WEEK FIVE
Beginning the Challenge

Lesson Goals

1. Team members have a clear understanding of the Challenge.
2. Team members know the tournament awards and the criteria they are based on.
3. The Challenge is broken down into tasks that small groups can work on.
4. Team members are assigned/volunteer for these various aspects of the Challenge.
5. Team members create a list of weekly goals so that a clear plan exists to get the team to the competition.
6. Coaches need to make arrangements to find/build a practice table if they haven't already done so.

Coach Preparation

Background Reading

- The Challenge.
- Miscellaneous reading (included below). You may recognize much of this reading from the rough draft of the coaches manual that Fred posted on the web.
- Tournament Award Guidelines (included below).

Preparation Time : 1 hour

Equipment Requirements

- Large writing area that everyone can see.

Documents

The Challenge.
Miscellaneous Readings.
Tournament Award Guidelines.

Once you complete the lesson goals outlined above, you can let the various smaller groups start brainstorming to accomplish their tasks. You may need multiple adults to help lead each of the smaller groups, however make sure that anyone who helps understands the FLL philosophy and the rules of brainstorming.

Included in this lesson is a list of the tournament awards with criteria guidelines and hints for the kids to meet those criteria. Make sure everyone in the team understands the criteria of the awards.

Acquired Skills Through the FLL Experience

As you begin to work on the Challenge, keep in mind the hard and soft skills that the kids should be learning through the FLL experience.

Hard Skills

Hard skills are the abilities we learn that produce direct, immediate physical results that we can see, hear, use, or otherwise enjoy. Generally FLL will be team member's first experience with programming and solving a problem with a technological device. Some of the hard skills they will learn with FLL are analysis, strategic planning, experimentation, problem solving, designing, building and basic mechanical design principles, basic programming principles, and the concept of an embedded computer controlling a device. These skills will add value to your robot and to future projects.

Soft Skills

Soft skills are the abilities we learn that produce results which are harder to perceive, but which are essential to support the hard skills. Examples of soft skills are being on time, tact (don't be negative with criticism), teamwork, focus, confidence, compromise, courtesy, perseverance (never give up), and respect for other solutions and teams. The successful team will learn all of these and more with FLL. These skills will add value to your team, your process, and your future. Don't overlook soft skills. They can make a big difference in the success of your team. You have probably heard athletic coaches talk about "good team chemistry". That means the soft skills on that team are working well.

A favorite story is from a coach asking his students at the end of the year what was the most important thing they learned from the season. One particularly bright child said, "I learned that other kids can have good ideas too."

Learn the Problem Solving Process

One of the most important skills to learn in FLL is the problem solving process. For kids, it can be simplified as following:

- Define the problem.
- Brainstorm solutions.
- Evaluate solutions and pick one.
- Implement the selected solution.
- Evaluate the solution once implemented.

The most important step in the problem solving process is defining the problem. This seemingly simple statement still fouls up everyday engineering problems. No matter what you are doing, know why you are doing it and what you are trying to solve. Is the problem to go around the course or to go around the course as fast as possible? Knowing what the problem is will make a big difference in how you approach a solution.

More Rules to Coach By

- **<u>First of all, relax!</u>**
  - Keep a positive attitude.
  - **Have fun!**

- It is your responsibility to instill the FLL spirit in your team. Remember, if you emphasize the score as a measure of their success, you could be setting up your team to feel like failures. Please emphasize that the learning process, the experience, and FUN are the worthwhile goals to achieve.
  - Always, always stress to the kids that getting a solution and showing it at the tournament makes the team a winner.
  - Never, never make winning a priority because you only end up with disappointed kids (and parents).
  - FLL is about the process of preparing the solution -- not the result of a competition. It is counterproductive to try to set goals like "win the competition". This way, the kids can feel they have met their objectives for the year regardless of the outcome of the competition.

- Remember, this is the kid's solution not yours.
  - Do not lose sleep at night worrying about your team's solution in comparison to others. The team is solving the problem, not you.
  - Do not rack your brain to think of your own solutions. This makes coaching more frustrating than it already is.

- Teambuilding is important. It is difficult to be creative "on schedule". Sometimes, just letting the kids have some fun together will allow them to develop better communication and respect – which will lead to smoother progress when work resumes.

