Aula 4

A biblioteca Turtle é uma coleção de módulos já programados que cria uma tela - ou usando o termo em inglês *canvas* - na qual podemos usar uma tartaruga (usando a lógica de programação) para desenhar e criar formas sobre a tela.
A biblioteca é super intuitiva e recomendada para iniciantes com a linguagem Python. Imaginemos que temos uma tartaruga que conforme ela ande vá deixando um rastro por onde passou, é exatamente isso o principal recurso da biblioteca. Com comandos como "avance" e “regresse”, podemos fazer ela andar e criar um rastro que será o desenho no canvas.

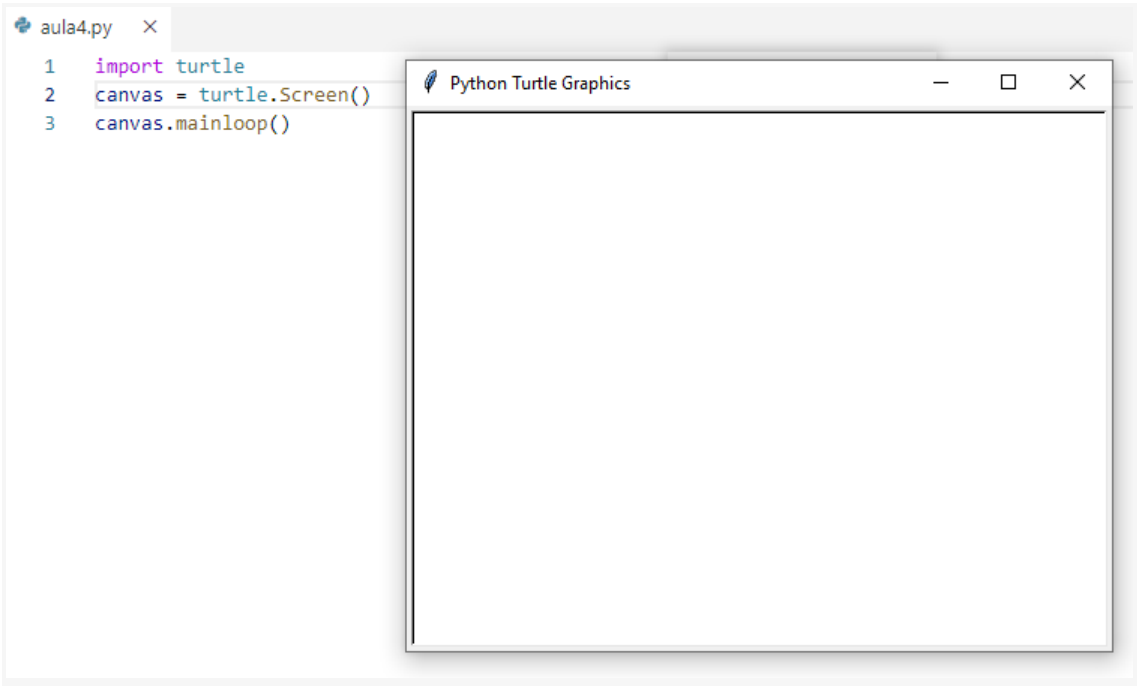
O primeiro passo a seguir é criar um arquivo com a extensão Python. Darei o nome do arquivo de “aula4.py” e abrimos ele no editor de código-fonte de desejo. Com o arquivo aberto, vamos importar para o código a biblioteca turtle:

```
>>> import turtle
```

A Tela

Agora podemos começar a usar os recursos da biblioteca. Primeiro vamos criar um objeto que será o canvas que iremos criar formas em cima. Para criar essa tela, criamos o objeto do tipo Screen e reverenciamos ela em uma variável para que podemos manipular o objeto. O nome da variável será “canvas”.
Podemos compilar o código para ver o canvas aparecer, mas ela não durará poucos segundos pois o código entende que o código já acabou e deve finalizar a execução do programa. Para evitar que a execução encerre, usamos o comando **mainloop**:

```
>>> canvas = turtle.Screen()
>>> canvas.mainloop()
```



