**T.C.**

**BAHÇEŞEHİR UNIVERSITY**

**FACULTY OF ENGINEERING AND NATURAL SCIENCES**

**Programmable toy - proto**

**Capstone Project Final Report**

**FURKAN BAYSAN (CMP)**

**MUSTAFA FUTTU (CMP)**

**UFUK ORHAN (EEE)**

**BERK SAÇKARA (CMP)**

**MUHAMMET SÜMBÜL (EEE)**

**Advisors:**

**DR ANDREW J. BEDDALL (EEE)**

**DR GÖRKEM KAR (CMP)**

**ISTANBUL, September 2020**

**STUDENT DECLARATION**

By submitting this report, as partial fulfilment of the requirements of the Capstone course, the students promise on penalty of failure of the course that

* they have given credit to and declared (by citation), any work that is not their own (e.g. parts of the report that is copied/pasted from the Internet, design or construction performed by another person, etc.);
* they have not received unpermitted aid for the project design, construction, report or presentation;
* they have not falsely assigned credit for work to another student in the group, and not take credit for work done by another student in the group.

# ABSTRACT

PROGRAMMABLE TOY

Furkan Baysan (CMP)

Mustafa Futtu (CMP)

Ufuk Orhan (EEE)

Berk Saçkara (CMP)

Muhammet Sümbül (EEE)

Faculty of Engineering and Natural Sciences

Advisor:

Dr Andrew Beddall (EEE)

Dr Gorkem Kar (CMP)

December 2020

The era we are in gives people a chance to control extremely various objects in their lives. But how can it be possible? The right answer is “Languages”, the languages of those objects. By learning that, people might have this chance. If we desire to spread this ability to control devices and the ability to manage them then our target user should be people who can easily adapt and embrace this ability by a quick learning step. That’s because we have chosen kids as our users. According to appropriate tests and experiments, it has been proven that children have a greater ability to learn than adults. That's why we decided that users should be kids. Thus, we can teach them the basics of programming with a toy easily.

In this project, we created a programming language and a device to control. Our main goal was creating a path for users to understand how to write a code with a programming language. But on the other hand, we believe that these languages must be as simple as possible; understandable easily, and containing core features of a programming language. The Project contains two parts: A programming language and a toy to either attract the kids’ attention and show them how a language controls a physical device. To be clear, the device has barrels on it and each barrel can be controlled by appropriate blocks of codes. The programming language has a simple syntax and with this syntax basics of programming can be taught. Also, we designed and created an interface that simulates barrels and the interface has a text area to write all of the codes.

# TABLE OF CONTENTS

ABSTRACT iii

TABLE OF CONTENTS iv

LIST OF TABLES vi

LIST OF FIGURES vi

LIST OF ABBREVIATIONS viii

1. OVERVIEW 1

1.1. Identification of the need 1

1.2. Definition of the problem 1

1.3. Standards and constraints 2

1.4. Conceptual solutions 2

1.4.1. Barrels (x6) 4

1.4.2. Digital Screen 4

1.5. Physical architecture 5

2. WORK PLAN 7

2.1. Work Breakdown Structure (WBS) 7

2.2 Responsibility Matrix (RM) 9

2.3 Project Network (PN) 10

2.4. Gantt chart 10

3. DESIGN PROCESS 11

3.1. Computer Engineering 11

3.1.1. Definition of the problem 11

3.1.2. Review of technologies and methods 11

3.1.3. Standards and constraints 12

3.1.4. Conceptualization 12

3.1.5. Risk assessment 13

3.1.6. Prototypes 14

3.1.7. Materialization 16

3.1.8. Evaluation 23

3.2. Electrical and Electronics Engineering 24

3.2.1. Definition of the problem 24

3.2.2. Review of technologies and methods 24

3.2.3. Standards and constraints 25

3.2.4. Conceptualization 25

3.2.5. Physical architecture 27

3.2.6. Risk assessment 29

3.2.7. Materialization 29

3.2.8. Evaluation 31

4. COSTS 32

5. CONCLUSION 32

ACKNOWLEDGEMENT 33

REFERENCES 34

APPENDIX A 35

APPENDIX B 39

# LIST OF TABLES

[Table 1. Responsibility matrix for the team 9](#_Toc60330481)

[Table 2. Example of a Use-Case scenario 11](#_Toc60330482)

[Table 3. Risk Assessment Tables 13](#_Toc60330483)

[Table 4. Risk Matrix from lecture notes 14](#_Toc60330484)

[Table 5. Table of comparison 25](#_Toc60330485)

[Table 6. Table of comparison 26](#_Toc60330486)

[Table 7. Table of comparison, 26](#_Toc60330487)

[Table 8. Risk Assessment Tables 29](#_Toc60330488)

[Table 9 Costs 32](#_Toc60330489)

# LIST OF FIGURES

Figure 1. Example code block from website scratch platform 3

Figure 2. Visual from Code Combat website 3

Figure 3. Barrels representation 4

Figure 4. LED Screen 4

Figure 5. Visualization of physical architecture 5

Figure 6. Visualization of physical architecture 6

Figure 7. WBS for the project 7

Figure 8. Explanation of the WBS 8

Figure 9. Project network 10

Figure 10. Timeline of the project 10

Figure 11. Prototype 1.0 14

Figure 12. Prototype 2.0 15

Figure 13. Prototype 2.1 16

Figure 14. txt file which contains all properties of proggraming language 18

Figure 15. Diagram which shows the execution steps of algorithm 19

Figure 16. Old prototype wich implemented with Swing library 20

Figure 17. New GUI with JavaFX library 20

Figure 18. Login Screen 21

Figure 19. Structure of single (Byte) barrel input 22

Figure 20. Hardware components and their connection 27

Figure 21. Motors and Cylinders 28

Figure 22. Final barrels 30

# LIST OF ABBREVIATIONS

*GUI Graphical User Interface*

*PL Programming Language*

*PT Programmable Toy*

*V Voltage*

*GND Ground*

*3D 3 Dimensional*

# 1. OVERVIEW

This project aims to make it easier for people to adapt to and control new technologies. And the target audience is children who are the source of new generations. Therefore, the project consists of two parts: a new basic programming language as the software and a physical toy that attracts children as the hardware part. Tasks are distributed into three main objectives which are “Creating a User Interface and Programming Language”, “Constructing a Toy Structure” and “ Establishing a Connection between these two parts”.

## 1.1. Identification of the need

The years human-beings live in are changing day by day. Technology affects and leads the lives. In order to handle this technology, it is necessary to know its language. In these consequences, there must be adaptation progress of human beings as well as all the times they have. This project aims at children whose progress of adaptation is better than adults'. New generations can be familiar with the logic of the programming in this enjoying and visualized way.

## 1.2. Definition of the problem

Both coding and programming have been involving and more affecting people’s lives every passing day. However, as much as we observed; people are not aware of the importance of coding and how it will be affecting our future. Also, old generations are not aware of it either. But the most important thing here is “youth” and “children” which are specified as “millennial generation”.

Nowadays many elementary schools have programming courses. As mentioned before we need to find a more enjoyable and physical way because the art of programming can be intangible, difficult, and boring in the minds of young ages.

So in order to make children familiar with coding and programming, also to make them love coding we designed a system which has an interface – helps children to understand the basic of coding entertainingly-. Then we created a programmable toy for educational entertainment.

## 1.3. Standards and constraints

This product provides society with an understanding of the importance of coding. Also increasing the awareness of it may shape the structure of society in terms of economical conditions.

Because of the target audience, the programming language had a constraint which is accessibility and ease of use.

The project has to be completed in a given time period and budget limit. In order to satisfy these constraints, the team should be loyal to the work plan and weekly tasks. Due to the COVID-19 pandemic and its consequences, we may use different communication methods. And the cost of the product should be reasonable.

In this case, we did not encounter the pandemic, testing, and reporting the results might be slightly easier than the current situation. So, for the integration part of the project, it is highly important to see any feedback by the physical structure when the codes are written.

