. What are the insights gained by the instructor as regards to the learning/teaching programming via music?
4. What are the students' perceptions regarding programming prior and after to their participation in the study?

### 1.1 The scratch 2.0 environment

Scratch was developed by Mitch Resnick at the MIT Media Lab who believed that the ability to code computer programs is an important part of literacy in today’s society. While acquiring programming abilities, students learn important strategies for solving problems, designing projects, and communicating ideas. Scratch is a block-based visual programming language, which is most suitable to young students but can be used by elder ones as well. Via scratch, one can create online projects using a block-like interface. In the present study, I decided to use Scratch 2.0. In this version, the blocks palette is located at the middle of the screen (Figure 1). Each category of commands has a different colour. For example, the colour of commands referring to sounds is pink.

Figure 1 demonstrates the different areas of the Scratch 2.0 screen.

a) The graphical area in which the objects ‘act’ following orders-instructions
b) The list of the objects that have been added to the program.
c) List of blocks according to categories listed at the upper part of this area.
d) The programming area where the user joins programming Blocks to create scripts.

The 'Motion' category includes motion blocks, changes angels and changes X and Y values.

The 'Looks' category controls the visuals of the sprite; speech or thought, change of background, enlarge or shrink transparency and shade.

Due to the research focus, the 'Sound' category is elaborated in Table 1.

The 'Pen' category includes blocks controlling the paint of objects motion on the graphic screen, the pen size, it colour and shade.

The 'Data' category includes the ability to add variables and lists to a program.

The 'Events' category includes blocks controlling the flow of a program.

The 'Control' category includes blocks of defined and undefined repetitive structures, and conditional structures.
The ‘Sensing’ category includes blocks enabling sprites to interact with the surrounding the user has created.

The ‘Operations’ category includes mathematical operators, random number generator, and-or statements that compares sprite positions.

The ‘More Blocks’ category includes custom procedures (adding new Blocks) and external devices control.

Learning how to program is done via music programming therefore programming commands belonging to the ‘Sound’ category are displayed hereby (Table 1.).

**Table 1: The command blocks included under the Sound’ category.**

<table>
<thead>
<tr>
<th>Command</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Play Sound" /></td>
<td>Plays a sound of a meow or any imported sound by the user.</td>
</tr>
<tr>
<td><img src="image" alt="Play Drum" /></td>
<td>Plays a sound of a dram. One can control the type of dram and the duration of its playing.</td>
</tr>
<tr>
<td><img src="image" alt="Rest" /></td>
<td>Pauses the playing music for specified number of beats.</td>
</tr>
<tr>
<td><img src="image" alt="Play Note" /></td>
<td>Plays a music note for specified number of beats. The left number refers to a certain note (varies from 0 to 127) and the right number refers to the duration of its playing.</td>
</tr>
<tr>
<td><img src="image" alt="Set Instrument" /></td>
<td>Sets the musical instrument by which the sound will be played.</td>
</tr>
<tr>
<td><img src="image" alt="Change Volume" /></td>
<td>Three instruction blocks responsible for the volume of the sound: change, set and activation of this function</td>
</tr>
<tr>
<td><img src="image" alt="Change Tempo" /></td>
<td>Three instruction blocks responsible for the tempo of the sound: change, set and activation of this function</td>
</tr>
</tbody>
</table>
In this study, the students learn programming principles and concepts via engagement in music. For that matter, I planned ten learning units in which, via the programming of melodies of simple familiar songs, the students will learn programming concepts such as variables, lists, methods, defined and undefined repetition commands, conditional commands and even recursion. The learning units are designed in the spirit of the spiral curriculum idea [7], meaning that the programming concepts are revisited several times in a way that their complexity level increases in each revisit.

2 METHODOLOGY

In what follows, a brief theoretical background presented on the following issues. The added value of music to the learning of programming, the spiral curriculum; and the importance of developing programming fundamental concepts and computational thinking.

