Kojo / Scala

Andrea Sterbini – sterbini@di.uniroma1.it

Kojo learning environment:
Scala language + turtles + math / (~~geometry~~)

Kojo IDE:

- HTML “stories” to guide you in simple programming tasks

- Code editor

- 2D canvas (turtle-based)

~~- 2D math and geometry cartesian canvas (Geogebra):~~ - ~~[OLD]~~
~~- lines, points, segments, angles, areas, formulas, plots, …~~

- Arithmetic exercises
- Music player (with Midi instruments)

- Game programming (Processing-like Stage) [NEW]

Other: Arduino programming (with an arduino driver/interface)

AI with Tensorflow: (https://github.com/litan/kojo-ai-2) [NEW]

Scala is a functional programming language

- compiles to Java VM
  (can use Java libs)
- has a simpler readable syntax
- functions + OOP
  - anonymous functions,
    map / filter / …
  - simpler classes
    (no getters/setters)

- implicit return (last instruction)
- operator overloading
- recursive structure matching
  (“similar” to unification
  in Prolog)
- multiple assignment
  (like Python)

- prefer immutable structures vs. mutable:
  - immutable (val) vs. mutable (var) vars
  - function args are always immutable

Scala resources

Programming in Scala

[by Martin Odersky, Lex Spoon, and Bill Venners]

https://www.scala-lang.org

the Scala language community

Good IDE for Scala:

- Eclipse
- Netbeans
- Idea

Programming style

Single-threaded?	**YES**	**Concurrency?**	**YES**
Procedural?	**YES**	**Lazy evaluation?**	**YES**
Functional?	**YES!**		

Statically typed:

- complex abstract types - metaprogramming

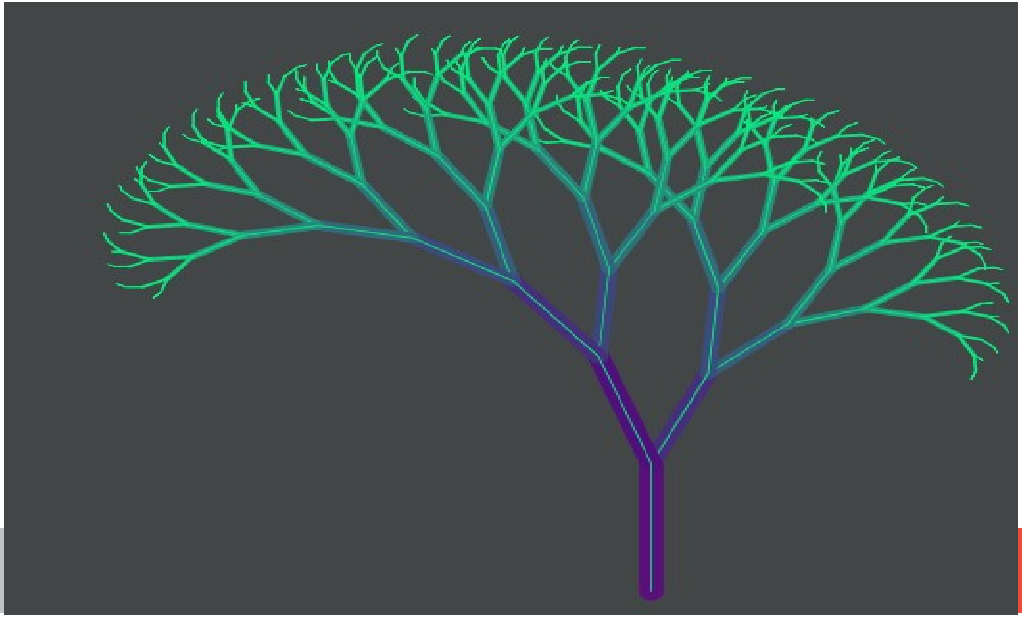
Data types:

- Objects + Classes (+ Singletons + Structs)
- sequential / parallel collections
- list-based operations (map / filter / ...)
- anonymous functions (code blocks)

