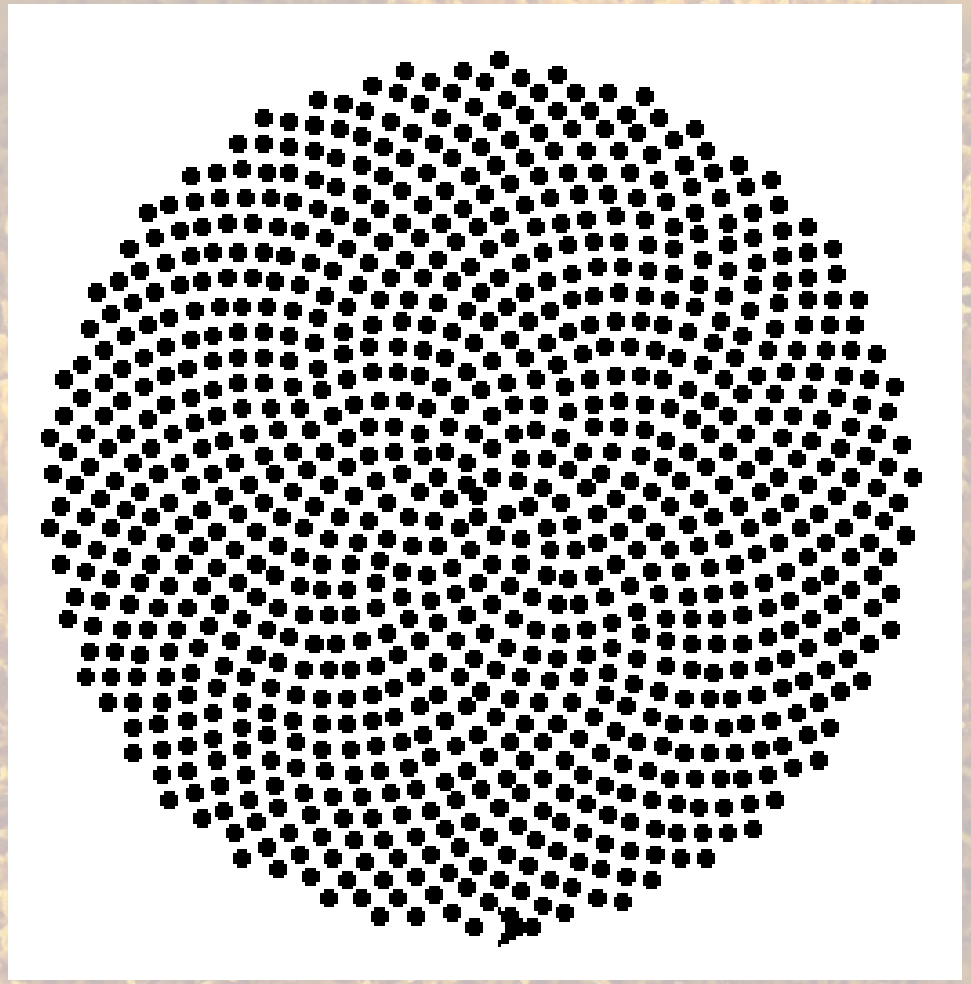
Programming Process

by Deborah R. Fowler



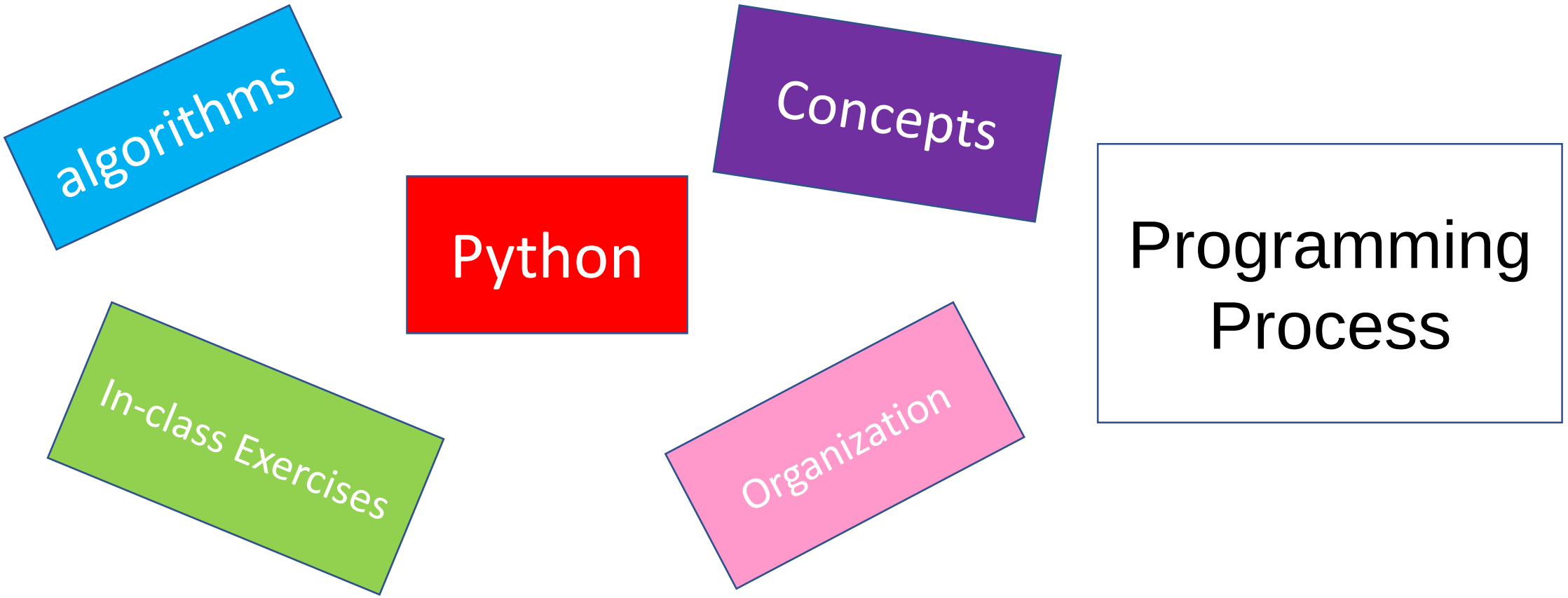
REVIEW

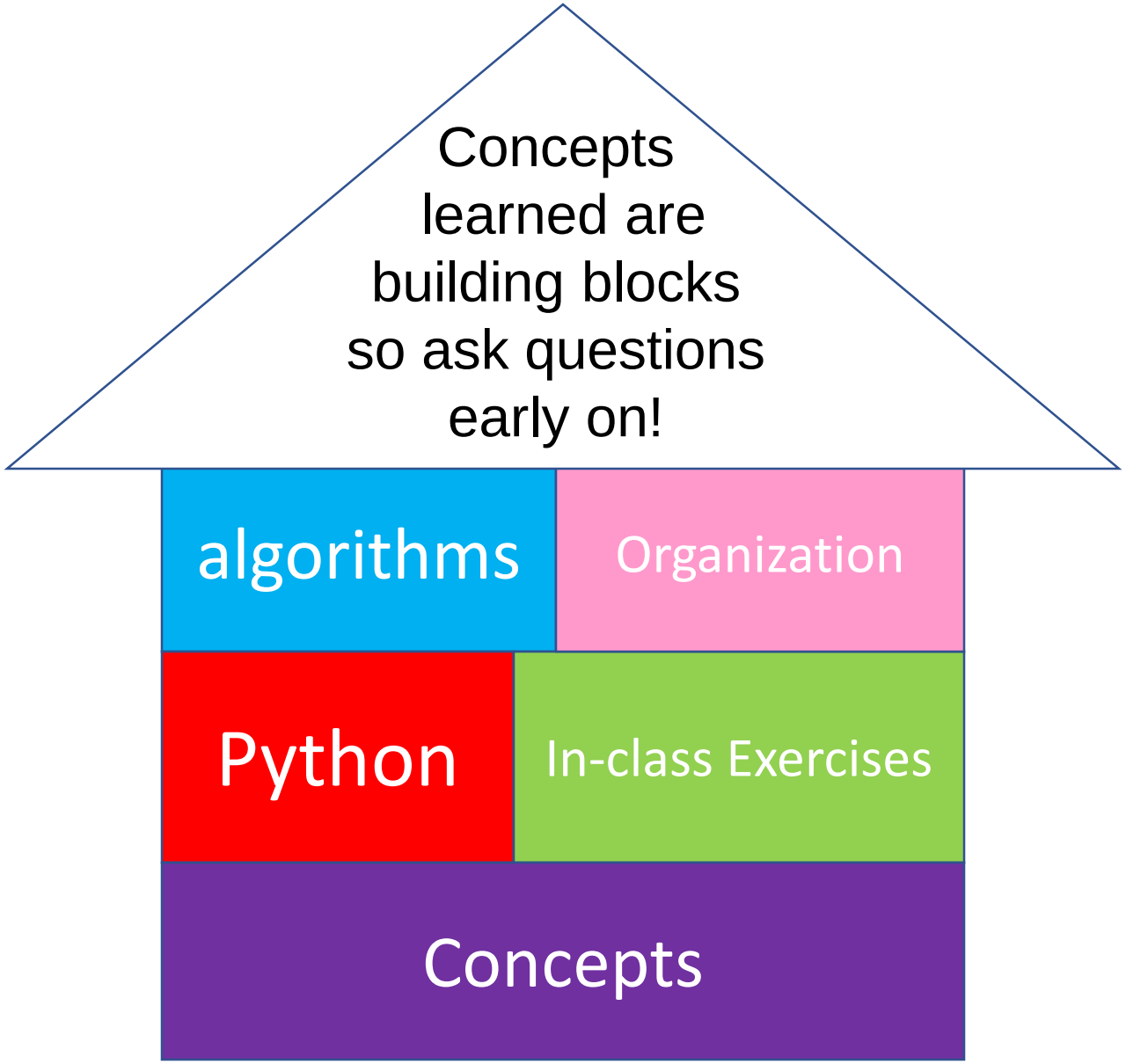
- variables
- truth statements
- looping
- functions
- I/O
- lists
- classes/objects
- OOP

Today

- Programming Process
  - Algorithms
- Code Habits and details
  - Variables
  - Syntactic sugar

Programming requires **clear** and careful **creative thinking**




Concepts
learned are
building blocks
so ask questions
early on!
algorithms
Organization
Python
In-class Exercises
Concepts

What is Programming?

Telling the computer what to do

- Problem solving
- Algorithms - plan

- Analysis
- Design – identify key concepts involved in a solution
- Program – express that solution in a language

… how do you learn it? Studying good examples, **practice**, and **experimentation**

Steps

Hide Images

1 Heat oven to 375°F.

2 Mix sugars, butter, vanilla and egg in large bowl. Stir in flour, baking soda and salt (dough will be stiff). Stir in nuts and chocolate chips.