2.1 The added value of music to the learning of programming

Most people of all ages enjoy listening to music and those who were 'blessed' with musical talent even create it for us. The music adds joy and colour to our lives. Because of the significant role of music in our lives, there are those who attribute to music therapeutic abilities [9].

Boulanger & Lazzarini [10] wrote "The audio programming book" for assisting learners how to make music and program audio on modern computers. Music has become an integral part of many multimedia environments like "MicroWorlds Project-Builder", "ScratchJr", "Scratch" and "Alice". "MicroWorlds Project-Builder" of LCSI Company which is based on the LOGO programming language, ScratchJr which is a programming environment, which provides a platform for the development of problem-solving skills in a playful way and can be ideal for the development of reading, writing, and arithmetic skills in preschool education. With ScratchJr preschool children learn fundamental programming principles and concepts via the creation of their animated stories and games in a developmentally correct and playful way [1]. Scratch, which is used in this study, is a block-based visual programming language, which is most suitable to young students but can be used by elder ones as well. There is also "Alice" which is a free 3D development environment designed to advance the teaching of computer science, with an emphasis on object-oriented programming for middle and high school students. Using Alice environment one can create 3D animations (videos and interactive games) easily and quickly, using a rich library of existing 3D models on a variety of topics. All the above environments include music to be used as part of projects students develop.

In the present study, music is the focus of learning. Meaning that via the programming of simple and familiar melodies, the students will be acquainted with fundamental programming concepts and principles such as variables, parameters, lists, methods, repetition and conditional commands and even recursion.

2.2 The spiral curriculum

Bruner [7] coined the term 'spiral curriculum' and wrote, "We begin with the hypothesis that any subject can be taught in some intellectually honest form to any child at any stage of development." Namely, even complex learning materials, if properly structured and presented, can be perceived even by young children. The main ideas underlying the spiral curriculum based on Bruner's work are: (A) Students revisit a topic, theme or subject several times throughout their school studies; (B) The complexity of the topic or theme increases with each revisit; and (C) Revisiting a topic or theme always relates with its previous learning. Spiral learning enables, among other things, the establishment and solidification of the learning topic each time the student revisits it. Spiral learning also provides a gradual learning process from simplistic ideas to complicated ones [11, 8].

Shneiderman [12] used the spiral teaching approach to teach novice programmers claiming that this approach make programming education more natural to students, alleviated 'computer shock' and promote the development of computer literacy. He also asserted that the spiral approach is the parallel acquisition of syntactic and semantic knowledge in a sequence, which increase student motivation by using meaningful examples, builds on previous knowledge, is in harmony with the student's cognitive skills, provides reinforcement of recently acquired material and develops confidence through successful accomplishment of increasingly difficult tasks.
2.3 Programming and computational thinking

Preparing future generations, called ‘digital natives’, to utilize technological developments is one of the main challenges educators face in the recent years. Among the actions to be taken, we have to enable students to gain mastery over computer languages and the sooner the better. That means that we should precede the teaching of programming to low grades of elementary school and even to preschool learning [1]. Especially for the future generations that will have to function in a digitized world in which technology is evolving rapidly, creating new fields of study, new forms of employment, requiring new skills and abilities [2].

The term computational thinking refers to a rich set of analytical methods that effectively involve the human and the machine element in the solution of various problems [13]. These methods include specific tasks such as programming, testing and debugging.

In today's world, more and more professional people possessing versatility capabilities are required. People who will be able to use technological tools together with the professional knowledge they have acquired in order to make a better world will be in demand. Friedman [14] claimed that Computer Science is the connective tissue that might yield the development of versatility capabilities to bridge between people professional expertise with technological innovation [15]. To be able to perceive the potential and limitations of both human and computers intelligence, one has to develop problem-solving abilities and design systems skills [16]. This is possible when a person develops his computational thinking, which is considered an important skill in preparation for future workplace and for better functioning in the modern digital world [1]. Computational thinking is a skill that can be developed in the process of problem solving. There is a common agreement in the scientific community that programming should be included in preschool education similar to reading and writing. In the recent years, the academic and scientific community recognizes to the importance of imparting programming skills to elementary and even preschool students. Computational thinking enables the use of Computer Science in all disciplines providing the means for analysing and developing solutions to all algorithmic problems [15].

