UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Development of a series of interactive projects for learning programming at school

Author:
Natalya PARANYAK

Supervisor:
Oksana PASICHNYK

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

I, Natalya PARANYAK, declare that this thesis titled, "Development of a series of interactive projects for learning programming at school" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Development of a series of interactive projects for learning programming at school

by Natalya PARANYAK

Abstract

There are many research studies on the education process. Even fewer studies are devoted to programming learning or the use of programming in the teaching of children. This work is another attempt to explore the possibilities of the modern world and use them in developing an application that will help children learn. We planned to use this application primarily to study programming at the senior (and secondary) schools. We succeeded in exploring the current achievements of the programming world, which are aimed at optimizing the learning process. We tested many of them during classes with children. We have succeeded in revealing key things that are both interesting and fun to educate children. The result of our work is the Space Lab application.

Acknowledgements

I would like to express my appreciation to Oksana Pasichnyk for providing supervision and consultations, and for sharing her experience in the education domain.

Contents

List of Figures

To my supportive family

Chapter 1

Introduction

The theme of my thesis is Development of a series of interactive projects for learning programming at school. I chose this topic not by chance, but because our world is constantly changing. And approaches to teaching children are no exception. It is worth noting that many of the things that we choose to teach children depend on their future lives. For example, the enthusiasm to learn something new, develop the ability to perceive information and self-search, when formal learning will end and preparation for the challenges of the modern world. By changing approaches, perhaps by redefining approaches, we can drastically change the future of the future generation - their attitudes, aspirations, and opportunities. In my opinion, it is important that education corresponds and prepares to the requirements of the time we live in now.

1.1 Motivation

There is a lot of research on this point. And according to each of these studies, there are applications that cover the conclusions of each study. Each of these studies is important in the context of this bachelor's thesis and the creation of a new application. They show a research format that is suitable for studying this and related topics. Also, these studies may explain why each of the existing applications looks like this. But despite a large number of studies on this topic, there is some hole in explaining what elements are key in existing applications: which elements are fundamental to the teaching of children, and which are only additional. There is also a small amount of research in recent years. Again repeating, the world is changing, especially the digital world. And many things that were 10 or even 5 years ago popular today are obsolete. We should use all the possibilities of the today for the creation of a new future for children. Also, a large number of studies give examples for teaching children in general and a very small number of studies (discussed below) gives examples for learning in the group. Therefore, I think that the relevance of this research exists.

1.2 Goals of bachelor's thesis

Thus, the purpose and specific objectives of this study can be summarized as follows:

- Explore the available learning tools
- Explore and discover which learning format is best for teaching children
- Identify key details of a successful education application

- Develop a teaching application

1.3 Structure of the thesis

- Chapter 2. Related works
  This chapter contains information on various research topics on programming, programming approaches for children and adolescents, and an explanation of why some programming languages...
- Chapter 3. Experiments
  This chapter is the key to understanding why the application we created looks like aforementioned. In this section, we will describe the testing of various applications and explain the impact they have on the future solution.
- Chapter 4. The Proposed Solution
  This chapter will discuss what things have been selected from research. Our thoughts about why these items are important in the context of solution development are described in detail. And the final part of this section will be the testing of a finished tool at the school during the lesson of programming (that is, in particular, in conditions that have been identified as the basis for the development of the application)

Chapter 2

Related Works

2.1 All I Really Need to Know (About Creative Thinking) I Learned (By Studying How Children Learn) in Kindergarten by Mitchel Resnick

For modern children, there are many challenges in the world. And the task of kindergarten and school is to prepare children for them. One method is to teach children to think critically. Approaches used when teaching children in a kindergarten should be used in education for different age groups. Under the approach of studying in a kindergarten is a spiral - Imagine, Create, Play, Share, Reflect and Imagine again. In this spiral, each process leads to another and eventually generates a project. One of the reasons why this approach is not used in further school education - the lack of "projects" in which students could use this spiral.
Details on what means every step of the spiral:

- **Imagine:** The purpose of this part is to provide children with applications for work with something, and the ways to solve the problem they will find themselves. That is, the child's imagination will allow one and the same tools to be used differently.
- **Create:** During this step, children should not only use existing applications but also be able to add something new to them.
- **Play:** The game is a very important aspect of learning since, during it, children conduct experiments and test link hypotheses. If you take video games / in which play children, then it becomes obvious that the degree of engagement is higher than, for example, during a class at a school. What is important is that this engagement was in the right direction. Since not all games convey the development of creative thinking, the process of creation and the principle of problem-solving.
- **Share:** Sharing projects foster other people's inspiration for new projects and develop community. If we take into account research data of Scratch programming language, then most projects are a remix (new refined copy) of an existing project.
- **Reflect:** The process of reflecting is very important as it helps children to identify the things that need to be refined and which can be improved upright once.

based on: *All I Really Need to Know (About Creative Thinking) ILearned (By Studying How Children Learn) in Kinder-garten by Mitchel Resnick*