## 1.4. Conceptual solutions

There is a couple of websites which helps to teach programming idea and structure to children. For example, a website called "Scratch" trying to implement the idea of creating algorithms with block structure. In the block, the structure user needs to put function blocks in a meaningful order to reach the desired solution.

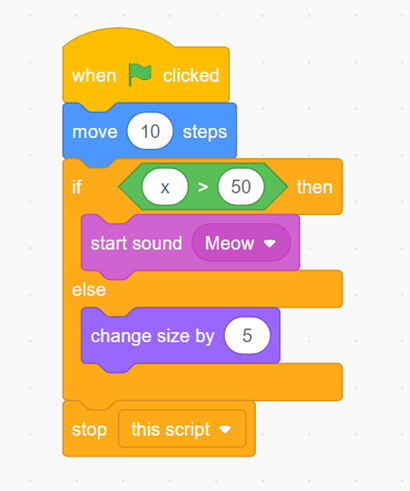


Figure 1. Example code block from website scratch platform

And some websites like “Code Combat” [1] teaches programming via games. In these kinds of games user - a child in these cases - needs to type basic code lines to give commands to the target character.



Figure 2. Visual from Code Combat website

This project offers its users an enjoying interface and a toy which is connected to this interface inside. Users can control the toy by writing some specific codes and instructions. In this way, users will be familiar with the logic of the programming languages and how strong talking their language helping them.

And also from the hardware side, we thought of different approaches like

### 1.4.1. Barrels (x6)



Figure 3. Barrels representation

### 1.4.2. Digital Screen

It is a basic led screen that shows 6 or 8 digits representing the hour, minute, and second respectively.



Figure 4. LED Screen

However, as barrels are easy and cheap to build, moreover barrels fit for 8 digits; we chose barrels to use in our product.

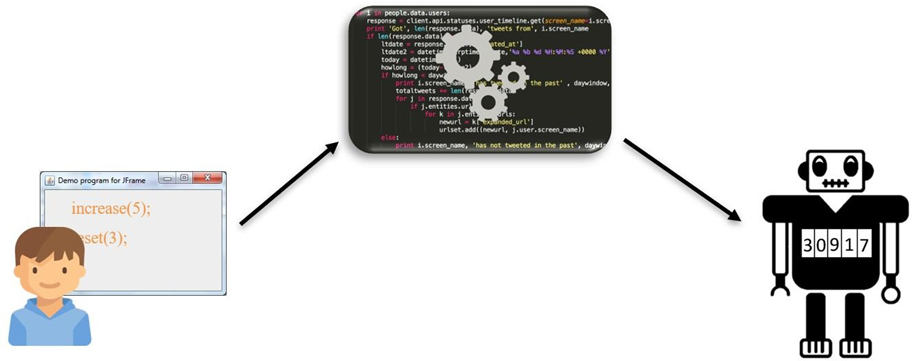


Figure 5. Visualization of physical architecture

## 1.5. Physical architecture

This project has three parts which are a toy application, programming language, and a toy.

**User console (Toy Application):** Interface which visible to the user and place that user uses to manipulate toy. Users can also write codes of programming language here.

**Programming language:** Collection of simple functions and logic operations. The user will write special functions in the special language of the toy.

**Toy:** Physical machine which has rotating barrels. Barrels are going to be manipulated by user code.

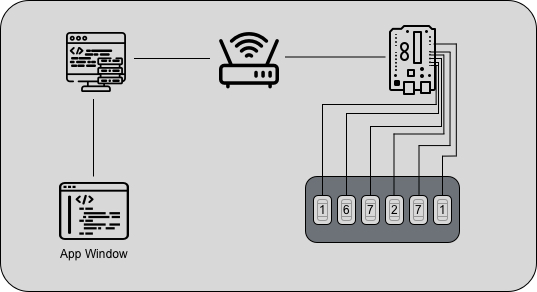


Figure 6. Visualization of physical architecture

# 2. WORK PLAN

## 2.1. Work Breakdown Structure (WBS)

1. **PROGRAMMABLE TOY**
   1. **Toy Application**
      1. ***User Interface***
         1. Design of User Interface
         2. Implementation of User Interface
      2. ***Programming Language***
         1. Determine and implement proper barrel functions
         2. Implement programming language
   2. **Physical Toy**
      1. ***Microcontroller***
         1. Order/Manufacture Parts
         2. Assemble and Circuit
      2. ***Implementation of Toy Hardware***
         1. Order/Manufacture Parts
         2. Assemble Manufacture
   3. **System Interface (Connections between toy and application)**

Figure 7. WBS for the project

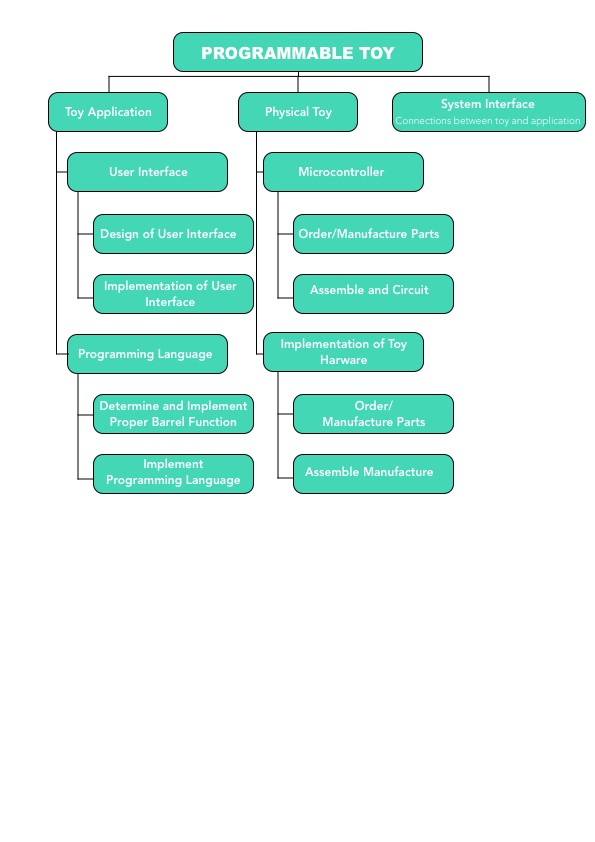


Figure 8. Explanation of the WBS

## 2.2 Responsibility Matrix (RM)

Table 1. Responsibility matrix for the team



Mustafa, Berk, and Furkan are from the CMP department. Their duties are to produce the software and to design programming language, language operations, and functions. Ufuk and Muhammed are from the EEE department and their duties are to implement the toy hardware and design of the circuit and microcontroller connections. Also, details are explained in the table above.

## 2.3 Project Network (PN)

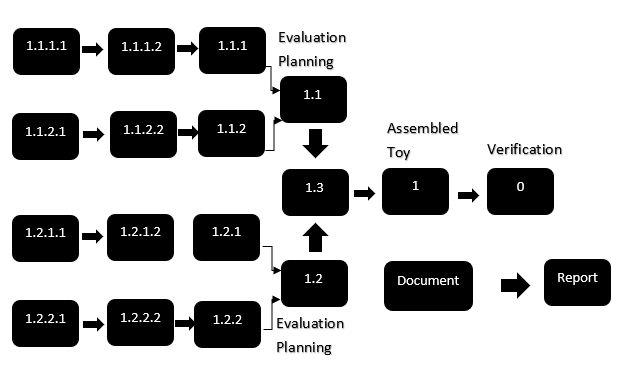


Figure 9. Project network

## 2.4. Gantt chart

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Week*** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| *Mustafa* | 1.1.2.1 | | 1.1.2.2 | | | | | 1,3 | |  |  |  |  |  |
| *Berk* | 1.1.2.1 | | 1.1.2.2 | | | | | 1,3 | |  |  |  |  |  |
| *Furkan* | 1.1.1.1 | 1.1.1.2 | | | 1.1.2.2 |  | 1.1.2.2 | 1,3 | |  |  |  |  |  |
| *Ufuk* | 1.2.2.1 | | 1.2.2.2 | | | | | 1,3 | |  |  |  |  |  |
| *Muhammet* | 1.2.1.1 | | 1.2.1.2 | | | | | 1,3 | |  |  |  |  |  |
| *Integration* |  |  |  |  |  |  |  |  | 1 | |  |  |  |  |
| *Verification* |  |  |  |  |  |  |  |  |  |  | 0 | |  |  |
| *Documentation* |  |  |  |  |  |  |  |  |  |  |  | Draft | Report |  |
| *Presentation* |  |  |  |  |  |  |  |  |  |  |  |  | Draft | Present |

Figure 10. Timeline of the project

# 3. DESIGN PROCESS

## 3.1. Computer Engineering

### 3.1.1. Definition of the problem

There is a technology in the World that affects the lives of people directly, and this developing technology continues to be a bigger part of their lives day by day. The tendency of humanity to adapt to this development is generally in the form of consumption. This situation causes certain kinds of drawbacks in society, especially for young ages.