2.4 Programming and self-efficacy

Educators have found a positive correlation between students’ beliefs about their academic abilities and their motivation to achieve. Self-efficacy has emerged as a highly effective predictor of students' motivation and learning [17]. The concept of "self-efficacy" was coined by Bandura [18], and the sense of self-efficacy is the degree to which one believes in his ability to perform successfully a task or behaviour under certain conditions in order to achieve the desired results. Bandura [19] concluded that this belief affects the behaviour of the individual, his choices, the efforts he invests, his ability to cope with different situations, and his perseverance in pursuing the goals he sets for himself. Studies have shown that a sense of self-efficacy is derived from the individual's perception of himself in terms of knowledge, personal ability, performance, and control [20, 2, 17].

For many, this belief serves as a guideline for life and a basis for action, influencing the individual's decision to behave in different situations, and determines his ability to persist in stressful situations. A person with a sense of high self-efficacy tends to respond to challenges and persevere in efforts to achieve his goals, believing that it depends solely on him and his determination. This has a significant impact on motivation and performance. According to Piaget's theory formal operational thinking abilities develop around age 11-12 [22]. Approximately, around this age students start to move from concrete thinking to logic/abstract thinking. Moreover, less than 20% of 7th graders and less than 25% of 8th graders reach formal operational thinking abilities [23]. Research showed that people that have reached Piaget's formal operational stage might have the ability to understand programming [3]. It is well known that programming languages are divided into different types: procedural, Object Oriented programming, Script languages and visual programming languages. The understanding of and learning of the above types necessitates different levels of thinking abilities. Due to the low percentage of formal operation thinker among 7th and 8th graders, we might say that majority of them are pre-formal operational thinkers and they will be able to cope with the learning of visual programming language such as Scratch since the cognitive characteristic this language has a concrete component.

2.5 Related works

An exploratory study was conducted by Papadakis et al. [1] that investigated the effect of Scratch programming on 5th grade primary school students’ problem solving skills. The researchers were curious to find out the above students’ attitudes towards programming. They found that programming
in Scratch platform did not cause any significant differences in the problem solving skills of the study participants. There was found only a non-significant increase in the mean of the factor of "self-confidence in their problem solving ability". However, they found that all the students liked programming and wanted to improve their programming and found the Scratch platform easy to use.

In this study, students are engaged with known melodies and are asked to break them into logical units. In the process of decomposition of these melodies, they learn basic concepts of programming. Later, the students are asked to compose melodies by themselves using the programming structures they had studied. In addition, they also learn additional programming abilities of the Scratch environment such as motion and can design and implement a musical project consisting of plot and appropriate animations.

3 THE STUDY

In what follows, information about the study participants, the course of the study, data resources and analysis tools will be presented.

3.1 The study participants

Ten students (five 7th graders and five 8th graders) participated in the study. In fact, more students demonstrated interest in participating in the group however, due to schedule limitations (after eight regular school hours) and technical ones (a need to parents' transportation from school) the number of participants decreased to ten students. The study was conducted within the framework of non-formal studies conducted after school hours. For ten successive weeks (except holydays), there were two hours meetings every week between 14:15 and 16:15. The students registered to the group activities on a voluntary base. The group activities took place at the Centre for Entrepreneurship, which was established in the school in the recent years. This school is a regional six-year middle and high school. During the group activities that were conducted by the researcher, each participant worked on a laptop on which the Scratch 2.0 environment was installed.

3.2 The course of the study

At the first meeting, the researcher presented the Scratch environment to the participants with a special focus on the music part. The music in Scratch is under the 'Sound' category. Each command was explained at extensively, demonstrating the various possibilities of using it. From the second meeting and on, in each meeting, the students were handed with a learning unit that includes several exploration tasks. Each study unit contains between four to eight exploration tasks while the level of complexity increases from one task to the successive one. The students worked individually on the exploration tasks and could turn to the researcher when he/she tackled difficulties. Twenty minutes before the end of each meeting, the researcher conducted a class discussion in which the new learnt concepts were summarized and students could share the insights they gained during the activity. At the last five minutes of a meeting, the students were asked to provide feedback about the unit; whether certain topics should be revisited in the following lesson and the activities they liked. The participants were acknowledged that they could express their opinion freely since it will have no influence on their grades whatsoever.