3 Drop dough by rounded tablespoonfuls about 2 inches apart onto ungreased cookie sheet.

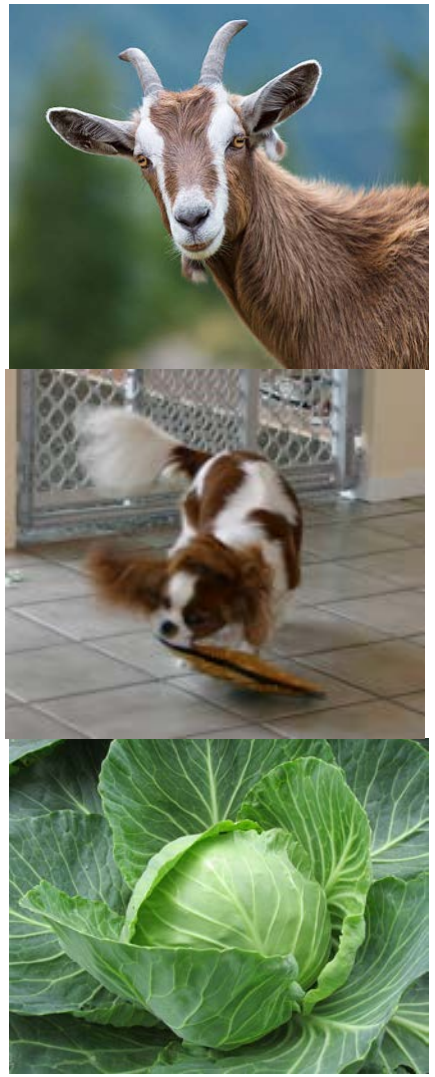
4 Bake 8 to 10 minutes or until light brown (centers will be soft). Cool slightly; remove from cookie sheet. Cool on wire rack.

As a programmer you are problem solving – defining the instructions (algorithm is the first step)

In-class Exercise



In-class Exercise



(can be found in 18th century writings, although they use a wolf)

Anaylsis – what can you leave behind? The dog does not eat cabbage

Algorithm

Start on side A
Take the goat over to side B
Return alone to side A
Take the cabbage over to side B
Return with goat to side A
Take the dog to side B
Return alone to side A
Take the goat to side B

Steps lead to completion

Unambiguous

Appropriate level of detail

Well ordered instructions

Covers all possible outcomes

Algorithm is a general term for a clear concise finite set of instructions to solve a problem

Vogel 1979

$r = c * \sqrt{n}$ $\theta = n * 137.508$

- r is the radius from center
- n is the number of the floret from center
- theta is the angle

