INSTRUCTORS MANUAL
Computational Loops

Goals:
Students should be able to:

- explain simple computational loop structure,
- write simple computational loops,
- explain how Euler-Cromer integration works, and
- write Euler-Cromer integration loops.

This tutorial is based on:

- Original work written by Danny Caballero and Steve Pollock.

Materials needed:

- Laptops or Workstations with Mathematica installed.

Special Instructions:

- None.

Homework Connections: Homework 4, Problem X

Reflections on this Tutorial:

We advertised this tutorial for 2 weeks, reminding students to bring in their laptops with Mathematica installed. On the day of the tutorial, 90% of students brought in their laptops. Students without laptops were asked to pair up with another student.

We completed this tutorial in about 30 minutes with most of the class completing the last page in class. Students struggled with the syntax of Mathematica `for` loops (i.e., the difference between semicolons and commas). We attempted to make this more clear by using carriage returns between the sections with commas. A few students framed the syntax in this way, "The commas separate the big ideas: initial conditions, testing, updating, and calculations; the semicolons separate similar actions (i.e., setting different initial conditions)". This explanation seemed to make discussion of the loop structure much easier for us.

In Parts 2 and 3, students were asked to explain how to change the algebraic representation of a differential equation into a computational one (i.e., $dx/dt = v \longrightarrow$ `x = x + v*dt`). They had no instruction preceding this, so they struggled with it. Some instruction to precede this would likely help and, in the wrap-up, extending it to second order differential equations would be helpful also. Students struggled with the `for` loop homework question; 25% chose not to attempt it.

⋆ TUTORIAL: Computational Loops ⋆

In this tutorial, you will learn about looping calculations and how to build a simple code that can perform numerical integration.

Part 1 – Understanding Looped Calculations

Consider the following piece of Mathematica code and its output.

Code:

```
In[1]:= For[i = 1; x = 0,
          i < 10,
          i++,
          x = i^2; Print[x]
           ]
```

Output:

```
Out[1]:= 1 4 9 16 25 36 49 64 81
```

(a) Explain what the code is doing. In particular, point out different statements in the code and their purpose in this algorithm. If any lines are unclear, try the Mathematica help.

(b) Now, write a piece of code using Mathematica that divides the number 1500 by the first 100 integers. Run that code.

Contact: Steven.Pollock@Colorado.EDU

Part 2 – Numerically solving a simple differential equation

(a) Consider an object that moves with a constant velocity of 5 m/s. Write down the differential equation which describes this motion.

We can use a `for` loop to numerically integrate this differential equation. Consider this piece of code:

Code:

```
In[2]:= For[v = 5; x = 0; dt = 0.01; t = 0,
         t < 20,
         t = t + dt,
         x = x + v*dt; Print[x]
          ]
```

(b) Explain what this code is doing. In particular, point out the different statements in the code and their purpose in this algorithm.
Make sure to think carefully (connect it to part a) about what this line does: `x = x + v*dt`.

(c) What would the output of this code look like? You can just write down the first few results. How many times would this loop be executed?

Contact: Steven.Pollock@Colorado.EDU

Part 3 – Storing computations and plotting

Printing the output from the loop is not particularly useful. What you really would like to do is visualize it (i.e., plot the results). We can start to do that by storing the results in a Mathematica table (essentially a list of numbers).
This piece of code will perform the numerical integration (constant 5 m/s velocity) we defined in Part 2 and plot the results:

Code:

```
In[3]:= xx = tt = vv = Table[0, {i, 1, 201}];
        dt = .1;
        ifinal = 200;
In[4]:= For[i = 1,
         i < ifinal,
         i++,
         vv[[i]] = 5;
         xx[[i + 1]] = xx[[i]] + vv[[i]]*dt;
         tt[[i + 1]] = tt[[i]] + dt
          ]
In[5]:= ListLinePlot[Table[{tt[[i]], xx[[i]]}, {i, 1, 200}]]
```

(a) Try it out! Explain what this code is doing. In particular, point out the different statements in the code and their purpose in this algorithm. Which differential equation(s) is this code modeling?

(b) How would the code change if instead we had an object traveling with constant acceleration?

(c) Modify the above code for an object starting with $x = 0$ m and $v = 0$ m/s, but accelerating at a constant acceleration of 1 m/s^2. Run it. Do the plots of x and v look right?

Contact: Steven.Pollock@Colorado.EDU