

Turtle – želva

Je grafické pero, které si pamatuje pozici, směr natočení a další parametry a v grafické ploše na základě pokynů kreslí vektorové obrazce.

`t = turtle.Turtle()` - vytvoří grafickou plochu a v jejím středu pero natočené na východ (doprava)

Při čemž platí:

- Střet souřadnicové soustavy je ve středu grafické plochy
- Souřadnice x a y mají tedy i záporné hodnoty
- Směr otočení určujeme v stupních (ne radiánech) v protisměru hodinových ručiček
- Pozice a směr je vizualizovaná natočením obrázku kreslicího per

Při práci mimo idle je dobré zadat příkaz udržující kreslicí okno ve smyčce – tudíž otevřené a je možné použít jednu z těchto metod:

- **`turtle.mainloop()`** – nejčastěji používaná smyčka – okno se uzavírá kliknutím na křížek
- **`turtle.done()`** – stejné jako `turtle.mainloop()`
- **`turtle.exitonclick()`** – okno se zavře kliknutím do grafické plochy

Vybarvování útvarů:

- začátek kreslení objektu, který chceme vybarvit označíme příkazem **`begin_fill()`**
- konec objektu, který chceme vybarvit označíme příkazem **`end_fill()`**
- barvu měníme příkazem **`fillcolor(farba)`** – defaultně je nastavena černá barva
- když nakreslíme křivku, která není uzavřený útvar, křivka se při vyplňování barvou uzavře
- pokud nechceme barvu obrysu, kreslíme objekt se zvednutým perem

Globální funkce (Mají vliv na všechny kreslicí pera):

`turtle.delay(číslo)` - zpomalení vykonávání metod (kreslení) na zadaný počet milisekund
- standartně je 10 (0 je nejrychlejší)

`t.speed(číslo)` - individuální zrychlení / zpomalení jednotlivé želvy
- číslo je od 0 do 10 (0 nejrychlejší, standartně je 3)

`turtle.tracer(číslo)` - zapne / vypne průběžné zobrazování změn v grafické ploše
- standartně je 1
- 0 vypne zobrazování změn – zobrazí se pouze výsledek v kratším čase
- 1 zapne zobrazování změn – vydáme všechny změny

`turtle.bgcolor(farba)` - změni barvu pozadí plochy

Seznam metod:

Pohyb:

metoda	varianty zápisu	význam	příklad
forward(d)	fd	jdi dopředu	t.fd(100); t.fd(-50)
back(d)	backward, bk	jdi dozadu	t.bk(50); t.bk(-10)
right(u)	rt	otoč se vpravo	t.rt(90); t.rt(-120)
left(u)	lt	otoč se vlevo	t.lt(90); t.lt(-45)
setpos(x, y)	setposition, goto	jdi na pozici	t.setpos(50, 70)
towards()		natoč se k...	t[i].towards(t[j])

Kreslení:

metoda	varianty zápisu	význam	příklad
penup()	pu, up	zdvihni pero	t.pu()
pendown()	pd, down	zpuť pero	t.pd()

Smazání kresby:

metoda	varianty zápisu	význam	příklad
reset()		smaž kresbu a inicializuj pero	t.reset()
clear()		smaž kresbu	t.clear()

Info:

metoda	varianty zápisu	význam	příklad
pos()	position	zjistí pozici pera	t.pos()
xcor()		zjistí x-ovou souřadnici	t.xcor()
ycor()		zjistí y-ovou souřadnici	t.ycor()
heading()		zjistí úhel směru	t.heading()
pencolor()		zjistí barvu pera	t.pencolor()
fillcolor()		zjistí barvu výplně	t.fillcolor()
color()		zjistí barvu pera a výplně	t.color()

Nastavení hodnot:

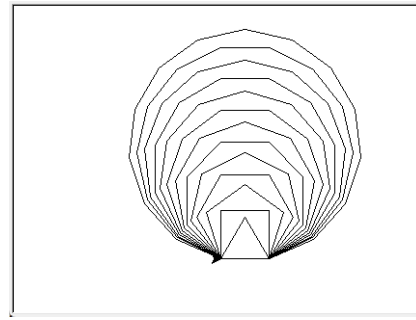
metoda	varianty zápisu	význam	příklad
setheading(u)	seth	nastav úhel směru	t.seth(120)
pensize(h)	width	nastav tloušťku pera	t.pensize(5)
pencolor(f)		nastav barvu pera	t.pencolor('red')
fillcolor(f)		nastav barvu výplně	t.fillcolor('blue')
color(f1, f2)		nastav barvu pera aj výplně	t.color('red', 'blue')
begin_fill()		začátek budoucího vybarvení	t.begin_fill()
end_fill()		konec budoucího vybarvení	t.end_fill()

Tvar korytnačky:

t.shapesize(sirka, vyska, hrubka) - změni vzhled kreslího pera
shape() - změni tvar kreslího pera
t.shape('arrow') - šipka
t.shape('turtle') - želva
t.shape('circle') - kruh
t.shape('square') - čtverec
t.shape('triangle') - trojúhelník
t.shape('classic') - hrot šipky

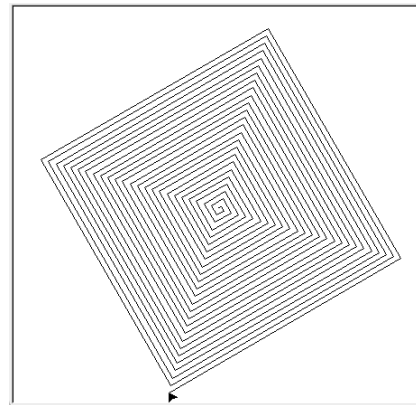
Kreslení n-úhelníku:

```
def n_uholnik(n, d):  
    for i in range(n):  
        t.fd(d)  
        t.lt(360 / n)
```



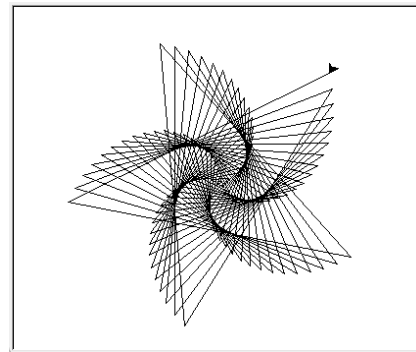
Kreslení spirál:

```
t.lt(30)  
for i in range(3, 300, 3):  
    t.fd(i)  
    t.rt(90)
```

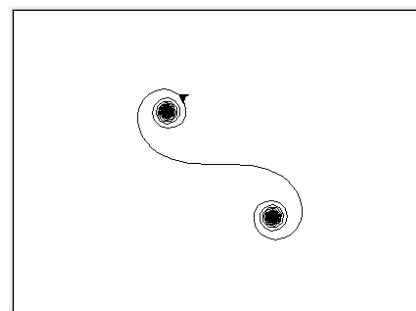


while True:

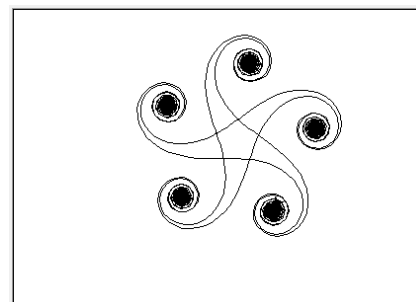
```
    uhol = random.randint(30, 170)  
    print('spirala s uhlom', uhol)  
    for i in range(3, 300, 3):  
        t.fd(i)  
        t.rt(uhol)  
    t.reset()
```



```
for uhol in range(1, 700):  
    t.fd(8)  
    t.rt(uhol)
```



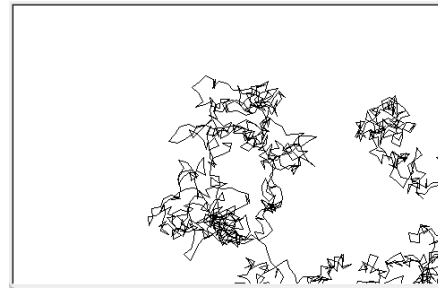
```
for uhol in range(1, 2000):  
    t.fd(8)  
    t.rt(uhol + 0.1)
```