Figure 1: The kindergarten approach to learning

FIGURE 2.1: Kindergarten approach to learning

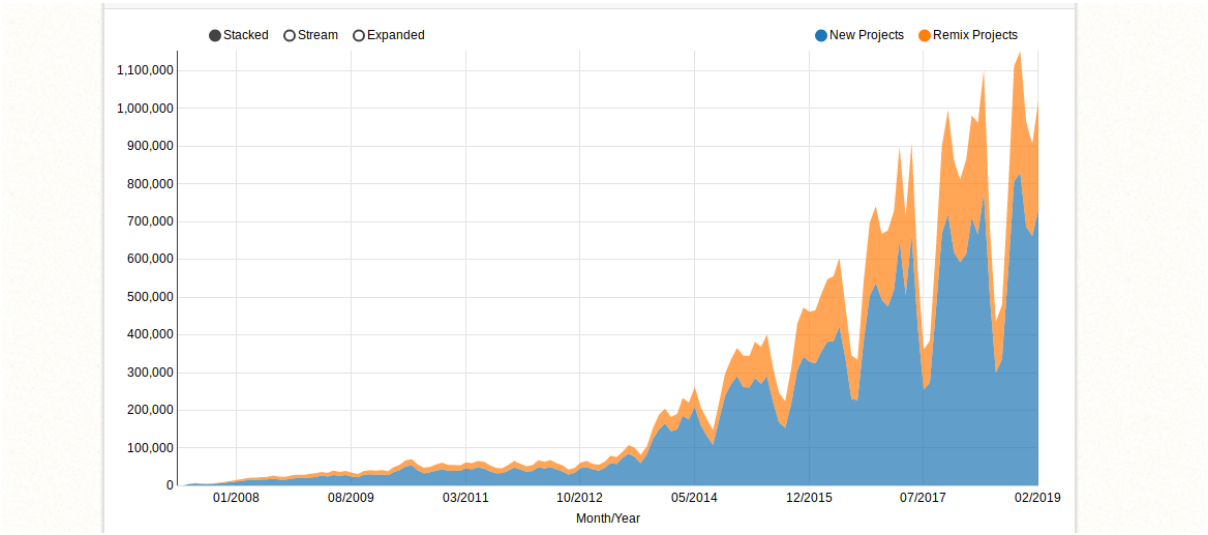

FIGURE 2.2: Amount of remixes and new projects

2.2 Some Reflections on Designing Construction Kits for Kids by Mitchel Resnick

Ten guiding principles for designing construction kits for kids::

- **Design for Designers:** People are involved the most in creating those things that are important to them or their surroundings. The goal of this principle is to develop technologies that not only engage kids in constructing tools, but also support them to create and embody their ideas.
- **Low Floor and Wide Walls:** This principle means that in order to master some kind of tools kids do not need much effort. But the frontiers for fantasy in the implementation of something is unlimited.
- **Make Powerful Ideas Salient – Not Forced:** "What is a powerful idea? In Mindstorms, Papert describes powerful ideas as ideas that "can be used as tools to think with over a lifetime."" The goal of this principle is to help children to grasp powerful ideas with the advent of a microcosm of origin. By creating a micro-world, children attach ideas from the outside world to him.
- **Support Many Paths, Many Styles:** Successful technology should be able to work for people with different approaches - hards and softs, patterners and dramatist. None of them should have a higher priority.
- **Make it as Simple as Possible – and Maybe Even Simpler:** Does it seem that it can be simpler? Is this not obvious? But there is no doubt that technological products are becoming more and more complex. One of the reasons is "creeping figure": new features in a particular industry are important. But this does not always simplify the product.
- **Choose Black Boxes Carefully:** Depending on the goals of the tools you are developing, you need to hide the functional from the user.
- **A Little Bit of Programming Goes a Long Way:** Children have no difficulty in learning how to use imperative (action-oriented) commands (for example, forward and right), simple control structures (eg, repeat), basic conditions, and simple procedural abstraction. Therefore, during the development of tools, there is a need to focus on what the children understand, convey to them the context and operate the familiar speech.
- **Give People What They Want – Not What They Ask For:** It often happens that the expectations of the product are different from the real desires of the users. During testing of the Scratch programming language, some differences were found between the "order" and the "real need".
- **Invent Things That You Would Want to Use Yourself:** The Scratch team found that we are doing a lot better work as designers when they really enjoy the use of the systems they build. By crossing their own wishes, they can design a system that will interest children. In addition, if developers like their own work, they will be motivated to develop it further.
- **Iterate, Iterate – then Iterate Again:** You always need to improve your product, never stop at the achievement

based on: *Some Reflections on Designing Construction Kits for Kidsby Mitchel Resnick*

2.3 Tynker EBook

Learning in this system is constructed in such a way that each student can independently and at its own pace. The main approach in Tynker is that children do not realize that they are learning something new during the game! The main element is the block, like in Scratch. From these blocks, there is easy to create the algorithm. Therefore, children do not need to know the syntax of some programming language, because they work with familiar actions. Only when the student is ready he can he start experimenting when writing code by switching between visual and text blocks. After the kids become familiar with the basics of programming and syntax, they can go to the full programming algorithm in different programming languages.

based on: *Coding for Kids Tynker*

2.4 Teaching Programming at Primary Schools: Visions, Experiences,and Long-Term Research Prospects by Giovanni Serafini

Studying programming (in this example, the Logo at Elementary School in Switzerland) is a didactic approach to Computational thinking, it allows you to create a certain bridge between programming and other extremes in school. The study describes the specific experience of teaching programming in the Logotype of Swiss elementary schools.In the study, the authors explore many points. I would not want to consider them all - but to stay on the most important in the context of learning.

Didactic Concept and Teaching Approach: The Logo programming language is comparable to other programming languages in the mini-language. It is great for primary school children because the instructions that children use are intuitive and do not require a rich study of synthesis. The German book *"Einfuhrung in die Programmingung mit Logo"* consists of six lessons of different levels and has many useful programming guidelines.The basics of the didact method during these classes are:

- A student himself, without constant instructions of a teacher, performs tasks - he has only a formal task and a field for the implementation of ideas
- Students without a teacher can check the correctness of the task and start solving problems independently (here the important role is played by the complexity of the algorithm being developed)
- A teacher plays an important role as an instructor who can give directions to a task, but he is not key to writing a task.

