

CMPT 120

Lecture 21 – Graphics and Animation

Python – Implementing **Recursion**
and introducing **Dictionary**

Last Lecture

- Investigated **Recursion**
- Solved the **factorial** problem using **recursion**
- **Visualized** the execution of our **recursive factorial** function

Today's Menu

- Let's try again: Solve another problem using **recursion**
- Close our unit on **Computer Graphics** looking into drawing **trees iteratively** as well as **recursively** using **turtle**
 - Introduce **Dictionary**

Let's try again!

Step 1 - Problem Statement

- Remember the **palindrome function** of Practice Exam 4?
- Solve the **palindrome problem recursively**

Step 2 – Design

Step 2 - Design

- Using an example: **kayak**

Using an example, we observe how we solve a palindrome problem by hand, **thinking recursively** and **looking for the pattern**. Then we **implement the pattern!**



Hum...
Easy for
you to say!

Source: <https://en.wikipedia.org/wiki/File:Man-scratching-head.gif>

Step 3 – Implementation

Step 4 - Testing

Drawing Trees

Rendering of wilderness nature reserve using 3D computer graphics

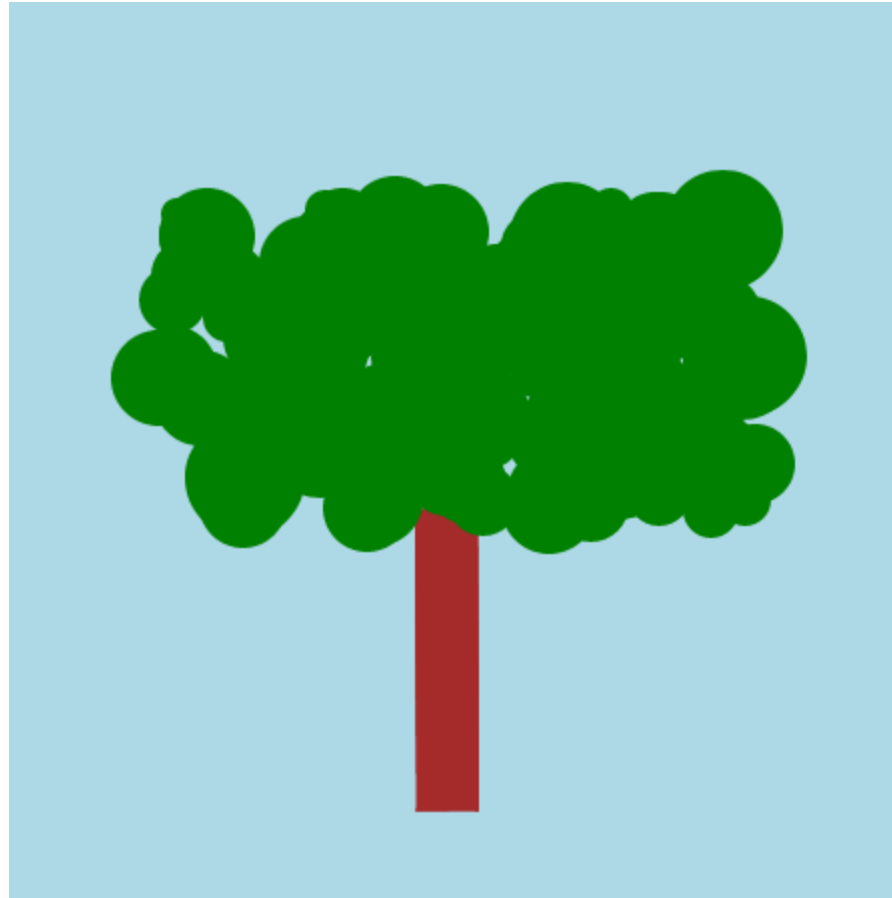


Source: <https://www.kisspng.com/png-rendering-wilderness-nature-reserve-3d-computer-gr-4461372/>

Iteratively -> using `for` loop ...

