

WEEK FOUR

Mini-Challenges

Lesson Goals

1. Team members will continue to learn to work together to solve a problem.

Technical Learning Goals

1. Team members will learn the advantages of breaking a problem into smaller pieces.
2. Team members will begin to learn how to break a problem down.
3. Team members will learn how to apply and integrate their learned skills in order to solve a large problem.
4. Team members will become familiar with the terms repeatable and robust.

Coach Preparation

Background Reading

- Rules to Coach By (included below).
- Solve the Problem a Little at a Time (included below).

Preparation Time : 0.5 hours

Lesson Preparation

Think about how you will help team members brainstorm solutions and divide the labor. (*You* don't need to solve the mini-challenges, you need to help *them* solve the mini-challenges.)

Preparation Time : 0.5 hours

Equipment Requirements

- At least one FLL Challenge Kit (2 RCX bricks would come in handy).
- A PC with the RCX software installed and the transmitter attached.
- A large area for the robot to run around in.
- A test area that has a black line that the robot can follow.
- A measuring tape.

Documents

Rules to Coach By

Solve the Problem a Little at a Time

Creative Problem Solving Exercise : Design a Team Logo

Technical Learning : Mini-Challenges

Rules to Coach By

ETHICS

- The intention of the FLL Program is to provide creative technical problems that should be solved by *only* the team members. Coaches and mentors can help advise on sticky technical problems and show skills.
- The coach should help set a good example of adult behavior, sportsmanship, and FLL spirit. Help students understand that winning is **not** the goal. The process of getting there is the important thing! The competition should not be the means to an end but a sharing and celebration of ideas!
- Guide the team using appropriate questioning techniques and allow discovery through experience of trial and error, research and skill learning.
- Avoid through direct instruction, any interference from parents who wish to give their ideas or solutions to a team.

DO ...

- teach your students the creative problem-solving process.
- teach your students the necessary computer skills if they don't have them (or find someone to do it for you).
- try always to answer a question with a question.
- be a quiet guide on the side not the sage of the stage.
- help them give and take constructive criticism of ideas without insulting and insensitive remarks.
- help them learn how to evaluate their ideas and progress.
- help them organize and learn the importance of keeping a schedule and meeting deadlines.
- help students recognize the abilities of each member and encourage them to capitalize on the individual strengths of ALL.
- help them expand their minds and brainstorm more creative ideas
- encourage growth through new experience.
- be willing to admit you don't know everything and encourage your team to seek other resources.
- let your team know you are human and also have needs.

DON'T ...

- tell them how to solve the problem, but ask questions which help them think it through.
- allow criticism of teammates personalities or physical attributes.
- step in on their disagreements, let them try to work it out as part of learning to work on a team, but also, don't hesitate to mediate disagreements so they have constructive outcomes.
- limit creativity by setting parameters which reflect your ideas.
- get upset when they make mistakes. This is part of the FLL learning process.
- make them feel like they failed if they don't score highly. FLLers only fail when they haven't tried
- complain about other teams, coaches, or judges.
- hesitate to ask for help. NO ONE can do this job alone!
- forget to have FUN with your team.

Solve the Problem a Little at a Time

be sure and share this information with the team

The first reaction of most kids is probably to build a robot to solve everything and then try to program it. This “big bang” approach rarely works well.

A better approach is to solve each problem independently. For example, you can prototype a robotic arm by building it and writing a simple program to test it. This is how products are designed and built because the earlier in the process that a problem is found, the easier and cheaper it is to fix. In scientific terms, you reduce the number of variables you are dealing with by building the one piece and writing the program to control it.

There are lots of statistics and papers in the engineering world that document the order-of-magnitude cost increase to fix an error each time you progress a step through the product lifecycle (requirements, design, build and test, manufacture, field support). Every day in the newspaper it seems there are reports of some type of security bug or other problem with some released software. It is much easier to fix those problems earlier. The way to do that is to build the project in pieces and verify each piece by itself. Once you know one piece is working, you can integrate it with another piece and then test how they interact. The same concept is true for FLL.

The team needs to decide how to break up the project. Maybe the whole team works on one area at a time, or maybe different groups work on different areas.

Creative Problem Solving Exercise

Design a team logo to go with your team name. You may want to tie it in with the Challenge theme. It would be great if you could get team shirts made up with the logo on it. This could take a long time to come up with the logo, so you may want to first open it up to the whole team then have a couple of members come up with one or two options that the team can vote on when they are complete.

Miscellaneous

To Create a Robot that Can Move Forward in a Straight Line

- Make sure the playing field is level.
- Make sure the robot is tight.
- Make sure the vehicle is built symmetrically.
- If the drive system has rubber bands, try switching them around.
- Make sure all axles rotate freely.
- If axles spin freely, try matching the motors.
- If no two of your motors are a speed match, call PITSCO LEGO Dacta for at least one replacement motor.
- You can also redesign your drive system to work on a single motor or hook up both motors through a single linkage.

Using the Rotation Sensor

You must set up the rotation sensor if you want to use it. You do this by selecting the following starting from the Main Menu (you will need your RCX brick to do this):

- **Getting Started**
- **Set Up Options**
- **Advanced**
- Check the **Unlock Rotation Sensor Watcher** option (a yellow checkmark appears when this option is selected).

Programming Restrictions

You cannot use multiple stack controllers (the red tiles) within a single program. You get around this by packaging them inside of the **my commands** and then stacking the **my commands**.

You cannot *nest* **my commands**. *Nesting* means that a **my command** is included in the stack of code for another **my command**.

We think that the 2.0 version has removed these two restrictions but you will have to test that on your own. We cannot recommend that you use this beta version (meaning it is not yet tested) as it may have bugs, additionally it may be difficult to find technical support for this version.

Mini-Challenges

As you begin to solve each of the mini-challenges listed below, keep this in mind:

- If the task is complicated, solve it in pieces.
- Always spend some time thinking about the problem before you start construction or programming.
- Before starting to write a program, write out an *algorithm* to solve the problem. Recall that an *algorithm* is a step-by-step plan on how to solve the problem but does not necessarily refer directly to your programming language.
- KEEP IT SIMPLE.
- Once you have completed the mini-challenge, test it several times in a variety of situations. If you can get the same results over and over under identical conditions, then the behavior is *repeatable*. If your solution works under a variety of situations, then your behavior is *robust*. Sometimes it is more important that a behavior is repeatable and sometimes it is more important that it is robust.

1. Build a robot that travels in a straight line. See how long you can get the robot to travel in a straight line. *Look at the hints above for moving the robot in a straight line.*
1. Go forward in a straight line, moving exactly 24 inches using:
 - the internal timer.
 - the rotation sensor (make sure the rotation sensor is activated).
 - the light sensor.

Which method is more repeatable? Which method is easier to program?

2. Follow a black line.
3. Move the robot in a square so that it stops at the exact location it started in. If you repeat this 5 times without stopping, does the robot still end up in the same location?
4. Climb over a textbook that is at least an inch thick.
5. Go forward, grab a small ball of tissue, come back and drop the tissue.
6. Go forward, grab a block, come back and drop the block.
7. Build a robot capable of *obstacle avoidance* using touch sensors. This means the robot will avoid obstacles by actually first bumping in to them, then backing up and moving around the object. Most likely, you will want a robot with bumpers. Refer to the last lesson or look in the Constructopedia under “special features” for bumper construction. This exercise will show the importance of building a sturdy robot as it bumps in to things around the room.