General experiments: This section tells about the general results of the experiments that occur in most schools and the conclusions of work in each individual school. I would like to devote special attention to these general features because they can be used to draw conclusions on which approaches can be considered useful in the context of my research.

- During each project, students need up to 30 minutes at the start of the 1st day to get adapted to autonomous learning.

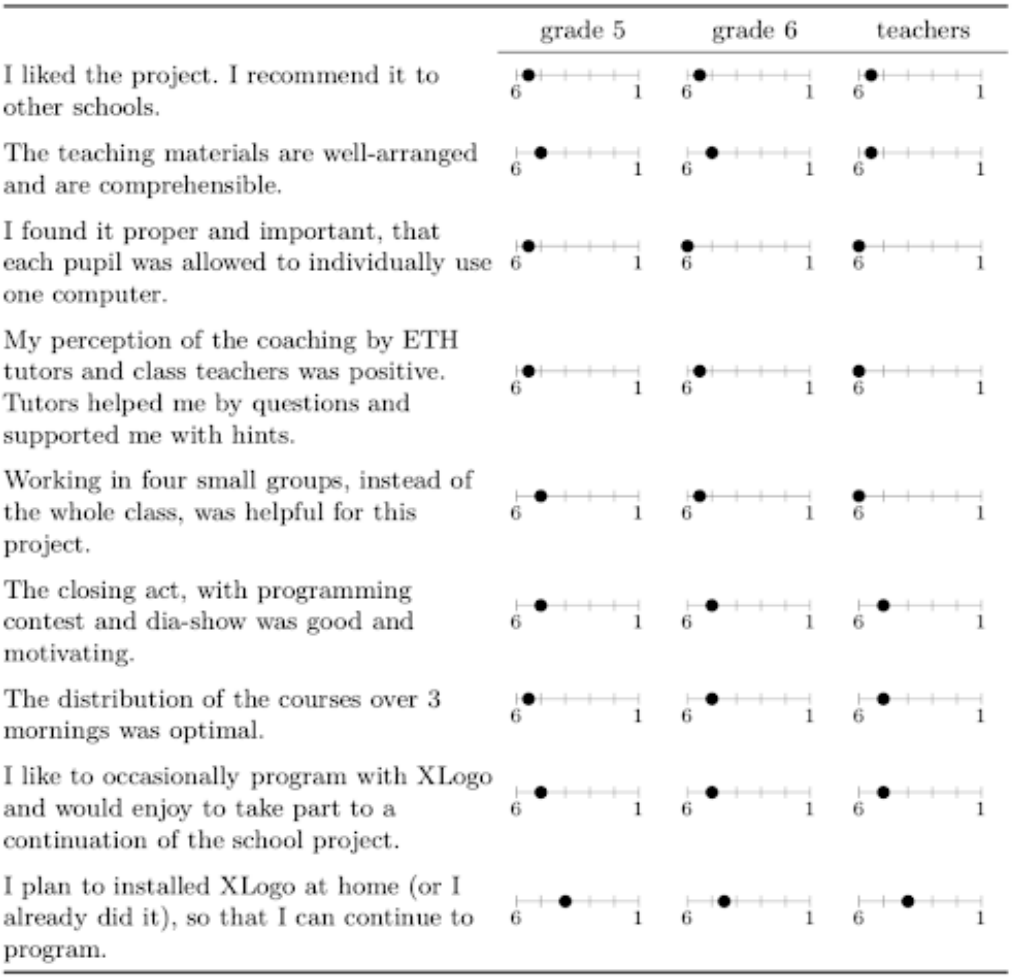

FIGURE 2.3: Pupils feedback
*Each item was graded on a scale from 1 (I totally disagree) to 6 a(I totally agree)
*16 pupils of grade 5 and 23 pupils of grade 6 were taught in 4 parallel groups during 3 half-day blocks, in the morning of March 14th, 16th, and 18th 2011.

- Pupils get quickly accepted to the Logo syntax. Typing the code by themselves is not a critical problem. From this point, we can conclude that the Logo programming language or its analogs can be applied in the development of this project.
- The pupils are really dedicated. They notice their progress and appreciate their results. For example, feedback from kids and teachers after taking classes at primary school of Attinghausen, in the Swiss Canton of Uri, near to the Saint-Gotthard Massif.
- The concept of a variable is usually the first major problem for beginner programmers of any age. Its complexity can be reduced by dividing the process into two training phases. Pupils first learn to work with a constant parameter, based on a simple abstraction of the filler. Elementary school children who have been selected for research already own this abstraction. And in the second stage, pupils will learn that the values of the parameters can change over

time and that the variable parameter is essentially related to the memory of the computer.

based on: *Teaching Programming at Primary Schools: Visions, Experiences,and Long-Term Research Prospects by Giovanni Serafini*

2.5 Scratch: Programming For All by Mitchel Resnick

Today, most people consider programming as a business for a narrow group of people. Even though it would seem to teach programming, even a child who is studying at a junior school. There are many different applications for kid's education, but in this captain, their main problems are highlighted. Such as:

- early programming languages are too complex to use (their syntax is hard for understanding).
- programming lessons are often introduced from speech, which is not interesting for those who are studying programming (that is, the children themselves).
- programming learning was often introduced in contexts where nobody could give guidance when things went wrong or encouraged deeper research when it was right.