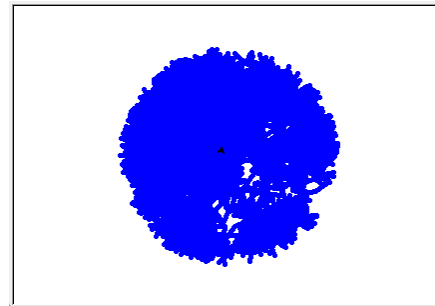
Náhodné procházky:

jsou zadání, při kterých jsou pohyb a úhel náhodně generovanými hodnotami:

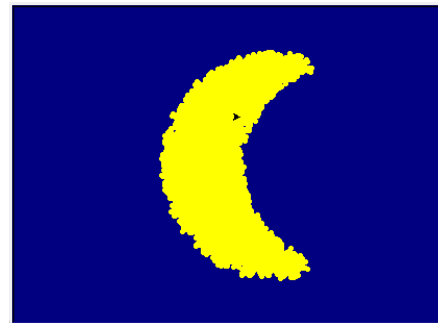
```
for i in range(10000):  
    t.seth(random.randint(0, 359))  
    t.fd(10)
```



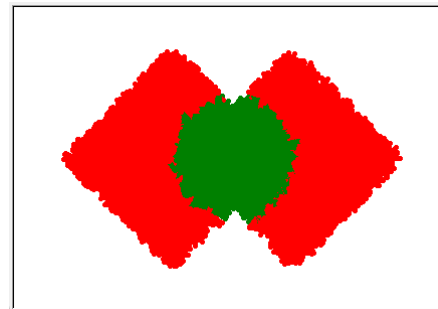
```
t.pensize(5)  
t.pencolor('blue')  
for i in range(10000):  
    t.seth(random.randint(0, 359))  
    t.fd(10)  
    if t.xcor()**2 + t.ycor()**2 > 50**2:  
        t.fd(-10)
```



```
turtle.bgcolor('navy')  
t.pensize(5)  
t.pencolor('yellow')  
for i in range(10000):  
    t.seth(random.randint(0, 359))  
    t.fd(10)  
    if t.distance(40, 0) > 100 or  
t.distance(100, 0) < 100:  
        t.fd(-10)
```



```
def fun(pos):  
    x, y = pos  
    if abs(x - 60) + abs(y) < 100:  
        return False  
    return abs(x + 60) + abs(y) > 100  
t.pensize(5)  
for i in range(10000):  
    t.seth(random.randint(0, 359))  
    if t.distance(0, 0) < 60:  
        t.pencolor('green')  
    else:  
        t.pencolor('red')  
    t.fd(10)  
    if fun(t.pos()):  
        t.fd(-10)
```