There is another point to be mentioned. Harlow et al. stated that while these technological developments keep going, a significant number of children do not meet programming which will be able to give them a chance to handle this technology, before high schools [4].

This project aims to convert a certain rate of younger users of the technology into people who will be able to handle and manage the technology.

Table 2. Example of a Use-Case scenario

|  |  |
| --- | --- |
| **Use-case Name** | Change state of barrels |
| **Use-case Description** | The normal procedure of allocating wanted numbers to toy |
| **Actors** | Child |
| **Pre-Condition** | The toy must be connected to the computer |
| **Post-Condition** | - |
| **Normal Flow** | **Step 1:** User opens the application of toy  **Step 2:** User enters the right codes to the writing area.  **Step 3:** System understands and converts entered code for hardware.  **Step 4:** User can observe changes in barrels of the toy. |
| **Alternative Flow** | **Step 2 Alternative:** If the user's code contains syntax errors program  will warn the user |

### 3.1.2. Review of technologies and methods

The Programmable Toy project has two main parts as we mentioned before. Software and hardware part. In order to handle the software side, we decided to use Oracle’s JAVA programming language. Java is useful and flexible enough for this project. We will use the library "Swing" for the graphical user interface. Then programming language will be implemented with JAVA too. Library "Vector "and some string functions will be used to divide, analyze, and convert Toy's language to java language. And also Array lists will store divided user codes before converting them one by one. These technologies and methods result in a desktop application for the toy.

### 3.1.3. Standards and constraints

The quality of the toy application is related to usability, efficient time usage, and completion of the budget. The application part of the project has no budget cost but coding must be done in a certain amount of time. In order to proceed safely and complete the project in time, some code standards must be declared.

**Code Standards:**

Coding Standards help the team to write safer and more understandable code. We can use Oracle's "Java Code Conventions" [2] which is published on September 12, 1997. Toy application does not require any internet or identity from the user so children can use it safely. And also there is no violation in any engineering moral theory. The application does not attack any right of an individual or society and lawful.

**GUI Standards:**

The application interface must be simply enjoyed as much as possible. In order to create a user-friendly GUI, AMERITECH's "Ameritech Graphical User Interface Standards and Design Guidelines" were adopted [3]. Created programming language must be also simple for young ages. Language should require a minimum amount of syntax rules and must be short.

Application and programming language might help to improve the development level of the individual. That results seem beneficial in the social aspect.

### 3.1.4. Conceptualization

**Designing Interface:**

First of all, considering the target mass which is kids in this Project causes certain kinds of difficulties for developers which might be easily distracted the attention of kids. This factor might be considered as a benefit at the beginning. Creating well-designed and cartoon-like interfaces might provide beneficial results in the start. But in the late stages, these themes of the interface can be significantly distracting. In order to get rid of this possible drawback, it is suggested to create a user-based level system. This level system is required to create an account and system stores using the time of the application. According to using times, the application changes its theme to less complicated and attractive ones.

**Programming Language:**

This Project offers users a new basic and understandable programming language. Even though the Project seems to limit this language with the barrels, the language which is developed provides users to go details of abstracts of programming. It is planned to add new features extra.

### 3.1.5. Risk assessment

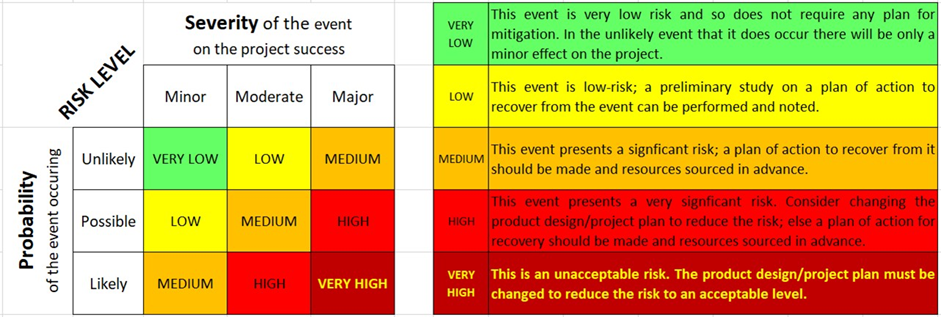
Table 3. Risk Assessment Tables

|  |  |
| --- | --- |
| **Task** | Design of application |
| **Associated**  **Risk** | User may found interface complicated and boring. As a result, product usability reduces and its purpose can't be ensured. |
| **Severity** | Critical |
| **Probability** | Medium |
| **Risk Score** | High |
| **Management of the Risk** | In order to prevent risk, GUI must be simple and enjoyable.  Created designs can be voted by children or multiple themes can be added. |

|  |  |
| --- | --- |
| **Task** | User coding |
| **Associated**  **Risk** | User needs to know the programming language of Toy. If the user doesn't know that, s/he cant use product. |
| **Severity** | Critical |
| **Probability** | Medium |
| **Risk Score** | High |
| **Management of the Risk** | There can be information or teaching section in the application  and at the first use, the program can direct the user to this page. Mini code booklet can be created too. |

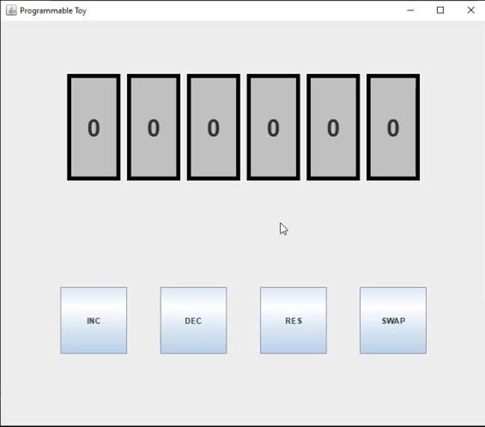
|  |  |
| --- | --- |
| **Task** | Runtime error handling |
| **Associated**  **Risk** | The application might be bugged and crashed like any other  software. |
| **Severity** | Critical |
| **Probability** | Low |
| **Risk Score** | Medium |
| **Management of the Risk** | To minimize application crashes, the application can be implemented through safer code design. Every user input variation  must be considered. |

Table 4. Risk Matrix from lecture notes



### 3.1.6. Prototypes

**Prototype 1.0:**

This is our first prototype which is created in the first weeks. In this prototype, the user can select barrels by clicking on them. After then he can manipulate selected barrels by pushing function buttons. For the swap function user have to select only 2 barrels, otherwise, the program prints an error.

In this Figure, we presented our first

prototype which is created by us to

work with and ease the test.

Figure 11. Prototype 1.0

1. **Prototype 2.0:**

In the second work, we added "set" and "randomize" functions to the prototype. Users can select barrels and give them random values. If the user pushes the "switch" button, function buttons change into code writing area.