3.3 Data resources and analysis tools

To explore whether or not there was a change in the students’ perceptions as regards to programming, pre and post questionnaires were handed to the students before and after the group activities. The questionnaire included utterances referring to programming and other relevant issues. The same questionnaire will be handed to the students at the last meeting as well. The research data also included the outcomes of the students’ explorations from all the meetings. They had to save their works and mail it to the researcher at the end of each meeting. Additional data source was the feedback the students provided at the end of each meeting, which was documented by the researcher. The last data source was the reflective journal written by the researcher in which she documented episodes occurred during the meetings and insights she gained as a result.

The present research is an action research [24], which in this case its purpose is to learn from the feedback given by the students and from the teacher’s insights in order to produce effective guidelines for successive lessons [25]. After each meeting, the data that includes the students’ outcomes of the inquiry tasks, the reflective notes taken by the researcher and the feedback given by the students at
the end of the meeting was analysed using content analysis method [26]. The obtained results and insights gained were used to plan and implement in the successive lessons.

4 RESULTS AND DISCUSSION

In what follows, I present preliminary results regarding the students’ expectations from the course, the students' involvement in the learning process and the effects of learning programming via music on their self-perception as regards to their ability to cope with the learning of programming.

4.1 Students’ expectations from the course

Within the pre questionnaire, the students were asked to specify their expectations from the course. In what follows, presented and discussed the students’ expectations from the course.

All the students wrote similar to the following: “I want it [the course] to be fun and I expect to enjoy the learning”. Majority of the students uttered that they expect the course will be interesting. They also uttered that they expect “to know how to program, to gain education, learn new things, open their minds and face challenges.” One student wrote, “I expect to know how to program in Scratch and get mastery over the programming language”.

Two seventh graders were sceptic as regards to their ability to cope with the programming tasks and wrote: “I expect it to be fun and that I will really be able to succeed the programming studies”. Another student wrote: “I am not very good in mathematics and I hope I will succeed to learn programming”. The only female student wrote: “I expect to be able to program something impressive without experiencing too many difficulties and without frustrating myself for many hours. But I most expect to enjoy”.

Content analysis [26] of the students’ utterance revealed that the students’ expectations referred to four main aspects: enjoyment, interest, gaining programming knowledge, and experiencing feeling of success. Since the course was given after school hours, and required efforts from their behalf, they expected that these efforts would be worthwhile. All the students specified that they expect to enjoy the learning and that it will be fun, which is quite uncommon expectation among students. This expectation can be attributed to the unique learning focus – the music.

The following references “I hope I will succeed” or “I am not very good in mathematics..” can imply on the student self-perception as regards to their ability to cope with the learning of programming. The students express their concerns as regards to encountering difficulties during the learning process. Nevertheless, the high frequency of references in the students’ utterances regarding their desire to learn how to program can indicate on the importance they attribute to possess programming knowledge. We may say that this importance increases their motivation to learn despite the concerns they expressed about their ability to cope with this task. The students’ concerns might indicate on a low self-perception about their ability to cope with the programming tasks.

As for the female student, she expects to be able to code a sophisticated program without experiencing frustration and without investing too many efforts. Her utterance points on an emotional attitude to the learning process as well on her self-efficacy [19] as regards to her abilities to cope with difficulties and frustration. She is eager to be able to get mastery over the programming language recognizing the importance of possessing such a knowledge, but expresses hesitations stemming from her fears to encounter difficulties that will be accompanied with feelings of failure and frustration. Nevertheless, she ends her utterance by expressing her feeling that what is most important to her is to experience enjoyment. The common thread between all participants is the expectation to enjoy the learning process, which is not common among students especially when it comes to learning of programming, which is considered, and a difficult task.

