# Team Management FAQ

## How to Work Together

## 1 FAQ's

1. **Q: What is a good number of programmers to have on a given team?**
   **A:** It is generally good to have two, or maybe three, of your best programmers writing the bulk of your code for your team. Two people helps avoid mistakes and working together will help give each other a constant reference for any questions or anything that is unclear. However, it would probably be a waste of time to have everyone on the team write a full program as lots of time could be spent debugging on trivial mistakes.

2. **Q: Can/should everyone test adjustments to the strategy on the IDE?**
   **A:** YES! Because of the ability to share your code with the rest of the team on the IDE, everyone on the team can easily get access to the master copy and make small adjustments to test strategy, decrease code size, etc. Ideally, everyone should be able to make adjustments to the main code, test specific strategy improvements, and be able to report back to the team. It would be annoying and a time sink if every single little strategy addition that may or may not be positive had to go through the main programmers in their code.

3. **Q: Would it be helpful to draw or write out strategies on paper before putting it into the code?**
   **A:** Absolutely. Debugging in the IDE can be frustrating at times as you may not be completely sure what goes wrong in the simulation, so planning out the logic of the code beforehand is very important. Groups of 2-4 people visually brainstorming strategies on a whiteboard, big pieces of paper, etc. usually works very well, and can then report their strategy for the programmers to simply plug into programming syntax. Especially because simulations take at least 3-5 minutes to run, breaking up into groups and having a robust code beforehand is encouraged for time purposes.

4. **Q: How should we go about writing the functions involving more math and physics concepts?**
   **A:** Like with strategy testing, this is not something you want to "brute-force-guess-and-check" on the IDE, as it will take up too much time and is unreliable. Having 3-5 people good with math and physics work together on a big writing display conceiving and implementing the proper math functions is an effective procedure. There is definitely some give and take with the functions of the game regarding code size and syntax, so make sure to be cognizant of your limitations for your math functions.

5. **Q: Based on how you split up your team, there may appear to be different roles that materialize. What are some examples and what are expected of those roles (these roles can overlap they do not have to be so cut-and-dry, but just for an example)?**

**A:**

- Programmer- The programmer should be comfortable with syntax and putting strategies into a correct, succinct programming format. Most of the thinking and logic should already be presented by the strategist and mathematician, so this job should not be any more involved than the other two, just putting ideas into the computer. Do not be mistaken that the programmer should have to do all the work!
- Strategist- The strategist's job is to be creative and keen about ways to navigate the game to maximize the team's utility. Strategists should be able to analyze other strategies and adjust accordingly, but also exhaustively consider as many strategies as possible. This is a very big picture job.
- Mathematician- There will likely be some tricky mathematics functions your team will need to employ in the game. The mathematicians should be able to use their math skills (coordinates, angles, vectors, etc.) to understand how the function should work and create a strategy for coding them, taking into account the limitations of the IDE/game environment.
- As a team, the strategist would create the sequence of actions for the satellite to follow, the mathematician would create the specific functions (if not pre-defined) that allow the satellite to perform such actions, and the programmer would be able to communicate this sequence and these actions to the computer.

6. **Q: Is it useful to run some of your different strategies against each other in the IDE simulations?**
   **A:** This is a very useful strategy to employ. Often times there are subtleties that come up when you are racing another live satellite instead of just going through the motions by yourself. Depending on how you break up your strategizing, 1-2 people running simulations against each other analyzing to report back to those strategy groups works very well. Be careful though, as it is easy to just get caught up in beating the other satellite in any given round- the goal of this should be to identify errors or discrepancies to your code in general.
7. **Q: It seems splitting up the group in teams is a good strategy. What is the importance of meeting as one big group?**
   **A:** Coming together offers the smaller groups to get an outside perspective as well as helping get everyone on the same page. In the end you are all working towards one final goal, so frequent reunions in one central location is essential for long term goals and unified teamwork. Presenting your work on a projector or bigger screen as a big group is a great way of reflecting and moving forward while adequately informing your teammates.

## 2 Additional Information

The team management resource on the Zero Robotics Website is a helpful tool to organize and delegate members. Check it out!