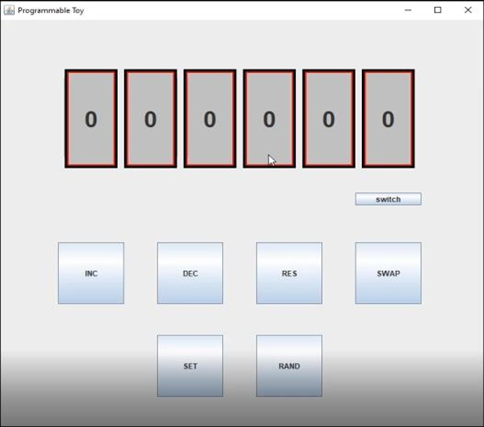


Figure 12. Prototype 2.0

This figure shows the second prototype with added new buttons.

**Prototype 2.1:**

Now user can perform basic functions which can be found at prototype 2.0 with code blocks. The program reads blocks line by line and manipulates barrels. The User has to type one function in each line. We will discuss and think more about the structure of the programming language. We tried to keep simple as simple. increase(a), decrease(a), reset(a), swap(a, b), set(a, x), randomize(a) can be used in this prototype where "a" and "b" are order(location) of a barrel, "x" is a value. Finally, the user has to click the run button at the bottom right.

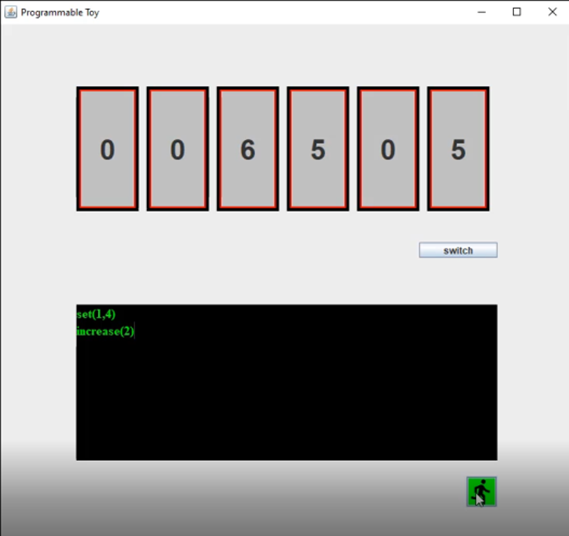


Figure 13. Prototype 2.1

### 3.1.7. Materialization

In the beginning, we started to design a programming language to create. Possible functions and structure of the language are discussed and declared. During the declaration process, we always considered, remembered the audience and the toy itself. We asked, what are the working principles of electronic toy and what are the possible operations to perform. With this perspective, we decided on the following structures and functions:

• Increasing barrel

• Decreasing barrel

• Resetting barrel

• Randomize barrel

• Setting barrel with given input

• Declaring integer variable

• Declaring string variable

• Conditional structures

• Simple mathematical operations

• Repetition structures

We declared those initial functionalities; new features can be added later.

On the other hand, we had to remember the target user of the toy. According to this constraint, the syntax of programming language must be easy to learn and to use. It must be simple. That's why we designed a simple syntax. Each barrel operation is designed as a single word function that takes barrel number as input. And also, each function should end with the ";" symbol.

• increase(x); for increasing barrel

• decrease(x); for decreasing barrel

• reset(x); for resetting barrel

• random(x); for randomizing barrel

• set(a, y) for Setting barrel with given input

• number keyword used to create integer variable

• name keyword used to create string variable

• if("condition") { "task" } and while("condition") { "task" } added

• =, +, - , \* , / operational signs

• repeat(x){ "tasks" } to repeat tasks x times

After syntax was declared we started to work on the compiler. We needed a compiler to take a bunch of written code text and understand them, then parse them into meaningful tasks.

While creating a compiler we had to remember all tasks, features of our language. We needed a common way to read and identify them inside the long text. That's why we created a .txt file that contains all the functions. The compiler reads the file line by line and creates a unique hash number. It makes string matching from start to end. It passes the whole string into smaller meaningful parts according to ";" and "{}" signs. Another string matching operation works for each chunk. When it found a string which identical to one of the functions in the .txt file, it pulls all the information of the function.

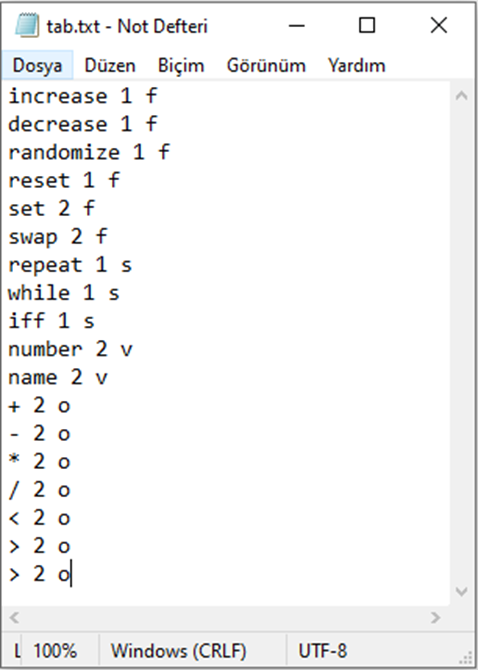


Figure 14. txt file which contains all properties of proggraming language

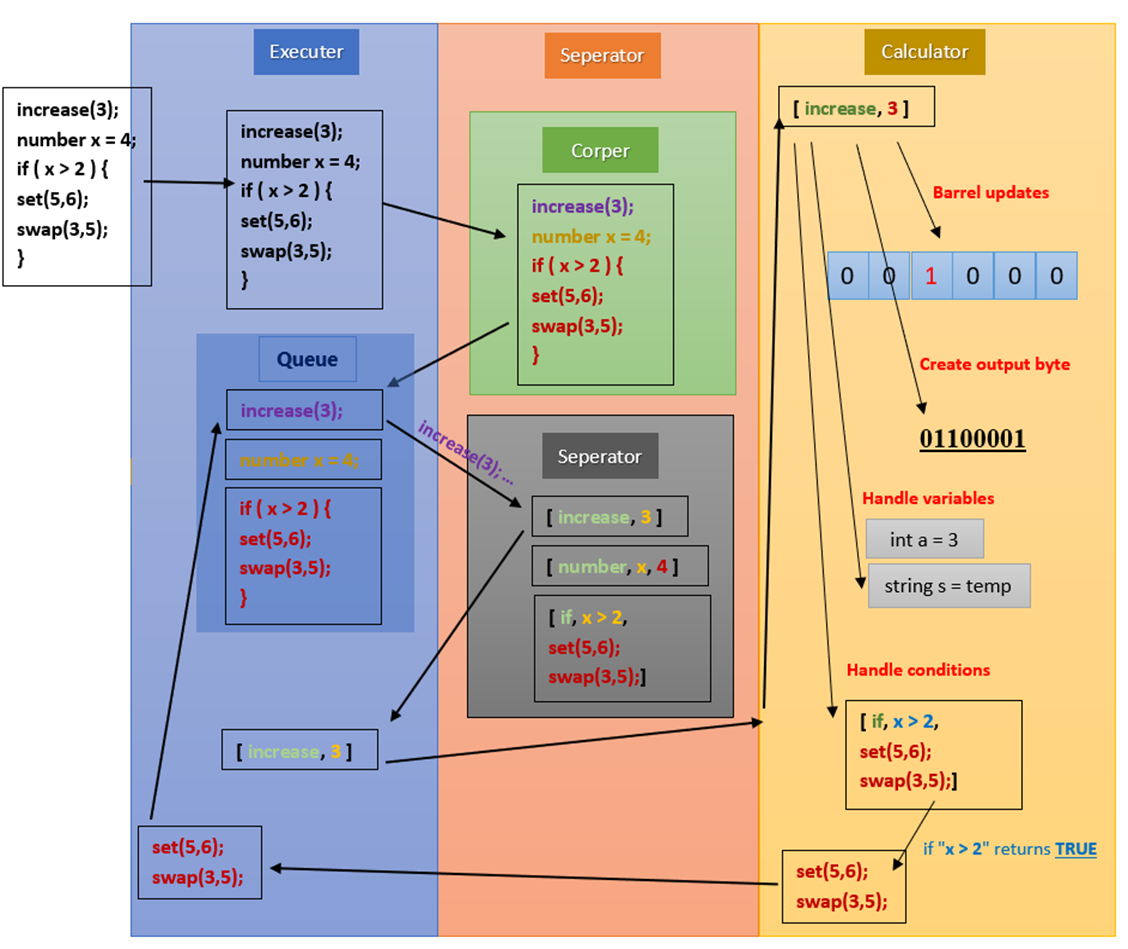


Figure 15. Diagram which shows the execution steps of algorithm

We faced some problems and bugs during the implementation of conditional statements. To solve this, we come up with a recursive approach. When conditions are satisfied, the task that inside the "{}" executed again. And also, we handled variable operations with generic type node class.