4.2 The students’ feedback

As was previously mentioned, the students were asked to provide feedback after each session. After the first session, they say it will help them to code melody notes if they will be provided with the numbers of note on a piano as they appear in the YouTube. To address their request I turned to a music teacher for help me with the notes numbers to address the students’ request. They also specified that the session was enjoyable and ended rapidly. They were also satisfied regarding the pace of progress of learning subjects in the unit.
From the second session and on, they were handed with learning units each includes several exploration tasks referring to programming concepts (see Table 1) and had to work individually on them. To enable them to focus on their explorations, they had to use earphones so that their programmed music will not distract other classmates’ attention. At the first learning units, the students tried to avoid the self-reading of the instructions and turned to me asking for oral explanations. I explained to them that one of the learning goals is to develop independent learner skills and that one of these skills is to be able to understand a given text and learn from it. They claimed that they were not used to such an activity, and that all the learning materials in other subjects are mediated to them by their teachers. To avoid frustration, and yet make them become responsible of their learning process, I decided to educate them to learn individually in a gradual manner. I insisted that they try first to figure out the tasks by themselves and when I saw that the level of frustration raised, I mediated part of the learning materials urging them to continue alone. After three or more weeks, the students' demands for my mediation decreased and this issue was not mentioned in their feedback anymore.

In their feedback, the students also referred to the spiral learning method [7]. They said that the revisiting of programming structures and concepts more than once helped them internalize and understand them thoroughly. The fact that they had an initial acquaintance with a programming structure made it easier to cope with the learning of it at a higher level.

To conclude, the students' feedback enabled me to improve the learning units and to be aware to difficulties raised and address them on the spot.

4.3 Students' engagement in the learning process

As described above, the students worked individually on the exploration tasks and near to the end of each session, a class discussion was held concerning the new learnt concepts. The students were very curious and enthusiastic; this came into fruition in their active participation during the class discussions. They turned independently to YouTube, searching for notes of melodies, and transferred these notes into Scratch commands, which was not an easy task. Some of them even emailed me their programmed melodies without being specifically asked to do so. Right from the first lesson, they asked how to use advanced programming commands that have not yet been learned, to address actions such as playing in different octaves or playing with two hands in concert.

Though these meetings were after school hours and the students were tired, it was not came to fruition in their functioning. Throughout the meetings, lively activity could be observed in class of questions asked by the students addressed to both the researcher and the other students. The students kept checking the progress and achievements of their classmates and the moment they encountered the use of unknown functionalities done by their colleagues, they were keen to gain mastery over these functionalities as well. They constantly asked questions referring to advanced functionalities that were not included in the present learning unit.

The significant insight from the students' behaviour was that learning of a certain functionality stemmed from their need to use it rather than from the teacher's decision to teach it. For example, after coding the notes of a known melody ('little Jonathan'), the students noticed that the octave in which the music way played was rather low. I asked them what we could do in order to play it in a higher octave. They said it is not logical to rewrite the whole melody commands by changing the notes' numbers because it will not solve the problem 'nicely'. They said we need something general to replace the number in the 'play note' command that will enable us to change the note number whenever we wish to do so. This will enable to play the melody in different octaves easily. At this point, they were ready to learn the concept of a variable, which in fact, came from them. After showing them how to define a variable and its related commands (initializing a variable, changing its value), they developed their idea saying that we should use two variables, one for the number of the note and one for its playing duration. Then they turned to other melodies they had already programmed and modified the commands to include the use of variables without being asked to do so.

In another case, in one of the melodies they chose on YouTube, they noticed that part of the notes are blue and part are green and asked for my explanation. After I explained to them that, the different colours were meant to distinguish between the notes played in each hand, they asked how we could translate it into programming. One of them answered, that we should write two different programs, one for the notes to be played in the right hand and one for the notes to be played in left hand and these two programs should be played in concert. Then another question was raised, how two programs can
be played together. Although it was at a preliminary stage of the course, I provided a basic explanation regarding parallel processes specifying that we will return to discuss this issue later in detail.

