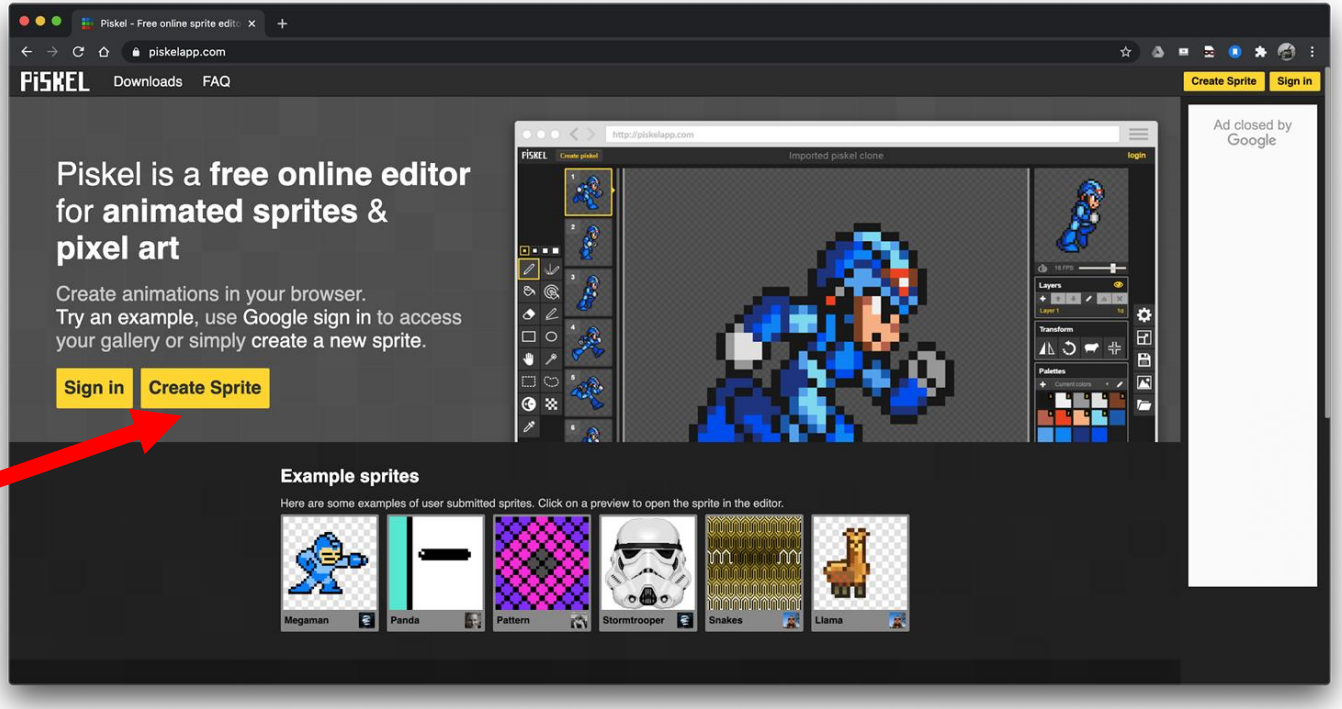