Agora vamos dar uma personalizada no objeto canvas. Daremos uma cor de fundo com **bgcolor**, um título com **title** e configuramos o tamanho da tela com **setup**, no qual terá .5 de largura e de comprimento:

```
>>> canvas.bgcolor("blue")
>>> canvas.title("aula 4")
>>> canvas.setup(.50,.50)
```

aula4.py

```
1   import turtle
2   canvas = turtle.Screen()
3   #cor de fundo canvas
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  #loop canvas
11  canvas.mainloop()
```

A Tartaruga

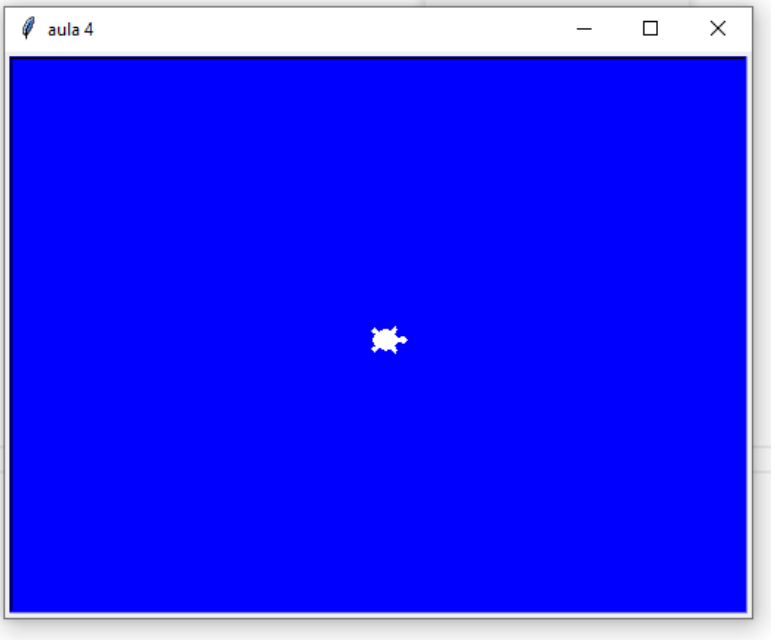
Agora chegou a hora de criar outro objeto dentro de nosso código, que será a tartaruga na qual iremos usar para criarmos formas no canvas, funcionando como uma caneta para desenhar. Assim como o canvas, criamos a caneta com a classe Turtle e armazenamos ela em uma variável que terá o nome “tartaruga”.

Por padrão, o objeto Turtle não começa com o formato de tartaruga, devemos então configurar isso como o método **shape** passando como parâmetro a string “turtle”. Além disso, vamos trocar a cor da tartaruga para branco com o método **color**:

```
>>> tartaruga = turtle.Turtle()
>>> tartaruga.shape("turtle")
>>> tartaruga.color("white")
```

aula4.py

```
1   import turtle
2   canvas = turtle.Screen()
3   #cor de fundo canvas
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  tartaruga = turtle.Turtle()
11  #configura formato para tartaruga
12  tartaruga.shape("turtle")
13  #cor da tartaruga
14  tartaruga.color("white")
15
16  #loop canvas
17  canvas.mainloop()
```



Desenhando Formas

Agora com tela (variável “canvas”) e caneta (variável “tartaruga”), podemos começar a criar formas e desenhos. Com comandos de movimento para a tartaruga, podemos criar essas figuras.

