Lekcja 5. Fraktale – Płatek Kocha

Uczeń
- omawia cechy charakterystyczne fraktala
- wykorzystuje do rysowania moduł turtle
- rysuje krzywą i płatek Kocha
- wyjaśnia krótko pojęcie rekurencji

Python Turtle to narzędzie służące do pisania aplikacji graficznych.
Biblioteka turtle jest dołączona do Python 3 więc wystarczy ją tylko zaimportować. Kolejna linika kodu tworzy obiekt zółwia i następna nadaje kształt kursorowi (zamiast turtle można napisać np. triangle lub square)
Dalsza część kodu to już sposób, w jaki ma poruszać się kursor i na końcu jest funkcja exitonclick(), która powoduje, że okienko się nie zamknie po wykonaniu poleceń:

```
import turtle
kursor = turtle.Turtle()
kursor.shape('turtle')

for ruch in range(36):
    for ruch in range(4):
        kursor.forward(200)
        kursor.right(90)
    kursor.right(10)

kursor.home()

for ruch in range(36):
t.circle(50)
t.right(45)

turtle.exitonclick()
```

Do kodu można dopisać kolejne linijki,któ®e pozwolą uzyskać także koła wpisane w dotychczasową strukturę kwadratów.

```
for ruch in range(36):
    kursor.circle(100)
    kursor.right(10)
```

Kursor może jeszcze poruszać się w tył po zastosowaniu funkcji back() (w skrócie bk()), może przenieść się do konkretnego punktu o podanych współrzędnych goto(x,y).
color(x) zmienia kolor rysowanej linii
bgcolor(x) lub bgcolor(x,y,z) zmienia kolor tła na podany po angielsku w apostrofach lub w RGB
pensize(x) ustawia grubość linii
penup(), przemieszcza się bez rysowania
pendown(),
hideturtle() i showturtle() sprawia, że kursor znika i pojawia się
write(tekst) kursor pisze
speed(x) ustawia prędkość rysowania
begin_fill() i end_fill() ustawione przedkodem rysowania i po nim sprawią, że figura zostanie pomalowana.
Aby się upewnić, jaka jest aktualna pozycja kursora, wystarczy skorzystać z funkcji position()
Poniżej jeszcze inny ciekawy przykład ze zmieniającym się kolorem rysowanych linii:

```
import turtle
import random

turtle.bgcolor('black')
kursor = turtle.Turtle()
kursor.speed(0)
for x in range(50, 300):
    kursor.color(random.choice(['red', 'green', 'blue']))
```

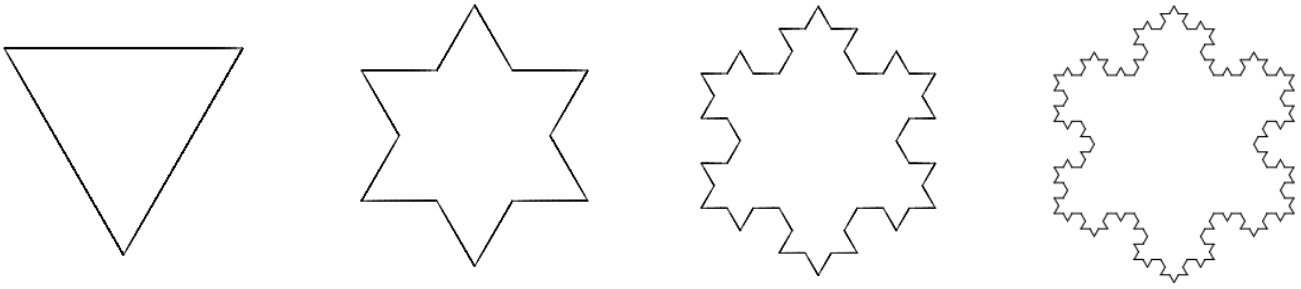
```
    kursor.forward(x)
    kursor.right(91)

turtle.exitonclick()
```