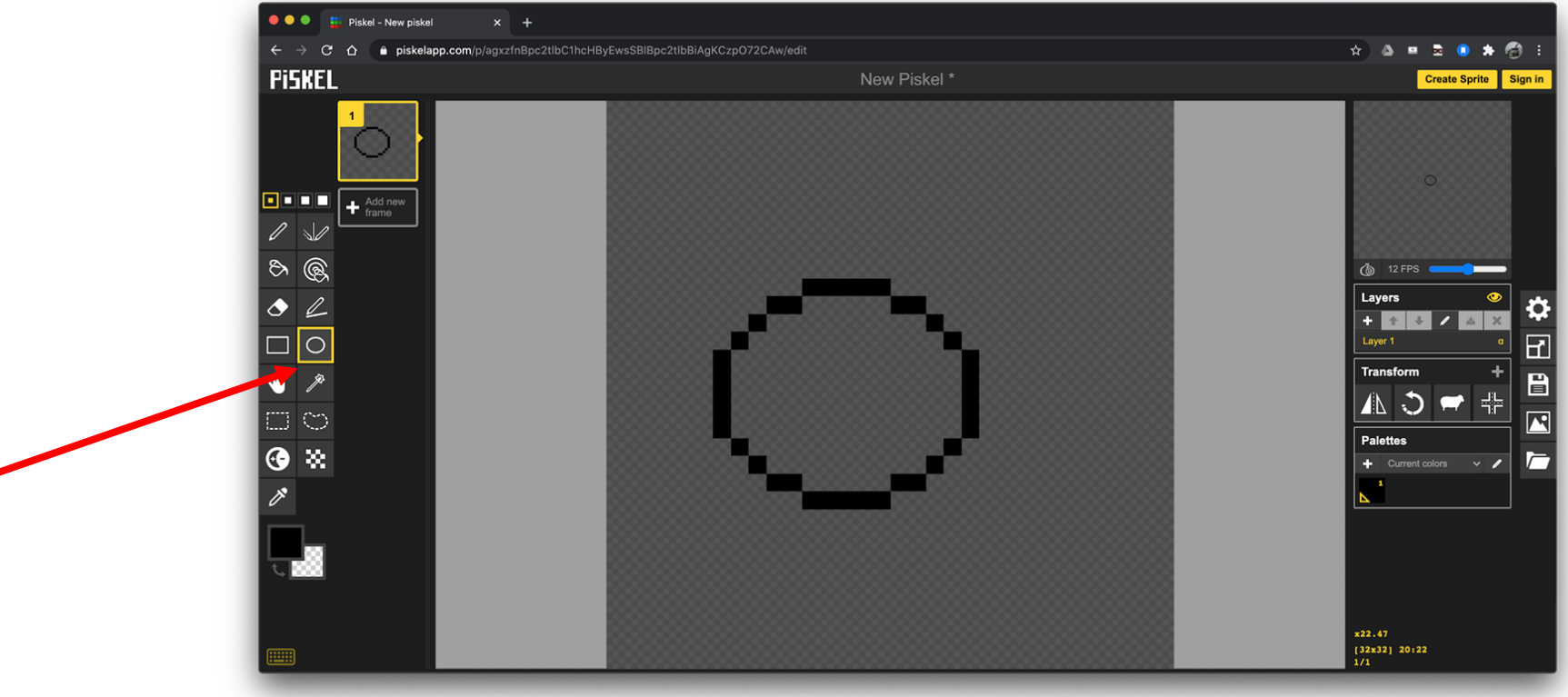
https://www.piskelapp.com/ 사이트