... and **random** module

Let's have a look
at the code!



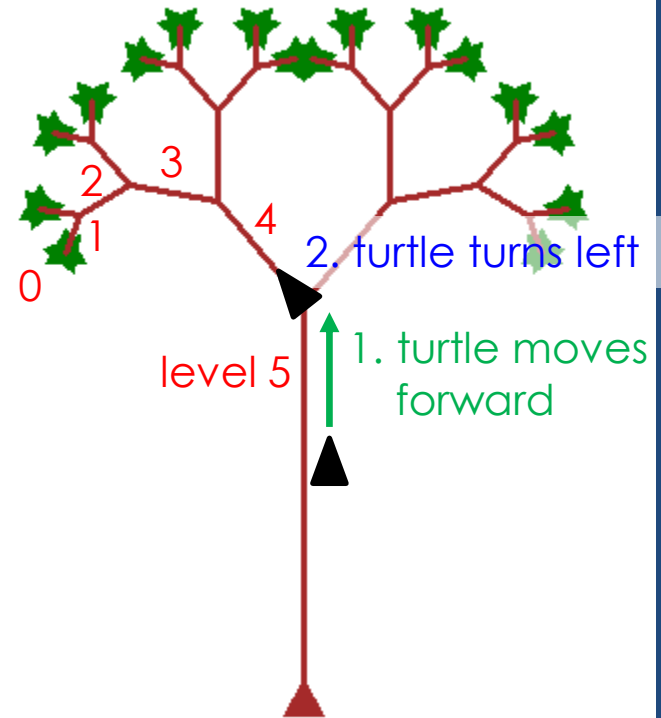
Turtle and Trees!

- **Step 1 - Problem Statement**

- Draw a tree recursively.

- **Step 2 – Design**

- Using this tree as an example, let's examine how it was created.
- This will give us a sense of the algorithm we need to code into Python. Then we will be ready to start our next step...



- **Step 3 - Implementation**

Let's see how we can repeat these two Python statements:
`aTurtle.forward( ... )`
`aTurtle.left( ... )`

Draw a tree recursively

```
# ***Main part of my program

# Creates a graphics window "canvas"
canvas = turtle.Screen()

# Create a turtle named "tt"
tt = turtle.Turtle()

# Set up our turtle "tt"
tt.color("brown")
tt.width(3)
tt.shape("triangle")

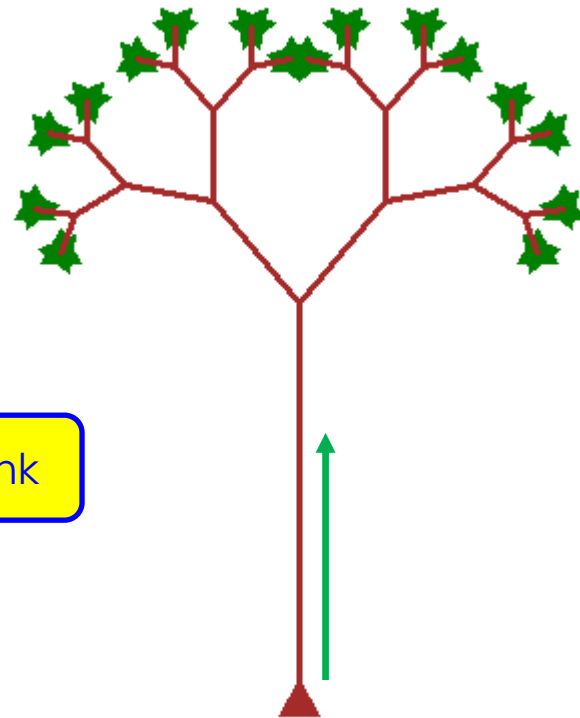
# Move our turtle "tt"
tt.speed(0)
tt.penup()
tt.goto(0, -180)
tt.left(90)
tt.pendown()
theBranchLength = 100
tt.forward(theBranchLength)

# Call our recursive function
theLevel = 5
drawTree(tt, theLevel, theBranchLength)
tt.back(theBranchLength)

# Click on the canvas to exit
canvas.exitonclick()
```

Setting the canvas and the turtle

Drawing the tree trunk



Calling **drawTree (...)** :
drawing the branches and
the leaves recursively!