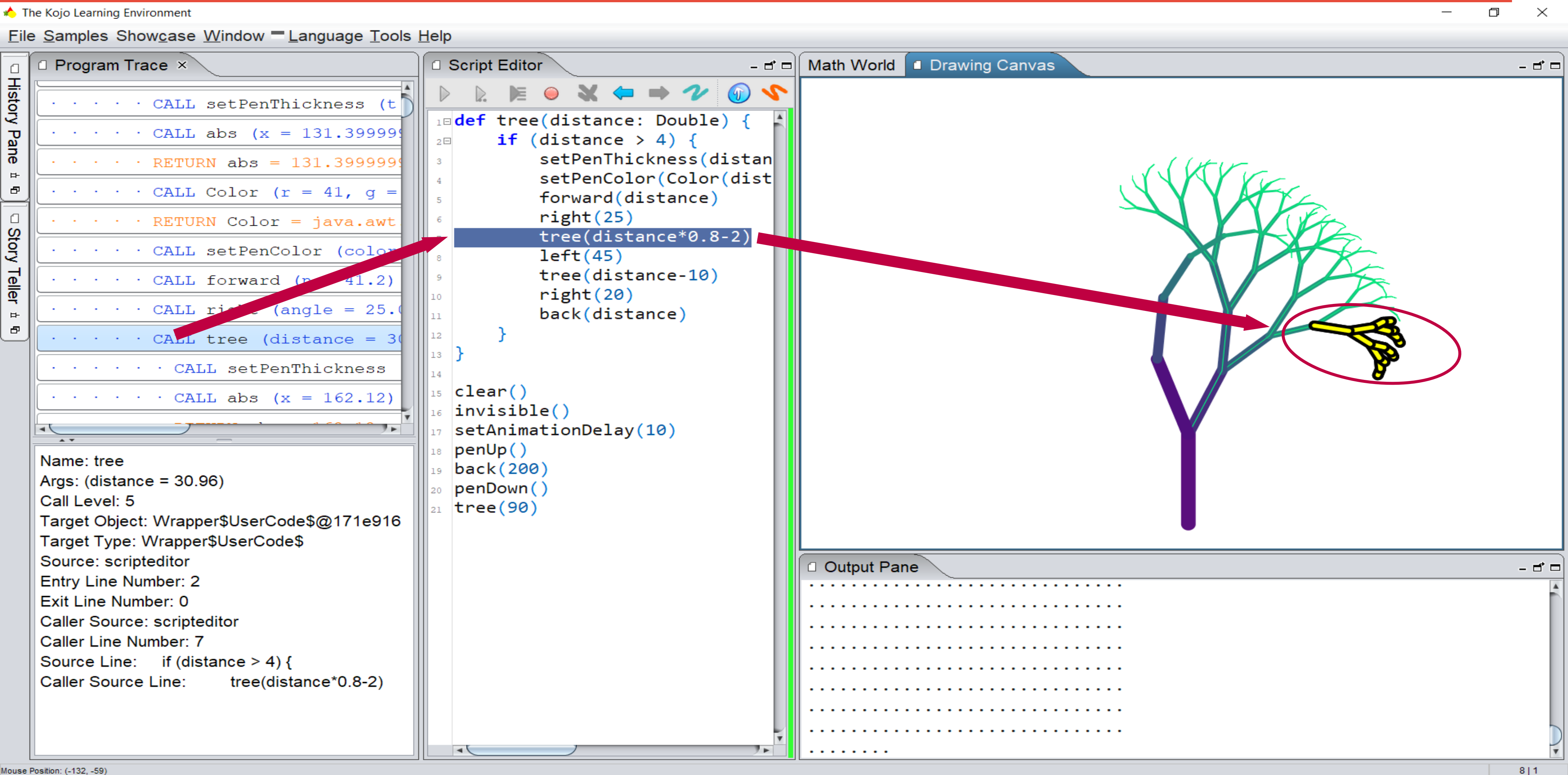
Demo 1: recursive tree by turtle graphics (as usual)

```
clear()
invisible()
setAnimationDelay(10)
hop(-200)
tree(90)
```

```
def tree(distance: Double) {
    if (distance > 4) {
        setPenThickness(distance/7)
        setPenColor(
              Color(distance.toInt, math.abs(255-distance*3).toInt, 125))
        forward(distance)
        right(25)
        tree(distance*0.8-2)
        left(45)
        tree(distance-10)
        right(20)      // 45-25
        hop(-distance)
    }
}
```



Good debugger with recursion trace / visualization



```
. . . . . CALL setPenThickness (t
. . . . . CALL abs (x = 131.39999
. . . . . RETURN abs = 131.3999999
. . . . . CALL Color (r = 41, g =
. . . . . RETURN Color = java.awt
. . . . . CALL setPenColor (color
. . . . . CALL forward (n = 41.2)
. . . . . CALL right (angle = 25.0
. . . . . CALL tree (distance = 30
. . . . . . CALL setPenThickness
. . . . . . CALL abs (x = 162.12)
```

```
Name: tree
Args: (distance = 30.96)
Call Level: 5
Target Object: Wrapper$UserCode$@171e916
Target Type: Wrapper$UserCode$
Source: scripteditor
Entry Line Number: 2
Exit Line Number: 0
Caller Source: scripteditor
Caller Line Number: 7
Source Line:     if (distance > 4) {
Caller Source Line:        tree(distance*0.8-2)
```

