ABSTRACT

This article describes a unique ongoing experience of learning programming via engagement in music programming of known melodies. Seventh and eighth graders participated in 10 lessons of weekly activity. Via the programming of melodies, using the Scratch 2.0 environment, they became acquainted with 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. The study units were designed in the spirit of the spiral learning method, in which the learning concepts are revisited several times while their level of complexity rises. From the data analyzed so far, four categories emerged: enjoyment, interest, gaining programming knowledge, and experiencing feelings of success. The students were enthusiastic during the learning lessons, and they were curious to learn and use advanced concepts. The music programming provided them with practical meaning for the learned programming structures and concepts.

KEYWORDS
Music Programming, Programming Concepts, Scratch Environment, Self-Efficacy, Spiral Learning

1. INTRODUCTION

There is now broad agreement that in order to prepare future generations to make the most of technological developments, we should introduce the teaching of programming to early grades of elementary school and even to preschool learning (Papadakis et al., 2016). This is especially important for the present generation of ‘digital natives’ that has to function in a digitized world in which technology is evolving rapidly, creating new fields of study and new forms of employment, requiring new skills and abilities (Yang et al., 2015). As the learners are exposed at a younger age to programming, it is easier for them to function more efficiently in a world where technology is gaining momentum and controlling almost everything in it. Learning programming structures and principles enhances problem-solving abilities, analytic thinking abilities, creative thinking, and computer literacy (White & Sivitanides, 2002; Thomas et al., 2007; Papadakis et al., 2016; Kalelioglu & Gulbahar, 2014). Developing problem-solving skills involves developing cognitive abilities and high order thinking that help a person face thinking challenges. Analytic thinking and creativity are needed during the process of looking for an algorithm that solves a given problem (White & Sivitanides, 2002). In Mindstorms: Children, Computers, and Powerful Ideas, Seymour Papert (Papert, 1980).
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 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 learning. He was the first researcher who paved the way to the idea that children will learn to code.

In the field of education, various teaching methods have evolved over the years. Among the effective ways of learning is the spiral method. Bruner (1960) 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 (Takaya, 2008). First, the topic is taught on a basic level, followed by several rounds such that in each round 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 thoroughly. Spiral learning provides a gradual learning process from simplistic ideas to complicated ones (Johnston, 2012). In light of what is written above, I found the spiral learning method to be suitable for teaching programming principles to young students. To enable the students to 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 principles 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 become acquainted with programming concepts such as variables, repetitive structures, conditional commands, methods, and so forth. Moreover, the teaching/learning process follows the constructivist theory in a way that the students will engage in personal exploration tasks. In the last fifteen minutes of each lesson, 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.

This study is part of a broader study that aims to explore various aspects of learning to program via music on different research populations. The present study aims to explore the effect of learning programming via music on the students’ perceptions as regards programming studies.

The research questions derived from this aim are:

1. How did students’ programming attitudes change as a result of participating in the lessons?
2. How did programming via music influence students’ programming self-efficacy?

2. A BRIEF INTRODUCTION TO SCRATCH WITH A SPECIAL FOCUS ON MUSIC BLOCKS

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 for young students but can be used by older 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 block palette is located in the middle of the screen (Figure 1). Each category of commands has a different color. For example, the color of commands referring to sounds is pink.

Figure 1 shows 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: The list of blocks according to categories listed in 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, changing angles, and X and Y values.

The ‘Looks’ category controls the visuals of the sprite: speech or thought, background, size, transparency, and shade.

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

The ‘Pen’ category includes blocks controlling the drawing of object motion on the graphic screen, the pen size, its color, 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 surroundings the user has created.

The ‘Operations’ category includes mathematical operators, a random number generator, and ‘and/or’ statements that compare sprite positions.

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

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

In this study, the students will learn programming principles via music. For that purpose, I have planned ten learning units in which, via the programming of simple familiar songs, the students will learn programming concepts such as variables, lists, functions, definite and indefinite loops, nesting loops, conditional commands, and parallel processes. The learning units are designed in accordance with the spirit of the spiral curriculum idea (Bruner, 1960), meaning that the programming concepts are revisited several times each time in an increasingly complex level.
3. THEORETICAL BACKGROUND AND RELATED WORKS

In the following, a brief theoretical background is presented on the following issues: music and programming, the spiral curriculum, developing programming self-efficacy, and related works.