Draw a tree recursively

```
# Define a recursive function that draws a tree
def drawTree(aTurtle, aLevel, aBranchLength):
    '''Draws a tree recursively where
    "aTurtle" is the turtle drawing the tree,
    "aLevel" is the number of levels of branches and
    "aBranchLength" the length of branch to draw.
    '''

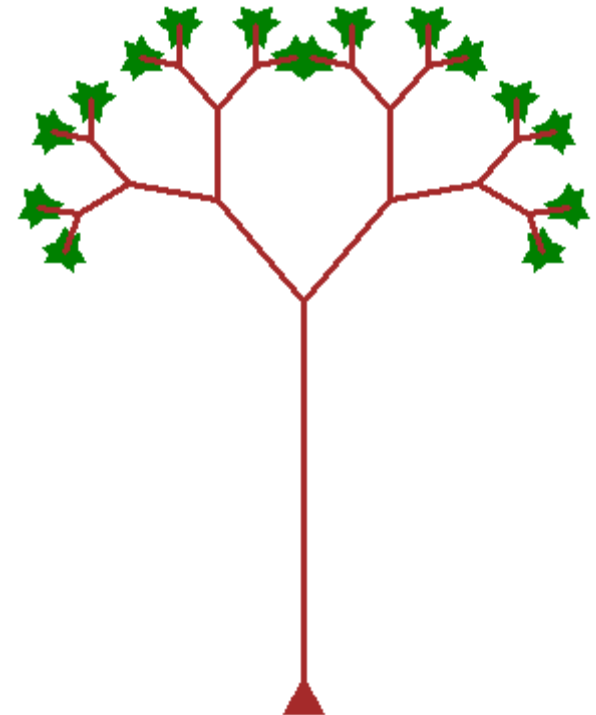
    # Base Case:
    # If we are at the leaf level (level == 0), draw a green leaf!
    if aLevel == 0:
        aTurtle.color("green")
        aTurtle.stamp()
        aTurtle.color("brown")
    else:
        # Recursive Case:
        # Draw a branch
        aTurtle.forward(aBranchLength)

        # Turn left and draw a smaller tree
        aTurtle.left(40)
        drawTree(aTurtle, aLevel - 1, aBranchLength/1.5)

        # Turn right and draw a smaller tree
        aTurtle.right(80) # 40 + 40
        drawTree(aTurtle, aLevel - 1, aBranchLength/1.5)

        # Go back
        aTurtle.left(40)
        aTurtle.back(aBranchLength)

    return
```



Question?

```
# ***Main part of my program

# Creates a graphics window "canvas"
canvas = turtle.Screen()

# Create a turtle named "tt"
tt = turtle.Turtle()

# Set up our turtle "tt"
tt.color("brown")
tt.width(3)
tt.shape("triangle")

# Move our turtle "tt"
tt.speed(0)
tt.penup()
tt.goto(0, -180)
tt.left(90)
tt.pendown()
theBranchLength = 100
tt.forward(theBranchLength)

# Call our recursive function
theLevel = 
drawTree(tt, theLevel, theBranchLength)
tt.back(theBranchLength)

# Click on the canvas to exit
canvas.exitonclick()
```

To what value must we set the variable **theLevel** in order to create this tree?



Question?

```
# ***Main part of my program

# Creates a graphics window "canvas"
canvas = turtle.Screen()

# Create a turtle named "tt"
tt = turtle.Turtle()

# Set up our turtle "tt"
tt.color("brown")
tt.width(3)
tt.shape("triangle")

# Move our turtle "tt"
tt.speed(0)
tt.penup()
tt.goto(0, -180)
tt.left(90)
tt.pendown()
theBranchLength = 100
tt.forward(theBranchLength)

# Call our recursive function
theLevel = 
drawTree(tt, theLevel, theBranchLength)
tt.back(theBranchLength)

# Click on the canvas to exit
canvas.exitonclick()
```

To what value must we set the variable **theLevel** in order to create this tree?



