Python Programming

Turtle graphics

Mariusz Dzieńkowski

Institute of Computer Science
Lublin University of Technology
m.dzienkowski@pollub.pl

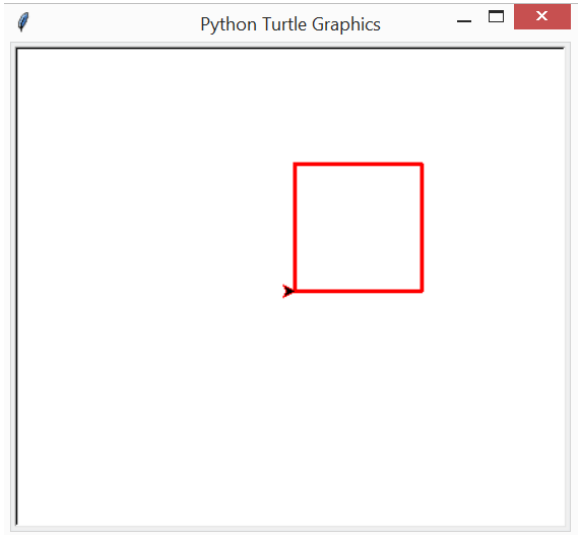
Turtle commands

- Turtle library
  - `from turtle import *`
- Turtle motion
  - `forward(distance) | fd(distance)`
  - `right(angle) | rt(angle)`
  - `left(angle) | lt(angle)`
  - `goto(x,y) | setpos(x,y) | setposition(x,y)`
  - `home()`
  - `dot(size,color)`
  - `home()`
- Pen control
  - `pendown() | pd() | down()`
  - `penup() | pu() | up()`
  - `pensize(number) | width()`
  - `pencolor(colorstring) | pencolor(r,g,b)`

Turtle simple graphics

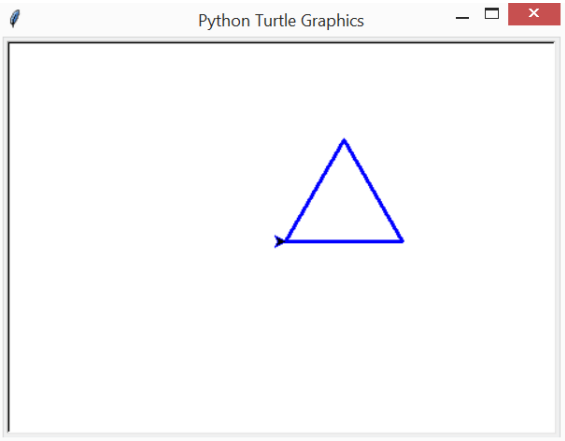
- Simple shapes
  - Square

```
from turtle import *

pensize(3)
pencolor("red")

forward(100)
left(90)
forward(100)
left(90)
forward(100)
left(90)
forward(100)
left(90)
```



  - Triangle

```
from turtle import *

pensize(3)
pencolor("blue")

forward(100)
left(120)
forward(100)
left(120)
forward(100)
left(120)
```



Loops

- ”for ” loop
  - triangle

```
from turtle import *

pensize(3)
pencolor("blue")

for i in range(3):
    forward(100)
    left(120)
```

