Methodological Guidelines

What is a game? My First Game

Basic concepts covered at the lesson

- Programming
- Algorithm and program
- Scratch, sprite, script
- Commands, appearance, background
- Cycles
- Random numbers

Electronic educational resources used at the lesson

- Internet-enabled computer
- preinstalled RobboScratch3, Zoom or Blue Jeans
- Own account in the community on https://scratch.mit.edu/

Features of the lesson's representation

A special thing about the course is that the material is presented visually in the form of slides. Explanations for each slide follow below.

The text of methodological guidelines doesn't need to be conveyed to students word-for-word. It can be either accepted or redefined, either elaborated or shortened by the teacher. The only thing to adhere to strictly is the order of slides.

The training neither supposes nor welcomes any creative deviations from the general

Introduction

Hello, everyone! Welcome to our RobboClub. New conditions grant us new opportunities! Due to remote learning knowledge has become more available, and you have done great by choosing ROBBO to study robotic science!

Our main task is to grasp laws of robotic science - the science that deals with designing, building and programming robots.

Learning in the RobboClub is based on mastering a skill of operating the Scratch universal platform.

While I'm speaking about Scratch, please, open the game via a link.

https://scratch.mit.edu/projects/397006337 It's here that sharks prowl across the sea, while you move a starfish using your mouse to destroy sharks, though they will re-emerge later. All the fuss is being observed from a submarine. It's gradually getting dark... Let's play!

By the end of the lesson you will complete a large part of the game.

You can play while I'm explaining the topic of the lesson. But estimate the quality of the game while playing. This is called testing.

I have mentioned that Scratch programming is universal. For us, universal means that our company has designed the RobboScratch3 platform based on the American Scratch programming environment.

We use RobboScratch3 to manage robots that have been designed by our team as well. So, one and the same programs written in the same language can be used either to manage robots or to create cartoons, mathematical models of real-life events or computer games.

The thing is, game characters are also managed with programs. The can be called 'virtual executives'. Why not even 'virtual robots'?

Once you learn to program actions of a virtual robot, it will be easier to make programs for actual robots.

The best way to learn is to implement challenging projects. In our case, it means inventing and creating a game on your own. By the way, it doesn't have to be a game. It can also be a cartoon or a project that clearly demonstrates some issue of the school curriculum in the form of animation. It's critical to master Scratch skills to the extent when any suggested or invented topic, any fantasy could be implemented by your own in the form of an exciting scratch project. You don't need to wait for a new version of Tanks or Zombies, you can make your own game!

Working with the computer one need to give their eyes some rest from time to time. It doesn't mean simply to close them. We are going to do special exercises.

https://scratch.mit.edu/projects/397009251 Eye training (45 seconds).

We will not try to surpass each other. Everyone studies in their own pace, because of different reasons. If you haven't done everything during the lesson, you can figure out things at home taking your time. Those who manage to catch up with the teacher will make the game with all the elements. Others will have simpler games, but these games will work anyway. That's why only those who are the quickest of apprehension will be able to do all things presented on slides. Others will have to catch up at home.

Question to the chat. Now, about grades. Please, open the chat and report about the results of game testing. Send me four figures:

a) Assess the complexity of the game from 1 to 5.

b) Assess the colorfulness of the game from 1 to 5.

c) I'd like to download it on my smartphone – 1 / I'll never download such a thing – 2.

d) I'll never manage to make such a game – 1 / I'll make it in a month – 2 / I'll make it by the end of the lesson – 3.

Slide No. 1

The Title of the Presentation. Open the chat in the conference. I will ask you occasionally, please, answer in the chat. You can also speak to answer, but please, raise a hand beforehand.

Before demonstrating each slide the teacher asks students via a chat or asks questions about the content of previous slides to repeat the material and solidify the knowledge. Advised questions are marked the following way: **Question to the chat.** *It doesn't limit the range of questions that may be asked by the teacher.*

Slide No. 2

Algorithm and Program (Code) Before you turn to making a computer game, you need to have an idea of what you would like to see in the end. You need to construct a scenario of the future game, distribute roles between characters and think their actions through. Every action is to be carried out in a certain order, in the correct succession. This succession of actions written down in a certain way is called an algorithm. What is the succession for drawing a line on a paper using a pencil? One can't draw a line before taking a pencil. The algorithm here is the following: take a pencil, put it on the paper, then draw a line.