Question?

```
# ***Main part of my program

# Creates a graphics window "canvas"
canvas = turtle.Screen()

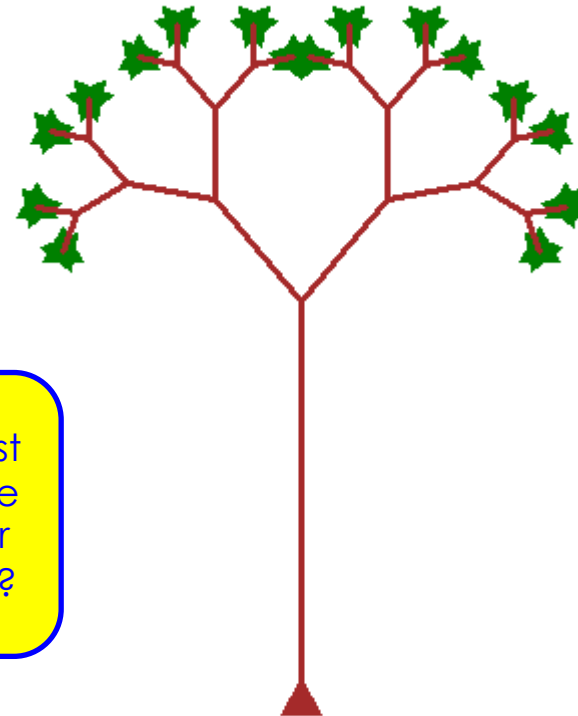
# Create a turtle named "tt"
tt = turtle.Turtle()

# Set up our turtle "tt"
tt.color("brown")
tt.width(3)
tt.shape("triangle")

# Move our turtle "tt"
tt.speed(0)
tt.penup()
tt.goto(0, -180)
tt.left(90)
tt.pendown()
theBranchLength = 100
tt.forward(theBranchLength)

# Call our recursive function
theLevel = 
drawTree(tt, theLevel, theBranchLength)
tt.back(theBranchLength)

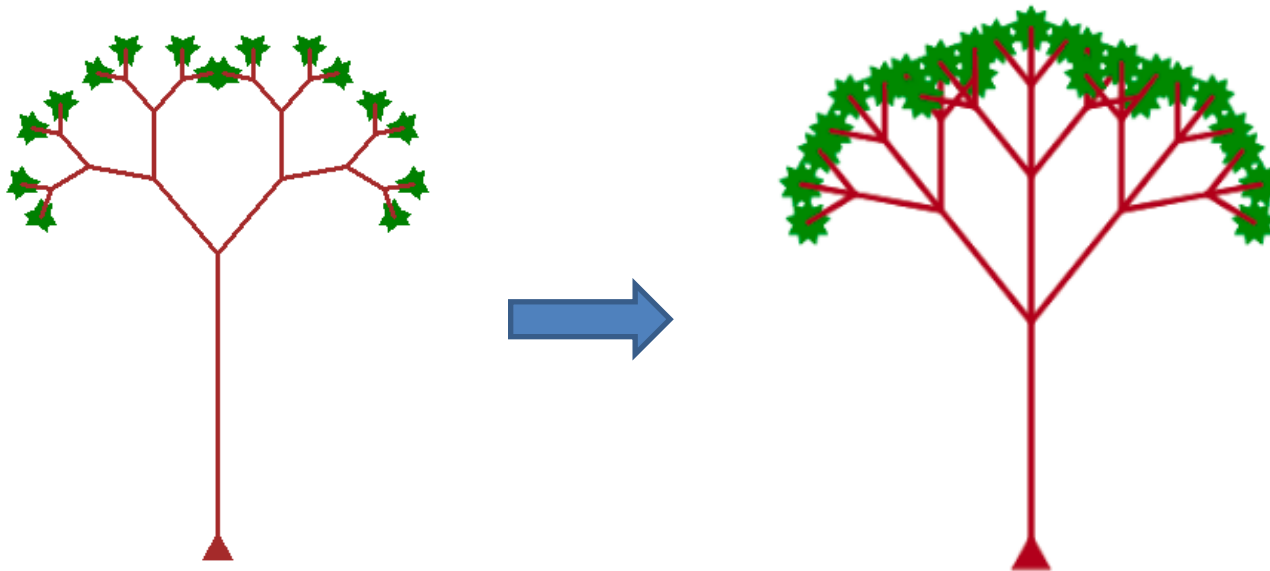
# Click on the canvas to exit
canvas.exitonclick()
```