When the students were asked to animate their melodies, an atmosphere of competition arose between them as regards to the sophistication level of their provided animations. The students kept checking the progress of their peers and the moment they realized that one of them, used a functionality they were not familiar with, they asked her or him to show them how to do it. As a result, they initiated the learning of new concepts without being explicitly asked to do so and were proud to present their outcomes to me.

In order to teach the subject of nesting loops, I planned an activity in which the students were required to compose accompaniment of different types of drums to a particular melody. After demonstrating my simple accompaniment design constituting of two different drums, the students were so enthusiastic and suggested that they build a drums accompaniment for each of the melodies that they had programmed so far and competed with each other regarding the complexity level of their created accompaniment.

4.4 Students' self-perception about programming

The change in the students' self-perception was reflected in their behaviour during class sessions. The engagement in music programming brought humour and colour into the sessions. They showed curiosity and were eager to learn advanced functionalities to improve their products. Observing the animations they created for the melodies revealed that a great deal of thought was invested in them. During the week, between two successive sessions they sent me by e-mail improved versions of their programmed animations specifying how satisfied they are. These results are in line with Bandura [19] who stated that a person with a sense of high self-efficacy tends to respond to challenges and persevere in efforts to achieve his goals.

After the sixth lesson, one of the students said: "I have to say that I had doubts at the beginning of the course if I could learn to program. What broke the ice was the engagement with music; it raised my motivation to learn how to program a sophisticated animation that will fit the melody". Another student said: "Music has made programming into something concrete and effective, which made me motivated to learn it. Not just a theoretical study of commands, but learning for concrete purposes, such as creating drums or playing in different octaves". The student girl uttered: "I don't believe I am saying it but after all my concerns at the beginning, thanks to the music, now I learn because I want it and not because I have to do it".

From the above utterances, it can be concluded that music played an important role on affecting their self-perception as regards to their ability to cope with programming tasks. This came to fruition in the raise of their motivation to succeed in providing solutions to concrete problems. The change in their self-perception might be attributed to the need to provide solutions to concrete problems and not be engaged solely on the theoretical aspects of the programming structures and concepts. Moreover, during the learning of programming structures and concepts they enriched their musical knowledge as well.

5 CONCLUDING REMARKS

This study describes a different learning approach to learning programming - learning via music programming. Using this approach, the need for existing programming concepts and structures such as variables, procedures, and parallel processes raises from the students behalf rather than imposed upon by the teacher. While engaging in a joyful activity of music programming, they asked for advanced programming structures to improve their work. The students' joyful came to fruition in sounds of laughter and joy when succeeding to compose known melodies and in creating humoristic animations to them. It came to fruition also in their engagement with the programming tasks during the time between two successive sessions. They used to email the teacher code of a melody and its improved animation without being deliberately asked to do so. During the class discussions conducted at the end of each session, they were active and eager to demonstrate their outcomes and were curious to see their classmates' outcomes as well. In cases someone use a functionality that were unknown to them, they asked him/her to show them how to use it. In my reflective journal, I documented unusual declarations of the students uttered during the lessons. Many of these utterances were similar to the following: "look at my work I am so smart!", "No one can compete my
work it is the best!" Such utterance might point of high self-perception [19] regarding their abilities to provide valuable programming outcomes.

Finally, I must admit that the most enthusiastic person on this process was myself. I enjoyed developing exploration tasks looking for suitable yet simple melodies trying to figure out what programming concepts can be attached to the different parts of the melody. I enjoyed to see the students' enjoyment during the sessions and learned from their feedback. I believe that studying programming that is considered a complex task via something as enjoyable as music can facilitate the learning process; make it more meaningful and especially enjoyable.

To establish the conclusions and insights gained so far from this study, I intend to analyse the research data at the end of the course and repeat the course in the following academic year with two whole classes. I also intend to adjust the learning units to fit the level of sixth graders and teach them as well.

REFERENCES