- ”while” loop
  - square

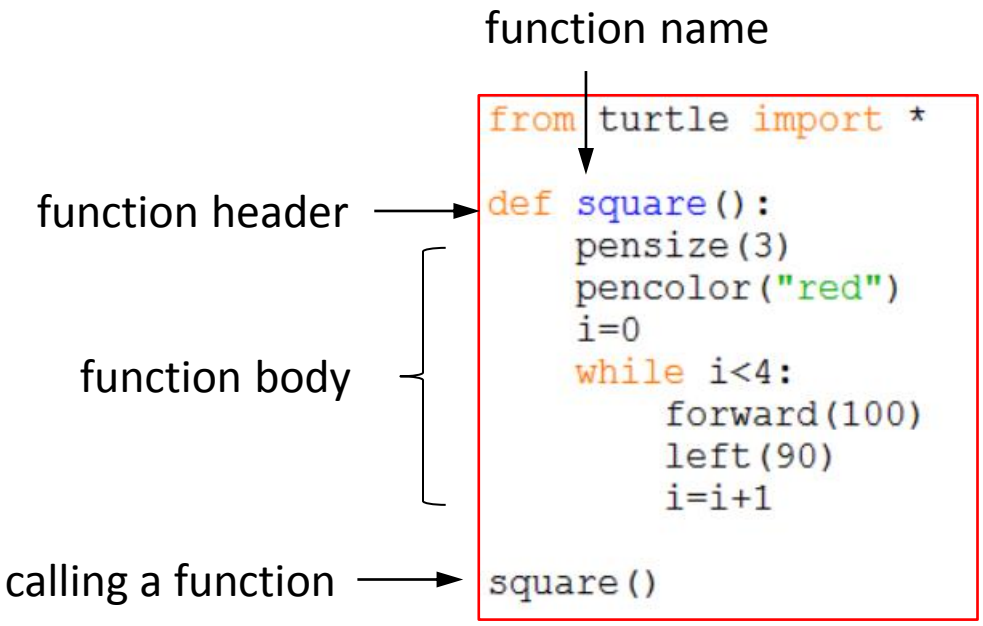
```
from turtle import *

pensize(3)
pencolor("red")

i=0

while i<4:
    forward(100)
    left(90)
    i=i+1
```

User-defined functions

- Function definition
  - keyword **def** followed by the function **name**
  - arguments inside the opening and closing **parentheses** ending the declaration with a **colon**
  - program **statements**
  - ending the function with/without **return** statement
- Calling a function - executes the code in the function
  - by **name** of the fuction
  - parentheses with/without **arguments**

User-defined functions

- Define a function

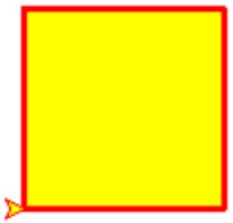


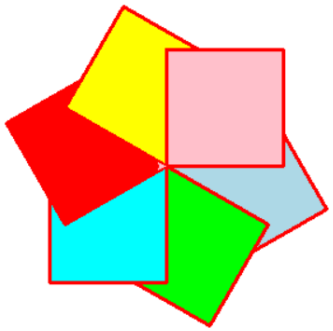
User-defined functions

- Function with arguments

```
from turtle import *

def square(color):
    pensize(3)
    pencolor("red")
    fillcolor(color)
    begin_fill()
    i=0
    while i<4:
        forward(100)
        left(90)
        i=i+1
    end_fill()

square("yellow")
```

```
def drawing(angle):
    rt(angle)
    square("light blue")
    rt(angle)
    square("lime")
    rt(angle)
    square("cyan")
    rt(angle)
    square("red")
    rt(angle)
    square("yellow")
    rt(angle)
    square("pink")

drawing(360/6)
```

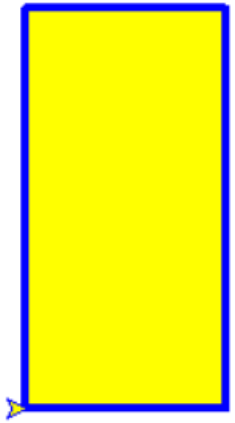




User-defined functions

- Function with arguments

```
from turtle import *

def rectangle(a,b,w,pc,f):
    pencolor(pc)
    width(w)
    fillcolor(f)
    begin_fill()
    for i in range(2):
        fd(a)
        lt(90)
        fd(b)
        lt(90)
    end_fill()

rectangle(100,200,4,"blue","yellow")
```

