Computational Models

A critical question for the Computational Thinking Competition is "what can we use as the computational model for our entry?" The answer is that there are a lot of computational models that can be used for most project entries. It really comes down to what do you already know and how clever is your approach. Below, we will look at a very simple example of a project and some of the computational models that you could use.

Note that the project and its solution given below ignore some science issues to make it easier for you to see what we mean by a "computational model." For example, this sample project looks at the 3-D problem in only two dimensions. Also the asteriods are assumed to be stationary.

Project Description: My science project is to determine the best maneuvering strategies for navigating a space ship through an asteroid belt in space. I am going to need to run several tests in order to be able to put together a convincing result. I decide that I am going to have to capture the layout of space and try different maneuvers for each test.

I want to use computational thinking to develop my science project. That means that I will have to think like a computer scientist. To do that, I first need to fully understand the problem I am trying to solve, before I worry about what kind of computational model I will use.

The first thing that I will do is to look at the requirements of my science project. Before I can start thinking about computational models, I have to truly understand my project. I will need:

1. to be able to create different layouts of the asteroids and the starting position of the space ship in space;
2. to be able to test different sizes and shapes of space ships;
3. to be able to "fly" the space ship.

Any computational model that I consider will have to support these requirements. As a computer scientist, I need to really understand these requirements. One way to do that is to define the problem so that we know what the different parts (entities) of the project are, and what sort of instructions can be used to manipulate those entities so that I can meet the requirements of my project.

Entities:

- I need to have an abstract model of space.
- Asteroid: I need to have an abstract model of an asteroid, including size and shape.
- Space ship: I need to have an abstract model of a space ship including size and shape.

Instructions:

- create space, create asteroid, create space ship
- place asteroid at a point in space, place space ship at a point in space
- a set of move instructions for flying the space ship (in 2D, for example, they could be moveUp, moveDown, moveForward, and moveBack).

Now that I have the requirements of the project, I can think about the different computational models I could use. What are my choices for the computational model?

In order to do the testing of the project (how the spaceship can move through space and not hit the asteroids) I need to have a concrete computational model. Depending on my background I have several choices. Four of the choices are shown below.

A. What if I don't know anything about computers, but I still need to get a good grade on the project?

I decide that one way that I can provide an abstract model of space is to use a white magnetic board (e.g., Figure 1) that has squares to represent points in space (I have just modeled the concept of space in two dimensions!) Using asteroid cutouts with a magnet glued to them gives me an abstraction of the real asteroids. I can do the same thing for the space ship. Properties like size and shape can be handled with construction paper and my scissors. I can show where the asteroids are in space by just placing an asteroid (or space ship) that I made at a point on the magnetic board.