Papert asserted that programming languages should have a "low floor" (easy to start) and "high ceilings" (the ability to create more complex projects over time). In addition, languages require "wide walls" (supporting many different types of projects so that people can have many different interests and learning styles). The builders of Scratch considered it important to make the floor even lower, and the walls were even wider while supporting the development of computational thinking. They set three basics for the design of Scratch - *"Make it more tinkerable, more meaningful, and more social than other programming environments"*. But what does each of those three basics mean?
More thinkerable: Scratch's developers watched the children make Lego blocks. They noticed that children are intuitive as to how they can form these blocks in the whole story. Therefore, developers also wanted in this programming language to achieve the same feelings and reactions from children.Regarding the design of the blocks themselves, the developers remark that the blocks which have syntactical contents (such as the loop, for example) have C-shaped shapes. This is done in order to intuitively understand that other blocks need to be placed in this C-shaped-block because it has a blank space inside.Blocks that output values are formed according to the types of values they return is different - ovals for numbers and hexagons for booleans. Conditional blocks (for example, if-else) have a hexagonal gap, indicating the Boolean is required.
More meaningful: Scratch designers point out that people are better at working on projects that matter to them. In this, they found two ways to apply in Scratch: diversity and personalization. Under the diversity means that the type of project itself can be any. For example, a game, a presentation, a project, a story, and a lot more. By personalization, they mean that a significant part of the content can be created and added by the user himself. For example, a user can add a photo of his room as a background or draw his own hero, etc.
More social: In order that any programming language existed and developed in the future, it is necessary that together with it develop a community. Scratch developers are convinced that feedbacks, "likes" and other methods of interaction with each

project can give the child the motivation to move on. For example, Scratch has the ability to post your finished project on a website. Thus, anyone on this site will be able to view the project and interact with it. Thus, knowledge and skills from one user are transferred to another.

based on: *Scratch: Programming For All by Mitchel Resnick*

Chapter 3

Experiments

3.1 Experiment hypothesis

"The format of the game takes precedence over other ways of learning programming." By way of understanding the form of assignment for children, the material mastering is faster.

3.2 Goal of the experiment

Test the hypothesis. Find the key stuff during an experiment to create the new application. In the opposite case, find the flaw in the hypothesis and the disadvantages of the approaches used in the experiments. Use the results for the prototype of the next application.

3.3 Conditions for research

- A group of 10 children aged 9 to 12 years
- All kids have previous experience in learning programming and available skills: the skills of the user PC or tablet, the experience of working with the Scratch (80%), knowledge of JS on entry level
- Time for each experimental application: 1 hour
- **Experiment format:** first of all the teacher tells the general essence of the application, the children together with the teacher begin to solve the tasks, the children after the passing of several levels continue to work independently, but in case of questions - they receive them and after passing several levels, the children continue their work independently (due to different previous experience)

3.4 Research results

The hypothesis about the effectiveness of using the format of the game in the study of programming has been confirmed. Additionally, it has been found that the use of the format of a full-fledged game is assessed by respondents (children) as "boring".
Key concepts to be used in developing the first version of the application:

- Tasks are divided into levels, where the complexity is evenly divided. No "jumps" has been applied.

- The text of the task is further supported by the visual part. That is, the task itself contains both the formal part - the question, and the result of the successful completion of the task.
- The key to doing the task is well visible to children, just like the visual component. The part for writing the code is intuitive and refers to the previous children's experience in real life. I should note that nobody has any problems understanding the process of solving the problem.
- The environment for the task has few parts. The main blocks of the game are clearly defined - they use contrasting colors, other fonts, etc. Parts for the advanced level do not fall into the eye immediately - that is, they do not have a bright image.
- The visual component of the code is also well thought out. The use of pulsing elements immediately gives the impression of an unfinished task, and its absence - about passing the level.

3.5 Comparative table

	Flexbox Froggy	Grid Garden	Codecombat
Understanding the task(1)	5	5	4
Presentation format of the tasks(2)	5	5	5
Overall impression of the presentation format *	4	4	3
Overall impression of tasks*	5	5	3
Things that distinguish students * (3)	The clarity of the tasks, the presence of the Ukrainian language, simple tasks	The presence of the Ukrainian language, understandable tasks	Interesting levels, the presence of the Ukrainian language, familiar principle
Average student score * (4)	9.7	9.2	8.2
The desire to continue (5)	5	5	1
Level of task corresponding to the level of students (6)	5	5	5
Material placement (7)	5	5	5

* students feedback

** 5 - totally agree, 1 - totally disagree

1. This estimation consisted of several parts. First of all, it included how students understood what to do in the task. As pointed out earlier - the students, along with the teacher, pass several levels and then continue on their own. That is, the first part of the evaluation is intuitive of tasks and how easy it is to use previous experience in the tasks. The second part was an estimation of the construction of the principle of work passing levels.

2. The format of the presentation is important because it depends on it: 1) whether students will perform tasks to the maximum of their capabilities; 2) whether they will master the material successfully. This estimation takes into account the first impression of children from the task (before the teacher began to do it with the students). Also included in the assessment was the intuition of the process of the task (but without the estimation of the placement of weighty blocks and UI experience)

3. Many children included the interface, characters, or overall impressions from an experiment in this block. Allocated only those things that can clarify the relationship creation and are important for the learning process.

4. The resulting score was after the anonymous survey. Children did not have any limitations or criteria for evaluation (only a maximum of 10 and a minimum of 1). In assessing the children could be guided by their individual prior experience.

5. Given that the experiment is allocated 1 hour, then each of the applications has not been completed. Children were offered to go through their homes or at least try to go through several levels at home. Grade 5 means that at least 10% of the group has expressed a desire to continue their studies. Score 1 means that none of the students has continued to go through the levels at home.