The algorithm may consist of words, just as we've demonstrated. Or of symbols. Programming specialists tend to present algorithms in the form of flow charts.

Wordings of commands and conditions are written in certain geometric figures within a flow chart. A rounded rectangle denotes the start and the end of the algorithm, simple rectangles are for execution of a command, and diamonds are for conditions. That will do for now. We can see our algorithm on this slide.

The next task of the programming specialist is to make a program for an actual or virtual robot based on this algorithm. To write a program code, or just a code.

There are no less programming languages in the world than human ones. A programming specialist is a translator from the human language to a programming one.

On the right of the flow chart we can see a fragment of the program written in C+, an 'adult' language. To code in this language one need to have good knowledge of English, mathematics and logics, and much more things. However, specialists from MIT (an American institute) have created a special programming language for children. Instead of a complicated code we will work with bright blocks - as if with pieces of the puzzle. And we'll make actual programs. These programs will be able to manage a robot, be it an actual or a virtual one.

Exercise No. 1 in a workbook.

Slide No. 3

Sprite and Script (Code) On the left we can see the structure of the program, the code in RobboScratch3. Color strips are various commands for a robot. Such a set of commands is conventionally called a Script. The script has its own executive. That's the character of the project we have in mind. It can be a presentation, an animation, a cartoon, an interactive cartoon, a simulator, a data base, or a computer game.

Question to the chat. Can an algorithm in Chinese be created? Recall shapes of the flow chart. What is the shape for 'End'?

Each character of the game is to have their own script. Otherwise it remains just an image. Characters are called Sprites. The word can be translated as 'fairy' or 'elf'... Sprites are located in a special Sprite Library where you can choose one. One sprite can be drawn in several poses. In Scratch these are called suits. As you might have guessed already, it will be used to animate and activate the sprite. You can also draw the sprite yourself. For example, we've drawn the submarine ourselves.

Slide No. 4

Choosing the Sprite 1. Point to the cat face under the scene; 2. Click on the zoom; 3. Get to the Sprite Library. You can either look through the whole library or turn straight to the necessary section: Animals, People, etc. Double click on the chosen sprite - and you can start coding. Choose any sprite, click. It will emerge on the scene, and its icon, a small image will be displayed below the scene. It is boxed with a bright blue contour, and there's a cross in the top right corner. Get an eyeful of the chosen sprite and delete it by clicking on the cross. You can explore two or three more sprites the same way.

Question to the chat. Will a robot understand the algorithm written in a human language?

Slide No. 5

Scratch Interface (RobboScratch3) So how can you interact with RobboScratch3? What way? A way the person interacts or communicates with a gadget or a program is called interface in our Information Technologies. RobboScratch3 interface is plenty of virtual buttons, windows, tabs. We get the desired result from the application by aiming a mouse cursor on them. All the scratch space of the program is divided into three large areas or fields.

Open the chat in your conference.

Now I'm going to name and show you various interface objects. Please, send pluses (+) to the chat when you find the mentioned object.

On the right we can see the Scene where the entire action with sprites managed by scripts takes place. As ant scene, it has a background selected in accordance with the chosen sprite in the Background Library. The background may be drawn as well.

Question to the chat. What is the easiest way to choose the sprite? (Surprise button).

There's a large area in the middle of the screen – a field for assembling scripts from different commands. Commands are located in the block of commands on the right. Commands themselves are agile, so they are easy to look through. Commands are displayed in the section after a click on the circle of a certain color. Because of multiple colors of the circles, the block of commands is sometimes called 'palette'. Each circle is a block as well, a container comprising dozens of commands. Or even hundreds. It's only passionate and solitary work on games and projects that allows mastering all the opportunities of Scratch. Mind that the number of opportunities is growing all the time. For now, though, we will have a look at what we need for today's game.

Slide No. 6

Question to the chat. Is the scene on the right or on the left?

Here we see a part of the top panel above the scene. Scripts are launched by clicking on a check mark. There's a red Stop symbol next to it. There are three buttons above the other top corner of the scene to fold or unfold the scene. There is the Sound button above the command section. It leads to the Sound Editor comprising the button to choose sounds from the Record Library.

If you move the cursor up, from the cat face under sprites to the Brush/Paint and click, you'll get to the Graphic Editor. If you like drawing, it will be impossible to pry you away with a crowbar.