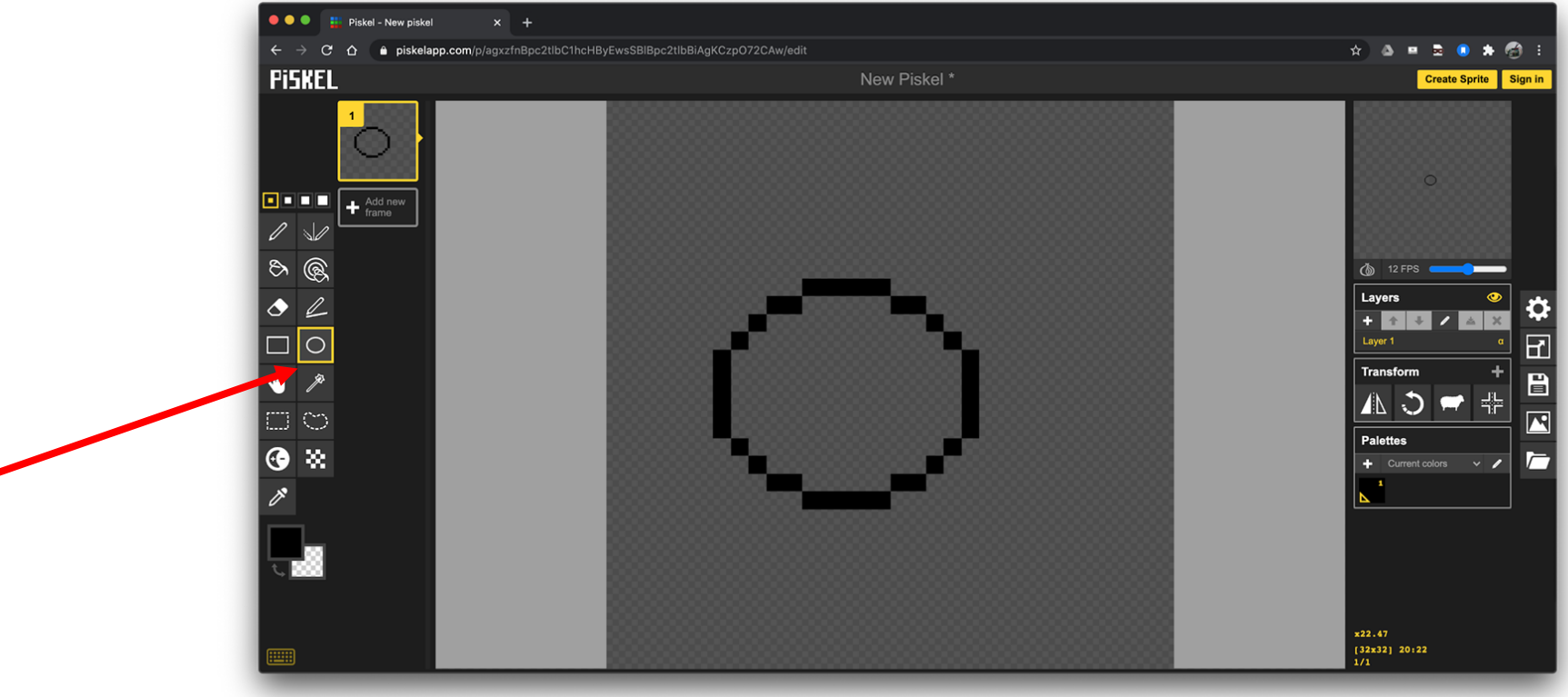
https://www.piskelapp.com/



https://www.piskelapp.com/

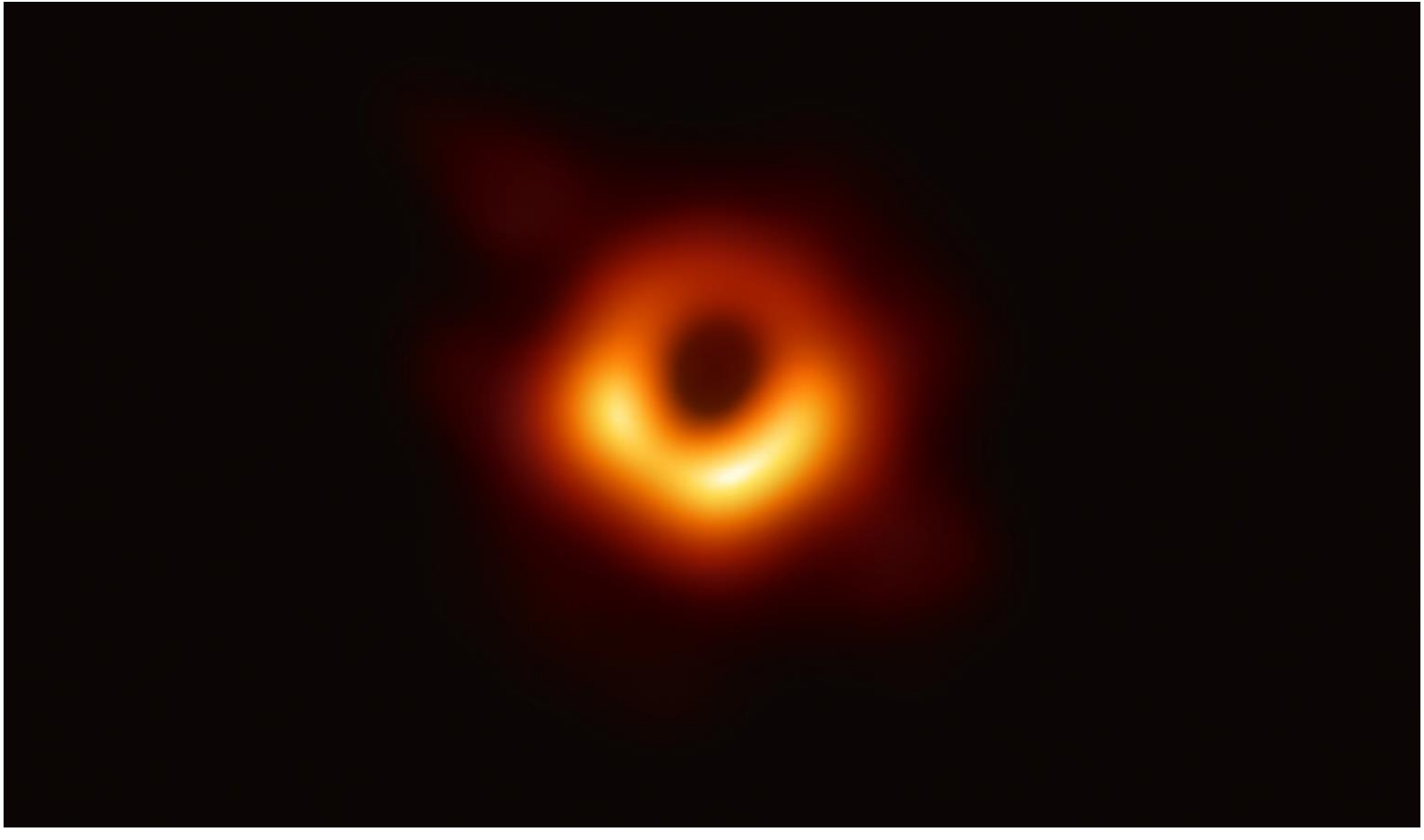


https://www.piskelapp.com/

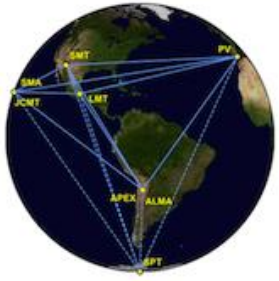



NumPy

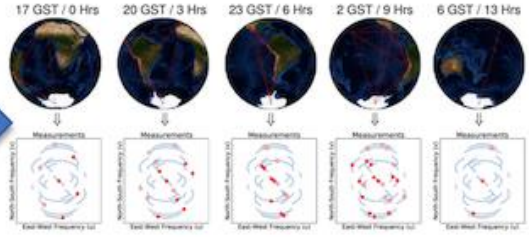
https://numpy.org/case-studies/blackhole-image/



https://numpy.org/case-studies/blackhole-image/



Event Horizon Telescope



Petabytes of data (measurements)

Data correlation, calibration, reduction pipelines

Scientific, Astrophysics & Computing Algorithms

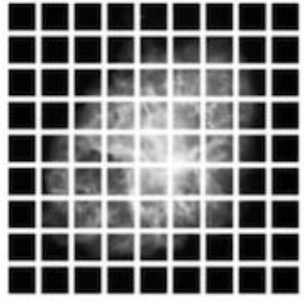


Image analysis

https://github.com/achael/eht-imaging/

Imaging, analysis and simulation software for radio interferometry

M87 – the first image of a black hole

Software: DiFX (Deller et al. 2011), CALC, PolConvert (Martí-Vidal et al. 2016), HOPS (Whitney et al. 2004), CASA (McMullin et al. 2007), AIPS (Greisen 2003), ParselTongue (Kettenis et al. 2006), GNU Parallel (Tange 2011), GILDAS, eht-imaging (Chael et al. 2016, 2018), Numpy (van der Walt et al. 2011), Scipy (Jones et al. 2001), Pandas (McKinney 2010), Astropy (The Astropy Collaboration et al. 2013, 2018), Jupyter (Kluyver et al. 2016), Matplotlib (Hunter 2007).

NumPy was one of the software used!