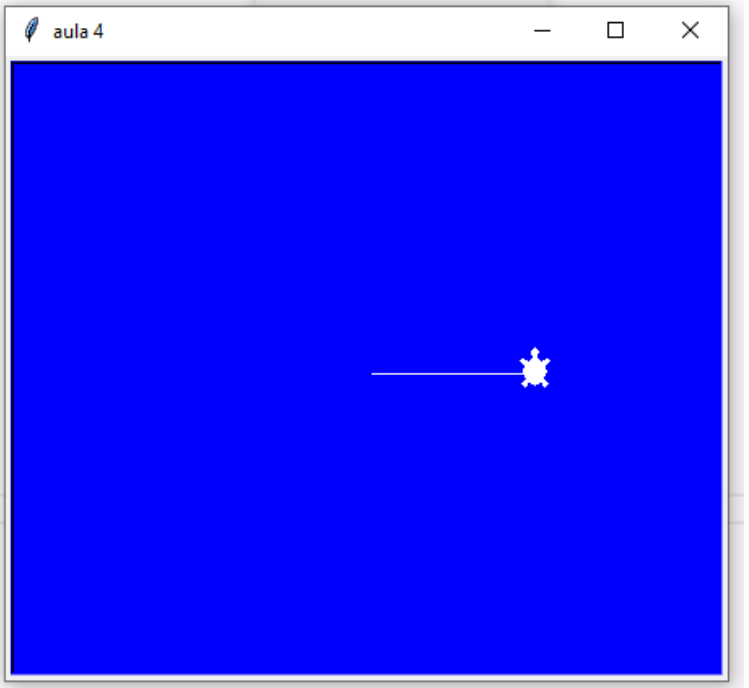
Para a tartaruga se movimentar, usamos os comandos **forward** e **backward** passando a quantidade em pixels (entenderemos pixels nesse contexto como passos) que queremos que o objeto avance.

Para rotacioná-la, usamos os comandos **right** e **left**. Passamos por parâmetros os graus que queremos girar a tartaruga. Por exemplo, para fazer que ela ande 100 casas usamos o forward e para que fique com a cabeça para cima giramos 90 em left:

```
>>> tartaruga.forward(100)
>>> tartaruga.left(90)
```

aula4.py

```
1   import turtle
2   canvas = turtle.Screen()
3   #cor de fundo canvas
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  tartaruga = turtle.Turtle()
11  #cria formato para tartaruga
12  tartaruga.shape("turtle")
13  #cor da tartaruga
14  tartaruga.color("white")
15  #andar para frente
16  tartaruga.forward(100)
17  #girar para cima
18  tartaruga.left(90)
19
20  #loop canvas
21  canvas.mainloop()
```



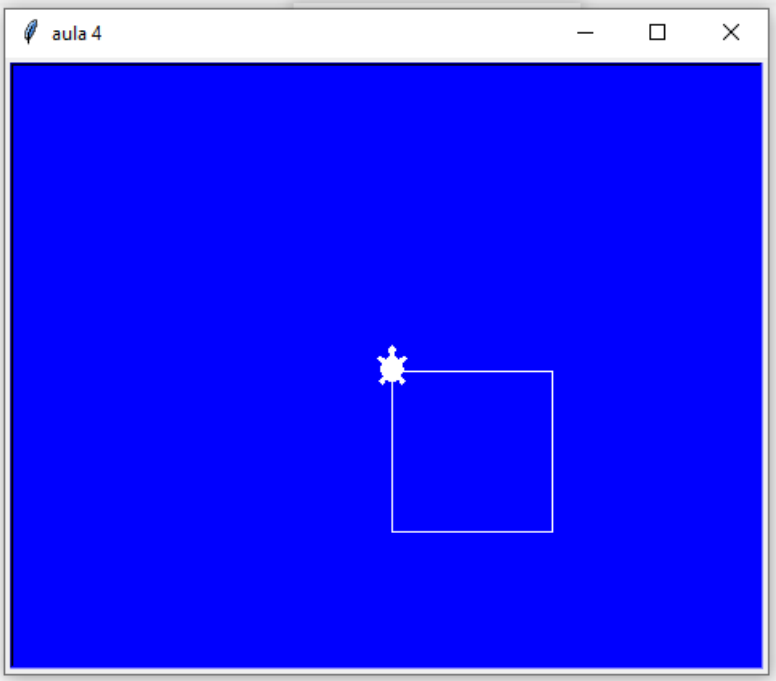
Com esses conhecimentos, podemos já começar a desenhar formas geométricas básicas com nossa tartaruga. Vamos apagar os métodos de movimento para que a tartaruga volte ao centro da tela e, em seguida, escrevemos os comandos para criar as formas.