To what value must we set the variable **theLevel** in order to create this tree?

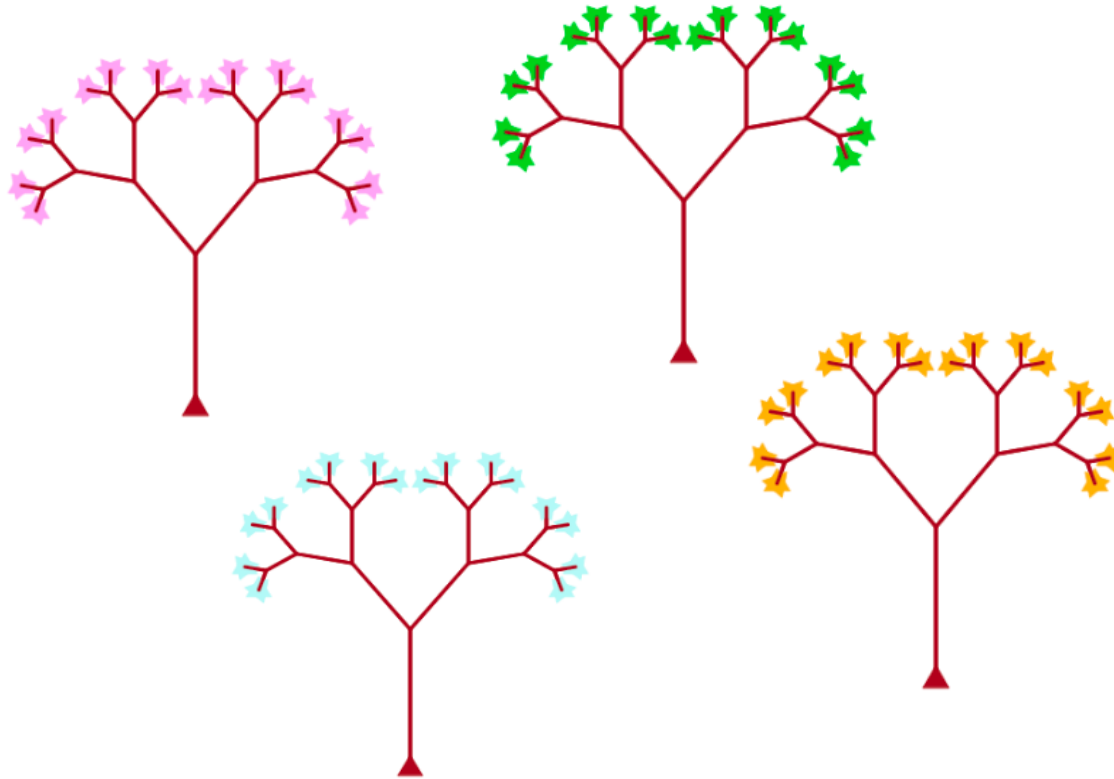
Challenge?

- How would you modify the recursive function `drawTree` to get this tree?