In order to test results and simulate changing barrel numbers, we used a simple array.

After language was completed we started to work user interface. First, we created a simple UI that contains barrel simulations and a text area. According to the research, we made last semester swing library was enough to create a GUI but as the depth of the research grows, we realized that we can create a better GUI for the application by using the JavaFX library. The library provides us with a simple visual design opportunity.

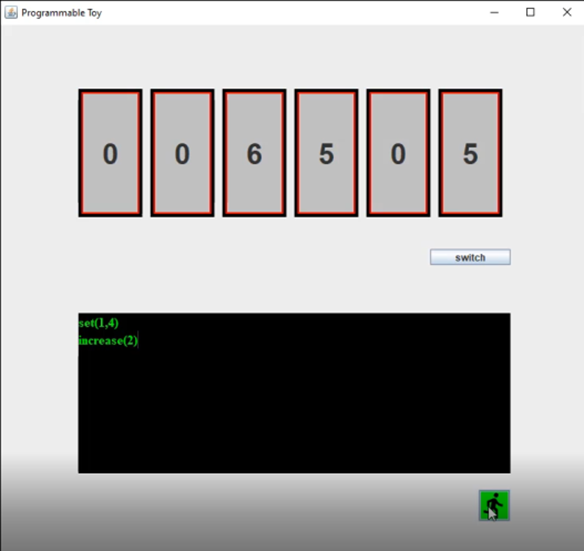


Figure 16. Old prototype wich implemented with Swing library

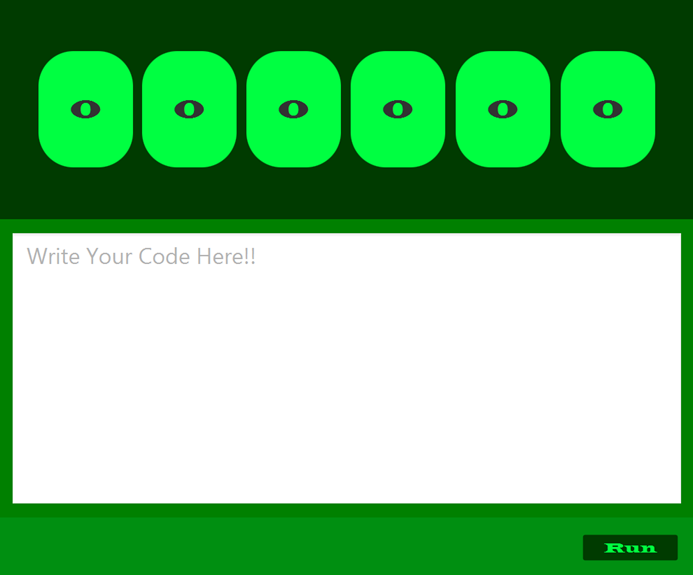


Figure 17. New GUI with JavaFX library

In addition to UI, the login page was designed and implemented by using the JavaFX library and components of this library. In order to construct this page, basically, three different labels, two buttons, two image views, and two text field components were used.

Labels were used for username, password, and login message.

Text fields were also used for username and password fields.

Buttons were implemented for login and cancel operations.

One of the image views (lock image view) was implemented to support the login operation idea, the other one was used to emphasizing the relation between thinking and programming who will be used this product, especially young generations**.**

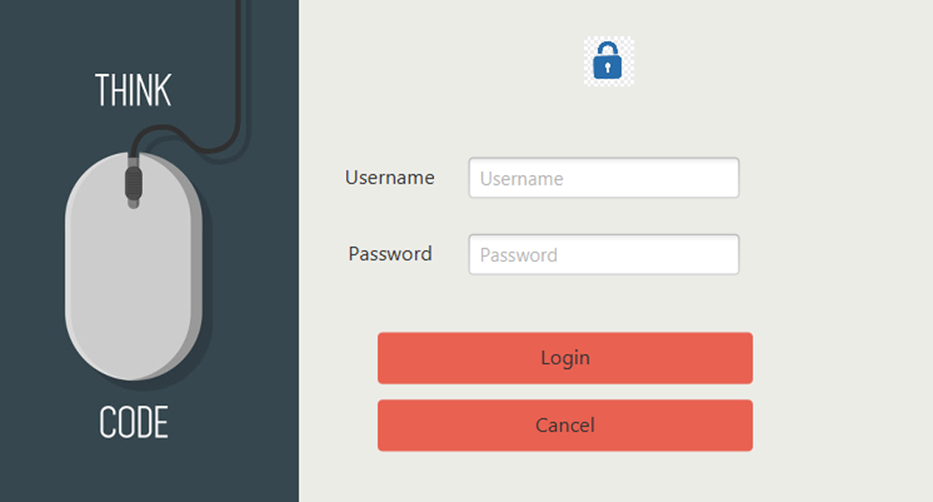
This login page is connected with the MySQL database. A table was constituted for users in the database and the validity of the login process has been examined. When users logged incorrectly, a “congratulations” message was displayed on the login page however when users logged in incorrectly, “invalid login, please try again” message was displayed on the login page. When users want to login without entering their username and password the message “enter username and password” was displayed.

Figure 18. Login Screen

In order to send the required input for the physical toy. We designed two different output formats.

The first format is an array which contains barrel\_number and rotation\_count.

For example: increase(3); -> [3,1], superrandom(1,6) -> [1, 1, 2, 6, 3, 3, 4, -6, 5, 2, 6, 3]

The second format is a single byte for the rotation of a single barrel. We encode the barrel and rotation information into 1 byte. First 3 bits responsible for selecting the target barrel (step motor). 4th bit responsible for the rotation side ( positive or negative). And last 4 bits are responsible for the rotation amount.

For example: increase(3); -> 01100001, superrandom(1,6) -> 00100001,01000110,01100011,10010110,10100010,11000011

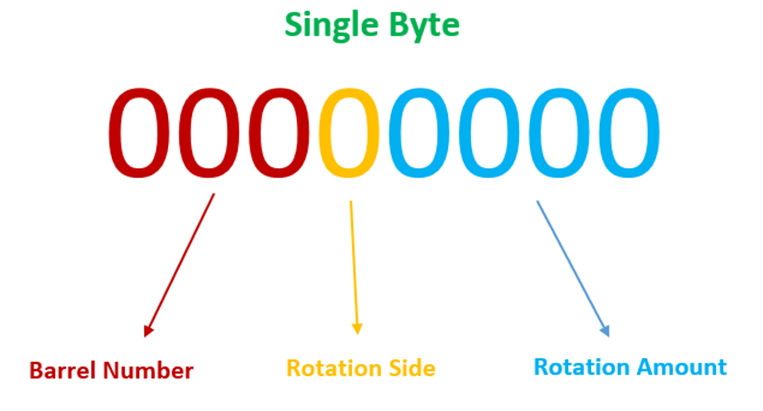


Figure 19. Structure of single (Byte) barrel input

We implemented three more functions for our programming language.

* superset(x); function sets all barrels with given input x.
* superreset(x,y); function resets all barrels between barrel x and y.
* super random(x,y); function randomises all barrels between barrel x and y.

The last part of the project was to send appropriate data to the physical side. To achieve that we decided to use the ESP8266 WI-FI module and because of the use of this module, we had to create a server and send all data to this server. We used the “Socket Programming” mechanics of Java for completing this part. The net library was going to be helpful for this task.