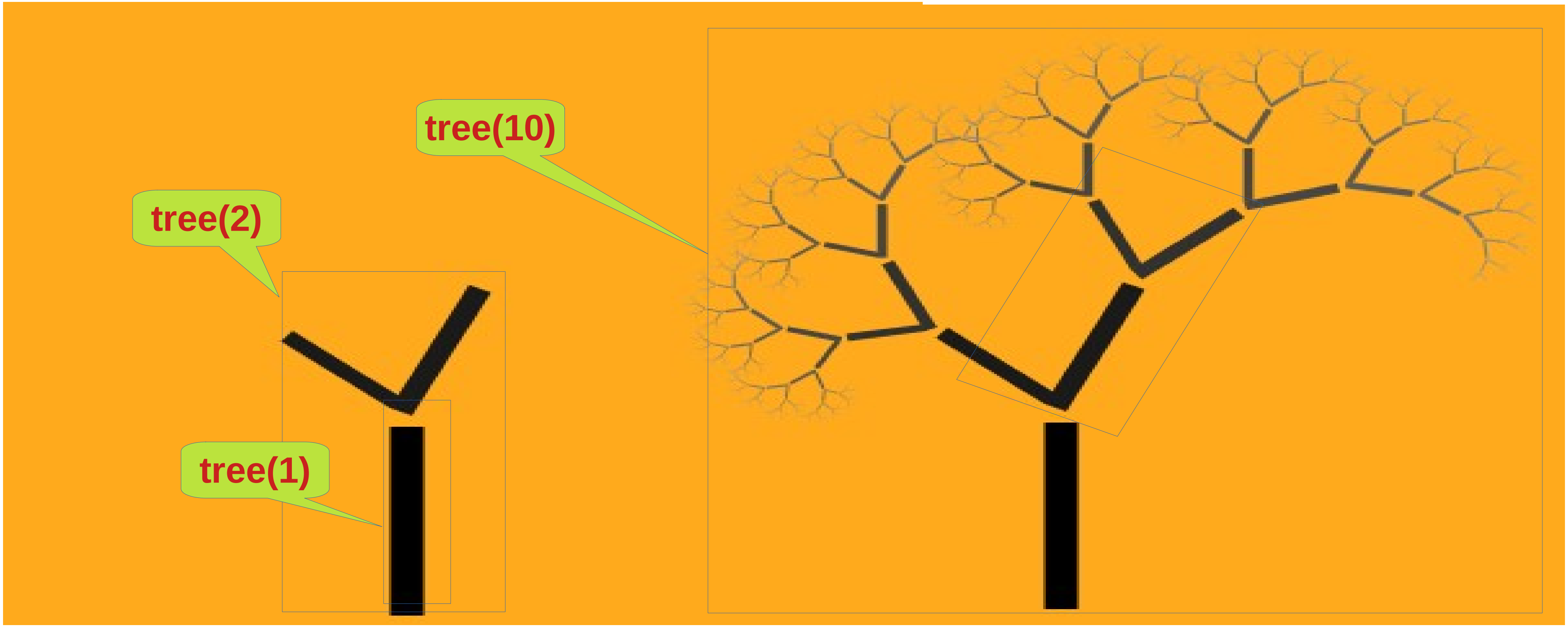
```
def tree(distance: Double) {
    if (distance > 4) {
        setPenThickness(distan
        setPenColor(Color(dist
        forward(distance)
        right(25)
        tree(distance*0.8-2)
        left(45)
        tree(distance-10)
        right(20)
        back(distance)
    }
}

clear()
invisible()
setAnimationDelay(10)
penUp()
back(200)
penDown()
tree(90)
```

Mouse Position: (-132, -59)

8 | 1

Tree2 = recursive transformations of a rectangle figure rotate(-25°) + scale(72%) + brightness(+10%)