6. In evaluating this part, how easy it is for children with different experiences to pass the application level is easy. The important thing was whether it was easy for them to understand the tasks and understand how to write it without the help of a teacher. None of the participating students used the applications in the experiments.

7. In my opinion, this evaluation block is the most important one. Its importance proves that one of the goals of the experiment was to find key things in each application. There is also a dependence between the interface and the understanding of information. During evaluating this block, questions were asked to the children in order to understand whether they understand the interface and work principle. For example, it was suggested to explain what some part of the application means, or how to move from one part of the game to another.

3.6 Why are these results important?

The most important part of these experiments was to understand what existing things in “virtual world” help children learn. Everyone knows the statistics of how much time the children spend on a computer. The purpose of this study was to discover, but what such an attractive can be taken from existing applications and used

or/and modified for use in a full-fledged task at school. This study showed both quite obvious results and quite unexpected. At the moment, there are a lot of solutions which are at the intersection of childhood education, childhood hobby and children's pastime for the computer. But there are few studies available within the school. In addition, it's worth noting that the time is changing at fantastic speeds. And this is especially true for children who absorb knowledge and solutions of the world today.

Chapter 4

The proposed solution

Our most important task is to create an education solution that will be most understandable to children and useful for their learning. Also, we must not ignore the important part of this study - to identify the key elements of the perfect solution. We present a new application, which is the result of the collected practices and approaches after the review of researches and experiments. It is worth noting that the proposed solution satisfies the conditions for teaching programming in the Ukrainian school for middle and upper secondary school children. The results of the experiment conducted on the basis of this solution are here.

4.1 Components of solution

As previously mentioned, one of the goals of this work is to identify the fundamental elements of the application. Methods for discovering this were experiments with children. An important contribution to this part was made by observing the behavior of children in these experiments and their feedbacks. Especially important was the part of the "Material placement" in which the children answered questions about the interfaces and at the same time made clear how they understood the parts of the program.

1. Animated elements to carry the state of the game at the moment.

   The use of animation not accidental is the first element I have selected for a successful application. The animation itself gives the user an insight into the picture of the world in which they got into the game. The first time they enter the scene, the animation gets their attention. The pulsing frog (in case of Flexbox froggy) "tells" that it expects that it will be programmed for a solution. It remains in this state until the user "brings" it to the next level. In my opinion, such an animation creates a two-way interaction between the user and the program.

2. Bright colors to get attention.

   The game is designed in such a way that all the blocks needed for it are well visible, and all the advanced levels are not noticeable. This strategy of using colors as a prompt is not new. It is important here to understand that children will not read the names of the buttons. All blocks should be intuitive and rely on previous experience. That is why the "Next" button is right below the input field. It changes its color (which attracts attention) if the level is resolved correctly. Let's take a look at the two examples below. They differ in the color of

FIGURE 4.1: Disabled Next button

FIGURE 4.2: Active Next button

the button (the real case in the game). In the screenshot above, the impression is that the button is not available. And in the photo below, it seems that the button can be pressed.

3. Use translucent color for disabled elements and non-important or advanced parts of application. (See example below)

4. An intuitive task even without reading the task itself.

   Have you guessed what to do if your task looks like a picture 4.3 ? Right! You need to place the frogs so that each frog is sitting on a leaf of its own color. This is the advantage of this application. Children do not need to read the assignment since the visual part of the tool hints it.

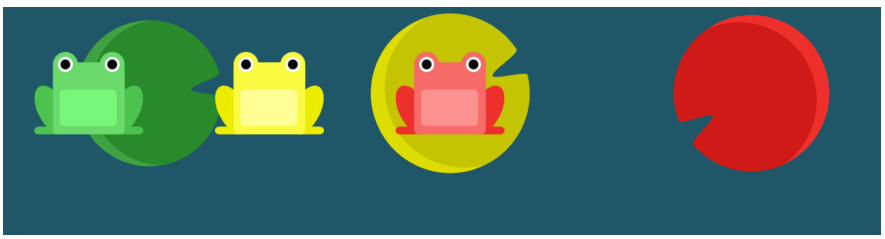

FIGURE 4.3: Visual part of task in FlexBox Froggy

5. The writing field uses the same color principle as mentioned above. It is clear where to write because there is an analogy with the input field, which is everywhere - a link to the previous experience.

6. Referring to a review of related works I found another indicator. I will take it as a backbone. To avoid repeating, I want to summarize what is important in the context of developing a new application. Children have to see things intuitive for them. We should use that, what surrounds kids in everyday life. For example, an analogy with blocks of lego or puzzles. It is easy and it is obvious.

4.2 Unique parts of the solution

Once I have identified the key elements of the solution - it's worth getting away from the disadvantages. During the experiments, I discovered some things that made me change the logic of the proposed solution.

First of all, I want to highlight that these details are related to children in middle and high school age. Since the applications used during the experiments were designed for a large-scale audience, we identified during experiments some limitations. It is worth reminding once again that we are considering solutions in the context of an application for full-time learning (for a particular subject, for example, computer science at school). Due to the fact that experimental applications were "hit" into a large target audience, it turned out that this smaller group is not in the field of interest of tools. That is, children could understand them as an alternative to playing computer or as an alternative to boring homework, but not as an application for full-time education.

I also want to note that tested applications do not have the ability to customize for different tasks. For example, changing the flex in the "Flex Box Froggy" will not turn out easy. Since the logic of writing this task is designated under the flex, and globally - only for the study of CSS styles.