### 3.1.8. Evaluation

In order to test and evaluate the program and programming language, we simulated use cases. Like we said in the use case. The target user runs the application and can easily write code to the text area. When the user enters the run button, he/she can easily observe the changes of the barrel from visual barrels on the program. In short, visual barrels are working correctly.

The output bytes which will send to the physical toy are tested. Programming language correctly transforms written code into communication bytes.

Users can combine programming language structures and functions and create complex algorithms. Compiler parses and understands written code correctly.

The user interface is clear and easy the understand. And also programming language is simple and easy to learn for young aged users.

We tested language and program on a couple of middle school students. After a small programming lecture, they used the product easily. They also keep learning knowledge of programming while using the product. With their positive feedback, we can say that our product meets its requirements.

## 3.2. Electrical and Electronics Engineering

### 3.2.1. Definition of the problem

There is a technology in our lives that affects us directly, and this developing technology continues to be a bigger part of their lives day by day. The especially young generation will need coding more and more each passing day. Therefore we aim to ease coding difficulties in an entertaining way. So, with this Project; Children will be using this product to improve their coding skills. Therefore coding toy should be as simple as possible for children to use.

According to the input numbers selected by children via coding (From 0 to 9), the motors rotate the barrels to adjust the number we want to see. The speeds and angles of the barrels may cause some problems while running. We need to supply enough voltage to each motor so that we have no asynchronous speed among the motors. Otherwise, it may cause slow or high speed which results in asynchronous speed among them. Also, the suitable motor type should be selected according to our design, which we chose stepper motors. Because all the motors should stay in the same vertical line.

### 3.2.2. Review of technologies and methods

The Programmable Toy project has two main parts as we mentioned before. The software part was already discussed before. In order to handle hardware, We decided to use Arduino which provides us to control our barrels and communicates between barrels and user inputs.

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button. We can tell our board what to do by sending a set of instructions to the microcontroller on the board.

Therefore to solve the problem which may occur we used 8 servo motors and the motor controller to be able to control our barrels and for each number 36 degrees were assigned, a total of 10 numbers (from 0 to 9).

The reasons, which made us choose the stepper motor, were its speed, accuracy, and reliability compared to the servo motor. On the other hand, it’s a bit expensive than a servo motor besides its quality.

And as the last component, we need a wi-fi module that provides us communication between user input and barrels output. These technological design created the devices we wanted to produce for a programmable toy.

### 3.2.3. Standards and constraints

The quality of the toy application is related to usability, efficient time usage, and completion of the budget. We have chosen the stepper motor because its speed, accuracy, and reliability are better compared to the servo motor. However, it costs more than a servo motor besides its quality. Also, we used Arduino because it is a microcontroller that’s taught us before how to code and control. Moreover, it can be easily found in industry and easily can be integrated into the system we built. Also regarding the safety issues and standards, we followed; motors are built and embedded into the toy so that children are protected in case of any danger or accident. We also covered our toy to make it better in terms of physical appearance.

### 3.2.4. Conceptualization

First of all, considering the target audience, who are considered as kids, causes certain kinds of difficulties which might easily distract their attention.

This factor might be considered as a benefit at the beginning. Creating a well-designed and cartoon-like interfaces and the product might provide beneficial results in the start. But in the late stages, these themes can be significantly distracting.

In order to get rid of this possible drawback, it is suggested to create a colorful toy which can easily get their attention. And also different kinds of product can be thought of as another solution such as;

-Barrels (x6)

-Digital Screen: It is a basic led screen that shows 6 or 8 digits representing the hour, minute, and second respectively.

Table 5. Table of comparison

|  |  |  |
| --- | --- | --- |
|  | **Stepper** | **Servo** |
| **Price** | Cheap | High |
| **Accelerate** | Good | Very Good |
| **Speed** | Medium | Fast |
| **Performance** | Medium | High |

Stepper motors are easier to control and can run at a slightly slower speed but with high torque like a servo motor, also stepper motor is cheaper than servo motors. That's why we chose a step motor in our project.

Table 6. Table of comparison

|  |  |  |
| --- | --- | --- |
|  | **Arduino** | **Raspberry Pi** |
| **Coding** | Easy | Medium |
| **Accessibility** | Easy | Good |

For Arduino, the coding part is easier for us, because we have already learned how it works and how to code. Also, the accessibility is better for us. Because during our university lives, we have bought and used lots of Arduinos, therefore we don’t need to buy or look for the new one.

Table 7. Table of comparison,

|  |  |  |
| --- | --- | --- |
|  | **Arduino Mega** | **Arduino Uno** |
| **Coding** | Easy | Easy |
| **Accessibility** | Easy | Easy |
| **Input Numbers** | High | Low |
| **Reliability** | Similar | Similar |

The only reason why we choose Arduino Mega is just that it has more inputs to use and the number of Arduino Uno’s inputs is not enough for us.

### 3.2.5. Physical architecture

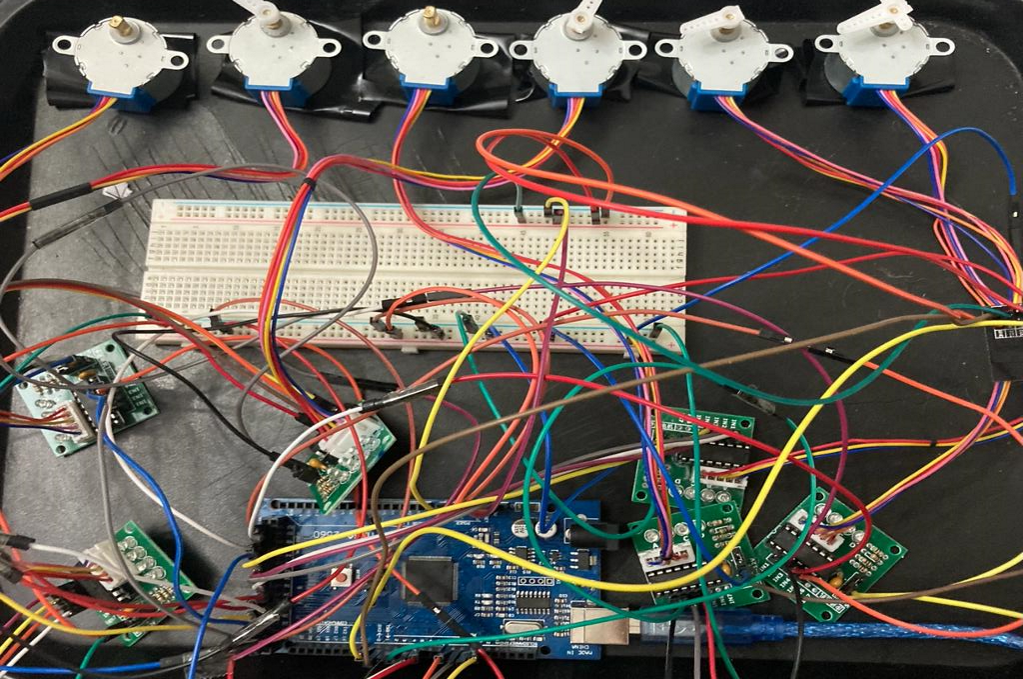
We have a microcontroller and we can call it the brain of the other parts that this toy needs to work, we used Arduino mega because it has more inputs and outputs than the others. We have two motor drivers connected to the microcontroller and the motors to which these motor drivers are connected, and these motors will rotate at certain angles according to the signals coming to the microcontroller and will make visible changes of the barrels. We used stepper motors and these motors normally will be inside of barrels this figure shows us the main idea of design.