3.1 Music and Programming

Scanning of the research literature yielded various studies that explore the learning of music using programming. However, no studies that explore the learning of programming via music, which is the focus of the present study, were found.

In recent years, there is a broad agreement that the circle of people that are experts in developing tomorrow’s computing technologies must be expanded. To expand this circle, the teaching of programming should be preceded to elementary school and even to preschool (Papadakis et al., 2016), and should strive to integrate as many female students as possible, many of whom are currently reluctant to program. However, exposing 7th and 8th-grade students to learning how to program, requires formal thinking abilities that only 20% of these students possess. Most of these students are still pre-formal operational thinkers and need the mediation of abstract concepts via concrete use (White & Sivitanides, 2002). And that’s where the music comes in. It incorporates emotional aspects into the learning process and ‘softens’ the rigidity of the programming language. During the programming of a song, the need for the use of programming structures arises. The music provides concrete meaning to abstract concepts and thus helps the students internalize and understand them.

3.2 The Spiral Curriculum

Bruner (1960) 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 to 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 (Harden, 1999; Johnston, 2012).

### Table 1. The command blocks included under the sound category

<table>
<thead>
<tr>
<th>Command</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>play sound meow</td>
<td>Plays a sound of a meow or any imported sound by the user.</td>
</tr>
<tr>
<td>play drum 14 for 608 beats</td>
<td>Plays a sound of a drum. One can control the type of drum and the duration of its playing.</td>
</tr>
<tr>
<td>play note 14 for 608 beats</td>
<td>Pauses the playing music for specified number of beats.</td>
</tr>
<tr>
<td>play note 14 for 608 beats</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>set instrument to 14</td>
<td>Sets the musical instrument by which the sound will be played.</td>
</tr>
<tr>
<td>change volume by 10</td>
<td>Three instruction blocks are responsible for the volume of the sound: change, set, and activation of this function.</td>
</tr>
<tr>
<td>change tempo by 100</td>
<td>Three instruction blocks are responsible for the tempo of the sound: change, set, and activation of this function.</td>
</tr>
</tbody>
</table>
Shneiderman (1977) used the spiral teaching approach to teach novice programmers, claiming that this approach makes programming education more natural to students, alleviates ‘computer shock’, and promotes the development of computer literacy. He also asserted that the spiral approach is the parallel acquisition of syntactic and semantic knowledge in sequence, which increases 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 the successful accomplishment of increasingly difficult tasks.

3.3 Developing Programming 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 (Zimmerman, 2000). The concept of “self-efficacy” was coined by Bandura (1977), and the sense of self-efficacy is the degree to which one believes in one’s ability to perform successfully a task or behavior under certain conditions to achieve the desired results. Bandura (1986) concluded that this belief affects the behavior 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 (Linnenbrink, E.A. & Pintrich, P. R., 2010; Goddard et al., 2004; Zimmerman, 2000).

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 high sense of 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 (Bitner, 1991). At approximately this age students start to move from concrete thinking to logical/abstract thinking. Moreover, less than 20% of 7th graders and less than 25% of 8th graders reach formal operational thinking abilities (Epstein, 1980). Research showed that those that have reached Piaget’s formal operational stage may have the ability to understand programming (White & Sivitanides, 2002). It is well known that programming languages are divided into different types: procedural, object-oriented, scripting, and programming and visual programming languages. The understanding and learning of the above types necessitate different levels of thinking abilities. Due to the low percentage of formal operational thinkers among 7th and 8th graders, we might say that the majority of them are pre-formal operational thinkers and they will be able to cope with learning a visual programming language such as Scratch since this language has a concrete component as the cognitive characteristic.

3.4 Related Works

An exploratory study was conducted by Papadakis et al. (2016) that investigated the effect of Scratch programming on 5th-grade primary school students’ problem-solving skills. The researchers were curious to discover the above students’ attitudes towards programming. They found that programming in the 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 in the programming of simple and known melodies. During this process, they cope with various problems such as finding and correcting a certain music note that does not match the song’s tune. To address this problem, they learn to appreciate the use of functions (blocks in Scratch) and understand the meaning of modular programming. Then, the students are asked to compose melodies by themselves using the programming structures they have studied. Besides,
they also learn additional programming functionalities of the Scratch environment such as animation motion and can design and implement a musical project that includes appropriate animations.