Demo 2: recursive tree = recursive pictures + graphic transformations

```
// we start with a square
def square = Picture {
    repeat (4) {
        forward(100)
        right()        // default 90°
    }
}

// a stem is a distorted black square
def stem = scale(0.13, 1) *
    penColor(noColor) *
    fillColor(black) -> square
```

```
def drawing(n: Int): Picture = {
    if (n < 2)
        stem
    else
        GPics(stem,
              trans(0, size + 10) * brit(0.1) -> Gpics(
                         rot(-25) * scale(0.72)  -> drawing(n-1),
                         rot(  50) * scale(0.55)  -> drawing(n-1)
                   )
        )
}

clear()
setBackground(Color(255, 170, 29))
invisible()
val pic = trans(0, -300) -> drawing(10)
draw(pic)
```

Demo 3 [S. Penge]
choose the correct article for an italian word

Type: definite/indefinite (determinativo/indeterminativo)
Gender: male/female
Number: singular/plural

1) deduce the word gender from final char

2) select proper gender/number from final char

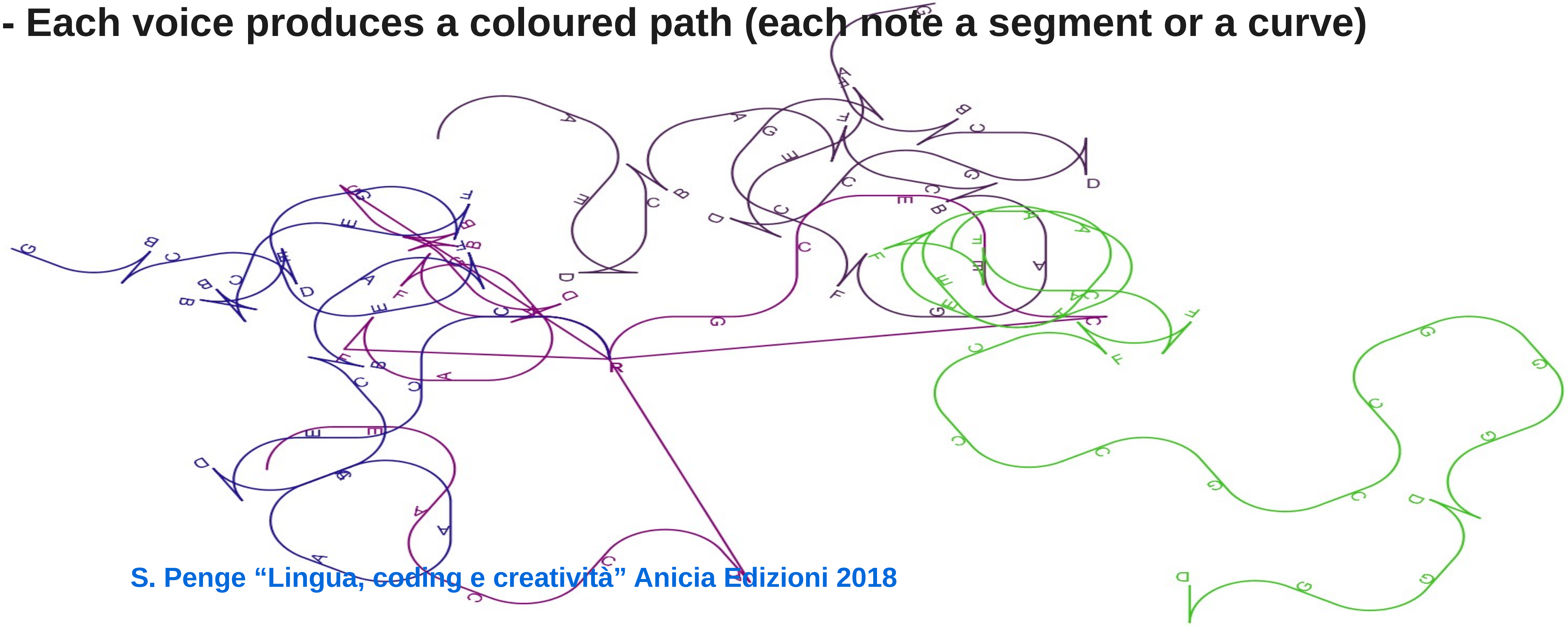
3) handle normality & exceptions (here for indefinite male singular only)
   - starts with vowel → ”un” (normal case)
   - starts with consonant → “un” (normal case)
   - starts with 2 vowels (‘ia’, ‘ie’, ‘io’, ‘iu’) → “uno” (special case)
   - starts with 1 or 2 special consonants → “uno” (special case)

   (“x”, “y”, “z”, “gn”, “pt”, “ps”, “pn”, “sc”, “sf”, “sq”, “st”)