Figure 20. Hardware components and their connection

### 

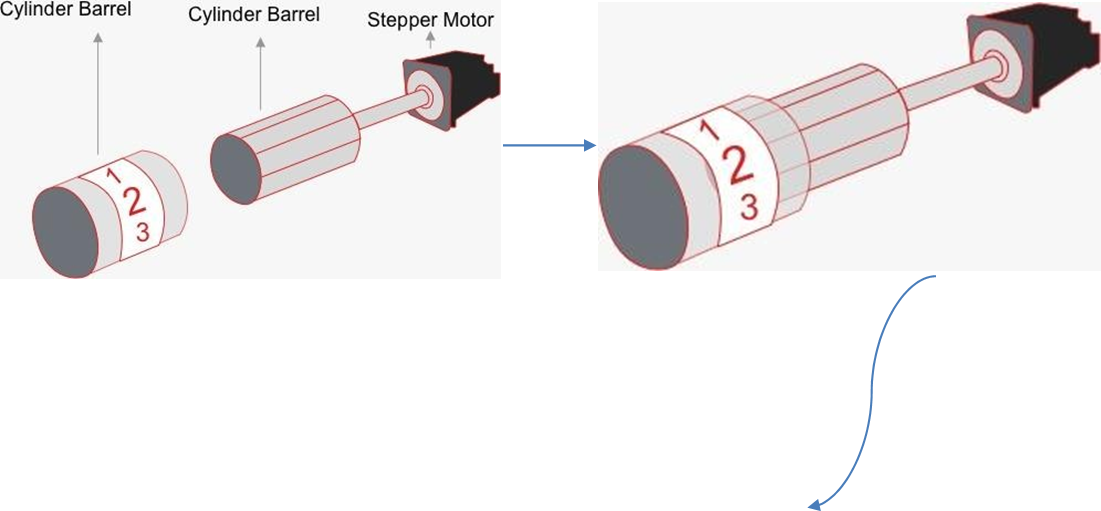
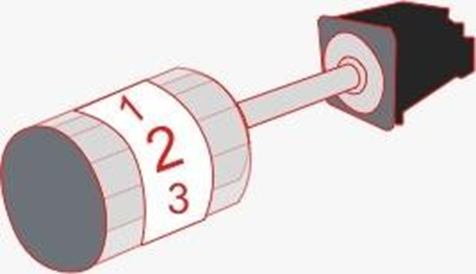


Figure 21. Motors and Cylinders

We will place 6 barrels next to each other by hiding them inside the box we design. Therefore nothing will be seen from the out of the box. And it’s not going to be very large as it is seen in Figure

We just showed it big to make people understand the Project report well.

The Stepper motor will rotate the cylinder barrel and cylinder barrel will be integrated into the main cylinder barrel which is showed above. Motor’s size approximately 4.2 cm for each barrel so that we may end up with something small and suitable to use.

### 3.2.6. Risk assessment

Table 8. Risk Assessment Tables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Failure Event** | **Probability** | **Severity** | **Risk Level** | **Plan of action** |
| **Stepper Motor Fails** | Possible | Moderate | Medium | Consider better quality motors, have spare motors on hand if  required. |
| **COVID-19**  **Lockdowns** | Likely | Moderate | High | Redesign the sub-system communication to work with the Internet, we can then integrate the subsystems over the internet if required. |
| **Arduino Failure** | Unlikely | Major | Medium | Have a spare Arduino at hand. |

### 3.2.7. Materialization

We started to work with a motor and Arduino, and after performing the step and speed control we wanted on the first motor, then we added the second motor and continued to work on two motors. After doing the necessary coding, we made the other 4 motor and their driver connections on the Arduino mega and started them one by one.

However, there was a problem when we connected all the motor drivers, after the 5th motor, the Arduino stopped working and when we removed the 5th motor driver, it was working again, the Arduino voltage was insufficient.

Then we decided to supply the power of the motors on the breadboard with the help of the connection from the Arduino's 5V and GND outputs then, each motor works according to the selected barrel number and the value to be increased.

We have 6 stepper motors, their drivers, wifi module, and Arduino. We get the required values from the user via the serial monitor, for example, by taking M1,3 inputs first motor will increase 0-3.

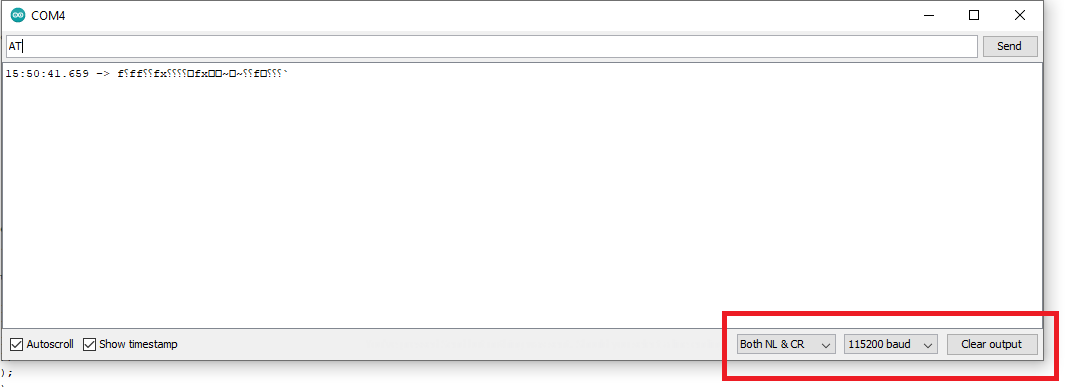
Another problem is that after starting the motors one by one, the motor drivers were still drawing current, so the Arduino stopped working after the 4th motor again, and thus we could not operate as we wanted, then we added the codes that cut the energies of the relevant pins after the motor drivers were connected through the Arduino.

Also, the motors have 2048 steps, which means that we had to get inputs between 0 and 2048 values for the rotation of the motor, so we needed the conversion of the step numbers, and we solved it by adding a string to int commands.

We decided that the 3D parts should be bigger than the motor to hide the motors and we revised their dimensions to be bigger, thanks to our friends Berk Saçkara and Mustafa Futtu who helped us in this regard.

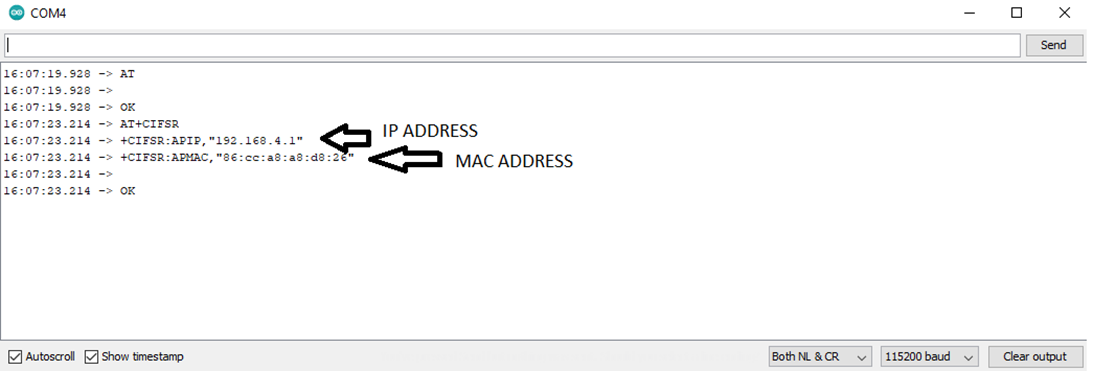


Figure 22. Final barrels

The problem caused by the wifi module circuit in our project is that we realized that the wifi module did not work when we connected it with the Arduino-related pins, Because the wifi module we had was not up to date, so when the AT command was entered on the serial monitor after connecting, we realized that it did not give us the (OK) message.

We made the necessary update with NodeMCU flasher.



We took the first step to connect over the internet by obtaining the necessary information of the wifi module. Then we solved all the problems we faced. Now it’s working properly.

### 3.2.8. Evaluation

The product was tested step by step. First of all, each project member finished their duties. After that, we started to integrate one part into another. Software students finished designing and coding the application and its interface. They tested their application and its processes. Electrical-Electronical Students designed the electrical system and the circuit. Then they tested their circuit and processes. After they all have been done, these parts (software and hardware) were integrated into each other. So application (software) and the product were working simultaneously. An unpredictible problem occured on WI-FI modüle, and we had to produce an alternative solution for it. We decided to use Serial Communication method. Then the last version of the product was tested with the cooperation of children who want to use it.

