Sean Connor 19 June 2018

1. Describe a scenario where you see recursion at play in your daily life. What are some of the key features of this process or activity that make it recursive; Reflect on the defining characteristics in the lecture. For each example, how is this different from an iterative process.

One possible example of recursion in my life is driving. Often, there are many possible routes to take that vary in quickness based on traffic levels. I recursively implement the method navigate(current location, destination). At each point of diversion, like an intersection, I call the navigate method to determine which is best until I reach my destination.
- if (current_location == destination) return arrived;
- return navigate(current_location + 1, destination);
This could also be implemented iteratively using a while-loop (i.e. while(current_location != destination) and if statements, but I tend to think of navigating as a dynamic activity that is constantly changing based on conditions.

2. Describe a second and different scenario where you see recursion at play in your daily life. What are some of the key features of this process or activity that make it recursive; reflect on the defining characteristics in the lecture. For each example, how is this different from an iterative process.

One thing almost every human does every day is walk. This can be thought of as an extremely simple recursive method. I call the walk() method, and continue calling walk until the base cases are satisfied – that is, until I have reached my destination or I am unable to continue walking. Again, this could be implemented with a while-loop and if statements, but it seems that every recursive function can be implemented iteratively. The defining characteristic in this case is that I keep calling walk until a base case is met.

3. Are there any scenarios you can envision in which recursion would really be impossible, that the process is so iterative that you could not look at it recursively at all?

From what I understand, this is highly dependent on the number of recursive calls that would have to be made and the computer hardware. All recursive functions can be made into iterative functions and vice versa, but the recursive functions will generally be less efficient.

4. What are some good, practical ways to organize your thoughts and ideas when working with recursion to manage the process a little more easily.

The most important idea in my opinion is to start by defining the base case. After this, it makes thinking of how to implement the rest of the cases recursively much easier. Once the base case is established, think of how the same function could be used one step simpler to move toward objective. For example, in the homework when calculating the average of the array, it was almost like recursively iterating through the array until the base case was reached.

Sean Connor 19 June 2018

5. A lot of things have a LIFO characteristic. What is something that does not have a LIFO characteristic that you could still effectively manage with a queue?

LIFO is last in first out, and generally refers to stacks. Queues are FIFO, and can effectively manage many things. The things I think of first is a line at a store. The first in line is the first to be served.