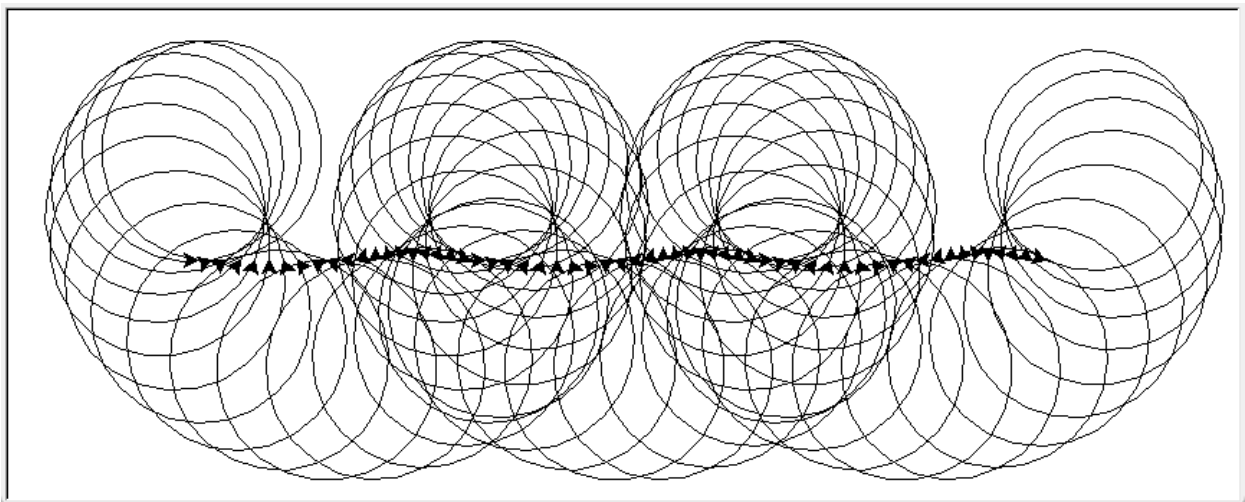


Výčero kreslících per:

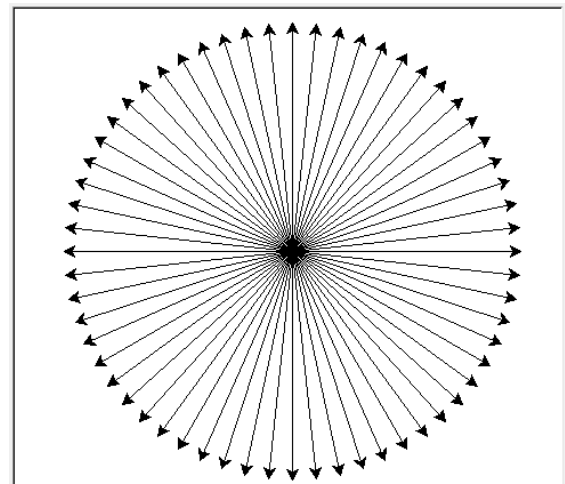
- Kreslících per je možné vytvořit libovolné množství
- Pokud chceme aby více kreslících per využívalo stejné globální proměnné, je potřeba ve funkci přidat nový parametr, který bude ve funkci zastupovat požadované kreslící pero.
- Vygenerované pera můžeme přidat do seznamu, který je pak možné procházet for-cyklem a měnit parametry všech per najednou.

```
zoznam = []  
for i in range(60):  
    t = turtle.Turtle()  
    t.pu()  
    t.setpos(-300 + 10*i, 0)  
    t.pd()  
    t.seth(i * 18)  
    zoznam.append(t)
```

```
for t in zoznam:  
    for i in range(24):  
        t.fd(20)  
        t.lt(15))
```



```
zoznam = []  
for i in range(60):  
    zoznam.append(turtle.Turtle())  
    zoznam[-1].seth(i * 6)  
  
for t in zoznam:  
    t.fd(200)
```



Nahánějící se želvičky:

Na náhodné pozice vygenerujeme kreslící pera a po té je necháme nahánět, podle těchto pravidel:

- Každá se natočí směrem k další v pořadí
- Každá se posune o setinu vzdálenosti
- Nakreslí se spojnice

```
import turtle
import random

turtle.delay(0)
while True:
    turtle.bgcolor('black')
    n = random.randint(3, 8)
    t = []
    for i in range(n):
        nova = turtle.Turtle()
        nova.speed(0)
        nova.pu()
        nova.setpos(random.randint(-200, 200), random.randint(-200, 200))
        nova.pencolor(f'#{random.randrange(256**3):06x}')
        nova.pd()
        nova.ht()
        t.append(nova)

    for k in range(100):
        for i in range(n):
            j = (i+1) % n          # index nasledovnej
            uhol = t[i].towards(t[j])
            t[i].seth(uhol)
            vzdialenost = t[i].distance(t[j])
            t[i].fd(vzdialenost)
            t[i].fd(vzdialenost/10 - vzdialenost)

    for tt in t:
        tt.clear()
# turtle.done()
```

