The Sieve of Eratosthenes

In this article, we will learn how to simulate the Sieve of Eratosthenes to find prime numbers.

This program is basically an animation program that shows how the sieve of Eratosthenes works. There is no user interactivity required once the program is started. The animation initially shows a table of numbers 1 to N and then using the technique of Eratosthenes starts dropping numbers until only prime numbers are left in the table.

Explore the game:

If you want to play with my final program to get a feel for this game, run the program specified at the end of the article. Try not to peek at the code yet, since we want to design them ourselves below.

1. Run the program in a Python environment.
2. Follow the instructions in the console (shell) window.
3. Click anywhere on the GUI to run the animation.

Download all the files mentioned in this document by clicking here.

Concepts used in this program:

- Algorithms
    - Using algorithms
    - Designing new algorithms
- Arithmetic
    - Expressions
    - Basic operators (+, -, *, /)
    - Advanced operators: mod, floor, ceiling, %, //, etc.
- Conditional statements:
    - Conditions: YES/NO questions
    - Relational operators (==, <, >, !=, <=, >=)
    - Conditionals (IF)
    - Conditionals (If-Else)
    - Boolean operators (and, or, not)
- Data structures – list
    - List operations
    - Using list as 2-D array
    - Lists of lists (Python)
    - List comprehension (Python)
- Data types – basic
    - Integers
    - Floats
- Data types – strings
    - String operations

    - String traversal
- Data type conversion
- Events
- GUI (graphical user interface)
    - Basic widgets: labels, buttons
    - Synchronizing backend logic with GUI
- Looping (iteration)
    - Looping - simple (repeat, forever)
    - Looping - simple (for)
    - Looping - nested
    - Looping - conditional (while)
- Procedures
    - Built-in
    - User defined (custom)
    - Simple
    - With inputs
    - With return value
- Program output
    - Text
- Sequence
- User input
    - Text
    - Click buttons
    - Input validation
- Variables
    - Simple
    - Local/global scope

High-level algorithm:

1. Make a list L of numbers 2 to N.
2. Take the first non-zero number in L as p. If (square of p) > N, stop.
3. Starting from p, replace all multiples of p (except p itself) in L by 0.
4. Go to step 2.

Version 1:

We will first implement this program in console version, i.e. without the graphical animation.

Design:

The above algorithm can be seen as 3 components: first component just doing step 1, the second one doing only step 3, and the final one putting all steps together.

First component (step 1):

First component is straightforward. It takes N as input and builds a list from 1 to N, and then replaces 1 with 0.

Algorithm BuildArray:

```
Input: integer N
Output: list L
I = 1
L = empty
Repeat N
      Add I to L
      I = I + 1
End-repeat
```

Second component (step 3):

Algorithm for ReplaceMultiples:

```
Input: list L (0, 2 to N), integer Q (location of P)
Output: L with all multiples of P (2xP, 3xP, …) replaced by 0
Steps:
P = item at location Q in L
# Multiples of P will be at Q+P, Q+2P, Q+3P, etc.
Q = Q + P
Repeat until (Q > size of L)
      item at location Q in L = 0
      Q = Q + P
End-repeat
```

Third component:

The following steps indicate the main algorithm for Eratosthenes and they use the 3 steps above (2, 3, and 4).

Algorithm for Eratosthenes:

```
Given:
L: list of integers in ascending order

Steps:
Last = square root of array size (according to Eratosthenes we need
to check multiples from 2 up to this number)
Q = index of first item in L
Increment Q until a non-zero item is found
Repeat until (Q > Last)
      P = item at location Q in L
      Replace all multiples of P in L by 0
      Q = Q + 1
      Increment Q until a non-zero item is found
End-repeat
```

Solution: Eratosthenes.py

Version 2:

Let's animate the algorithm using the "Number table" idea (see the "Number-table" program in "Practice CS Concepts with Scratch" to understand how to design it).

Follow these steps to animate the algorithm:

- Take the Number-table program. Modify it work for the range N=5 to 324 (in anything bigger the numbers would look too tiny).
- Import the code of version 1 above. You will need to call the "Eratosthenes" script after displaying the initial array of numbers. As the algorithm goes on replacing numbers with 0s, you will need to update the display. See below for the "refreshGrid" algorithm. Instead of writing 0, we will just use a blank square.

Algorithm refreshGrid:

```
Steps:
For each cell in the GUI grid, fetch the corresponding value from the
"primes" list and change the cell's text property to reflect this
value.
```

Solution: Eratosthenes-GUI.py

Author: Abhay B. Joshi (abjoshi@yahoo.com)
Last updated: 12 February 2020