- **Quadrado**

```
>>> tartaruga.forward(100)
>>> tartaruga.right(90)
>>> tartaruga.forward(100)
>>> tartaruga.right(90)
>>> tartaruga.forward(100)
>>> tartaruga.right(90)
>>> tartaruga.forward(100)
```

aula4.py

```
16  #quadrado
17  tartaruga.forward(100)
18  tartaruga.right(90)
19  tartaruga.forward(100)
20  tartaruga.right(90)
21  tartaruga.forward(100)
22  tartaruga.right(90)
23  tartaruga.forward(100)
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

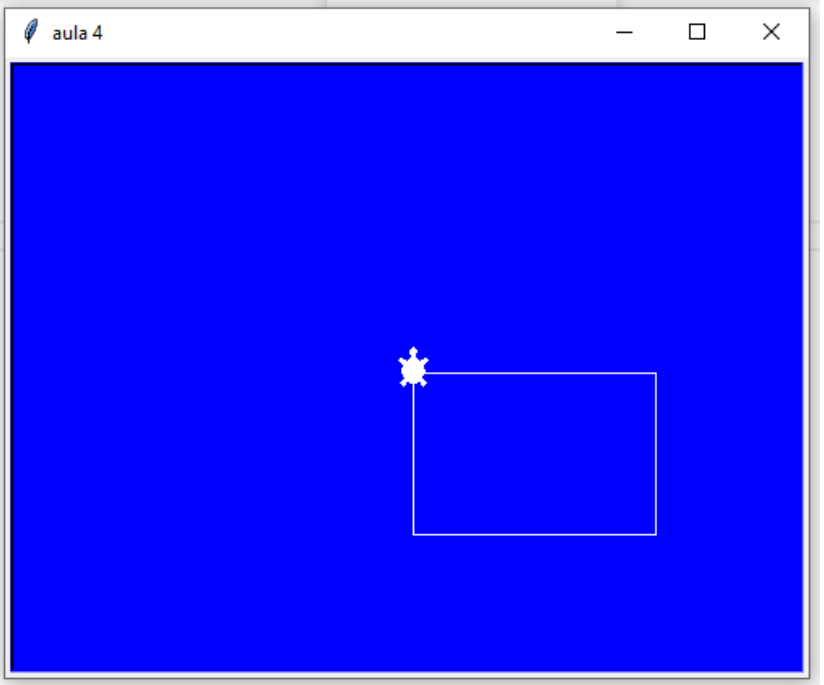


- **Retângulo**

```
>>> tartaruga.forward(150)
>>> tartaruga.right(90)
>>> tartaruga.forward(100)
>>> tartaruga.right(90)
>>> tartaruga.forward(150)
>>> tartaruga.right(90)
>>> tartaruga.forward(100)
```

aula4.py