Describes the head of a sunflower

phyllotaxis

by drspiral

SCRATCH

4 scripts
2 sprites

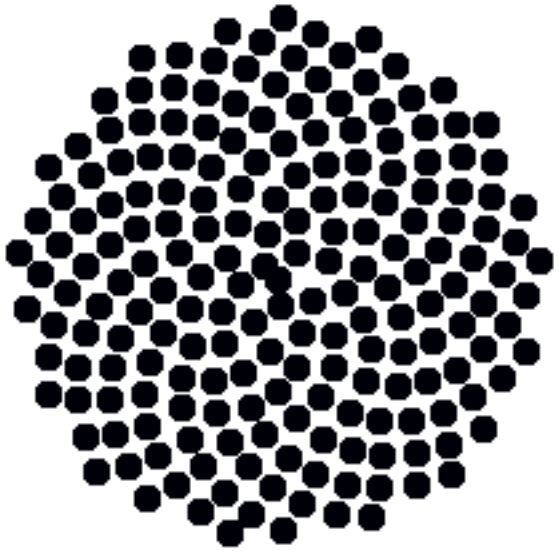
See inside

v450.1

total 200

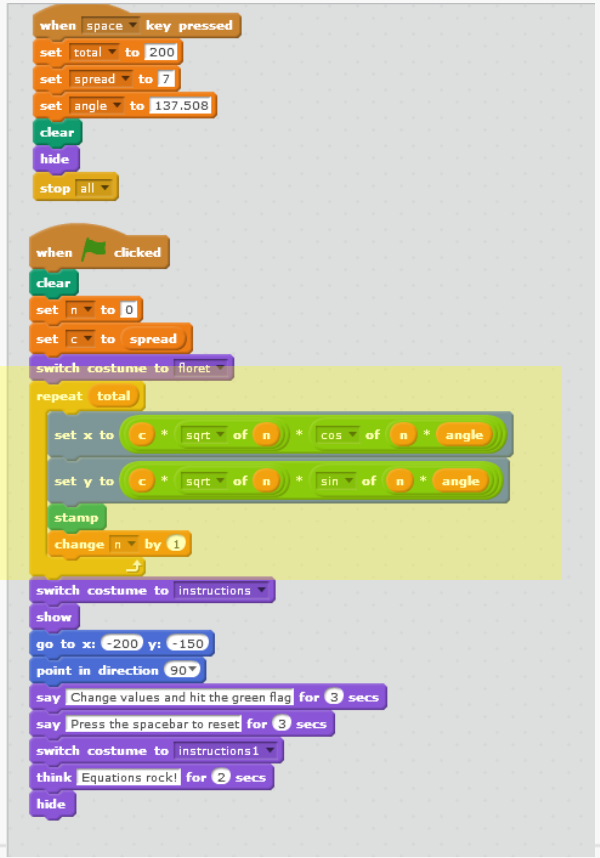
angle 137.508

spread 7



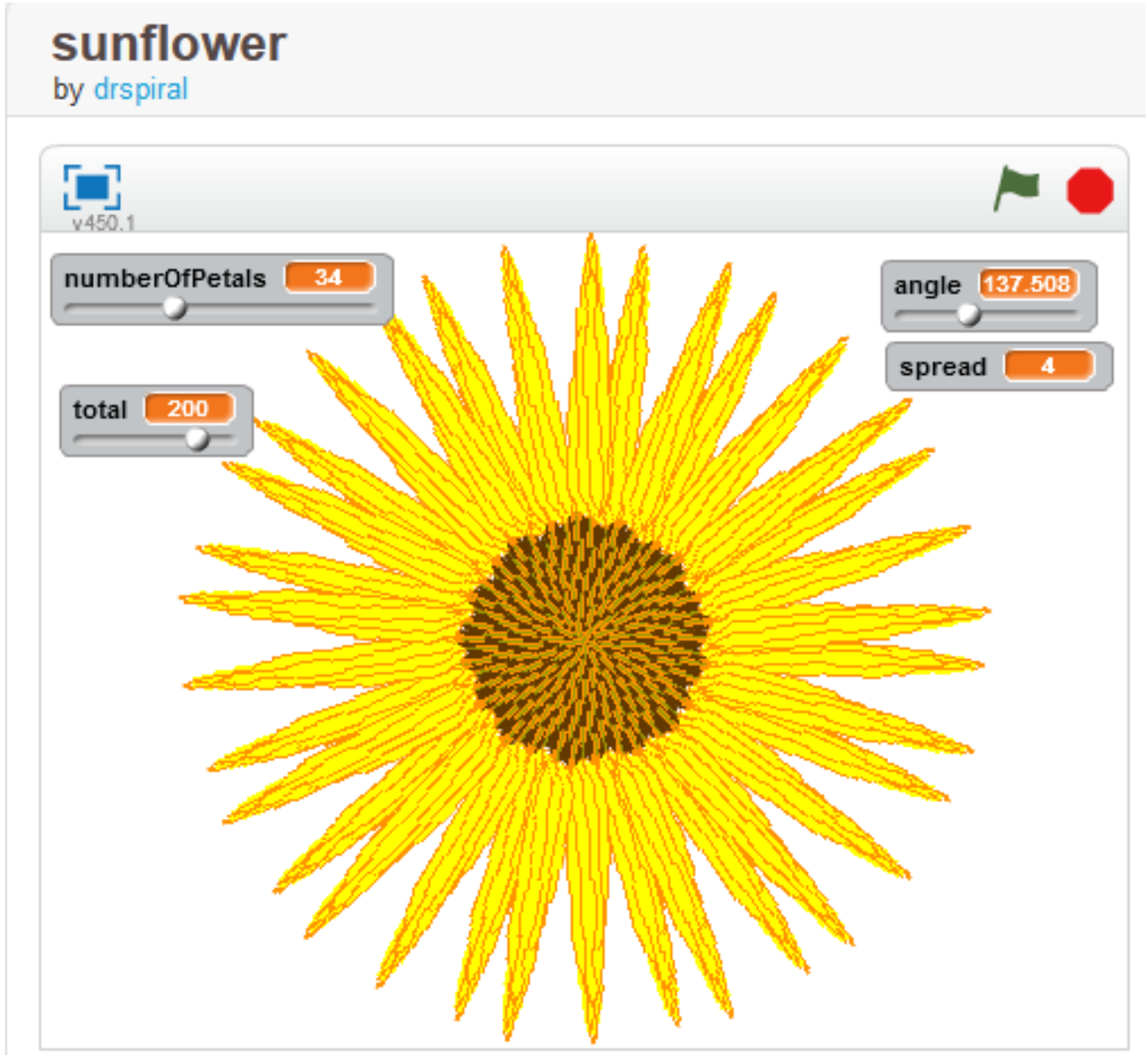
Notes and Credits

Demonstration of the spiral phyllotactic pattern as described by Vogel 1979 and shown in The Algorithmic Beauty of Plants.



Shared: 11 Sep 2011

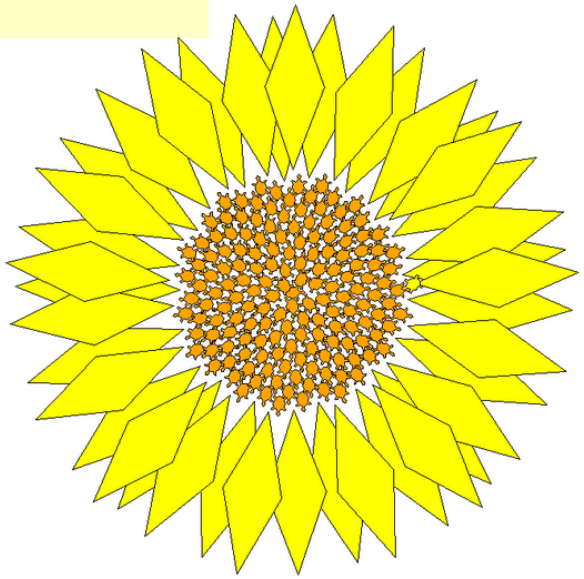
Modified: 11 Sep 2011



Here the pattern has different geometry to appear more like a sunflower

```
# Spiral Phyllotaxis Demo
#
# Example for VSFX 705
# Turtle Sunflowers - Introduce Phyllotactic Pattern
#
# Author: Deborah R. Fowler
#
# March 21, 2013
# Based on original code in C 1989 using Silicon Graphics Workstations and gl

import math
import turtle

def drawPhyllotacticPattern( t, petalstart, angle = 137.508, size = 2, cspread = 4 ):
        """print a pattern of circles using spiral phyllotactic data"""
        # initialize position
        turtle.pen(outline=1,pencolor="black",fillcolor="orange")
        # turtle.color("orange")
        phi = angle * ( math.pi / 180.0 )
        xcenter = 0.0
        ycenter = 0.0

        # for loops iterate in this case from the first value until < 4, so
        for n in range (0,t):
                r = cspread * math.sqrt(n)
                theta = n * phi

                x = r * math.cos(theta) + xcenter
                y = r * math.sin(theta) + ycenter

                # move the turtle to that position and dra
                turtle.up()
                turtle.setpos(x,y)
                turtle.down()
                # orient the turtle correctly
                turtle.setheading(n * angle)
                if n > petalstart-1:
                        #turtle.color("yellow")
                        drawPetal(x,y)
                else: turtle.stamp()
```

```
def drawPetal( x, y ):
        turtle.up()
        turtle.setpos(x,y)
        turtle.down()
        turtle.begin_fill()
        #turtle.fill(True)
        turtle.pen(outline=1,pencolor="black",fillcolor="yellow")
        turtle.right(20)
        turtle.forward(100)
        turtle.left(40)
        turtle.forward(100)
        turtle.left(140)
        turtle.forward(100)
        turtle.left(40)
        turtle.forward(100)
        turtle.up()
        turtle.end_fill() # this is needed to complete the last petal


turtle.shape("turtle")
turtle.speed(0) # make the turtle go as fast as possible
drawPhyllotacticPattern( 200, 160, 137.508, 4, 10 )
turtle.exitonclick() # lets you x out of the window when outside of idle
```