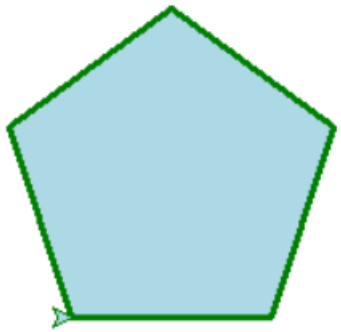


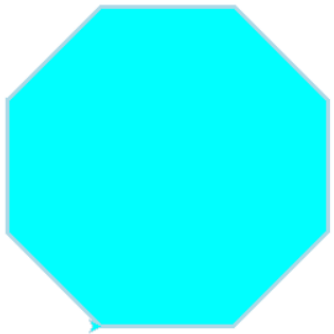
Problem 1

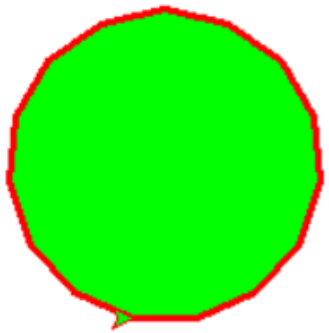
A polygon as a turtle drawing

- App that draws an n-sided polygon by means of a turtle
- Problem solution
  - Enter values: number of sides, length of sides and filling color
  - Calculate an angle using the formula:

    360 / number of sides
  - Use the turtle module to provide a graphical display
  - Using a for-loop statement, draw a polygon







Problem 1 - solution

- Source code

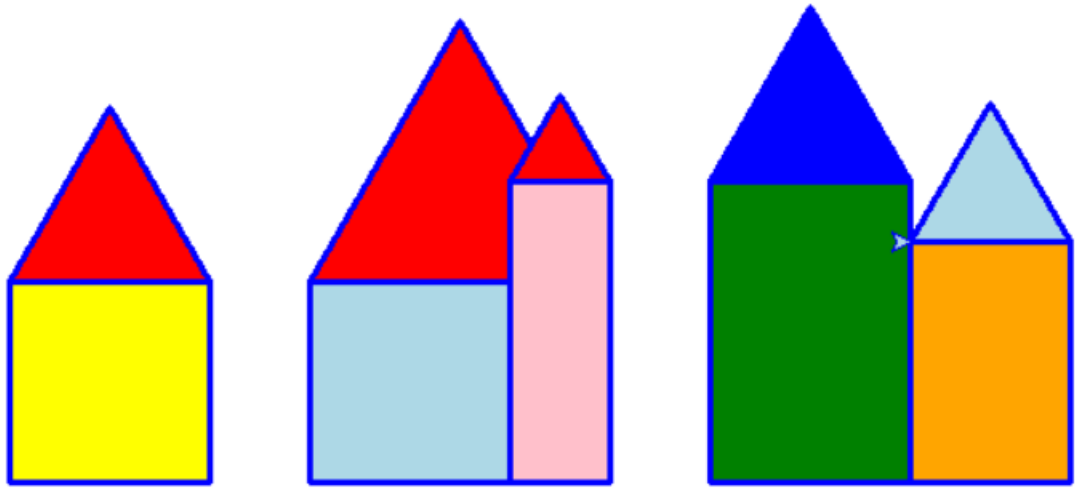
```
from turtle import *

def polygon(sides,size,color):
    width(3)
    pencolor("blue")
    fillcolor(color)
    begin_fill()
    for i in range(1,sides+1,1):
        fd(size)
        lt(360/sides)
    end_fill()

sides = int(input("Number of sides: "))
size = int(input("Length of a side: "))
color = input("Color: ")

polygon(sides,size,color)
```

Problem 2

Picture

- App that draws the picture by using user-defined functions with arguments:

```
rectangle(a,b,col), polygon(a,col), house(a,b,c1,c2),picture()
```