- Remind the team that respect for teammates is important. No criticism is allowed!

- Understand the rules. Ask for clarifications if you don't. Help the kids to understand the Challenge as thoroughly as you do.

- If you feel lost, ask for help online. Chances are someone else is in the same position.

- Celebrate your accomplishments!!

- *This is so important, we had to list it twice*: **Have fun!!**

Comments from Coaches

“Anyhow, from the competition we learned one very important lesson - don't worry about implementing the "ideal" solution - just get the job done!”

“I think there's a trap if you let the kids try to chase some sort of elegant solution -- they spend so much time on design and construction that they don't have the time to test and practice before the contest. Also, the kids have less input as a team if you freeze the design too early. I pushed the kids to get something (anything!) built ASAP that moved reliably under its own power. Then we worked on the missions, modifying the device to do what the missions required. This gave everyone a chance to make observations about how to make it work better, though a few leading designers did indeed emerge.”

Gears vs Pulleys:

I think it's easier to build a robust and accurate device using gears. If the kids match the gear sizes correctly, the whole assembly snaps together tightly. When the motor turns a gear to turn a wheel, there's no risk of inaccuracies due to slippage. There's also no risk of elastics pulling the device apart when the kids put them on.

Straight Lines:

I think there are two crucial tricks behind straight lines. First, you must build the whole assembly of wheels, gears, and motors as rigidly as possible. Slack, backlash, and loose parts just won't work. Don't use pulleys because they aren't as rigid as gears, and any slippage will throw it out of line. Second, use exactly the same gears to drive both wheels. If you do these right, the robot should run straight and true when you run the
built-in RCX program #1. The program runs both motors at the same speed and in the same direction. If the wheels and gears are matched and tight, then the robot runs straight.

Our team played a variant of the "hot potato" game that tested this: the kids sat in a large circle, one kid would aim the robot at a "gateway" another kid made with hands or feet, and try to run the robot through the gateway. The other kid would intercept the robot, turn it around, and send it off to another kid. This gave the kids experience with the moving robot and (important) experience dealing with parts coming loose or falling off.

Measuring Motion with the Rotational Sensor:

One of our team members, Alex, had taken several LEGO Logo classes at the Science Museum and had soaked up one of their cardinal rules – there should always be a gear or pulley connecting a motor to a wheel. I don't know the rationale for this, and the Constructopedia suggests that this isn't true for the RCX. Alex insisted, and the resulting robot worked well.

As a practical matter, you need to drive your wheels with gears or pulleys if you are using the rotational sensor to measure your motion. To get the best accuracy, the rotational sensor should rotate in conjunction with one of the driving wheels. Then the motion of the wheel on the ground is fed directly into the rotational sensor. It's less accurate to measure motion any other way.

My own bias is in favor of gears, since they yield tighter and more reliable devices than pulleys.

Precise, accurate turns:

Our robot made sharp 90 degree turns by pivoting around a center point located between the two powered wheels. This was the easiest and most reliable way to do it. The robot would essentially rotate in place when we ran the two motors at the same speed but in opposite directions.

We made sharp 90 degree turns by counting rotations using the rotational sensor, which was geared to one of the driving wheels. It took some trial and error to find the right number of rotations for various turns (we used specialized turns to grab the escape latch) but the sensor provided the degree of accuracy we needed.

Another trick - we used rubber tires on the wheels attached to the motor, but did **not** use rubber on the other wheels. The rubber tires ensured accurate motion by the powered wheels and the omitted rubber prevented the non-powered wheels from causing too much drag when the robot turned.

This worked well enough to earn us fifth place at State, though it didn't get us into the final four. We noticed that two of the top scoring robots (the ones we would have had to beat) didn't make much use of 90 degree turns.

Software

Back in the dark ages of programming they taught programmers to use flow charts. When planning out your program, use a couple of methods to help your kids visualize this. Have one of the kids pretend he/she is the robot. Have them walk through the course. What do they do each step at a time? What program step should I use to do this? Then use a simple flow chart or something to write it on the board.

If you are using RoboLab, you'll have to go to at least *inventor level three* to get the same functionality as the standard RIS programming environment. What you get when you go to *level four inventor* is what they call "containers" (variables to the rest of us). RIS 2.0 also supports variables.

