Programming languages make extensive use of program variables. Program variables are a bit different than the variables you've used in algebra or in logic. A program variable is really just a name for a piece of computer memory. Programs include assignment instructions to cause data to be stored in the cell(s) of memory associated with a variable. In this lab you are going to answer several questions about variables and data as used in Alice (and most other programming languages).

An Initial Scene

Begin your experimental program by creating a scene as shown below, allowing the three fish to use their default names of blueTang, pajamaFish, and clownFish (left to right).

Q1: How is a variable declared in Alice?

In Alice a program variable can only be used after it has first been declared. Declaring a variable is accomplished by dragging the "variable..." tile from the bottom of the window and dropping it into your program at the desired position. (Note that a variable must be declared prior to being used in any other instruction, and that you only declare a variable once.) Every variable must have a type. The type restricts the kind of things that the variable can store. For example, a variable of type WholeNumber can store only integers and a variable of type Quadruped stores only Quadruped kinds of objects.

Begin by creating a variable named bluefish to store a Fish object. Your variable declaration should be dropped at the top of your program and be defined according to the window to the right. The class for this variable comes from the Gallery Classes and is called Fish. Note that Alice requires every variable to be assigned an initial value and it uses an arrow (near the top of this window) to indicate what value has been assigned (blueTang in this case). When you click OK you will see the instruction that declares and assigns the bluefish variable.

As you might guess, we can declare multiple variables in our program. To verify this create a second variable with a value type of Fish, a name of redfish and an initial object of clownFish. Your program should end up looking like the program to the right.
Since variables name memory that contains data, we can use a variable name to refer to the data it stores. For example, you can select either bluefish or redfish from the pull-down menu just above the “Procedures” tab. Select bluefish (not blueTang) and insert the following instruction at the bottom of your program.

Run this program to verify that bluefish must be storing the blueTang object, because blueTang rolls.

**Q2: Can a variable store different things at different times?**
Once you have created a variable, it is possible to insert assignment instructions, using that variable, into your program. In Alice you insert an assignment instruction by dragging and dropping the assign tile. Insert the following instruction immediately before the bluefish roll instruction (i.e., make it the second to last instruction in your program).

When you run the program, notice which fish rolls.

**Q3: Can a variable store more than one thing at a time?**
Next insert another assignment instruction and a second roll procedure, so that the complete program is as shown to the right.

Notice the impact that an assignment instruction has on previous assignments. You should conclude that variables do not "remember" data that were assigned prior to the most recent assignment. In other words a variable stores its value only until the time that the variable is assigned a different value. This is consistent with the idea that a variable simply names a piece of computer memory, and that memory can only store one thing at a time. If you change what is stored, then the old content of the memory is replaced.

**Q4: Can different variables store the same thing?**
At the end of your current program’s execution the bluefish variable has been assigned pajamaFish. Now let's see what happens if a second variable (namely redfish) is also assigned that same pajamaFish object. We can check this by appending the following three instructions to the bottom of the program.
Q5: What happens when one variable is assigned to another?
You may have noticed when you create assignment instructions, that variables can be assigned the value of other variables. Try executing a program once the following instructions are appended to the preceding program and you should discover what happens when one variable is assigned to another.

When this code executes redFish and blueFish are both synonyms for clownFish. However, recall from before that assigned values are transient (no memory). So assigning a new value to either variable (redFish or to blueFish) will have no impact on the value stored by the other variable. You may wish to add a couple more lines to your program to convince yourself of this. The key thing to remember about assigning one variable to another is that it is the data that is assigned and not the variable. Following such an assignment, the variables store the same data (object or value), but there is no other relationship. (I.e., future assignments or use of one variable are independent of the other.)

Q6: What about built-in (primitive) data types?
Most programming languages have two kinds of data types - those that are simple, primitive data type that are built into the programming language and those that are general purpose objects that are declared from libraries. In Alice the general purpose libraries are those used in set-up mode, including the fish. However, Alice also makes use of four primitive types: (WholeNumber for integers, DecimalNumber for real numbers, TextString for segments of text and Boolean for logical values). You have actually been using primitive data in several procedures. Add the following two instructions at the end of your prior program.

It is best to think of a primitive variable, such as distance, as storing the number it is assigned. Therefore, moving blueFish forward by distance is the same as moving by 1.0 units.

Q7: What about manipulating primitive data?
As you probably suspect, numeric data can be manipulated with basic arithmetic, such as addition, subtraction, multiplication, division, and so forth. You can use arithmetic any place that a numeric value is permitted. For example, if you click on the distance variable in the last instruction above and select Math from the menu, then you will see how Alice incorporates arithmetic operations. Change the last instruction so that it looks as follows:

How far does blueFish move when this instruction executes?
Q8: What happens when a primitive variable is updated?

Just like other variables, you can insert assignment instructions to alter a value assigned to a primitive variable during program execution. Insert the two instructions below to the end of your program:

In order to accomplish the assignment instruction above, you must first assign, distance the value of distance as a place holder, and then click on "distance" to select the Math options.

You should notice that this assignment changes the value stored in the distance variable to a negative value. Also, note that moving forward by a negative amount means to move backward. It is common for the values assigned to primitive variables be altered during program execution. For example, a variable to store your credit card balance must change every time you make a new charge or payment.

The last thing to notice about primitive variables is that each one is independent and stores its own value. When you assign one primitive variable to another it gets a copy of the same value, but changing the value of either variable has no impact on the other. Try declaring a second DecimalNumber variable and assign it a value like the following:

Now devise an experiment or two to convince yourself of what value is stored in each variable. Just like with Q3, note that variables do not remember the results of older assignments; only the most recent assignment matters.