1. Children do not like to read the task
   It's time to admit that children do not read the task carefully. As I mentioned earlier in paragraph 4 of the previous section - the intuitive interface is important. And I even highlighted this item as a key point to the future application. But we should understand that not every task can be correctly given in the animation or visual form.
   When comparing heavy and light levels of tools during an experiment, we can conclude that:

   - At simple levels, the children did not spend time reading the task (the exception are first one)
   - On difficult tasks, they did not have an understanding of the animation that needs to be done in the task. Therefore, they tried to read the task from another part of the application and then they had difficulties.

   To put it simply, the children at first had a connection with previous experiences in their lives. Then, this experience did not suffice for the interpretation of new tasks, and this meant attracting third-party assistance.

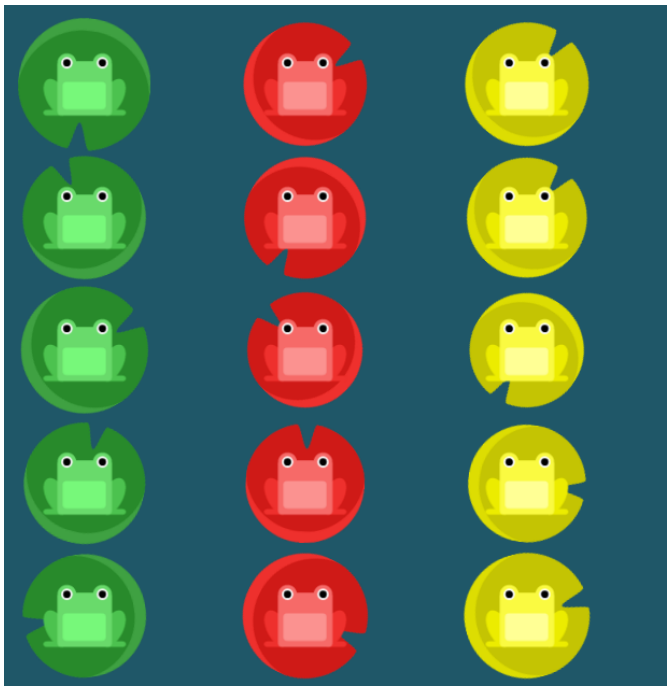

FIGURE 4.4: Heightening animation

2. Children do not type text well
   It would seem that this statement is strange. Everyone knows that children play computer games or that they are much closer to modern technology than their parents. But surprisingly, the kids are badly typing on the keyboard. In my opinion, this is due to the fact that their interaction with the keyboard is limited to a few buttons (in terms of games). That is, the interaction with the computer is well-tuned and does not require knowledge of the keyboard layout.

3. Too many animations distract
   Simply imagine that all these animation objects in picture 4.4 are moving. (Although, it takes place in the real game). This reminds you of a situation where you try to focus on the task, and someone is talking around you. If this is one person, then you can not pay attention to it. Now imagine that you are in a crowd where everyone speaks.
   Like an example above, in my analogy with the crowd, children were distracted by the lake of "pulsating" frogs.

4. If the way of presenting the tasks does not change, the process of passing the levels becomes automatic
   If you use the form of a game during learning, then the main point for children becomes "discover what's next." In this approach, children are passing level by level, trying to do it as quickly as possible. Also, it's worth not ignoring that children do not read the task well. They use the experience achieved earlier. For example, a student in a previous task copied a bold text in a task and passed the level. At the next level, they will try to do it again.