```
16  #retângulo
17  tartaruga.forward(150)
18  tartaruga.right(90)
19  tartaruga.forward(100)
20  tartaruga.right(90)
21  tartaruga.forward(150)
22  tartaruga.right(90)
23  tartaruga.forward(100)
24
25
26
27
28
29
30
31
32
33
34
35
36
37
```

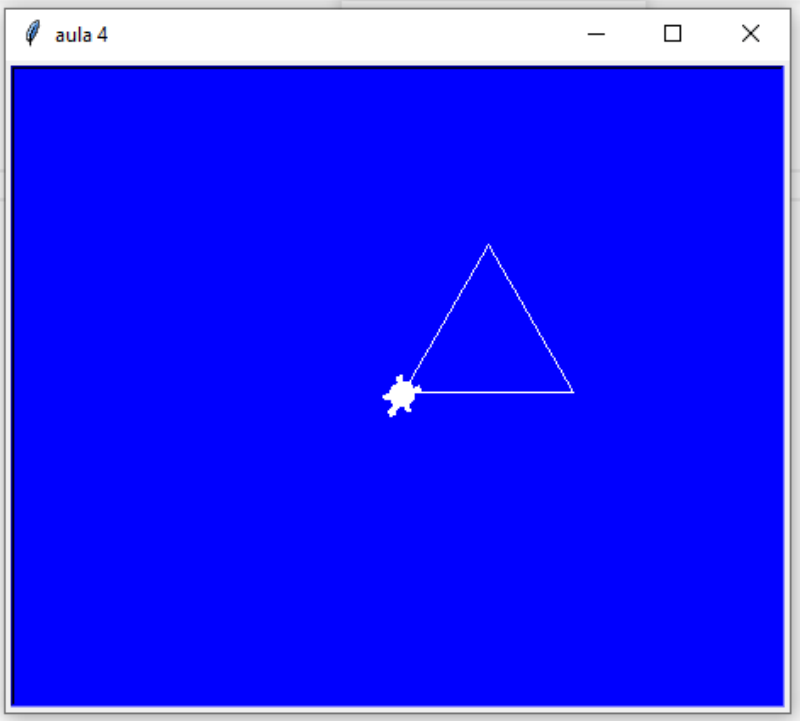


- **Triângulo**

```
>>> tartaruga.forward(100)
>>> tartaruga.left(120)
>>> tartaruga.forward(100)
>>> tartaruga.left(120)
>>> tartaruga.forward(100)
```

aula4.py

```
#triângulo
tartaruga.forward(100)
tartaruga.left(120)
tartaruga.forward(100)
tartaruga.left(120)
tartaruga.forward(100)
```

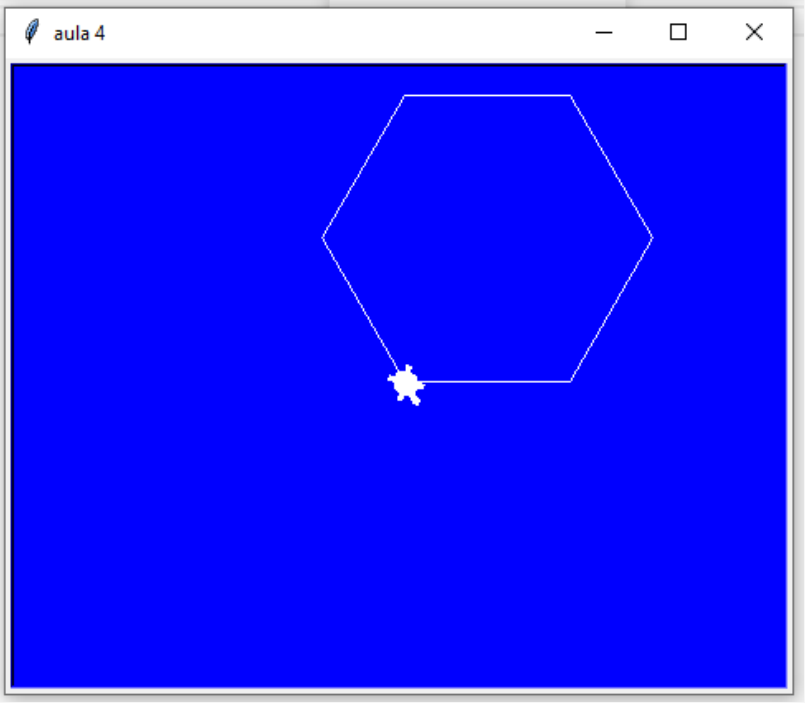


- **Hexágono**

```
>>> tartaruga.forward(100)
>>> tartaruga.left(60)
>>> tartaruga.forward(100)
>>> tartaruga.left(60)
>>> tartaruga.forward(100)
>>> tartaruga.left(60)
>>> tartaruga.forward(100)
>>> tartaruga.left(60)
>>> tartaruga.forward(100)
>>> tartaruga.left(60)
>>> tartaruga.forward(100)
```

aula4.py