### 3.2.9. Tests and Results

# 4. COSTS

Table 9 Costs

|  |  |
| --- | --- |
| **Component** | **Cost (TL)** |
| Arduino Mega | 50 |
| Stepper motor & Dc motor driver x 6 | 85 |
| Wifi Module | 25 |
| Breadboard, cables and other materials | 50 |
| **TOTAL** | **210** |

# 

# 5. CONCLUSION

In conclusion, we prepared our implementation path and declared important features. While implementing the design, teams stuck with standards and defined considerations. Written code blocks were clean and every possibility of error was countered. GUI was simple and enjoyable at the same time. The programming language of the toy was also simple for young aged users. Syntax and rules of language were simple and the number of them was minimized as much as possible. Teams worked synchronously to prevent any connection error between software and hardware. User code will be transformed into required hardware inputs. These inputs and input types were clear during the implementation process.

There were some errors that the team encountered. In the integration, part WI-FI module did not work as we wished. We decided to change the way of connection and chose to build a serial connection between Arduino and Java Application. We succeeded to send proper data and the toy could understand data.

There are some features that we want to improve. For the GUI side, we want to create a dynamically changing interface and colorful text area. For programming language, we want to add some variable types and some functions for users.

# ACKNOWLEDGEMENT

We wish to thank our advisers Dr. Beddall and Dr. Kar for their precious supports in this project.

We have to present our special thanks to Yüksel E. Bezci and Deniz Tan for their precious help.

# REFERENCES

[1]. Code Combat, Retrieved on May 22, 2020, from https:/[/www](http://www.codecombat.com/).[codecombat.com/](http://www.codecombat.com/)

[2]. Java Code Conventions, Retrieved on May 29, 2020, from https:/[/www](http://www.oracle.com/technetwork/java/codeconvtoc-136057.html).[oracle.com/technetwork/java/codeconvtoc-136057.html](http://www.oracle.com/technetwork/java/codeconvtoc-136057.html)

[3]. Ameritech Graphical User Interface Standards and Design Guidelines, Retrieved on May 31, 2020, from

https:/[/www](http://www.theuxbookmark.com/wp-content/uploads/2010/08/ameritech-gui-std-design-).[theuxbookmark.com/wp-content/uploads/2010/08/ameritech-gui-std-design-](http://www.theuxbookmark.com/wp-content/uploads/2010/08/ameritech-gui-std-design-) guidelines.pdf

[4]. Harlow, Danielle & Dwyer, Hilary & Hansen, Alexandria & Hill, C. & Iveland, Ashley & Leak, A.E. & Franklin, D.M., Computer programming in elementary and middle school: Connections across content, USA, 2015.

[5]. Java Documentation, Retrieved on May 29, 2020, from https://docs.oracle.com/en/java/

[6]. Arduino Basics Retrieved on August 28, 2020, from https:/[/www](http://www.arduino.cc/en/guide/introduction).[arduino.cc/en/guide/introduction](http://www.arduino.cc/en/guide/introduction)

# APPENDIX A

Arduino Side:

// Arduino stepper motor control code

#include <Stepper.h> // we include the header file

// we change this to the number of steps on our motor

#define STEPS 32

// we created an instance of the stepper class using the steps and pins

Stepper stepper1(STEPS, 8, 10, 9, 11);

Stepper stepper2(STEPS, 22, 26, 24, 28);

Stepper stepper3(STEPS, 30, 34, 32, 36);

Stepper stepper4(STEPS, 38, 42, 40, 44);

Stepper stepper5(STEPS, 2, 4, 3, 5);

Stepper stepper6(STEPS, 47, 51, 49, 53);

int val = 0;

void setup() {

Serial.begin(9600);

Serial.println("Please insert the motor and the wanted degree in this format :");

Serial.println("Motor,Degree");

Serial.println("Motor can take the following values: M1, M2, M3, M4, M5 or M6");

Serial.println("If you want to increase the value, insert a number between ]0,10] ,If you want to decrease the value, insert a number between [-10,0[ ");

stepper1.setSpeed(200);

stepper2.setSpeed(200);

stepper3.setSpeed(200);

stepper4.setSpeed(200);

stepper5.setSpeed(200);

stepper6.setSpeed(200);

}

void loop() {

String rxString = "";

String strArr[2];

if (Serial.available()) {

String MotorValueString = "";

String degValueTmp = "";

int degValueString;

int choice;

//Keep looping until there is something in the buffer.

while (Serial.available()) { //while there is something written, keep getting it

//Delay to allow byte to arrive in input buffer.

delay(2);

//Read a single character from the buffer.

char ch = Serial.read();

//Append that single character to a string.

rxString += ch; // this keeps adding the characters to have the full string ex : "A" += "B" = "AB"

}

int stringStart = 0;

int arrayIndex = 0;

for (int i = 0; i < rxString.length(); i++) { // this is a loop, it helps you to do the same thing over and over without having to write the code many times

//Get character and check if it's our "special" character.

// since our String will be something like "Motor,Value", we will have to look for the "," so we can take all the characters before it, and put them in one String, and put the next characters together on another String

if (rxString.charAt(i) == ',') {

//Clear previous values from array.

strArr[arrayIndex] = "";

//Save substring into array.

strArr[arrayIndex] = rxString.substring(stringStart+1, i);

Serial.println(strArr[arrayIndex]); //Set new string starting point.

stringStart = (i + 1);

arrayIndex++;

strArr[arrayIndex] = rxString.substring(stringStart, rxString.length());

break;

}

}

//Put values from the array into the variables.

MotorValueString = strArr[0];

degValueTmp = strArr[1];

//Convert string to int.

degValueString = degValueTmp.toInt();

Serial.println(MotorValueString);

Serial.println(degValueString);

choice = MotorValueString.toInt();

switch (choice)

{

case 1:

Serial.println(degValueString);

stepper1.step(-degValueString\*204.8);

stepper1.step(0);

// , 8, 10, 9, 11

digitalWrite(8, LOW);

digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(11, LOW);

break;

case 2:

stepper2.step(-degValueString\*204.8);

stepper2.step(0);

digitalWrite(22, LOW);

digitalWrite(26, LOW);

digitalWrite(24, LOW);

digitalWrite(28, LOW);

break;

case 3:

stepper3.step(degValueString\*204.8);

stepper3.step(0);

digitalWrite(30, LOW);

digitalWrite(34, LOW);

digitalWrite(32, LOW);

digitalWrite(36, LOW);

break;

case 4:

stepper4.step(degValueString\*204.8);

stepper4.step(0);

digitalWrite(38, LOW);

digitalWrite(42, LOW);

digitalWrite(40, LOW);

digitalWrite(44, LOW);

break;

case 5:

stepper5.step(degValueString\*204.8);

stepper5.step(0);

digitalWrite(2, LOW);

digitalWrite(4, LOW);

digitalWrite(3, LOW);

digitalWrite(5, LOW);

break;

case 6:

stepper6.step(degValueString\*204.8);

stepper6.step(0);

digitalWrite(47, LOW);

digitalWrite(51, LOW);

digitalWrite(49, LOW);

digitalWrite(53, LOW);

break;

default:

Serial.println("Please insert a value between M1-M6");

}

}

}

int\* decode(string s) {

int byteInt = stoi(s);

int types[3];

types[2] = (byteInt & 0x0F); // rotation amount

//cout << "RA: " << types[2] << endl;

byteInt = byteInt >> 4;

types[1] = (byteInt & 0x10); // rotation side

//cout << "RS: " << types[1] << endl;

byteInt = byteInt >> 1;

types[0] = (byteInt & 0xE0); // barrel index

//cout << "BI: " << types[2] << endl;

return types;

}

# APPENDIX B

Java Side (Because of its length we shared it as a GitHub link.):

https://github.com/CappyRock/Proto-Programmable\_Toy-.git