Problem 2 - solution

- Source code

```
from turtle import *

def polygon(sides,size,color):
    width(3)
    pencolor("blue")
    fillcolor(color)
    begin_fill()
    for i in range(1,sides+1,1):
        fd(size)
        lt(360/sides)
    end_fill()
```

```
def rectangle(x,y,c):
    width(3)
    pencolor("blue")
    fillcolor(c)
    begin_fill()
    for i in range(2):
        fd(x)
        lt(90)
        fd(y)
        lt(90)
    end_fill()
```

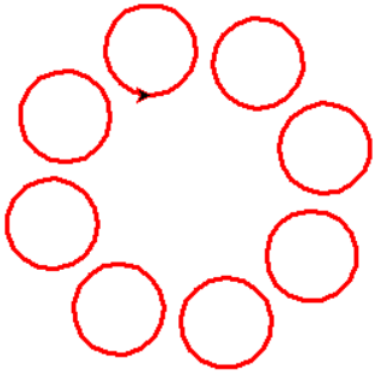
```
def house(posX,posY,sideA,sideB,fillcolor1,fillcolor2):
    pu()
    goto(posX,posY)
    down()
    rectangle(sideA,sideB,fillcolor1)
    goto(posX,sideB)
    polygon(3,sideA,fillcolor2)
```

```
def picture():
    house(-300,0,100,100,"yellow","red")
    house(-150,0,150,100,"light blue","red")
    house(-50,0,50,150,"pink","red")
    house(50,0,100,150,"green","blue")
    house(150,0,80,120,"orange","light blue")

picture()
```

Problem 3

Turtle drawing

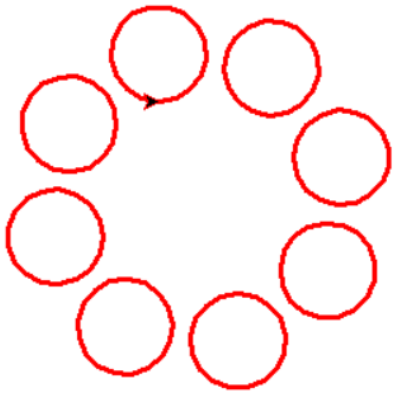
- Program that creates a drawing containing 8 circles
- Problem solution
  - Use the turtle module to provide a graphical display
  - Use a for-loop to repeat drawing of 8 circles
  - Use turtle's methods:
    - `circle(diameter)`
    - `penup()`
    - `forward(distance)`
    - `right(angle)`
    - `pendown()`



Problem 3 - solution

- **Source code**

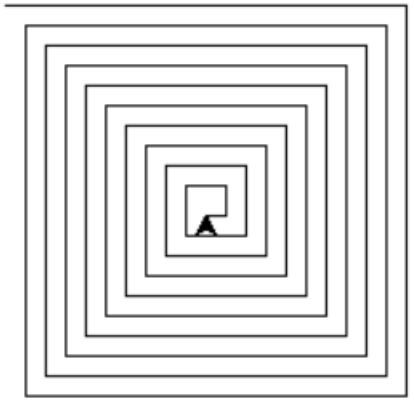
```
from turtle import *

pen(fillcolor="black", pencolor="red", pensize=3)

for i in range (8):
  circle(30)
  penup()
  forward(50)
  right(45)
  pendown()
```

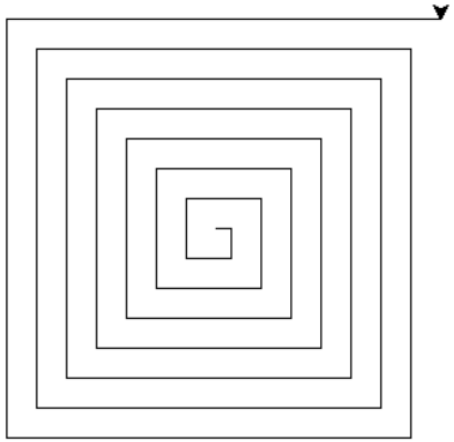


Problem 4

Spiral

- Program that drawes two spirals
- Problem solution
  - Create two functions: `spiral1(number of steps)` and `spiral2(length of sides)`
  - In the first function use a while-loop to repeat drawing parts of a spiral
  - In the second function use a recursion method