4. THE STUDY

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

4.1 The Study Participants

Ten students (five 7th graders and five 8th graders) from a regional school in Israel participated in the study, two female students, and eight male students. More students showed interest in participating in the study. However, due to schedule limitations (after school hours) and technical ones (a need for parents’ transportation from school), the number of participants decreased to ten students. The study was conducted within the framework of informal studies taking place after school hours. For ten successive weeks (excluding holidays), there were two-hour meetings every week between 14:15 and 16:15. The students registered for the group voluntarily. The group activities took place at the Center for Entrepreneurship, which was recently established in the school. 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.

4.2 The Course of the Study

The Scratch environment was presented to the participants at the first lesson with a special focus on the music blocks. The music in Scratch appears under the ‘Sound’ category. Each block command was explained broadly, demonstrating the various possibilities of its use. From the second meeting on, the students were handed a learning unit that included several exploration tasks. Each study unit contained four to eight exploration tasks, in an increasing level of complexity. The learning units were designed according to the spiral learning, meaning that concepts and programming structures were revisited several times in an increased complexity level. 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, a class discussion was initiated in which the newly learned concepts were summarized, and students could share the insights they gained during the activities. In the last five minutes of each meeting, the students were asked to provide feedback on the unit. In their feedback, they were asked to address the following issues: whether certain topics should be elaborated in the next lesson; whether the exploration tasks were clear; whether there was a need to add/omit part of the tasks; and to refer, in general, to their level of satisfaction from the lesson. The participants were instructed that they could express their opinion freely since it would not influence their grades whatsoever. Table 2 shows a schematic description of the weekly lessons.

4.3 Data Resources and Analysis Tools

The study is exploratory in nature and it is part of broader research that explores the effectiveness of a suggested learning method for teaching mid-school and elementary school students how to program via music. The data resources of the present study were:

1. **A student completed a questionnaire administered pre and post-program:** The questionnaire included 20 utterances referring to programming and related issues (see Table 3) and the students were asked to indicate their level of agreement on a 5-scale ranging from full agreement to no agreement at all. The questionnaire also included an open question in which the students were asked to specify their expectations from the course. This data used to address the first research question.
Table 2. Schematic description of the learning lessons

<table>
<thead>
<tr>
<th>Week</th>
<th>Lesson’s topics</th>
<th>Content</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introducing the Scratch 2.0 environment; Introducing the programming commands relating to music</td>
<td>Notes, octaves, music bar, tempo</td>
<td>Program a simple known melody and email it to the teacher.</td>
</tr>
<tr>
<td>2</td>
<td>Repetition commands, variables</td>
<td>Avoid repetition of code segments in melodies; be able to play a melody in different octaves</td>
<td>Look for repetitive code segments in the melodies they had previously programmed and use repetition commands to replace them. Replace the number of the note in the ‘Play note’ command by a variable and modify the whole melody accordingly.</td>
</tr>
<tr>
<td>3</td>
<td>Methods, motion</td>
<td>Define a method; create a ‘main’ program constituted from method blocks: ‘move’, turn right/left, and pen position to draw a movement path</td>
<td>Divide a melody into different logical blocks using methods. Create a simple animation of the sprite using movement blocks to accompany a melody</td>
</tr>
<tr>
<td>4</td>
<td>Animation of a known melody</td>
<td>‘movement’ and ‘sensing’ blocks, customize sprite and background</td>
<td>Provide an enhanced animation to the melody ‘Pink Panther’.</td>
</tr>
<tr>
<td>5</td>
<td>Animation of known melody played by two hands</td>
<td>Parallel processes</td>
<td>Animate several objects moving in concert on the screen, each playing the same melody in different octaves.</td>
</tr>
<tr>
<td>6</td>
<td>Repetition commands – round two</td>
<td>Nesting of loops</td>
<td>Create drums accompaniment to melodies using nesting of loops and different kinds of drums.</td>
</tr>
<tr>
<td>7</td>
<td>Conditional commands</td>
<td>Conditional blocks, logic expressions</td>
<td>Use conditions to stop infinite loops to coordinate between animation and a certain melody.</td>
</tr>
<tr>
<td>8</td>
<td>Methods with parameters</td>
<td>Modular programming, methods with parameters</td>
<td>Animate several objects moving in concert on the screen playing the same melody in different octaves.</td>
</tr>
<tr>
<td>9-10</td>
<td>Parallel processes – round two</td>
<td>Programming a melody that included several</td>
<td>Using the block ‘announce’ and activating several sprites which play in concert</td>
</tr>
</tbody>
</table>
2. **The student outcomes from each lesson:** Tracing the students’ managing their projects could shed light on their curiosity, enjoyment, and motivation to learn and can provide additional information as regards to the second research question.