S. Penge “Lingua, coding e creatività” Anicia Edizioni 2018

Demo 4 [S. Penge]
Music transposition and art

- Many voices/instruments - With arrangement
- Transposed to a different key/mode (major/minor)
- Each voice produces a coloured path (each note a segment or a curve)

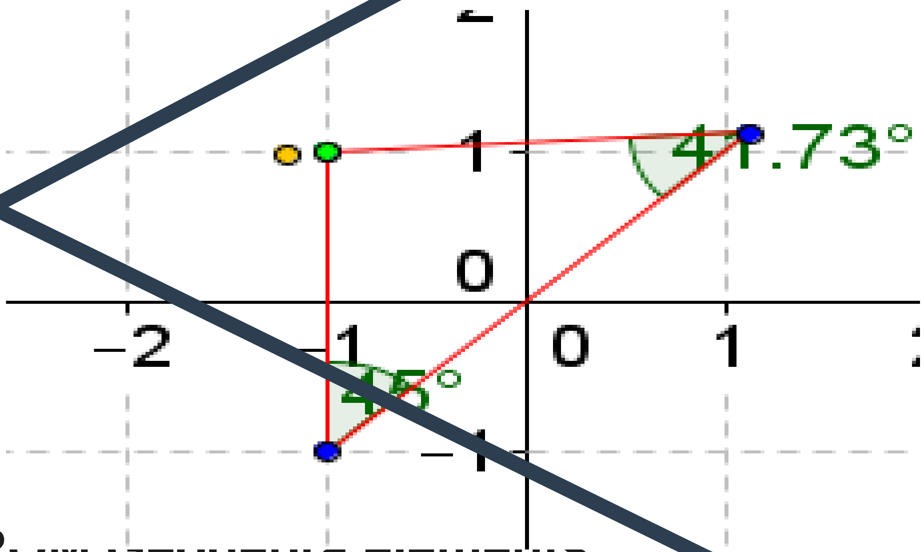


S. Penge “Lingua, coding e creatività” Anicia Edizioni 2018

~~Demo 5: MathWorld pane = Geogebra~~ [V 2.7]

Just use turtle graphics with the Mw class to create lines and angles

```
val t = Mw.turtle(-1, 0)
t.showExternalAngles()
t.forward(-2)
t.right(45)
t.forward(3)
t.moveTo(-1,0)
```



Full GeoGebra with spreadsheet

Not (yet?) possible to get all properties FROM Geogebra elements

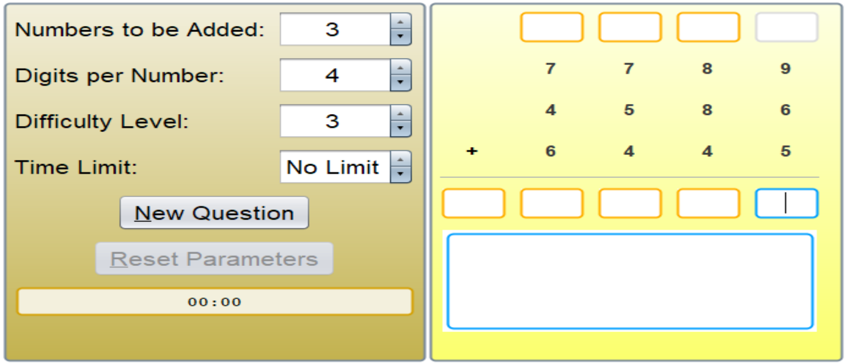
NEW
Game programming with “Staging”   [V 2.9]

```
import Staging._                                // import the Staging library
import Staging.{circle, clear, animate}    // explicitly import names that clash
clear()
gridOn()
val width  = 300
val height = 200
rectangle(0,0,width,height)
val ball = circle(100, 100, 5)
var y = height/2 ; var x = width/2    // ball position
var dy = 10; var dx = 5                    // ball speed

animate {                          // animation is around 30 - 50 frames per second
   ball.setPosition(x,y)
   // update ball speed, detecting out of bounce area
   dx =  if(x < 0 || x > width) -dx else dx
   x += dx
   dy =  if(y < 0 || y > height) -dy else dy
   y += dy
}
```