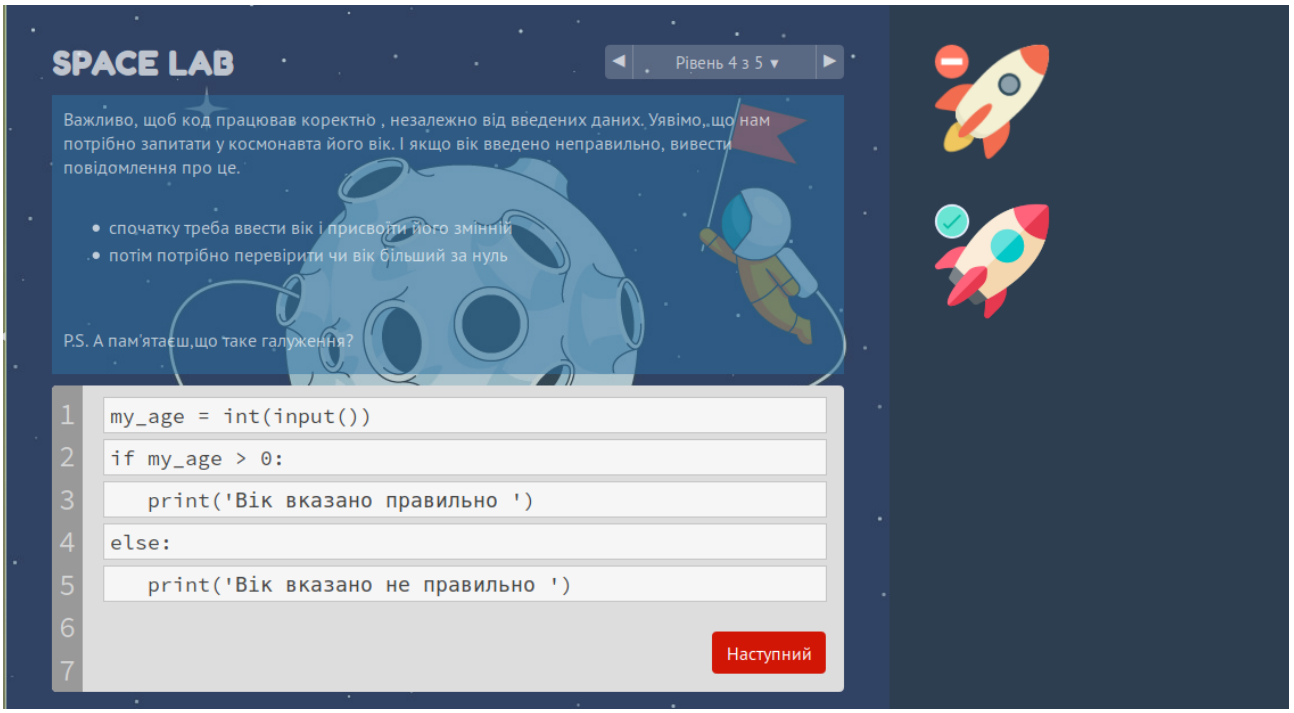

FIGURE 4.5: Created tool for teaching IT

4.3 Details of created tool

Using the results of previous studies and related work, we made an application. We can talk about the created application for a very long time. But we would like to focus on the 3 most important (different from other applications) things.

1. The main topic.
   It was important to be interesting to children who represented our target audience. After a long reflection, the topic of space was chosen. We invented a legend in which a student should help a team launch a rocket. They need to do this by performing the tasks that are in the application.

2. One of the most important things is the process of solving tasks.
   * Of course, the tasks format is very important. But we proceed from the considerations that the representation of tasks can be changed.
   Before we thought about solving the problem of "writing a code", we appreciated a lot of possible options. We decided to reject the approach in which the student writes the code. Reasons for this are described in the section above. The next idea was an example of puzzles, but we rated it as too simple (boring for kids from our target audience) and one that could confuse students. Since particle puzzles are like things with the same purpose, and not all parts are identical in code writing. For example, the condition of the cycle and the body of the cycle is not a puzzle, it is an attachment or dependence. The analogy with the puzzle can give a non-accurate representation of the code. We focus on an example with simple blocks that need to be replaced. Obviously, this solution satisfies all the conditions that Mitchel Resnick described (chapter 2). It's simple and bound with previous experience.

3. Animation.
   I had my personal desire to create the application where an animation that presents everything that's done in the code. Of course, as much as it's possible. It means that if there is a task in which the loop occurs, then the animation should simulate its behavior. Running forward, the children also appreciated the animation positively in the application and highlighted it in the TOP-3 best things in the application.

4.4 Testing the chosen solution

4.4.1 Conditions for research

- A group of 10 children aged 13 to 16 years
- Children have previous experience in computer programming at school. Technologies studied: Python, Scratch.
- Time for testing application: 45 minutes
- **Testing format:** The format in which the application was used is the exam. That is, the students studied the material in their usual format, finished their homework. After some time, the teacher used this tool to check how the children understood and memorized the material.

In order to maintain maximum objectivity, we will analyze this application in the same way as others in chapter 3. Also, one more, which is a review of the children about their experience of using the application in the school, is added to this table.

4.4.2 Comparative table

	Proposed solution
Understanding the task(1)	4.53
Presentation format of the tasks(2)	5
Overall impression of the presentation format	5
Overall impression of tasks	5
Things that distinguish students (3)	Animation, colors, simple tasks
Average student score	9.0
The desire to continue (4)	9.3/10
Level of task corresponding to the level of students	5
Material placement	5

The study was conducted among children of lyceum aged 13 to 16 years. Children studying in the 8th form - 30.8%, students of the 9th grade - 57.7%, students of the 10th form - 11.5%. The total number of children is 52.
After completing the tasks, the children had the opportunity to fill out an anonymous form in which they answered different questions.

1. 86.5% of children indicated that they understood the task. 3.8% of the children answered that they understood only some of the questions. And accordingly,

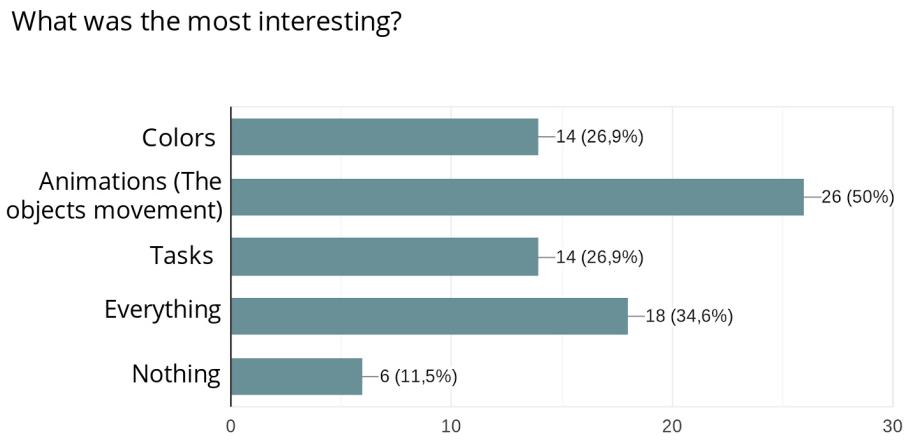

FIGURE 4.6: Student Survey Results - What's Most Interesting?

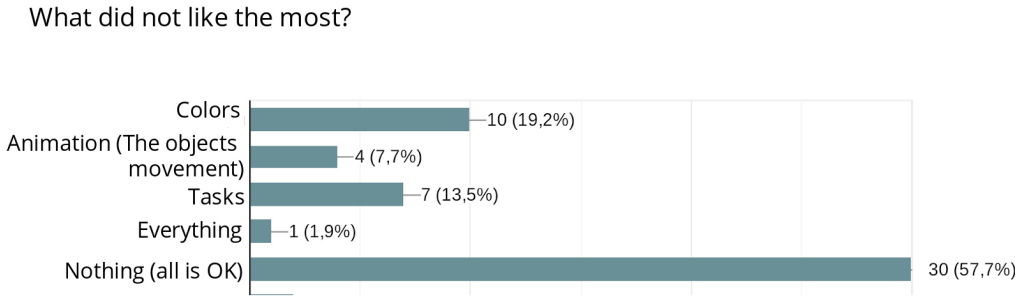

FIGURE 4.7: Student Survey Results - What did not like the most?

