**Start les.** (5 min)

* Doel van de les: Introductie recursie met behulp van turtles

**Opdracht 1** (10 min)

* Herhaling functies. Bespreek met studenten eerst de verwachte output.
* Laten zien hoe python tutor werkt

**Opdracht 2** (20 min)

* Introductie tot Turtle. Bespreek de code met studenten stap voor stap voordat ze de code uit proberen en aan de opdracht werken.
* voor de sq() functie kunnen studenten gebruik maken van de tri() functie voor inspiratie.
* Voor de bril() functie wijs ze op de drie P’s. Probeer, plan, programmeer
* De functie triRec() gaat oneindig door met tekenen. Bespreek met de studenten hoe ze de functie zouden kunnen stoppen.

**Opdracht 3** (15 min)

* a. Bespreek mogelijke resultaten met de studenten.
* b. gebruik de Python tutor om stap voor stap het programma door te gaan.
* c. Bespreek mogelijke resultaten met de studenten
* d. gebruik de python tutor om stap voor stap door het programma te gaan.

**pauze**

**Introductie recursie** (5 min)

* Wat is recursie
* Belang van de noodstop/base case

**Opdracht 4** (15 min)

* leg de signature triRec(n=3) uit.
* bespreek resultaat sqRec(n=4)

**opdracht 5** (20 min)

* Nadruk op probeer, plan, programmeer.

**Antwoorden**

**Opdracht 1**

Blaat maakt een lijst van gegeven lengte.

Ouput is [0, 1, 2, 3, 4, 0, 1, 2]

**Opdracht 2**

import time

from turtle import \*

from random import \*

def main():

    """

    Main functie. Roept de andere functies op om hun werk te doen.

    """

    art()

    done() # tell turtle the drawing is done.

def testing():

    """

    Test functie. Hier staan alle assertions om de functies te testen.

    """

def tri(lengte):

    """Draws 100-pixel sides of an equilateral triangle.

    """

    width(5)    # width of the line to draw

    clr = choice(['darkgreen', 'red', 'blue'])  #choose a random color

    color(clr)  # set the color of the line

    shape('turtle') # set the shape of the pencil

    dot(10, 'red')  # set the endpoints of the lines

    forward(lengte)    #move forward

    right(120)       #turn 120 degrees right

    forward(lengte)    #move forward

    right(120)       #turn 120 degrees right

    forward(lengte)    #move forward

    right(120)       #turn 120 degrees right

def sq():

    """Draws 100-pixel sides of a square.

    """

    width(5)    # width of the line to draw

    clr = choice(['darkgreen', 'red', 'blue'])  #choose a random color

    color(clr)  # set the color of the line

    shape('turtle') # set the shape of the pencil

    dot(10, 'red')  # set the endpoints of the lines

    forward(100)

    right(90)

    forward(100)

    right(90)

    forward(100)

    right(90)

    forward(100)

    right(90)

def bril():

    sq()

    forward(150)

    sq()

def art():

    bril()

    forward(100)

    right(90)

    bril()

    forward(100)

    right(90)

    bril()

    forward(100)

    right(90)

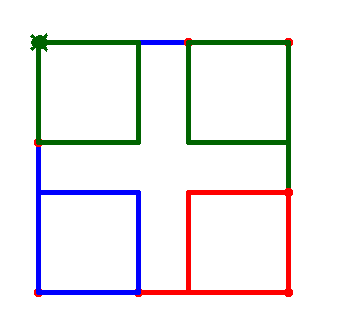
    bril()

    forward(100)

    right(90)

main()

testing()



art()

(kleuren kunnen verschillen.)

**Opdracht 3**

1. print uiteindig door. 5, 4, 3, 2, 1, 0, -1, -2, -3….
2. De python tutor geeft een foutmelding dat de recursie infinite is, maar laat nog steeds zien hoe de code werkt.
3. printen eindigt nu wel. 5, 4, 3, 2, 1, 0

**Opdracht 4**

def sqRec(n=4):

    """Draws 100-pixel sides of a square

    """

    if n == 0:

        return

    width(5)    # width of the line to draw

    clr = choice(['darkgreen', 'red', 'blue'])  #choose a color

    color(clr)  # set the color of the line

    shape('turtle') # set the shape of the pencil

    dot(10, 'red')  # set the endpoints of the lines

    forward(100)    #move forward

    right(90)       #turn 120 degrees left

    sqRec(n-1)

**Opdracht 5**

def polygoon(r, n):

    """Draws 100-pixel r sides of a n polygoon

    """

    if r == 0:

        return

    width(5)    # width of the line to draw

    clr = choice(['darkgreen', 'red', 'blue'])  #choose a color

    color(clr)  # set the color of the line

    shape('turtle') # set the shape of the pencil

    dot(10, 'red')  # set the endpoints of the lines

    forward(100)    #move forward

    right(360/n)       #turn 120 degrees left

    polygoon(r-1, n)