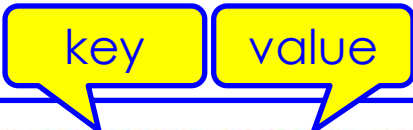
Here is another tree problem!

- **Step 1 - Problem Statement**
 - Draws a tree for every season!



Dictionary

- A **look-up table**
- A compound data type that associates a **key** to a **value**



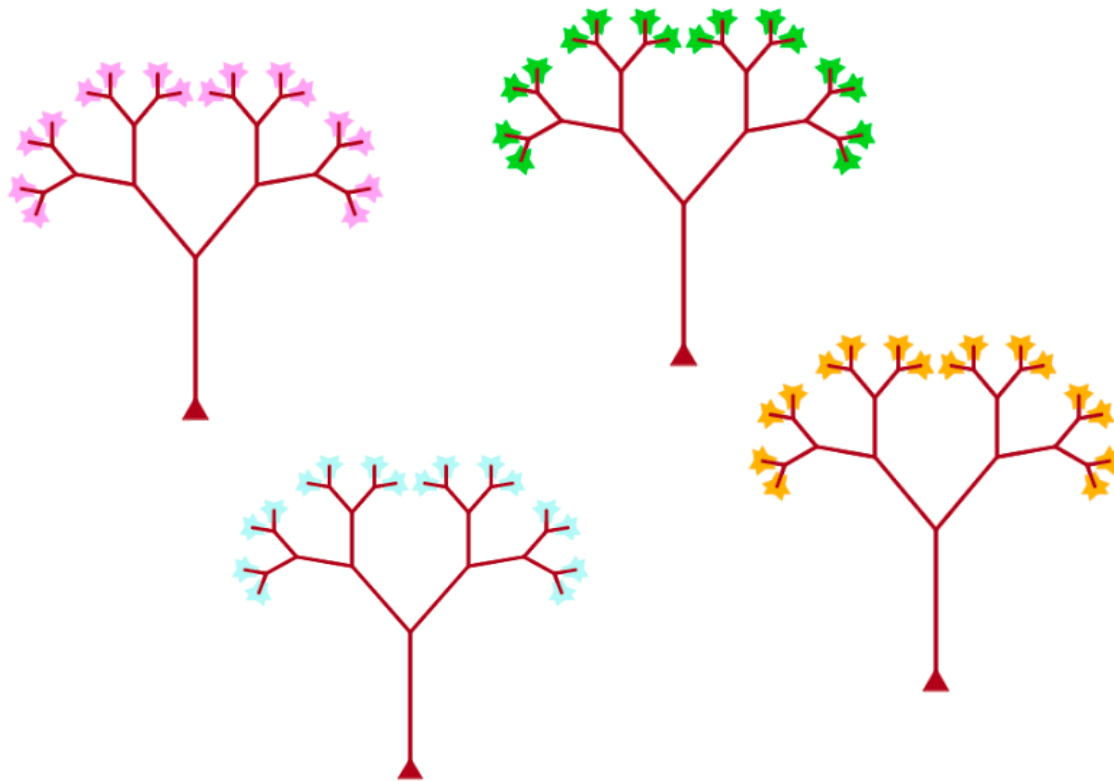
The diagram consists of two yellow speech bubbles with blue outlines. The first bubble contains the word 'key' and has a tail pointing to the word 'bonjour' in the code snippet below. The second bubble contains the word 'value' and has a tail pointing to the string 'hello' in the same code snippet.

```
french_dictionary = { "bonjour":"hello", "au revoir":"goodbye"}  
print(french_dictionary["bonjour"])
```

- Not ordered
- To access elements of the dictionary we use the **key**, not an **index**
- Mutable

Let's get coding with Dictionary!

- **Step 3 - Implementation**



Next Lecture

- **Practice Exam 6** on Wednesday!
- Please, bring paper, pens/pencils and all your **questions** to our lecture on Wednesday!
- Our midterm is Friday!
- After our midterm, we shall start investigating another field of Computing Science
Computer Vision!