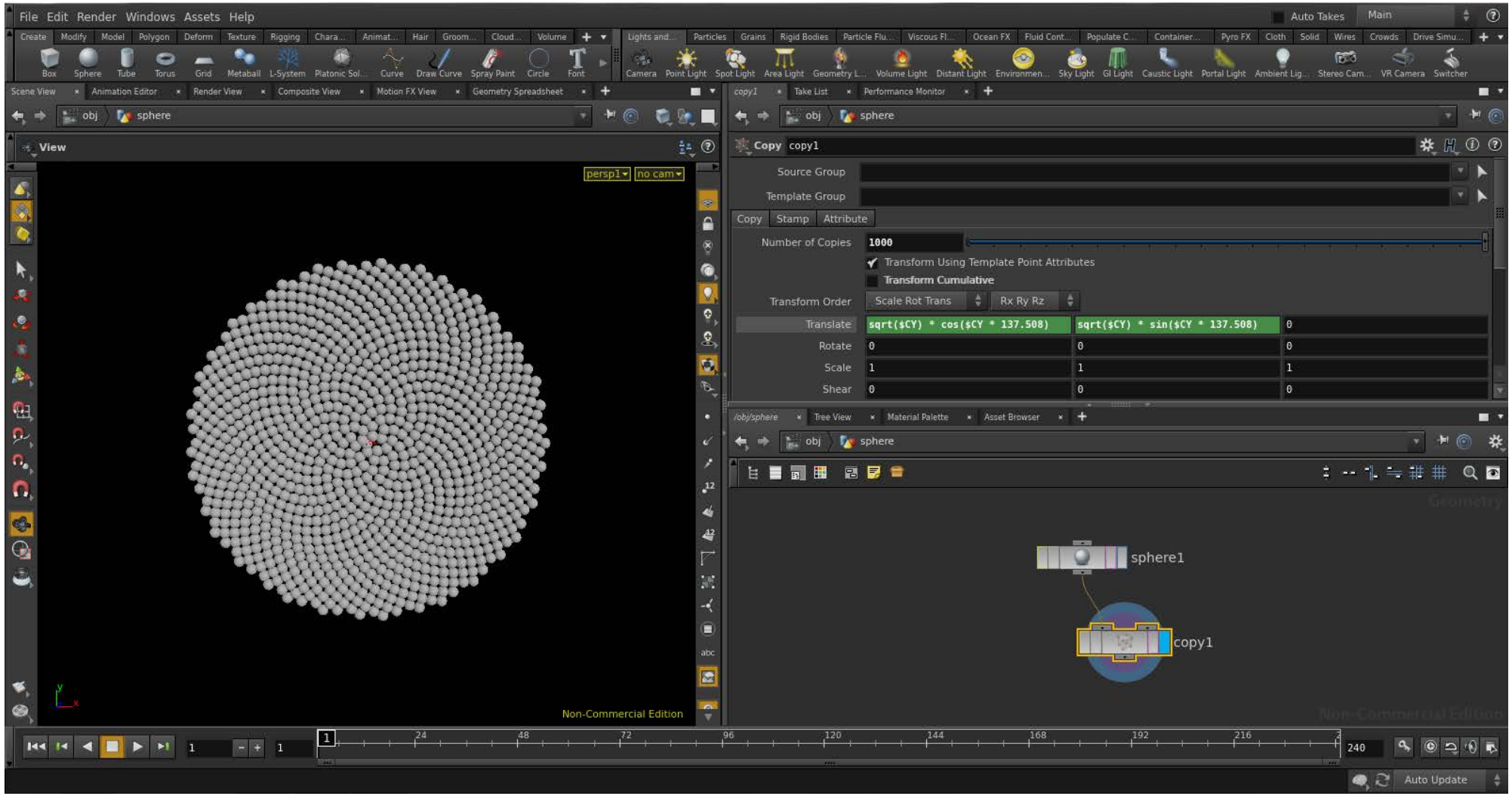
Created in python

```
//  Date: March 22, 2013
//
// This is the phyllotactic pattern as described by Vogel in Biomathematics 1979, and used in The Alorithmic Be
//
//  Inputs: GUI Mel interface for amount, size, spread and angle
//  Output: Pattern of spheres on a disc optimally packed
//

if ( `window -exists myWindow` ) deleteUI myWindow;
   window -title "Spiral Phyllotaxis" -widthHeight 500 200 myWindow;
   columnLayout -adj on;
   intSliderGrp -label "amount" -min 1 -max 5000 -value 100 -field true -changeCommand "phyllotaxis" total;
   floatSliderGrp -label "size" -min 1 -max 60 -value 2 -field true -changeCommand "phyllotaxis" size;
   floatSliderGrp -label "size" -min 1 -max 20 -value 2 -field true -changeCommand "phyllotaxis" spread;
   floatSliderGrp -label "angle" -min 0 -max 360 -value 137.508 -pre 3 -field true -changeCommand "phyllotaxis" angle;
showWindow myWindow;
```