Problem 4 - solution

- **Source code**

```
from turtle import *

def spiral1(step):
    p=10
    x=1
    while x<step:
        fd(p)
        rt(90)
        p=p+10
        x=x+1
```

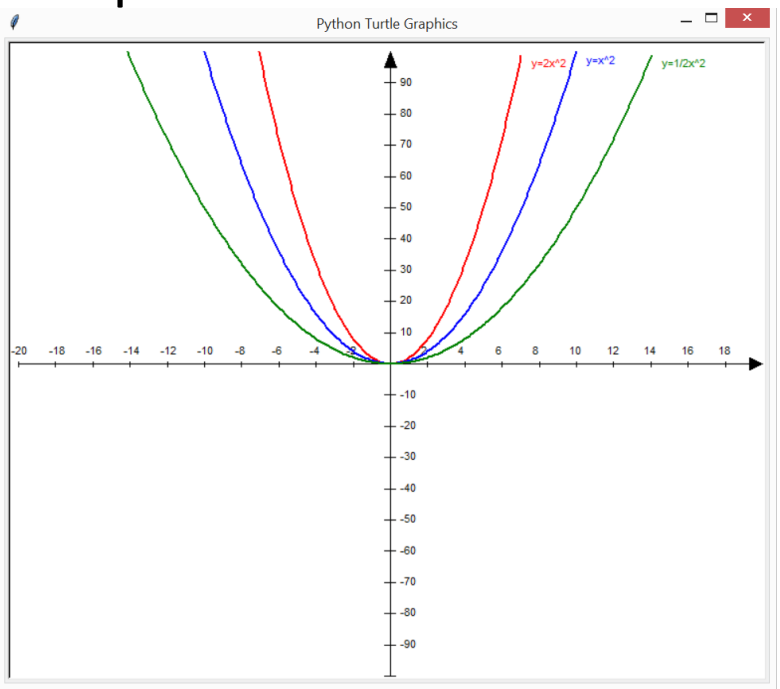
```
def spiral2(side):
    if side<10: return
    fd(side)
    rt(90)
    side=side-5
    spiral2(side) #recursion
```

```
def main():
    spiral1(20)
    up()
    goto(-100,100)
    down()
    spiral2(200)

main()
```

Problem 5

Quadratic equation plot

- Program that draws graphs of three quadratic functions
  - y = x^2
  - y = 2x^2
  - y = 1/2x^2
- Write the following functions:
  - `axisX(), axisY(), leftArrow(), topArrow()`
  - `drawGraph(a,color,str)`



where:

- argument ”a” is the equation coefficient - y=ax^2
- `color` – the color of a graph of a function
- `str` – string e.g. ”y=x^2”

Problem 5 – solution (1)

- **Source code**

```
import math
from turtle import *

def axisX():
    width(1)
    pen(pencolor="black")
    penup()
    goto(20,0)
    pendown()
    goto(-20,0)
    for i in range(-20,20,2):
        penup()
        goto(i-0.3,2)
        if i!=0:
            write(i)
        goto(i,-1)
        pendown()
        goto(i,1)
```

```
def leftArrow():
    penup()
    begin_fill()
    goto(19.3,-2)
    pendown()
    goto(19.3,2)
    goto(20,0)
    goto(19.3,-2)
    end_fill()
    penup()
```

Problem 5 – solution (2)

- **Source code**

```
def axisY():
    goto(0,100)
    pendown()
    goto(0,-100)
    for i in range(-100,100,10):
        penup()
        goto(-0.3,i)
        pendown()
        goto(0.3,i)
        penup()
        goto(0.6,i-2)
        pendown()
        if i%10==0 and i!=0 and i>-100 and i<100: write(i)
```

```
def topArrow():
    fillcolor("black")
    penup()
    begin_fill()
    goto(-0.3,95)
    pendown()
    goto(0,100)
    goto(0.3,95)
    goto(-0.3,95)
    end_fill()
```

