**Modelling Memory and Foraging Lesson Plan**

**Learning Goals:** Learn the basic building blocks programming. Students should be comfortable writing functions, using iterative loops, recursive functions, and R objects, and using these tools to flexibly solve problems. At the end of the quarter, students should have all the tools to write an agent based model in which an agent (represented by an R object) moves between and interacts with several spatially explicit resources. Students should also be comfortable with R troubleshooting: using the CRAN and stackoverflow resources, identifying error sources, and identifying solutions to programming problems.

**Week 1 – Introductions, Set-up, and Data-Structures**

Goals: Be able to identify what the structure of an R variable is and what type of data it contains. Be able to manipulate and convert data structures. Understand indexing for various structure types.

Set up a GitHub account and learn how to share, clone, and fork projects with other team members.

Functions: concatenate(), vector(), list(), data.frame(), array(), as.vector(), is.vector(), as.list(), etc. . ., numeric(), integer(), character(), logical(), is.numeric(), etc . . ., length(), str(), class(), head(), summary()

**Week 2 – Writing a Function, If/Else Statements**

Goals: Be able to write functions that can run a series of operations. Be able to control the logical flow of these functions using If/Else Statements. Learn best practice for documenting functions, including inputs, outputs, and a function description. Practice using “?” to find and understand R functions.

Functions: function(), return(), if(), else(), logical operations (>, <, ==, &, ||, !, etc. . .), mean(), sd(), sum(), range(), ?, ??

**Week 3 – Writing a Recursive Function**

Goals: Learn to write a function that calls itself. Understand how to use the functions to break problems into the smallest possible increments and solve in steps. Be able to use stop commands effectively, and understand how to avoid maximum recursion. Learn to use additional functions for manipulating data structures.

Functions: paste(), nrow(), ncol(), cbind(), rbind(), names()

**Week 4 – Recursive Functions, cont., Plotting**

Goals: Gain additional practice writing recursive function and solving more complicated problems. Learn some additional R functions that are generally helpful. Be able to create basic scatter, line, and bar plots.

Functions: sort(), order(), unique(), diff(), append(), rev(), rep(), identical(), plot(), hist(), lines(), segments(), curve()

**Week 5 – Writing a Loop: For & While**

Goals: Be able to use “for” and “while” to create loops that repeat a series of operations over a series of values. Understand how to loop over both values and their indices effectively.

Functions: for, while, in, %in%, which(), abs(), seq(), sequence(), floor(), ceiling(), round(), signif(), trunc()

**Week 6 – Using the apply function and its variants**

Goals: Learn how the apply function iterates a loop, and understand why it is faster than using “for” or “while”. Understand the limitations of the apply function, and when it can’t be used. Be able to select between variants on the apply function, and understand their differences in usage.

Functions: apply(), sapply(), lapply(), mapply()

**Week 7 – Object Oriented Programming: Initializing Objects and Setting Class**

Goals: Learn the about the pros and cons of using OOP. Understand how classes work and how to initialize objects. Be able to create class specific methods and use objects of different classes simultaneously.

**Week 8 – Object Oriented Programming: Attributes, Slots**

Goals: Learn about object slots and attributes. Be able to use these tools to achieve more sophisticated tasks using OOP including: Create objects that move through space gathering energy and reproducing, simulate a hawk-dove game, create objects that can adjust their movement based on their environment.

**Weeks 9 & 10 – As Needed . . . 2plyr, ggplot?**