9.6% of respondents said that the tasks were not clear to them. In the same section below will be a review of the teacher's commentary, which will explain that it was not clear to the children.

2. There was no specific question about the format of the tasks because the form was intended to get an honest answer about the children's impression of the application. Too many questions could be made so that students would not give a true answer. Therefore, many of the items we are considering in the table are combined in the form in a single question. For example, the conclusion of the figures 4.6 and 4.7 can be done as follows:

   - children like the application
   - the format of information they also satisfied

   The numbers speak for themselves.

3. As previously mentioned, children were given an opportunity to appreciate what things they most like and do not like. It is deserving noting that the choice was plural.

4. 51.9% of respondents indicated that they would like to continue to complete tasks in this format. 40.4% said they would like to do homework or classwork in this tool but not at all subjects. And only 7.7% of children expressed their reluctance to do the assignment so.

It should be highlighted that the difference between the results may be due to the fact that the groups have different practice and are from different age groups. Therefore, the following table, which expresses the attitude and overall impression of children from the application, is more presentable.
Also, it should be indicated that this group that participated in the testing above represents our target audience. The final way of using the application will be the same or really close to what is being tested.

A review of my supervisor and teacher of computer science in children in which I was testing:

> The students already started to work, the idea would already have some feedback. In general - they liked it. And I liked to join the serious business. Perhaps there will be ideas in the course of the day.
> Level 1: In a task, it is necessary to reformulate: not to "print this variable", but "to display the word Hello and this variable"
> At level 3 I would advise stopping the rocket after reaching the number 8. Or start the cycle counting again (if you leave an infinite animation).
> Levels 4 and 5, may indicate missiles with numbers to make it clearer that they relate to different parts of the team conditions?
> And the general one - it is worth a bit to reduce the transparency of the block with the text - a little hard to read and interrupts the image of the background. There's an opacity of 0.5, maybe 0.7-0.8.
> And as P.S. - a cool idea with rockets-space. Potentially "unfolds" on very interesting and diverse tasks!

And some reviews from children (punctuation is saved)

"The awesome thing is better than just writing codes"
"Good animation and project implementation. Very interesting website for renewal of gathered information) Happiness in the following achievements)"
"Cool tasks ... In the form of the game is interesting."

4.5 Technical part

The application was based on "Flexbox Froggy". From the outside, they may seem similar, but in reality, many parts of the code are rewritten. It is worth noting in advance that only the "skeleton" of code was used. That is the construction and contents of the documents and the process of their interaction.
First I would like to tell you what exactly was changed in the code, and then go through each of these documents separately.

- The principle behind "Flexbox Froggy" logic is flex styles and their use. Any code that the student wrote has been applied to animated frogs. Then was checking if placing of the frogs is the same as if the student wrote the code as in the solution document.
  It was also a check if the length of the code written by the student is equal to the length of the solution code.
  Given that we do not use the idea of writing code, this check did not satisfy our method. We have lines of code that which are shuffled every time before loading the level. Each time it is checked that all lines are "in the right place".
- Again, going back to check if the student wrote the code correctly. In the case of Flexbox Froggy: when checking the correct code execution applied styles and the student sees the "animation". Although in reality, this is not a comprehensive animation.
  Given our approach, we had to re-write the "successful scenario". Each task has its own animation. For each one, it was necessary to describe it in the code and add additional elements.So we created real animation in the application.
- And of course we had to write all the tasks and an explanation to the code.

And now more about individual documents. It is important that each one of them is more or less rewritten or appended.

- index.html - frame of the document, only elements and connection of other documents.
- docs.js - here is an explanation for the code (the items in the task are highlighted).
- dragAndDrop.js - a function that is executed when a student moves blocks in a task.
- game.js - the most important document that contains all the events in the program. Here is the checking process, loading levels, animation, etc.
- levels.js - all information about the level - name, task, solution, etc. Static information that is only used in the game.js.

Chapter 5

Conclusions

This was one more work on the implementation of modern technologies in approaches to educating children at school. During this research, we were responsible to develop an application that was tested by pupils at school. Its success can be confirmed by a survey and, accordingly, positive feedback from students. But the most important part of this bachelor thesis is the determination of key details of a successful educational tool. We were not only able to investigate and analyze previous research. We explored existing applications, evaluated the reactions of children and identified the fundamental elements that represented in real (or better, digital) life.

Bibliography

All I Really Need to Know (About Creative Thinking) ILearned (By Studying How Children Learn) in Kinder-garten by Mitchel Resnick. https://web.media.mit.edu/~mres/papers/kindergarten-learning-approach.pdf.

Coding for Kids Tynker. https://www.tynker.com/content/Coding-for-Kids-eBook.pdf.

Scratch: Programming For All by Mitchel Resnick. https://m-cacm.acm.org/magazines/2009/11/48421-scratch-programming-for-all/fulltext?mobile=true.

Some Reflections on Designing Construction Kits for Kidsby Mitchel Resnick. https://web.media.mit.edu/~mres/papers/IDC-2005.pdf.

Teaching Programming at Primary Schools: Visions, Experiences,and Long-Term Research Prospects by Giovanni Serafini. https://link.springer.com/chapter/10.1007/978-3-642-24722-4_13.