Problem 5 – solution (3)

- **Source code**

```
def drawGraph(a,color,str):
    pen(pencolor=color)
    width(2)
    penup()
    y=100
    x=math.sqrt(y/a)
    goto(-x,y)
    pendown()
    i=-x;
    while i < x:
        y = a*i*i
        goto(i,y)
        i=i+0.1
    up()
    goto(i+0.5,y-5)
    write(str)
```

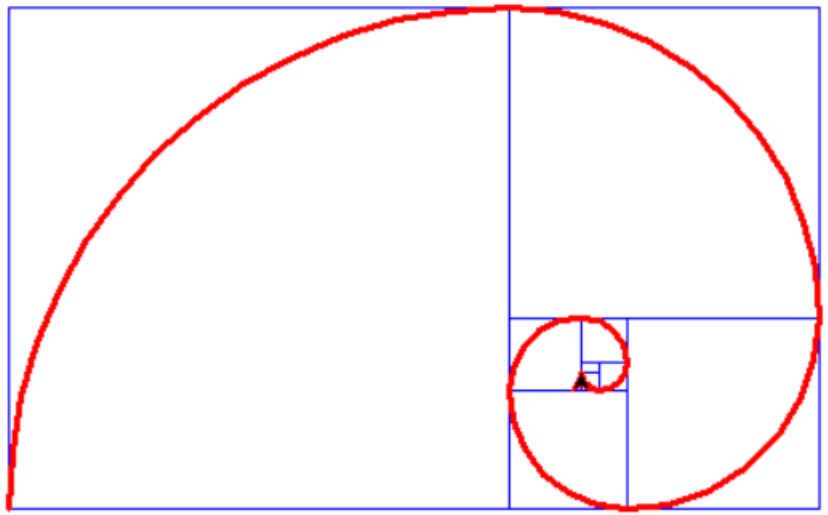
```
def main():
    setworldcoordinates(-20,-100,20,100)
    ht()
    axisX()
    leftArrow()
    axisY()
    topArrow()
    drawGraph(2,"red","y=2x^2")
    drawGraph(1,"blue","y=x^2")
    drawGraph(1/2,"green","y=1/2x^2")

main()
```

Problem 6

Golden Rectangle

- Program that draws golden rectangle, subdivides it into squares, and then draws a pseudo-spiral by connecting the corners of the squares with arcs.
- Side lengths of golden ractangle are in the golden ratio

$$\varphi = \frac{1+\sqrt{5}}{2} \cong 1.618 \ldots$$



Problem 6 - solution

- **Source code**

```
from turtle import *
from math import sqrt

phi = (1+sqrt(5))/2

def square(side_length):
  for i in range(4):
    forward(side_length)
    right(90)
```

```
def goldenRect(size):
  speed(6)
  penup()
  goto(-190,-100)
  left(90)
  pencolor("blue")
  pendown()
  tsize = size

  for i in range(8):
    square(size)
    forward(size)
    right(90)
    forward(size)
    size = size/phi

  penup()
  goto(-190,-100)
  pendown()
  width(3)
  pencolor("red")
  radius = tsize
  for i in range(8):
    circle(-radius,90)
    radius = radius/phi

goldenRect(250)
```

Homework 1

Aquarium animation

- Program that simulates a fish motion in an aquarium
- Problem solution
  - Download 2 files: fishR.gif and fishL.gif from: dune.pol.lublin.pl/~mdz/erasmus2018/
  - Use turtle's methods:
    - `shape('fishR.gif')` – to set the turtle's shape
    - `addshape('fishR.gif')` – to register the image with the screen
    - `setup(width,height)` – to set the size of the window
    - `hideturtle() | ht()`
    - `showturtle() | st()`