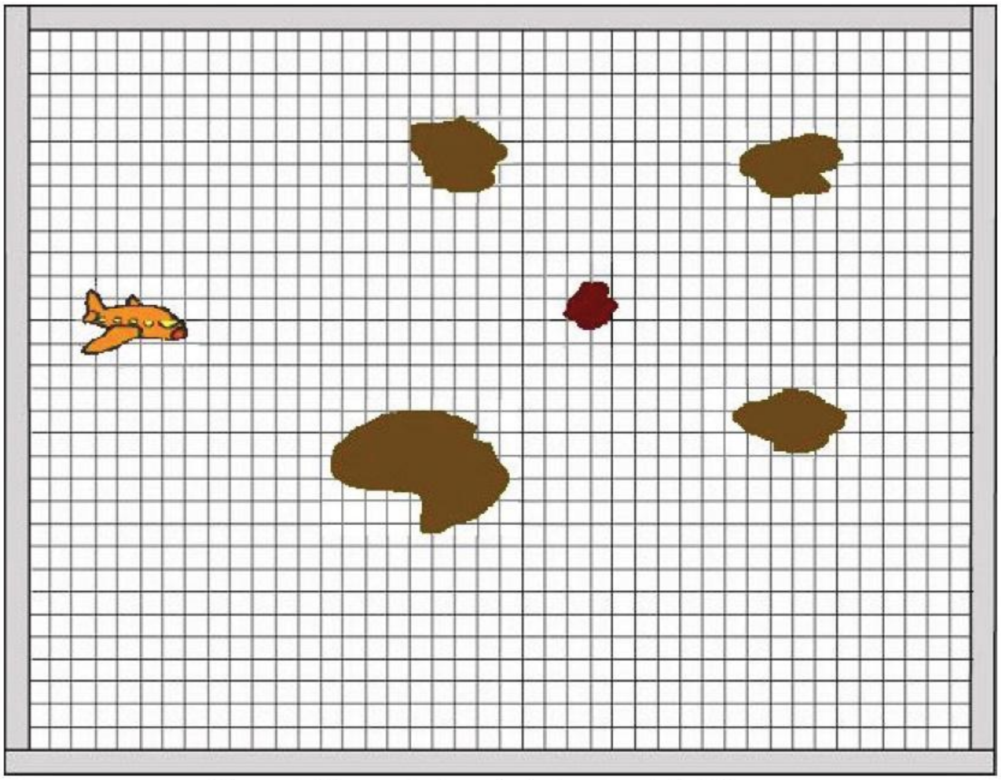


Figure 1. Magnetic board abstract model of Space.

The move instructions can be shown, for example, by using "moveForward(3)" as moving 3 squares forward on the magnetic board. The same can be done for the other move instructions.

Note that now I can define a maneuver in terms of a set of instructions, e.g.,

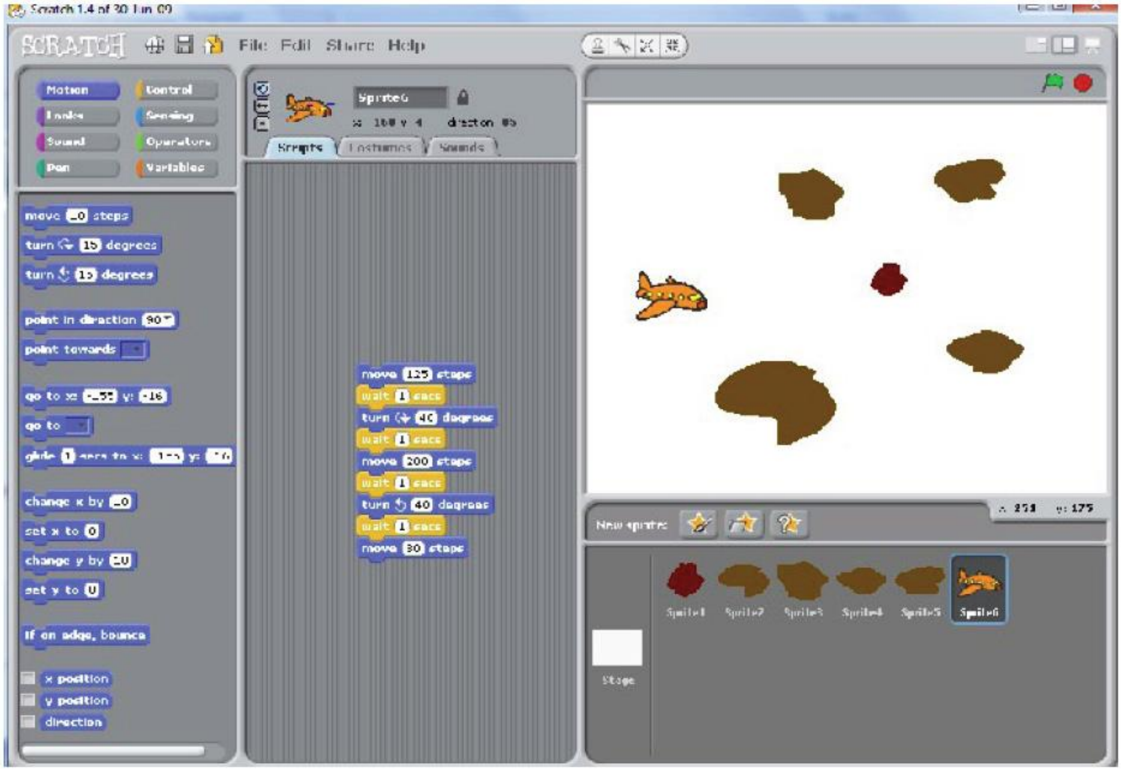
```
moveDown(3)
moveForward(5)
moveUp(6)
moveForward(4)
```

I can move the space ship entity on the magnetic board based on the instructions and see what happens. If the maneuver misses the asteroids and makes it through the asteroid belt, I can define it as a successful maneuver. If I collide with an asteroid, it is probably not so good. At the end of my testing I will have a set of maneuvers and how they worked. By keeping track of the starting positions of the asteroids and space ship on the board, I will also have the test conditions for each maneuver.