```
16  #hexágono
17  tartaruga.forward(100)
18  tartaruga.left(60)
19  tartaruga.forward(100)
20  tartaruga.left(60)
21  tartaruga.forward(100)
22  tartaruga.left(60)
23  tartaruga.forward(100)
24  tartaruga.left(60)
25  tartaruga.forward(100)
26  tartaruga.left(60)
27  tartaruga.forward(100)
28
29
30
31
32
33
34
35
36
37
```



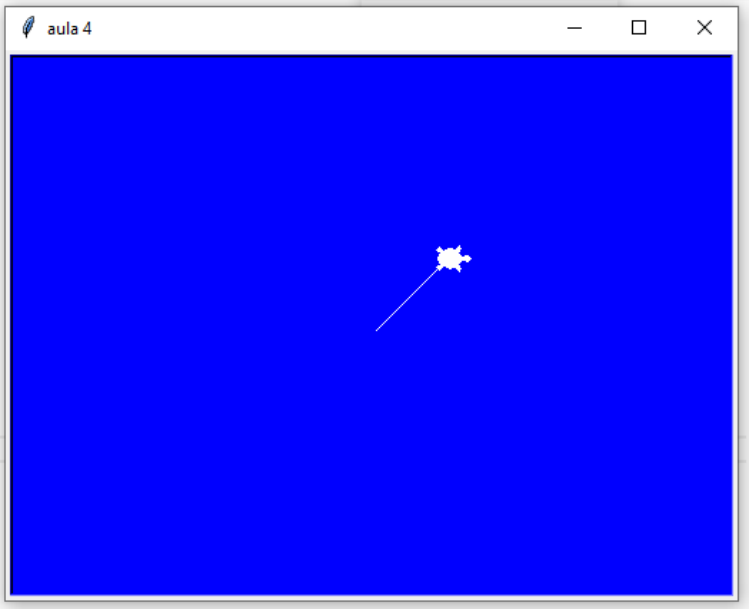
Com coordenadas

Além da movimentação com **forward** e **backward**, podemos movimentar a tartaruga passando coordenadas x e y no método **goto(x, y)**. Devemos imaginar o canvas como um plano cartesiano na qual, usando as coordenadas podemos posicionar o pincel:

```
>>> tartaruga.goto(50,50)
```

aula4.py

```
1   import turtle
2   canvas = turtle.Screen()
3   #cor de fundo canvas
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  tartaruga = turtle.Turtle()
11  #cria forma para tartaruga
12  tartaruga.shape("turtle")
13  #cor da tartaruga
14  tartaruga.color("white")
15  #movimentar por coordenadas (x, y)
16  tartaruga.goto(50,50)
17
18  #loop canvas
19  canvas.mainloop()
```



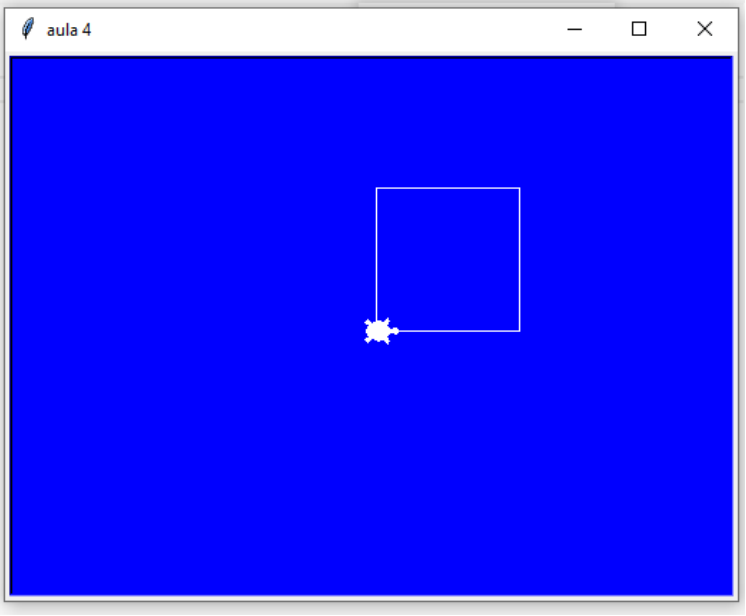
Para criar formas com o método goto, devemos ter em mente que ele sempre usará o ponto do plano cartesiano onde o pincel está. Por exemplo, o código para criar um quadrado ficaria:

```
>>> tartaruga.goto(0,100)
```

```
>>> tartaruga.goto(100,100)
>>> tartaruga.goto(100,0)
>>> tartaruga.goto(0,0)
```

aula4.py

```
15
16  #movimentar por coordenadas (x, y)
17  tartaruga.goto(0,100)
18  tartaruga.goto(100,100)
19  tartaruga.goto(100,0)
20  tartaruga.goto(0,0)
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
```

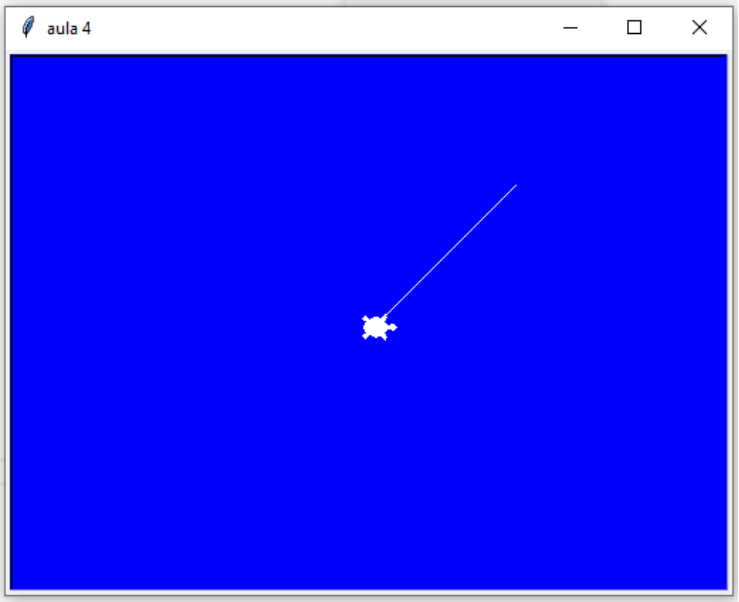


Para voltarmos a origem das coordenadas, usamos o método **home**:

```
>>> tartaruga.home()
```

aula4.py

```
1   import turtle
2   canvas = turtle.Screen()
3   #cor de fundo canvas
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  tartaruga = turtle.Turtle()
11  #cria forma para tartaruga
12  tartaruga.shape("turtle")
13  #cor da tartaruga
14  tartaruga.color("white")
15
16  #movimentar por coordenadas (x, y)
17  rtaruga.goto(100,100)
18  #volta para a coordenada (0,0)
19  tartaruga.home()
20
21  #loop canvas
22  canvas.mainloop()
```



Manipulando rastro pincel

Outros comandos interessantes que podemos promover é a retirada do rastro que o pincel faz ao se movimentar, ou seja, não deixa traço pelo canvas. Por padrão, o pincel está no modo **pendown**, ou seja, como se a caneta estivesse encostada no papel para deixar um rastro de desenho. Para tirarmos o rastro, usamos o método **penup**:

```
>>> tartaruga.pendown()
>>> tartaruga.goto(100,100)
```

```
>>> tartaruga.penup()
>>> tartaruga.goto(200,0)
```

aula4.py

```
4   canvas.bgcolor("blue")
5   #titulo canvas
6   canvas.title("aula 4")
7   #tamanho canvas
8   canvas.setup(.50,.50)
9
10  tartaruga = turtle.Turtle()
11  #cria forma para tartaruga
12  tartaruga.shape("turtle")
13  #cor da tartaruga
14  tartaruga.color("white")
15
16  #caneta no papel
17  tartaruga.pendown()
18  tartaruga.goto(100,100)
19  #caneta flutuando
20  tartaruga.penup()
21  tartaruga.goto(200,0)
22
23  #loop canvas
24  canvas.mainloop()
```