Physical Activity Break. We've done good while sitting. Now let's jump a little bit. We are going to make a sea-themed game, so let's swim instead. The teacher opens

Slide No. 26.

Physical Activity Breaks. Everyone can choose how to swim. Just mind that water is thicker than air, so try tensing muscles a bit while making swimming moves. The music is on. 1.5 minutes.

Slide No. 7

Question to the chat. Who was the finest swimmer on the slide?

Commands. Motions. Cycle. Click on the Choose the Sprite button. Choose the Shark among Animals in the library. Find the blue Motions circle in the Commands section. The first blue command is Go 10 Steps. It's a blue strip with pretty inserts. Pull the Go 10 Steps command to the middle field – the one for scripts, the assembly shop.

Click (LMB) straight on the command. Again. Again. Watch the shark moving. Share your impressions.

Click on the command with the RIGHT mouse button (RMB). Select Duplicate. Click with LMB now. Here we have two identical commands. It will be handy.

On one of the strips write '100 steps' command instead of '10 steps' in the input field. Now the command is: Go 100 Steps. Click, watch, share.

Click on the orange circle, which is Management. Find the Repeat 10 Times bracket. These brackets are also called C-blocks, can you see why?

Question to the chat. What letter does a C block resemble?

Throw the bracket to the script field. Pick the Go 10 Steps command with a mouse and pull it closer to the bracket. Then the bracket opens its jaws that get a bit greyer. That's the time to let go of the mouse. Now, let's insert! Do it a few times. Try inserting.

Well done! Here we have an assembled construction (Repeat 10 Times Go 10 Steps).

Click on the bracket or the command, i.e. the assembly, with the LMB. Watch. Compare teleportation and movement.

The bracket is a Cyclic Operator. The cycle is a motion in circle or repetition of one and the same. In this case, the same move of 10 steps is repeated 10 times.

Question to the chat. When has the Shark moved farther: making 100 steps or 10 times as much as 10 steps? Is it the same distance? Or not? Send the message to the chat.

Slide No. 8

Question to the chat. Who has noticed how many Suits the Shark has? Send the number to the chat.

Commands. Cycles. Conditional Operator. Change 10 times for 77 times, for example. How far has the Shark moved? Try different numbers. Conclusion: the sprite doesn't move beyond the edge.

Question to the chat. What is a single word to replace 'the biggest number of times'?

Go to Management. Find the same bracket, just without a field and without 'times'. This one is to ALWAYS Repeat. Throw and insert. No, it doesn't move beyond the edge again.

Get in Motions. Try to find the If Touches the Edge, Push Off command.

Focus on the word 'IF'. The command is carried out UNDER a certain CONDITION only. In this case, the condition is the EDGE. If there is the edge – push off! If there is no edge, the motion goes on. That's why it is called 'Conditional Operator' instead of a 'command'.

Insert this command in the assembly that already contains the Going 10 Steps command. Jaws of the bracket are getting even wider, so now let the mouse go, and the command will take its place. Click on the assembly. Now the Shark is raging from edge to edge, PUSHING OFF like a ball. It turns over occasionally, though.

Slide No. 9

Question to the chat. How does a Conditional Operator look in the flow chart? Is it a rectangle? Yes? No?

Commands. Direction in degrees. Sprites move in the direction they look. And the direction they look is the motion direction.

Changing the motion direction is a turn. The turn can be bigger or smaller. How can the angle of the turn be measured? It is measured in degrees.

Turning around to come to the initial place in the end is to turn on 360 degrees.

Question to the chat. Can there be a 1 degree turn?

In Scratch directions are a little bit different: Up, Right, Down and Left. If the sprite isn't given any direction, which is called 'by default', it will always move to the right. That's 90 degrees. Down is 180 degrees, and up is 0. Left is minus 90 degrees (-90). It's conventional for Scratch. So, when we launched the sprite 'by default', it went 90 degrees to the right and after touching the edge changed face to go in the opposite side. The opposite side means substracting 180 degrees from the direction. For example, it was moving down, which is 180, and then – up (which is 0). When the sprite goes to the left, it's -90.

Exercise No. 2 in a workbook.

Slide No. 10

Question to the chat. If you turn 180 degrees from you camera, what will others see on their screens?