3. **Student feedback at the end of each lesson or meeting:** This data source helped to modify the following learning unit and to address reported difficulties.

4. **In-depth semi-structured interviews:** Conducted with the study participants to reveal their qualitatively different ways in which they experience, perceive, and understand the learning to program via music. The leading questions in the interviews were based on Ornek’s study (2006) and were modified to suit the present study. This data was used to address the second research question.

5. **An instructor reflection journal:** After each lesson, the researcher documented episodes that occurred during the meetings and the insights she gained as a result.

The present research was planned to suit the form of action research (Stringer, 2013), which aims to learn from the feedback given by the study participants and from the researcher’s insights to improve successive learning units (Denscombe, 2014). Action research includes the following stages: (1) identify a change; (2) observe the present situation; (3) plan different possible interventions; (4) carry out the intervention; (5) observe the effects of the intervention; (6) evaluate against original goals; and (7) share findings.

In the current study, the understanding that traditional methods used to teach programming deter students (especially female ones) from choosing programming studies has led to the development of a learning method that integrates enjoyable activity (music) with programming (addressing stages 1–3). Within the weekly lessons, the developed units were taught to the study participants (stage 4). In each lesson, participant observation was conducted by the researcher and documented in detail in the researcher’s reflective journal. The researcher’s gained insights and the students’ feedback given at the end of each lesson was reviewed and evaluated (stages 5 and 6). The insights gained from the evaluation process were implemented in the successive learning unit (stage 7).

To analyze the received data, the phonomyography method (Marton, 1986) was found to be the most suitable. This method suits educational research that aims to discover the qualitatively different ways in which students experience, perceive and understand various aspects of a new phenomenon (Bowden et al., 1992) which in our case, concerns learning how to program via music. According to Sjöström (2002), Phonomyography analysis includes certain steps: (1) familiarization, of the researcher with the research data. It is done using reading through the transcripts of the gathered data; (2) compilation of answers from participants to a certain question. The researcher should identify the most significant elements in answers given by participants; (3) condensation, or reduction, of the individual answers to find the central parts of a dialogue; (4) a preliminary grouping or classification of similar answers; (5) a preliminary comparison of categories; (6) naming of categories; and (7) a contrastive comparison of categories. It includes a description of the character of each category and similarities between categories.

In the following section elaboration on the categorization process will be presented.

## 5. RESULTS AND DISCUSSION

The data analysis revealed information regarding two main interconnected themes: self-perception as a learner and perception of the knowledge domain (see Fig. 2). As to the first theme, cognitive and emotional aspects were detected. Cognitive: A2, A3, emotional: A1. Both cognitive and emotional: A4, and the change of the students’ self-efficacy regarding their ability to cope with the programming learning which is the amalgamation of A1-4. As to the second theme aspects relating common perceptions about the programming domain such as being a complex and masculine domain (B1);
suitable for only excellent students (B2); require persistence and patience (B3), and it is in a high demand profession (B4).

5.1. Students’ Programming Attitudes Change as a Result of Participating in the Lessons

Table 3 presents the utterances of the pre and post questionnaires and the average score of each (the maximum score is 5). Utterances A-I refer to the students’ perceived abilities to cope (or not) with learning how to program (marked in italics). Utterances J-P refer to the students’ views about programming in general (marked in bold). Utterances Q-T refer to views and prior knowledge of music (marked in regular font).