넘파이 실습

```
#넘파이 라이브러리 임포트 및 별칭 설정
import numpy as np

# 넘파이가 가지고 있는 속성과 함수를 리스트로 저장한다.
x = dir(np)
print(len(x))
print(x[540:550])
```

```
620
['sometrue', 'sort', 'sort_complex', 'source', 'spacing', 'split', 'sqrt', 'square', 'squeeze', 'stack']
```

넘파이 실습

```
#넘파이 라이브러리 임포트 및 별칭 설정
import numpy as np

```

```
#1차원 배열 정의
x = np.array([1, 2, 3])
print(x)          # x 배열 출력
print(x.shape)    # x 배열의 모양 출력
print(x.ndim)     # x 배열의 차원 출력

```

넘파이 실습

```
#넘파이 라이브러리 임포트 및 별칭 설정
import numpy as np

```

```
#1차원 배열 정의
x = np.array([1, 2, 3])
print(x)          # x 배열 출력
print(x.shape)    # x 배열의 모양 출력
print(x.ndim)     # x 배열의 차원 출력

```

```
[1 2 3]
(3,)
1
```

넘파이 실습

```
#2차원 배열 정의
y = np.array([
                [1,2,3],
                [4,5,6]
            ])
print(y)          # y 배열 출력
print(y.shape)    # y 배열의 모양 출력
print(y.ndim)     # y 배열의 차원 출력

```

```
[[1 2 3]
 [4 5 6]]
(2, 3)
2
```

넘파이 실습

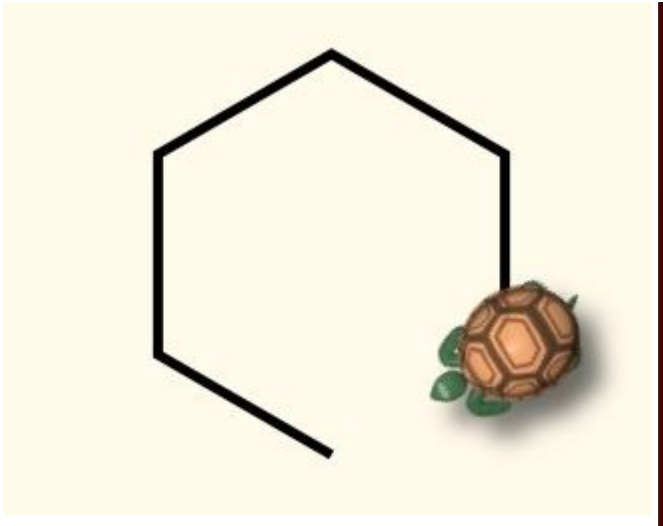
```
#2차원 배열의 합과 차
a = np.array([
                    [1,2,3],
                    [4,5,6]
                ])
b = np.array([
                    [1,2,3],
                    [4,5,6]
                ])
c = a + b
d = a - b
print(a)
print()
print(b)
print()
print(c)
print()
print(d)

```

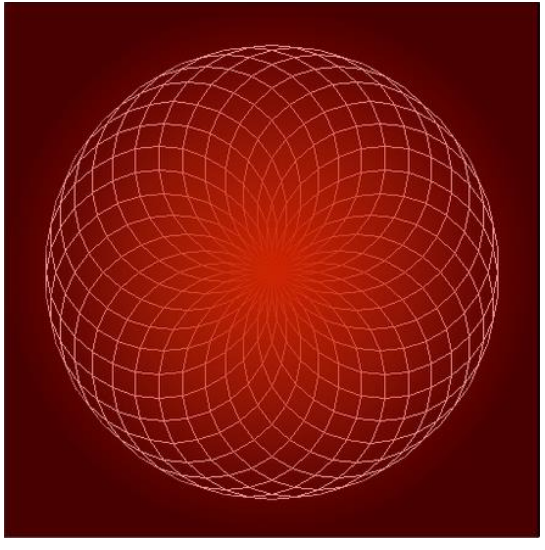
넘파이 실습

```
# 2차원 행렬의 실수배
a = np.array([
                [1,2,3],
                [4,5,6]
            ])
b = 0.5*a
print(a)
print()
print(b)