Commands. Direction. Scoping. To stop the Shark from turning over when taking the opposite direction, we need just to restrict turning over the head and enable it to turn and move sideward. We are going to curb the freedom of overturning. Find the restricting Set Turn Mode command in the Motions section and click on the white triangle. You'll see three options in the drop-down box. We've already tried circular turns without any restrictions. Now select No Turning. The Shark will push off to swim tail first. It doesn't suit us. Then click on Left-Right. Now the Shark swims back and forth. But fishes rarely swim in one and the same depth.

We remember that the sprite always has the initial direction – on the right, or 90 degrees. Add the blue Turn In 90 Direction command to our assembly. If you click on 90, the clock-face will appear. It has a hand, like any watch. And the hand points to the right. I.e. 90 degrees is a direction to the right. Below the scene, in the Direction field of the Sprite Properties field we can also see 90. This applies to the Shark when it moves to the right only though. When it turns, the window features -90.

So, let's say we would like to make the Shark not only rage from the right to the left, but also change the height. Click on the 90 field. Here comes the clock-face as well. Let's tamper with the hand a bit with the LMB, so that it points to 60 degrees. It's neither to the right, nor up, more like... criss-cross, kitty-corner. Now we can see how the sprite's direction is changing in the field in terms of degrees, as long as the sprite moves. Numbers from 0 to 180 will flash there, sometimes with a minus though, and sometimes without one, which means + in mathematics. When the sprite moves to the left, all marks are minuses. And vice versa.

Well, now the Shark swims by the rules.

Do you know that Sharks never sleep? They are heavier than water, so they'll just be drowned if they stop moving. And we'll ruin our eyes if we don't let them rest.

https://scratch.mit.edu/projects/397009251 Eye training (45 seconds).

Slide No. 11

Commands. Rotation. Center. X and Y. Let's make the Shark swim not only in a kitty-corner manner, but in a tortuous as well. Add the blue Turn 15 Degrees command. Change 15 for 1. Now look how degrees are changing in the Direction field.

To make the Shark start its way from the same point every time define this point. Let it be the center of the scene. Each point of the scene has its own x and y. The point in the center has X = 0 and Y = 0. They are so-called coordinates. Coordinates attach the sprite to a certain place on the scene. At the start the Shark will have (0; 0) coordinates. Just remember it for now.

As you can see on this slide, we've finally put the coverage with the check mark on the script. It is located in the light yellow Event container. The check mark on the script is related to the shared check mark above the scene. Click on the shared one to launch the script and make the sprite work. If there are several sprites, each of them should be under the coverage with a check mark. Then a click on the shared check mark activates all the sprites at the same time.

You should also remember that all the commands between the check mark and the Always Repeat cyclic operator are carried out once, straight after a click on the check mark. They define initial properties of the sprite with a single permanent order. The commands inside the body of the cyclic operator are constantly changing their orders based on circumstances, always influencing the sprite's behavior.

Question to the chat. Solve the problem. A crab was crawling in the direction of 12 degrees. Then he turned another 12 degrees left. What direction is crab crawling now?

Slide No. 12

Question to the chat. What does pushing off mean?

Background. It's critical for events to take place against a beautiful background. Of course, the background should be aligned with our game. So, let's go to the Background Library to choose Underwater. The background has all the same options, but one. The background doesn't have the Motions block of command. That's because the background is always still. It's everything else that moves against the background.

Slide No. 13

Question to the chat. What is a check mark above the scene for?

The program works! Let's assemble the script and verify it via the slide. If everything is correct, click on the check mark. Look how elegantly our beauty Shark is swimming.

But it's so bored. Why not adding some friends to it? Just click on the button of our lonely sprite below the scene with the RMB, then click on Duplicate, when it's popped up, with the LMB. Now we have a copy of the Shark, the second sprite. With the copied program. Click on the shared check mark, and both sharks will begin to bustle on the screen. However, both of them start at (0; 0), where they practically run into one another. You can pull them apart with a mouse for now. Or you can make the second Shark pause with the check mark. It's the yellow Wait For 1 Second command in Management.

Go on duplicating. How many Sharks are there now? 5? 10? Five is enough. Click on the check mark. A funny cartoon is ready.

Physical Activity Break, go to **Slide No. 26** Select music.

Slide No. 14

Commands. Any movement depends on others. Our underwater kingdom, on the other hand, is filled with Sharks only. Let's add some Starfishes.