As can be seen from Figure 3, about perceived learning capabilities, there is a significant change in the students’ views relating to the connection between mathematics and programming (C); only excellent students can learn to program(G) and expressions of concerns regarding the difficulty to learn how to program (H). The views expressed in the pre questionnaire regarding the connection between mathematics and programming are in line with Gomes et al. (2006) who found several reasons underlying the difficulties novice programmers encounter during their programming learning, among them are the lack of mathematical problem-solving competences. The views expressed in the pre questionnaire regarding the difficulty to learn how to program are known and investigated by several researchers (Bosse & Gerosa, 2017; Piteira & Costa, 2013; Derus & Ali, 2012; Bennedsen & Caspersen, 2007).

About general views regarding programming, a significant change between the pre and post questionnaires is observed regarding most of the utterances of this category (except “I don’t know what programming is” (K). These students’ pre attitudes stem from prevailing views on programming learning that is considered a complex subject requiring analytical thinking (Tsalapatas, et al., 2011) and abstraction skills (Kramer, J. (2007; Bennedsen & Caspersen, 2006). About the third category referring to views and prior knowledge of music, a significant change was observed regarding the utterance: “I don’t like music”(U) and “Don’t understand the connection between music and programming”(V). this change may be attributed to their exposure to learning how to program via music.
The following presents the results obtained from the open question referring to the students’ expectations of the course. All the students wrote similarly to the following: “I want it [the course] to be fun and I expect to enjoy the learning.” Most of the students wrote that they expect the course to be interesting. They also uttered that they expect “to know how to program, to gain an education, learn new things, open their minds, and face challenges.”

Two 7th graders were skeptical as regards their ability to cope with the programming tasks and wrote: “I expect it to be fun and that I will be able to succeed in the programming studies.”

Table 3. Utterances and average scores (pre and post)

<table>
<thead>
<tr>
<th>utterance</th>
<th>pre</th>
<th>post</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. I’m good at mathematics</td>
<td>3.75</td>
<td>3.75</td>
</tr>
<tr>
<td>B. I’m good at English</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C. To be a good programmer one must be good at math</td>
<td>3</td>
<td>0.2</td>
</tr>
<tr>
<td>D. It will be easy for me to learn to program</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>E. To be a good programmer one needs to know how to solve problems</td>
<td>4.8</td>
<td>4.8</td>
</tr>
<tr>
<td>F. It would be difficult for me to learn programming because of my poor English</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>G. Programming is only for excellent students</td>
<td>3.5</td>
<td>1</td>
</tr>
<tr>
<td>H. I can’t learn to program because it’s hard</td>
<td>3</td>
<td>0.5</td>
</tr>
<tr>
<td>I. Programming requires a lot of patience</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>J. Programming can be learned in a jolly way</td>
<td>0.5</td>
<td>4.8</td>
</tr>
<tr>
<td>K. I don’t know what programming is</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>L. Knowledge of programming will help me achieve a good profession in life</td>
<td>4.6</td>
<td>4.8</td>
</tr>
<tr>
<td>M. Learning to Program is boring</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>N. Not everyone needs to know how to program</td>
<td>2.6</td>
<td>1</td>
</tr>
<tr>
<td>O. Programming can be learned at any age</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>P. I’ll be able to succeed in life even without knowing to program</td>
<td>2.9</td>
<td>0</td>
</tr>
<tr>
<td>Q. I don’t like music</td>
<td>2</td>
<td>0.1</td>
</tr>
<tr>
<td>R. Don’t understand the connection between music and programming</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>S. I play/played a musical instrument</td>
<td>3.3</td>
<td>3.3</td>
</tr>
<tr>
<td>T. I know how to read music notes</td>
<td>2.5</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Figure 3. Distribution of results of the pre and post questionnaires
Another student wrote: “I am not very good at mathematics and I hope I will succeed to learn how to program.” One of the female students wrote: “I expect to be able to program something impressive without experiencing too many difficulties and without frustrating myself for many hours. But mostly I expect to enjoy.”

The students’ expectation to experience enjoyment during the learning process is unexpected. Usually, students’ expectations from a new learning method are that they will acquire new knowledge, not encounter difficulties and understand the material being taught (Jin & Hill, H, 2001; Littlejohn et al, 2010). The expectation of enjoyment probably stems from music. Music is an important source of enjoyment for people (Batt-Rawden & DeNora, 2005). When the teaching method was introduced to the students, they expected the music to bring enjoyment into the learning process.