RoboLab vs. RIS

Many people think RIS has a shorter learning curve.

RIS works only on Windows machines, whereas Robolab works in Windows or Mac environments.

You can print your program using the RoboLab environment. With RIS, you can do screen prints only with the PrintScreen key on your keyboard.

RoboLab uses something called *jumps*, rather than the loop type structures the RIS environment uses. You can do the same things so there is no advantage to either one except that the RIS method is probably more clear to kids.

RoboLab allows you to use multiple threads like the RIS environment (you connect the sensor watchers on to the regular program) except you use forks/splits in RoboLab. Again there is no advantage either way.

-

As you start to think about the design of your robot, remember

- Use whatever resources you have. Using a design found on the internet or in a book is not cheating. Looking at a variety of designs can really help generate ideas for your own design as well.

- Look to the real world for examples. Look for real-life machines that do some of these things. How do they work?

- Design, build, and test one thing at a time.

- Spend time thinking about a good design for the Challenge but not too much. You will probably have to change the design several times no matter how well you plan the design. It is better just to build something to see how it works, and then perfect the design as you go along (this is called *prototyping*).

- You may want to add or remove parts between missions.

- Communicate with the programmers. Each group has to know what the other is doing to be successful.

- You will have to exchange batteries often, so design your robot so that the batteries are easily accessible.

- You have three motors. Do you need them all?

- KEEP IT SIMPLE. The simpler the design is, the easier it will be to build, to take apart, to rebuild, and to redesign the robot.

Programming

As you start to think about the design of your program, remember:

- Make an algorithm before you start to code. A good way to do this is to have one member act like the robot performing each of the steps necessary to fulfill the mission. Write down those steps.

- Create ***mycommands*** whenever the same code sequence is used repeatedly throughout the program or a stack of code can be packaged together to make the code easier to read.

- There are often multiple ways to achieve the same thing. Think about which option is best *given the situation*. If you find the best solution through experimentation (e.g. you find that using the timer is better for turning than the rotation sensor), make sure that you talk about this when presenting your program to the judges. It will show that you know the various functions of the language.

- You may (and probably will) need multiple programs for the various missions. Tackle them one at a time. Decide which mission is most important to solve first and go from there. Don't assume that the one that achieves the maximum points is the one you should accomplish first. It's worth a lot of points because it is *hard*, and probably not a good place to start. (When you download a program, you can indicate which of the 5 program slots to save it to. So far we have been saving to just one slot.)

- Design, build, and test one thing at a time.

- Communicate with the robot designers. Each group has to know what the other is doing to be successful.

Presentation

This award celebrates understanding of the underlying science behind the FLL robotics challenge. The objective is to create a well-rounded team.

Judges will be looking for the following things in your team's presentation:

- What type of material are you collecting in the samples you are retrieving with your robot? Could these samples be used to predict volcanic activity? If so, how?
- Your team is using lava barriers to help protect the village from lava flow. Could these types of barriers work in real-life? What would they need to be made of?
- In one of your missions you will be attempting to place a gas sensor in the crater. What type of gases would be useful to detect? How would you use the data?
- How would you construct a sensor in real-life to withstand the environment within volcanic crater?
- Based on your research and your assumptions about the volcano you are studying, what type of eruption, which you expect?
- Did what you learn about volcanoes affect the way, or order, in which you performed your missions within the challenge? How?
- Are marbles is a good simulation of the lava flow, eruption you would expect? If not, what would you suggest?

This is a lot of questions and we don't expect teams to answer them all, but this is what the judges are looking for. There are many different answers here and in some cases, there is no right answer. Also there are other questions the team may decide to present. Mostly, the judges are looking for a clear development of a hypothesis backed by research and an indication that your team had clear steps in their research. **It would be best for you to present less information that you understand very well, than to present a lot of information without the understanding**.

As you put your presentation together keep in mind what makes a good presentation:

- It has a beginning, middle, and end.
- It is clearly understood by the presenters and in a language they understand.
- It is in the presenter's words, not quotes of references.
- Information presented is relevant.
- Information is clearly presented (i.e. the presenters don't mumble and faces are to the audience).
- Main points are clear and the judges don't have to hunt to find them
- You tell a story.

More can be found at http://www.hightechkids.org