```

```
[[1 2 3]
 [4 5 6]]

[[0.5 1.  1.5]
 [2.  2.5 3. ]]
```

터틀 그래픽

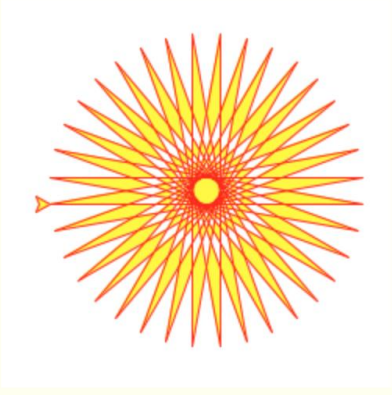




Python Turtle

Turtle star

turtle은 간단한 움직임을 반복하는 프로그램을 사용하여 복잡한 모양을 그릴 수 있습니다.



```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```

turtle.**begin_fill**()

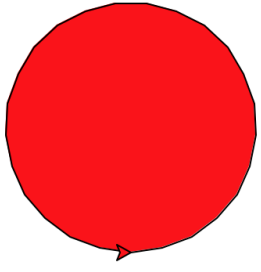
채울 모양을 그리기 직전에 호출됩니다.

turtle.**end_fill**()

`begin_fill()`을 마지막으로 호출한 후 그린 모양을 채웁니다.

스스로 교차하는 다각형이나 여러 도형의 겹치는 영역이 채워지는지는 운영 체제 그래픽, 겹침의 유형 및 겹침의 수에 따라 다릅니다. 예를 들어, 위의 거북이 별은 모두 노란색이거나 일부 흰색 영역이 있을 수 있습니다.

```
>>> turtle.color("black", "red")
>>> turtle.begin_fill()
>>> turtle.circle(80)
>>> turtle.end_fill()
```



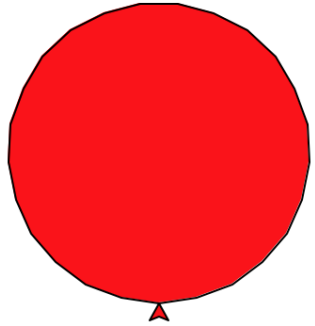
Python Turtle

`turtle.`**`setheading`**(*to_angle*)
`turtle.`**`seth`**(*to_angle*)

매개변수: **to_angle** -- 숫자 (정수나 실수)

거북이의 방향을 *to_angle*로 설정합니다. 다음은 몇 가지 일반적인 도(degree)로 나타낸 방향입니다:

표준 모드	로고 모드
0 - 동	0 - 북
90 - 북	90 - 동
180 - 서	180 - 남
270 - 남	270 - 서



```
>>> turtle.setheading(90)
>>> turtle.heading()
90.0
```

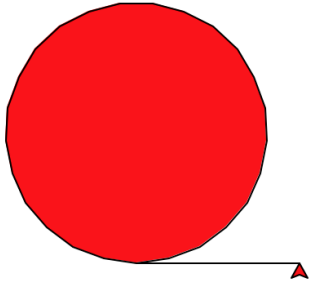
Python Turtle

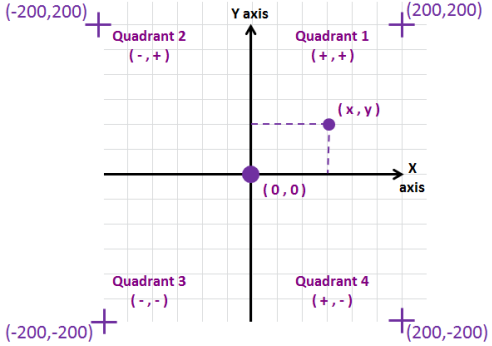
`turtle.`**`goto`**(*x*, *y=None*)
`turtle.`**`setpos`**(*x*, *y=None*)
`turtle.`**`setposition`**(*x*, *y=None*)

매개변수:
- **x** -- 숫자나 숫자의 쌍/벡터
- **y** -- 숫자나 `None`

*y*가 `None`이면, *x*는 좌표 쌍이거나 `Vec2D`여야 합니다 (예를 들어 `pos()`에서 반환된 것과 같은).

거북이를 절대 위치로 움직입니다. 펜이 내려져 있으면, 선을 그립니다. 거북이의 방향을 바꾸지 않습니다.





Python Turtle

```
turtle.pendown()
turtle.pd()
turtle.down()
```

펜을 내립니다 -- 움직일 때 그립니다.

```
turtle.penup()
turtle.pu()
turtle.up()
```

펜을 올립니다 -- 움직일 때 그리지 않습니다.