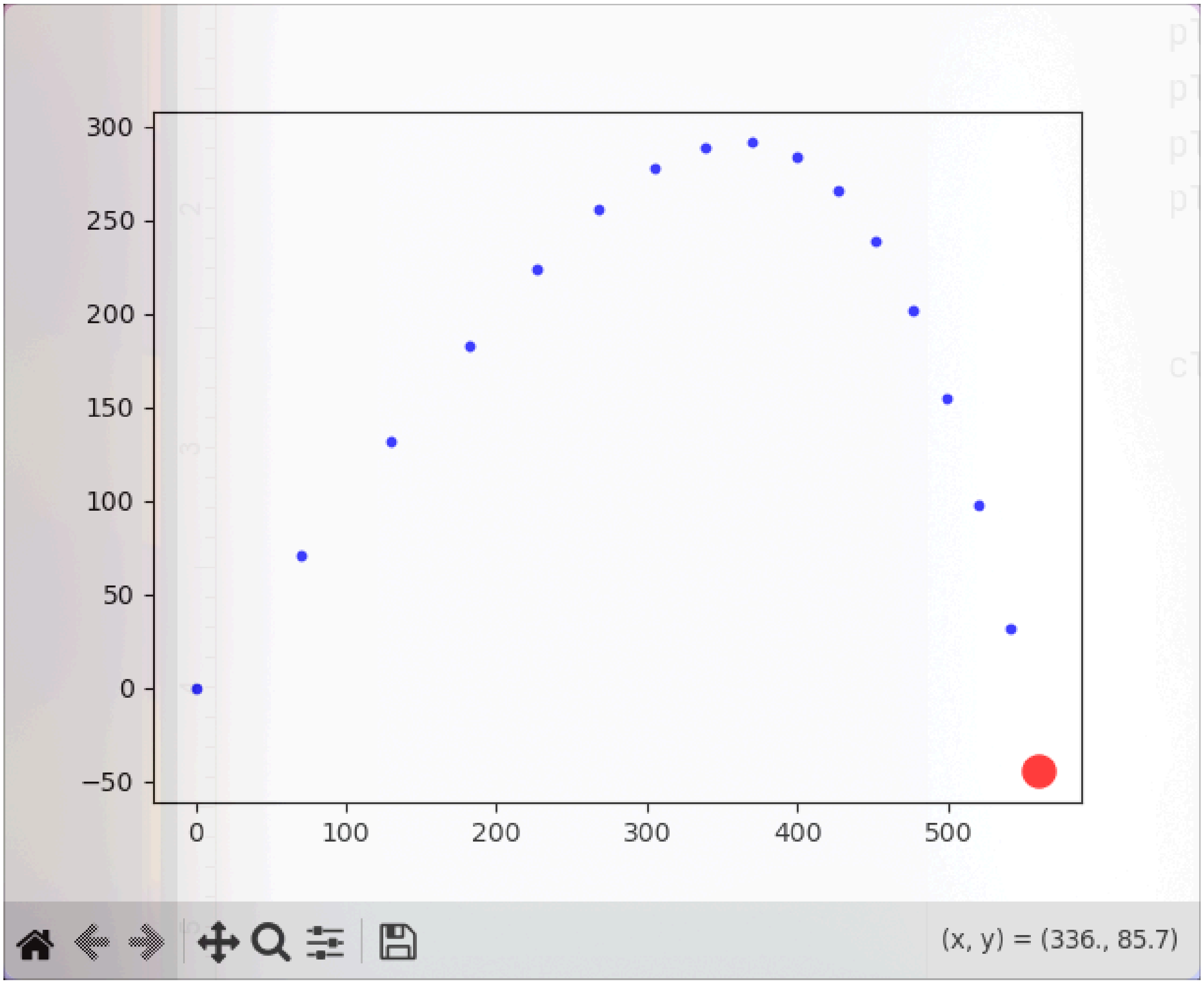

```
from matplotlib import pyplot as plt
import math

from numpy import double

plt.gca().set_aspect("equal", adjustable="box")
plt.xlabel("distance (m)")
plt.ylabel("altitude (m)")
plt.grid(True)


class Integrator:
      sum = 0

      def calc(self, x):
```

```
    self.sum
    self.sum = self.sum + x

    return self.sum

    def reset(self):
    self.sum = 0


def toComponents(direction, magnitude):
    x_component = magnitude * math.cos(math.radians(direction))
    y_component = magnitude * math.sin(math.radians(direction))

    return x_component, y_component


class Drag:
    variables = {"density": 0, "area": 0.0, "Cd": 0.0, "mass": 0.0}

    def values(self, **kwargs):
    dragCoefficients = {
        "sphere": 0.47,
        "half-sphere": 0.42,
        "cone": 0.50,
        "cube": 1.05,
    }
    for index, value in kwargs.items():
        self.variables[index] = value
    self.variables["Cd"] =
dragCoefficients.get(str(kwargs.get("drag_coefficient")))

    def determineDrag(self, velocity):
    return (
        (
            self.variables["Cd"]
            * (
            self.variables["density"]
            * ((velocity**2) * self.variables["area"])
```

```
                / 2
                )
            )
            / self.variables["mass"]
            * -1
        )


XDistanceIntegrator = Integrator()
XVeloIntegrator = Integrator()
YDistanceIntegrator = Integrator()
YVeloIntegrator = Integrator()
Dragon = Drag()


def ballySticksTime(**kwargs):
    XDistanceIntegrator.reset()
    XVeloIntegrator.reset()
    YDistanceIntegrator.reset()
    YVeloIntegrator.reset()

    variables = {
    "x_component": 0,
    "y_component": 0,
    "xDistance": 0,
    "xVelo0": 0,
    "xVelo": 0,
    "xAccel": 0,
    "yDistance": 0,
    "yVelo0": 0,
    "yVelo": 0,
    "yAccel": -9.8,
    "drag": False,
    "pause": 0.05,
    "color": "b",
    }

    for index, value in kwargs.items():
```

```
    variables[index] = value

    if (kwargs.get("angle") is not None) & (kwargs.get("magnitude")
is not None):
    variables["x_component"], variables["y_component"] =
toComponents(
        kwargs["angle"], kwargs["magnitude"]