In what follows elaborates on A1 (enjoyment from the learning process), A4 (motivation and curiosity), A2 (acquiring programming knowledge and experiencing feelings of success which is the amalgamation of A1-4), will be presented and discussed.

5.2 The Students’ Enjoyment From the Learning Process

Herein representative excerpts from the interviews:

To tell the truth, at first I didn’t believe I would enjoy the learning process. However, music sounds began to be heard in the classroom right from the first lesson in which we learned the music blocks. These music sounds created a liberating and pleasant atmosphere and the dry command blocks became clear and meaningful. The freedom to choose to work on one of my favorite songs reinforced my feelings of joy. When I was able to build the program so that the song was played accurately, I felt both joy and pride.

The learning method in all the school subjects is similar. The teacher explains a new topic and related concepts, and then we are asked to practice them. In cases where complicated concepts are involved, I have to say that most of the time I don’t understand them thoroughly. This comes to fruition in my difficulties to use them properly. The feeling of partial or even lack of understanding does not add to my mood and certainly does not make me happy in lessons. Learning how to program via music made me feel different. Providing musical meaning to the complicated programming concepts made the difference. There is something about music that gives you a sense of release and maybe that’s what makes me enjoy it. The practical meaning of the programming concepts helped me understand them and motivated me to complete the melody to one of my favorite songs. Feelings of success always make me happy.

What was special about the course is that the music I used was written by me. I could import ready-made music from YouTube for my programmed animation but it’s not the same. Here I can say that I programmed everything [both animation and music]. Successfully programing the music had a great impact on my enjoyment of the process.

From the researcher’s reflective journal:

The students were very curious and enthusiastic and demonstrated feelings of enjoyment. These feelings were demonstrated in verbal expressions such as: “Wow! [proudly] Look what my animation looks like!” or “I like the effect of music on the programming. It revives the ‘dry’ commands”; “I don’t believe in what I am going to say but I like what I am doing.” These expressions and more like these were repeated every lesson. The students’ enjoyment could be also observed in their body language. They were giggling during their work and looked relaxed. There were no sounds of frustration heard in class. They arrived at class early before its start time and remained after it was due to end. During
the class discussions, they were asked to specify the positive and negative aspects of the learning method. All of them uttered in similar words that the music part made the difference and that without the music if they had to learn the programming commands the way they learn other school subjects; they would probably cease to attend to these lessons.

Jenkins (2002) said:

Learning (or perhaps here ‘being taught’) programming can be very dull. Lectures covering the details of syntax are never going to be especially inspiring, and exercises that involve simple mathematical manipulations of collections of student marks, stock levels, baseball statistics, or bank account details are never going to set the pulse racing. Yet a glance in many programming texts will yield many turgid examples of each of these.” Referring to enjoyment and interest he added: “At its best programming can be an enjoyable, creative activity, and many students derive great enjoyment from their programming. They enjoy it even more (and learn more) when they can work on assignments that inspire them. It is a shame that so few assignments do indeed inspire. (p. 56)

From the above excerpts and the evidence from the researcher’s reflective journal, it might be said that the learning of programming via music made it happen.

Varila & Viholainen (2000) (in Rantala & Maatta, 2012) referred to two kinds of joy in learning, active and passive. The passive joy of learning means contentment with a pleasant state. The learner’s immediate reaction issued by an external factor beyond one’s control. The active joy of learning is a state that results from the students’ effort. The emotion of achievement that results from sudden and surprising success. Learning how to program via music nurtured both kinds of joy. Learning how to program via music, which was planned as a joyful activity, adds to the passive joy while the students’ success in creating animation or a computer game with their programmed music playing in the background adds to the active joy.

5.3 The Students’ Curiosity and Motivation During the Learning Process

Representative excerpts from the interviews:

Right from the first lesson when we were asked to program a song we chose; several questions came up immediately such as: How to find an incorrect note in the melody when all the command blocks are listed in a sequence? How to treat a repeated group of notes? How to play the same melody in different octaves? How do you play with both hands? And since I wanted my melody to play correctly, I wanted to know the answers to all of those questions right away.