Created in Maya/mel

```
proc phyllotaxis()
{
   // Delete any existing geometry - this is intended as a stand alone demo
   select -all;
   delete;

   // Get the values from the GUI interface
   int $total = `intSliderGrp -q -value total`;
   float $size = `floatSliderGrp -q -value size`;
   float $c = `floatSliderGrp -q -value spread`;
   float $angle = `floatSliderGrp -q -value angle`;

   // Calculate the positions in the spiral phyllotactic pattern
   float $phi;
   float $r;
   float $theta;
   float $x, $y, $xcenter, $ycenter;
   float $PI = 3.14159265359;

   $phi = $angle * ( $PI/180.0 );
   $xcenter = 0.0;
   $ycenter = 0.0;

   for ( $n = 0; $n  <  $total;  $n++ )
   {
         $r = $c * sqrt( $n );
         $theta = $n * $phi;
         $x = $r * cos( $theta ) + $xcenter;
         $y = $r * sin( $theta ) + $ycenter;

         // draw a sphere or whatever object you'd like at this position
         sphere -r $size -p $x $y 0;
   }
}

phyllotaxis();
```

Houdini … with two expressions right in the interface



… and more

- Hscript expressions (houdini script)
- Python expressions (as above but with python)
- Python with HOM (Houdini Object Module) think PyMel
- Vex (wrangle nodes/vex code) think rsl or C++
- L-systems (formal grammar)
- And so on

CODE HABITS

Variable names

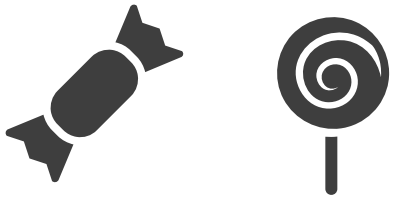
- meaningful
- add to the code readability
- (self-documenting code)

Not Good: hps, av, s

Good: hitsPerSecond, average, score

king_snake

camelFace



Syntactic sugar

```
>>> x = 0
>>> x = x + 1
>>> x += 1
>>> print x
2
>>>
>>>
>>> x++
SyntaxError: invalid syntax
>>> ++x
2
>>> print x
2
>>>
```

shorthand

Order of Operators

- * / % then
- + -
- Left to right in expression
- Use () when in doubt!

Precedence

```
>>> 5 + 2 * 3
11
>>> (5 + 2) * 3
21
>>>
```

Build modularly!

Test as you go!

Summary

Algorithm – a clear concise plan, not specific to a particular language syntax

Habits

- Human readable code – variables, comments
- Modularity – build in chunks

homework:

You may start on the quilting exercise (E1)