B. What if I have been playing with Scratch (scratch.mit.edu)?

Could I use it for my project and still be using a computational thinking approach? Sure, I just need to be able to define the parts of the project and the instructions in terms of what's available in Scratch.

I can use the stage (sprite window) to create the abstract model of space. The "create asteroid" instruction can be implemented by adding an asteroid sprite to the sprite window. The same approach can be used for creating a space ship. Properties of the asteroids and the space ship can be defined by changing the size of the sprites. The placement of the asteroids/space ship can be done manually to set up what I want for the starting condition for each maneuver test. The move instructions defined in the first example of a computational model (with the magnetic board) are replaced with the Scratch program.



```
move 125 steps
wait 1 secs
turn ↻ 40 degrees
wait 1 secs
move 200 steps
wait 1 secs
turn ↺ 40 degrees
wait 1 secs
move 30 steps
```

Figure 2. A simple maneunver using Scratch.

Maneuvers would be defined as Scratch programs. The Scratch screen print (Figure 2) illustrates the current state of space (what we see in the sprite window) and the "space ship sprite" program defines the maneuver. I can document each test by taking a screen print just before I start the maneuver for the particular asteroid layout. Notice that there are a variety of ways that I could program the maneuvers. Figure 2 uses a pretty simple (but not very clever) approach to maneuvering the space ship through the asteroid belt. Note that Scratch is just one of many possibilities. Software like Greenfoot (available at http://www.greenfoot.org/), could be used in a similar fashion.

C. What if I have been working with a robot as a hobby?

I decide to use the robot to represent the space ship and an area of the floor to represent my abstract model of space. I can use any object (rock, stack of books, waste can, etc.) to stand for the asteroids. The size of the object will be the same as the asteroid size property. The "place asteroid" and "place space ship" instructions can be done manually by physically placing the asteroid objects and the robot in the layout that I want to test.

Since I have to use the robot instructions to actually move the robot, I replace the move instructions with the move instructions that I use for my robot. "Space ship maneuvers" would be defined as the programs used to maneuver the robot through the obstacles used to represent the asteroids. I would be able to document the layout of space either by taking a picture of my model of space (the floor area) or by sketching the layout on paper just before starting each maneuver.

D. What if I know how to program in a programming language like Java (or Python, etc.)?

In this case the computational model is straight forward. The programming language provides the ability to define the entity abstractions I need (space, asteroids, and space ship) and the language's instruction set gives me all of the instructions I will need to implement the requirements of my project.

I can define the entities required by my project in terms of classes and the instruction set of the computational model can be defined by the methods assigned to the classes representing space, asteroid, and space ship.

My program would have to printout the layout of space and the java statements executed to implement the maneuver in order for me to document each test. There would be many ways to create the program to demonstrate how the space ship maneuvers worked for each test. Notice that some approaches to writing a program for my project would be considered clever, but not all programs I might create would automatically be more clever than any of the other three approaches discussed above.

E. Additional Note:

Another type of computational model that doesn't work very well for this project example, but is a viable choice for some projects makes use of existing software that is directly related to the project that I have chosen. For example, a project in a mathematics class might successfully make use of the MatLab software package as the basis of the computational model. Projects that require a computational model based on statistics could make use of software packages, like Jmp, SAS, or SPSS as the basis of the computational model.

COMPUTATIONAL MODEL CHOICE:

Now that I know there are numerous choices for my computational model, the critical question is which computational model will give me the best chance of winning the competition? Is one better than the other? Not really, in the end the question comes down to how clever you are and what experience you already have. You can develop a very clever computational model that doesn't use a computer at all. You can also develop very clever computational models that make use of computers and even programming languages. Your solution to the problems posed by your project can be clever or mundane!

The competition judges will judge your project on four issues:

1. **the difficulty of the project you have submitted;**
2. **how clever you have been in selecting and implementing your computational model for your project;**
3. **how clever your solution is; and**
4. **how well you can describe your project.**