While working on my project, I saw that the animation of one of my classmates includes functionality that I was not familiar with. I was eager to get to know it and to include it in my work. I asked him to show me how to use this functionality. This resulted in a significant upgrade of my animation.

From the researcher’s reflective journal:

Right from the first lesson, the students asked how to use advanced programming commands that had not yet been learned, to address actions such as playing music in different octaves or playing with two hands in concert.

The students demonstrated curiosity and motivation during the lessons. These were reflected in their behavior. They kept checking the progress and achievements of their classmates, and the moment they encountered unknown functionalities used 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 students’ curiosity to learn a certain concept stemmed from the need to use it in a specific context. For example, after coding the notes of a known melody (‘Little Jonathan’), the students noticed that the octave in which the music played was rather low. The researcher asked them to suggest a solution to this problem. They said it is not logical to rewrite the whole melody commands by changing the notes’ numbers because it will not provide an elegant solution. They said there is a need for a general element to replace the specific number in the ‘Play Note’ command block that will enable us to change the note number whenever we wish to do so. The use of this general element will enable us to play the melody in different octaves easily. At this point, they were ready to learn the concept of a variable, which came from their request. After demonstrating how to define a variable and its related commands (initializing a variable, changing its value), they said 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.

Curiosity is defined as a tendency to inquire, investigate, or seek after knowledge. It is simply the frame of mind in which you want to learn more about something. It also provides a source of internal motivation that comprises the foundation of education (Binson, 2009). It is widely accepted that curiosity provides the motivational fuel for learning at each step of the educational process. Curiosity motivates students to learn more and more about their world and as a result, a deeper understanding of the interactions and the relationship between the various elements is achieved (Loewenstein, 1994; Binson, 2009).

The study units included in the proposed learning method, include a collection of inquiry tasks that will arouse students’ curiosity to deepen and understand the concepts involved.

To get students to appreciate the value of nesting loops, the students were asked to compose accompaniment of different types of drums to a certain melody. After the researcher demonstrated a simple program in which two different drums play in concert, the students were so enthusiastic and suggested that they build a drum accompaniment for each of the melodies that they had programmed so far and competed with each other regarding the complexity level of their created accompaniments.

5.4 Gaining Programming Knowledge

From the interviews:

During this course, I gained basic programming knowledge. To be able to find errors in my programs, I realized the importance of using a function. To generalize a solution, I learned to appreciate the importance of a variable. To avoid code segment repetitions, I learned about loops and how to build drums accompaniment, I learned about nesting of the loops. Learning to handle simple loops helped me to understand the nesting loops.

I learned command blocks that would help me to design an animation. For the animation to do what I had planned, I had to probe the meaning of relevant command blocks. This investigation process certainly helped me to internalize the role and meaning of each block command.

From the researcher’s reflective journal:
To facilitate the understanding of some of the concepts, the learning units were designed according to the spiral learning method. For example, the issue of loops was addressed first referring to definite loops (the number of repetitions is known), then they were introduced with indefinite loops (the ‘while’ command), and finally, they learned nested loops. Moreover, the learning of programming structures was associated with music characteristics. For example, nesting of loops was associated with drum accompaniment, and variables were associated with playing a melody in different octaves. These associations have a dual role. First, they enable the students to connect new concepts to concrete objects, and second, they may develop the students’ appreciation and understanding of these programming structures and concepts, hoping that when they will be exposed to these structures and concepts in the future they will be able to cope with them easily.

When the students were asked to animate their melodies, an atmosphere of competition arose between them as regards the sophistication level of their provided animations. The students kept checking the progress and achievements of their classmates, and the moment they encountered the use of unknown programming functionalities used by their colleagues, they were keen to gain mastery over these functionalities as well. As a result, they initiated the learning of new concepts without being explicitly asked to do so.

The Researcher literature discusses broadly the positive effects of competition on the learning process (Burguillo, 2010; Fülöp et al., 2007; Sheridan, S. & Williams, P. 2011). They say that friendly competitions provide a strong motivation for students, helping to increase their learning performances. Fülöp (2004) defined constructive competition as a social and cultural phenomenon that enhances children’s abilities, develops their ambitions, and motivates their learning. In the context of the present research, the obtained results point to constructive competition that was created between students. This competition motivated them to gain mastery over unfamiliar programming concepts and resulted in impressive outcomes.