Click on Choose Sprite. Choose the Starfish among Animals in the library (also SS, Sea Star). You can copy the program of any Sharks for the Starfish.

That's how to do it. When you'd chosen the Starfish, an empty script field has been opened. There are no commands here.

Exit this sprite's profile: click on the button of any Shark. The Starfish button has got tarnished, and the button of the selected Shark stands out now. Point the mouse on the check mark of the script, click and - without letting LMB go – pull the entire code from the script field toward sprite buttons below the scene. You'll notice that each sprite on the mouse's way is flashed on. But we move strictly toward the Starfish. When the check mark and the mouse cursor are both on the Starfish button, the latter will start dancing merrily – that's your cue to let the mouse go. The script will jump back! So, what have we accomplished? Here it is: get in the Starfish sprite and you'll see that the copy of the Shark script is already there!

Click on the shared check mark. Now the sea is a mess. Sharks are hustling and bustling, and so are Starfishes. But we have another concept in mind.

Take the Shark script by the check mark and pull it to the field of command. When the cursor and the check mark and are both there, let the LMB go: the script will disappear, and the Starfish will be left without the program again.

Make a new script. On the slide, it's in the left. You need to click on the white triangle of the Switch Over To... command and select where 'to switch over'. It's the mouse cursor for us. Have you noticed how long the list is?

Let's start (click on the shared check mark). The Starfish responds to the mouse cursor immediately. Moving the mouse we manage the Starfish! Meanwhile, Sharks are raging on their own.

Now, let's make the right script. Before starting, though, throw the first script away. Or disengage it from the check mark. Otherwise, at the start both scripts will be launched to hinder each other.

So, let's compare two scripts. Which variant is better for the game?

Question to the chat. 1? 2?

Note the differences between scripts for Sharks and Starfishes. All the actions of the Shark depends solely on the Shark. Actions of the Starfish depends on the mouse, i.e. the gamer, though.

Slide No. 15

Interactive Cartoon. So, we still don't have a game, but it's no longer just a cartoon as well. We can participate in our cartoon instead of simply observing. Let's call it an interactive cartoon. Interactivity is an opportunity to exchange signals, communicate and influence. Let's start. Watch the Sharks and chase them. That's all for now though! Nothing else happens. That's because sprites don't have sense organs.

Question to the chat. What will happen if you click not on the shared check mark, but on one, for example, one of Shark No. 3?

Slide No. 16.

If you influence characters of the cartoon... that's when this cartoon turns into a game. Something needs to happen when the Shark and the Starfish meet, and we need to figure it out.

Question to the chat. What would you like to observe when the Shark and the Starfish meet?

However, first, we need to make sprites feel each other somehow.

Slide No. 17

Question to the chat. Which block of commands does Background lack?

Sprite Sensors. How do we sense the world? What do we feel? How?

We have eyes to see. Ears – to hear. Fingers – to touch (palp). These are our sense organs. That's what doctors say. Engineers say – detectors or sensors. Detectors can gather data about the outside world: this is hard, that is hot.

How can sprite feel? With sensors as well.

Let's open the Sensors container. Azure strips. We'll need the topmost strip – Touches... There's a whole list here as well.

So, the Starfish Touches the Shark. Something must happen.

Slide No. 18

Question to the chat. Which sensor is the most important for dogs?

Who has sensors? So, the Starfish touches the Shark. But the Shark touches the Starfish as well. Whom should we provided with a sensor? Both? It's easier to explain each Shark that it is supposed to disappear if it feels a Starfish. The thing is, we can do it for the first Shark only to duplicate it afterwards.

The trick is not to get things mixed up: 1. Delete all the Sharks, except the first one. 2. Get in the first Shark (not the Starfish!!) 3. Provide the first Shark with a sensor. 4. Make new Sharks.

You can also choose not to delete the Sharks. In this case though, after you make a sensor for one, you need to copy it for others as well. It often leads to confusion. So, it's easier to delete and then duplicate.

Slide No. 19

Create YOUR world!

You will be asked, why a Starfish is chasing Sharks to destroy them? In real life it's usually the other way round. Because it's my world! I'm creating it the way I prefer.

Well, the world is in place, now what about sensors?

A sensor looks like an azure strip. Also, it has the form of a neat stretched hexagon.

One more time: focus! We are in the sprite of the First Shark.