Można też sprawić, aby kursor reagował na naciśnięcie klawiszy. Wtedy to użytkownik może nim sterować. Trzeba stworzyć definicje, za pomocą których będzie możliwa zmiana położenia kursora. Funkcja onkey() pozwoli na wykonanie zadania (podanej jako pierwsza – definicji) za pomocą klawisza, który zostanie wpisany jako parametr.
W poniższym kodzie ważne są jeszcze funkcje listen() i mainloop(), które sprawią, że program będzie oczekiwał na użycie określonego klawisza wielokrotnie.

```
import turtle
kursor = turtle.Turtle()
def naprzod():
    kursor.forward(10)
def do_tylu():
    kursor.back(10)
def lewo():
    kursor.left(10)
def prawo():
    kursor.right(10)
def koniec():
    turtle.bye()
turtle.onkey(naprzod,'Up')
turtle.onkey(lewo,'Left')
turtle.onkey(prawo,'Right')
turtle.onkey(do_tylu, 'b')
turtle.onkey(koniec,'q')
turtle.listen()
turtle.mainloop()
```

Pozostaje już teraz wykonać projekty, związane z często prezentowanymi krzywymi fraktalnymi. Krzywa Kocha ma nieskończoną złożoność, ale mieści się na skończonej powierzchni.
Płatek Kocha o złożoności n=0 składa się z trzech takich krzywych, połączonych ze sobą w trójkąt równoboczny.
Kolejne przykłady złożoności są zaprezentowane poniżej:

Złożoność n=0 Złożoność n=1 Złożoność n=2 Złożoność n=3



Kod dla złożoności płatka n=0

```
import turtle
kursor = turtle.Turtle()

def krzywaKocha(bok):
    kursor.forward(bok)

def platekKocha(bok):
    for figura in range(3):
        krzywaKocha(bok)
        kursor.right(120)

kursor.penup()
kursor.goto(-200, 150)
kursor.pendown()
```

```
platekKocha(500)
turtle.exitonclick()
```

Kod dla złożoności płatka n=1

```
import turtle
kursor = turtle.Turtle()

def krzywaKocha1(bok):
        kursor.forward(bok)

def krzywaKocha2(bok):
    krzywaKocha1(bok/3)
    kursor.left(60)
    krzywaKocha1(bok / 3)
    kursor.right(120)
    krzywaKocha1(bok / 3)
    kursor.left(60)
    krzywaKocha1(bok / 3)

def platekKocha(bok):
    for figura in range(3):
        krzywaKocha2(bok)
        kursor.right(120)

kursor.penup()
kursor.goto(-200, 150)
kursor.pendown()
platekKocha(500)
turtle.exitonclick()
```

Kod dla złożoności płatka n=2

```
import turtle
kursor = turtle.Turtle()

def krzywaKocha1(bok):
        kursor.forward(bok)

def krzywaKocha2(bok):
    krzywaKocha1(bok/3)
    kursor.left(60)
    krzywaKocha1(bok / 3)
    kursor.right(120)
    krzywaKocha1(bok / 3)
    kursor.left(60)
    krzywaKocha1(bok / 3)

def krzywaKocha3(bok):
    krzywaKocha2(bok/3)
    kursor.left(60)
    krzywaKocha2(bok / 3 )
    kursor.right(120)
    krzywaKocha2(bok / 3)
    kursor.left(60)
    krzywaKocha2(bok / 3)


def platekKocha(bok):
    for figura in range(3):
        krzywaKocha3(bok)
        kursor.right(120)
```

```
kursor.penup()
kursor.goto(-200, 150)
kursor.pendown()
platekKocha(500)
turtle.exitonclick()
```

Ostateczna wersja programu rysującego płatek Kocha wygląda następująco:

```
import turtle
kursor = turtle.Turtle()

def krzywaKocha(bok, zlozonosc):
    if(zlozonosc== 0):
        kursor.forward(bok)
    else:
        krzywaKocha(bok /3,zlozonosc-1)
        kursor.left(60)
        krzywaKocha(bok /3,zlozonosc-1)
        kursor.right(120)
        krzywaKocha(bok /3,zlozonosc-1)
        kursor.left(60)
        krzywaKocha(bok /3,zlozonosc-1)

def platekKocha(bok, zlozonosc):
    for figura in range(3):
        krzywaKocha(bok, zlozonosc)
        kursor.right(120)

zlozonosc = input("Podaj złożoność płatka: ")
turtle.tracer(0,0)
#kursor.speed(0)
kursor.penup()
kursor.goto(-200, 150)
kursor.pendown()
platekKocha(500, int(zlozonosc))

turtle.exitonclick()
```

Ciekawa strona do odwiedzenia:
https://docs.python.org/3/library/turtle.html