Импортируем все необходимое:

```
from tkinter import *
from math import cos, sin
```

Также сохраним все константы для корректной работы программы:

```
size = {'width': 600, 'height': 600} # Размеры окна
radius = 200 # Радиус окружности
ball_radius = 20 # Радиус шарика
omega = 10  # Скорость вращения

_t = 0
```

`_t` - служебная константа

Далее создадим белое рабочее окно сохранив нужную длину и ширину

```
root = Tk()
c = Canvas(root, bg="white", **size)
c.pack()
```

`c.pack()` - необходим для корректного обновления объектов в родительском окне

Также сохраним служебный словарь со всеми размерами, деленными на два

```
size_div = {key: value // 2 for key, value in size.items()}
_x0, _y0 = size_div.values()
```

Создадим саму «рамку» и шарик, который будет ездить по ней:

```
circle = c.create_oval(
    size_div['width'] - radius,
    size_div['height'] + radius,
    size_div['width'] + radius,
    size_div['height'] - radius,
    fill='white',
    outline='black',
    width=5
)

ball = c.create_oval(
    size_div['width'] - ball_radius,
    size_div['height'] + ball_radius,
    size_div['width'] + ball_radius,
    size_div['height'] - ball_radius,
    fill='orange',
    width=10,
)
```

И наконец-то сама логика движения:

```
def motion():
    global _t, _x0, _y0
    x = radius * cos(omega * _t) + size_div['width']
    y = radius * sin(omega * _t) + size_div['height']

    dx = x - _x0
    dy = y - _y0

    _x0 = x
```

```
    _y0 = y
    _t += 0.001

    c.move(ball, dx, dy)
    root.after(10, motion)
motion()

root.mainloop()
```

Циклически мы обновляем наши координаты и получаем новые, но в `Tkinter` метод `move` принимает изменение координаты, поэтому важно сохранить и старые значения, а в дальнейшем находить дельту. После этого вызываем функцию каждые 10 миллисекунд

По итогу получаем Рис. 1:

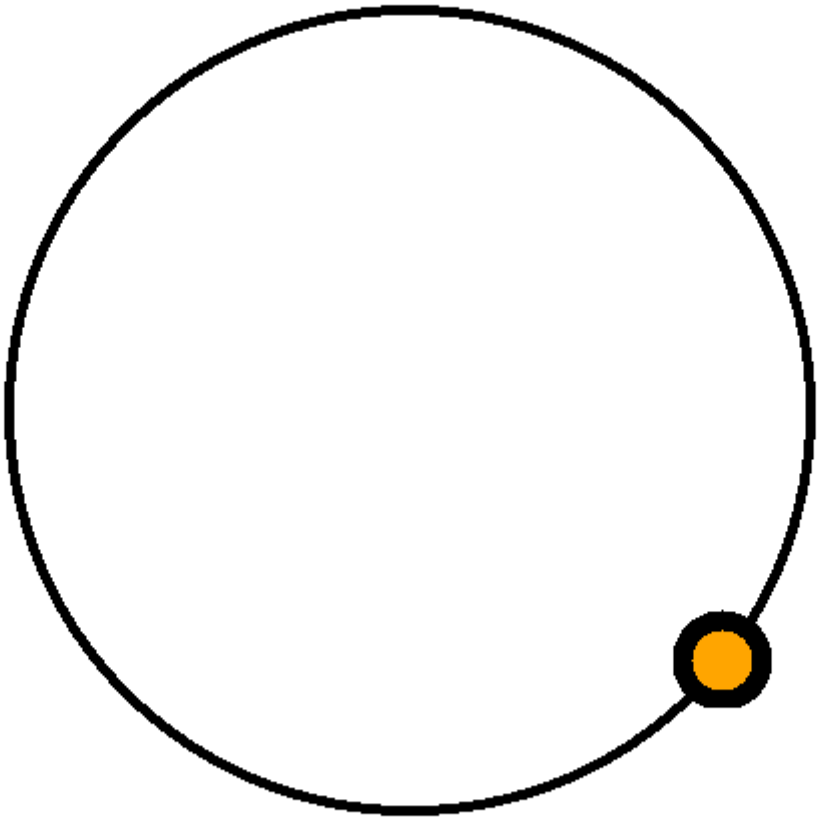

Рис. 1. Работа программы