Get in Management and find the Conditional Operator bracket with the words 'If ... then'. Make a conditional operator of three parts. Pick up the 'If ... then' C-block and put it to the script field. Let it stay away from the general script for a while. Then find the Touches... sensor. Select Starfish from the list. Pull it to the C-Block. Try to get a left corner of the hexagon inside the hexagon of a bracket which is empty for now. When the empty polygon is illuminated with a white framework, let the LMB go. The sensor is almost ready. What does it give the program though? How does the Shark disappear?

Question to the chat. Does disappearance belong to the properties of appearance (1) or motion (2)?

Get in the lilac Appearance block. Find the Hide command. Insert it into the C-block. There a sensor in the assembled condition: If (Shark) Touches Starfish (Hide). Pull this assembly by the yellow block with the mouse, take it to the script in the area of the blue If Touches the Edge command and put this new assembly right under this command without leaving borders of the yellow Always Repeat C-block. In the end you are supposed to get a Conditional Operator inserted into the Cyclic Operator like a Russian nesting doll.

Duplicate the Shark 5 times. Start! Now Sharks swim and disappear, and if you start again, no one appears at all.

Sharks need to reappear though, for example, in three seconds. Delete all the Sharks. Insert the Wait For 3 Seconds command into the Conditional Operator and put the Appear command below. Start! Nothing again. That's because we haven't set the starting condition: Appear. The thing is, in the beginning of the game all Sharks need to be visible, and in the end of the previous session they disappeared, and this state has been saved in the script. Insert Appear under the check mark. Remember: We have one Hide command and two – Appear! Duplicate the Shark 5 times. Start! Everything works.

Slide No. 20

Question to the chat. Who defines the rules for the game you invent?

The Program for the Shark is Getting More Complicated. The final intermediate illustration.

Slide No. 21

Question to the chat. What does the word 'random' means?

Random Numbers. Size. The uniformity of Sharks will give the hump to anyone. Who will buy such a game?

Let's deal with starting conditions. Let all the Sharks be of different sizes. Find Set 100% Size in the Appearance block. Set a smaller number instead if 100. Click. The Shark gets smaller. Set a bigger number. Click – it gets bigger. You can set this command for each Shark with various values by entering them manually. Programming specialists never do this though.

Question to the chat. Write down any number above 20, but below 40. Good.

Correct. Such 'any' numbers are called 'random' numbers in mathematics.

Question to the chat. Write a random number from 1 to 6. Also correct.

Get in the Operators block. That's what the longest strip suggests: Give Out a Random Number from 1 to 10. Any number can be entered in fields. For example, Give Out a Random Number from 44 to 99%. Insert this command into the Set the Size ...% command. The resulted assembly is to be put under the very check mark. Click on the check mark of the script (not the shared one). We are in the first Shark, aren't we? Let's watch it. With every click the size of Shark No. 1 changes.

What does it mean? It means we should delete all the Sharks, except... Well, you know what next.

Slide No. 22

Question to the chat. Give a random number from 59 to 60.

Random Numbers. Color. We study digital technologies. Everything comes down to digits and numbers in robotic science. Even colors. The Set Effect Color command offers a list of various effects. But it's Color that is denoted by numbers from 0 to 255 in the field. Then act as per the same algorithm as with the size.

We'll get a shiver of sharks, and each time each shark will be of a different size and color. What a fine aquarium you've made!

Slide No. 23

Random Numbers. Direction. Insert the Give Out a Random Number from 77 to 122 operator in the Turn 60 Direction command.

Slide No. 24

Question to the chat. How many colors are there in Scratch?

Random Numbers. 'Personality'. All things you've already done with random numbers are about starting conditions. Now let's give random speeds and random leverages to the Sharks.

Slide No. 25

The game has become more spectacular. You can add the Delete Graphic Effects check mark as a starting condition. It doesn't have any influence though. The game will be developed further at the next lesson.

Slide No. 26

Physical Activity Break. Arranged a few times during the lesson at the teacher's discretion.

Of course, you know that one shouldn't work with the computer for a long time, because eyes and back get tired. Please tell, did you like the lesson? *(children's answers)*

Today we have learned about programming of computer games in RobboScratch3. All the characters of our game are virtual robots. When we learn to manage them as we wish, we'll learn to manage actual robots as well.

Have you understood why robot technicians need to learn how to make computer games? *(Children's answers)*

You've helped a lot today. You've been attentive and carried out tasks precisely. Well done!