    )

    if kwargs.get("xVelo") is None:
    variables["xVelo"] = variables["x_component"]
    if kwargs.get("xVelo0") is None:
    variables["xVelo0"] = variables["xVelo"]
    if kwargs.get("yVelo") is None:
    variables["yVelo"] = variables["y_component"]
    if kwargs.get("yVelo0") is None:
    variables["yVelo0"] = variables["yVelo"]

    if variables["drag"] == True:
    Dragon.values(drag_coefficient="sphere", density=1.225,
area=0.785, mass=100)

    while variables["yDistance"] >= 0:
    plt.scatter(
        variables["xDistance"], variables["yDistance"], s=10,
c=variables["color"]
    )
    plt.pause(variables["pause"])
    if variables["drag"] == True:
        variables["xAccel"] =
Dragon.determineDrag(variables["xVelo"])

    variables["xDistance"] =
XDistanceIntegrator.calc(variables["xVelo"])
    variables["xVelo"] = (
        XVeloIntegrator.calc(variables["xAccel"]) +
variables["xVelo0"]
    )
```

```
      variables["yDistance"] =
YDistanceIntegrator.calc(variables["yVelo"])
      variables["yVelo"] = (
            YVeloIntegrator.calc(variables["yAccel"]) +
variables["yVelo0"]
      )

      plt.scatter(variables["xDistance"], variables["yDistance"],
s=150, c="r")


# program 1
print("\nprogram 1\n")

ballySticksTime(
      angle=((double)(input("angle of elevation (degrees): "))),
      magnitude=((double)(input("magnitude: "))),
)
plt.show()

print("\nprogram 2\n")
colorList = ["red", "orange", "yellow", "green", "blue", "purple"]

for i in range(0, 91, 15):
      ballySticksTime(
      angle=i,
      magnitude=100,
      pause=0.01,
      color=colorList[int(i / 15) % len(colorList)],
      )
plt.show()


# program 3
print("\nprogram 3\n")

ballySticksTime(
    angle=((double)(input("angle of elevation (degrees): "))),
    magnitude=((double)(input("magnitude: "))),
    xAccel=((double)(input("x acceleration: "))),
)
plt.show()


# part 3
print("\npart 3")

ballySticksTime(
    angle=((double)(input("angle of elevation (degrees): "))),
    magnitude=((double)(input("magnitude: "))),
    drag=True,
)
plt.show()

# Dragon.values(drag_coefficient="sphere", density=1.225, area=1)
# print(Dragon.variables)
```