5.5 Experiencing Feelings of Success

Representative excerpts from the interviews:

I came to the course with low expectations. I’m not that good at math, so I was afraid I would run into difficulties. What lessened my concerns was the music. The immediate feedback to any command I used, helped to navigate my learning and to result in the correct melody. With every note I managed to play correctly, my self-confidence in my abilities rose. You know that there were a lot of students who were afraid to join the course because they were not strong in math. Tell them they don’t have to be good at math to learn how to program.

Music has made programming into something concrete and increased my motivation to learn it. Not just a theoretical study of commands, but learning for concrete purposes, such as creating drums accompaniment or playing in different octaves.

I don’t believe I am saying it but after all my concerns at the beginning, thanks to the music, I understand the programming structures. As if the music provided a practical meaning to them. I learn because I want to do it not because I have to do it.

From the researcher’s reflective journal:

The change in the students’ self-perception was reflected in their behavior during the lessons. The engagement in music programming brought humor and color into the lessons. Curiosity and
eagerness to learn advanced functionalities to improve designed products were observed. During the week, between two successive lessons, they sent me by email improved versions of their programmed animations specifying how satisfied they are.

These results are in line with Bandura (1986), 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.

From the above data, it can be concluded that music played an important role in affecting their self-perception as regards their ability to cope with programming tasks. This resulted in raising their motivation to succeed in providing solutions to concrete problems. Through the engagement in music, the students had immediately successful experiences that helped to increase their self-perception about their ability to cope with the learning of programming. 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. Also, according to Piaget’s theory, most of the 11-12 students are pre-formal operational thinkers and still in the phase of concrete thinking (Bitner, 1991), the mediation of the abstract programming concepts through concrete use of them in music enhanced the internalization and understanding of the programming concepts and helped in developing feelings of success.

Success and a sense of self-efficacy are interrelated (Bandura,1977). Success increases a sense of self-efficacy and a high sense of self-efficacy affects the behavior of the individual, her motivation, her choices, the efforts she invests, her ability to cope with different situations, and her perseverance in pursuing the goals she sets for herself (Bandura,1986). Learning programming fundamentals via music provided the students with success opportunities which enabled the increase in their self-efficacy regarding their abilities to cope with programming.

5.6 Gained Insights From the Learning Process and Concluding Remarks

The willingness to learn a new concept came from the students rather than being imposed by the teacher. For example, during the programming process of a melody they chose, a need to understand and to use the following concepts was created. The need to use function and understand its advantages (errors in the code can be easily detected); to recognize, understand and implement repetition structures (to deal with repetitive code segments); to recognize, understand and implement loop nesting (to program drum accompaniment); to understand the meaning of a variable and be able to use it (playing a melody in different octaves), and understand the meaning of parallel programming (playing a melody and its accompaniment simultaneously).

A connection between abstract programming concepts and one of their concrete implementations was established (e.g. nesting loops and drums accompaniment).

The music induced a relaxed class atmosphere and “softened” the students’ coping with the programming that is considered a difficult and complex issue. They walked around freely in the lab, checking the progress of their friends. Once they discovered the use of programming functionality they did not recognize, they were determined to learn it and implement it in their code. A friendly competitive atmosphere was established that contributed to the learning process.

Finally, I must confess that the results are above and beyond my expectations. I watched the students’ enthusiasm with pleasure during the lessons, the enthusiasm they demonstrated while working on their projects between successive class sessions, their willingness to stay far beyond class hours, their voluntary search on YouTube for melodic for notes of melodies without being asked to do so, and their curiosity that was expressed by active participation in classes. I believe that this is the kind of learning Jenkins (2002) was referring to.
To establish the conclusions and insights gained so far from this study, I intend to repeat the course in the following academic year with sixth-grade students. For that matter, I intend to adjust the learning units to fit the level of 6th graders and teach them programming fundamentals via music as well.

I hope that when the students encounter programming structures in the future, they will associate them with what they experienced in this framework. For example, when they encounter nested loops, they will recall drum accompaniment.
REFERENCES


ENDNOTE

1 A popular children’s song in our country.