Arithmetic exercises

Sum, Multiplication, Division, Subtraction



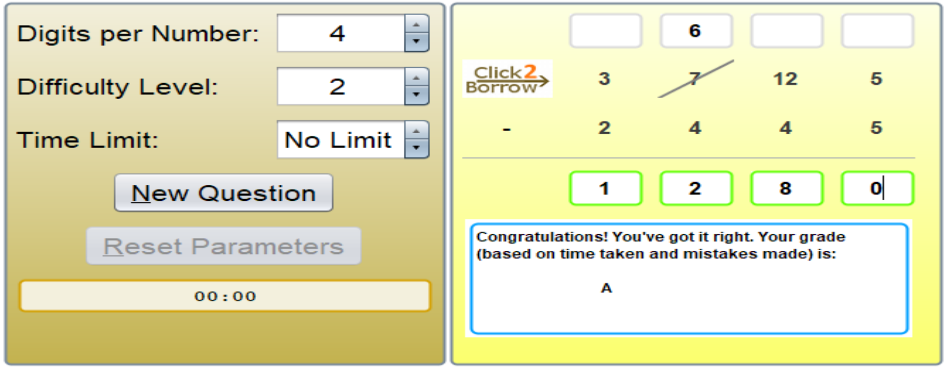
Numbers to be Added: 3

Digits per Number: 4

Difficulty Level: 3

Time Limit: No Limit

```
   7  7  8  9
   4  5  8  6
+  6  4  4  5
```



Digits per Number: 4

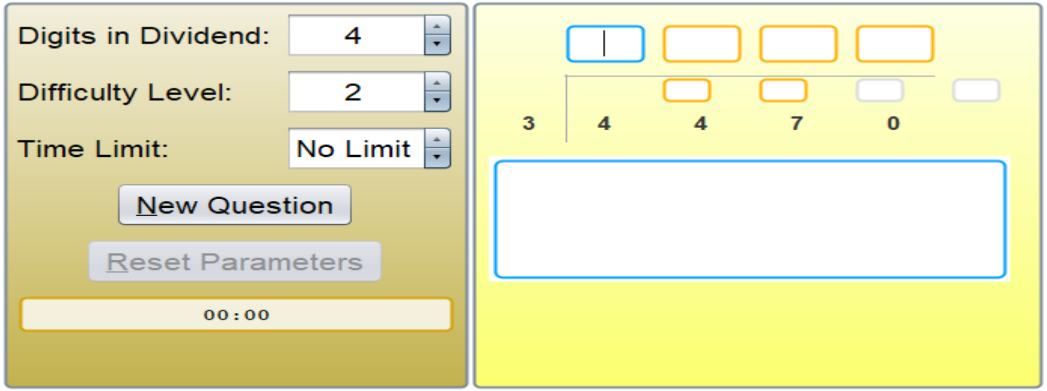
Difficulty Level: 2

Time Limit: No Limit

```
          6
   3  ~~7~~  12  5
-  2  4   4   5
   1  2   8   0
```

Congratulations! You've got it right. Your grade (based on time taken and mistakes made) is:

A

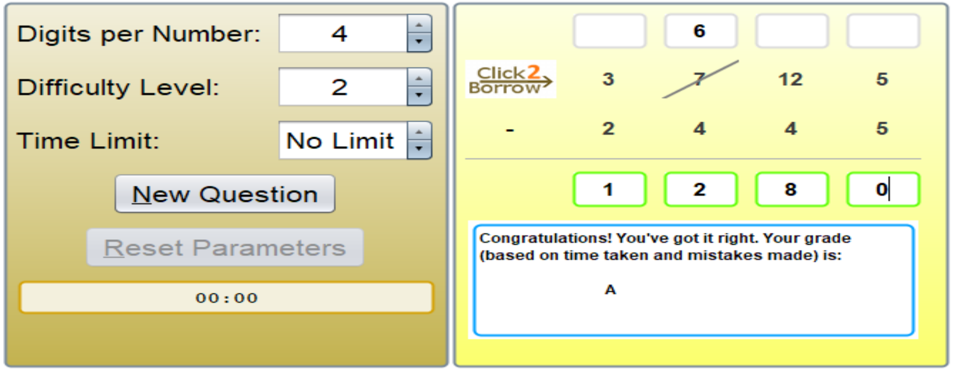


Digits in Dividend: 4

Difficulty Level: 2

Time Limit: No Limit

```
3 | 4  4  7  0
```



Digits per Number: 4

Difficulty Level: 2

Time Limit: No Limit

```
          6
   3  ~~7~~  12  5
-  2  4   4   5
   1  2   8   0
```

Congratulations! You've got it right. Your grade (based on time taken and mistakes made) is:

A

Demo

DEMO