Week-6:

1. a. Write a function called draw_rectangle that takes a Canvas and a Rectangle as arguments and draws a representation of the Rectangle on the Canvas.

Note: To install swampy in command prompt

```
#pip install swampy
from swampy.World import *
class Canvas(object):
    """Represents a canvas.
attributes: width, height, background color"""
a_canvas = Canvas()
a_canvas.width = 500
a_canvas.height = 500
class Rectangle(object):
    """Represents a rectangle."""
box = Rectangle()
box.color = 'orange'
box.bbox = [[-100, -60],[100, 60]]
def draw_rectangle(canvas, rectangle):
    drawn_canvas = world.ca(canvas.width, canvas.height)
    drawn_canvas.rectangle(rectangle.bbox, fill=rectangle.color)

world = World()
draw_rectangle(a_canvas,box)
world.mainloop()
```

1. b. Add an attribute named color to your Rectangle objects and modify draw_rectangle so that ituses the color attribute as the fill color.

```
from swampy.World import *
class Canvas(object):
    """Represents a canvas.
attributes: width, height, background color"""
a_canvas = Canvas()
a_canvas.width = 500
a_canvas.height = 500
class Rectangle(object):
    """Represents a rectangle."""
box = Rectangle()
box.color = 'orange'
box.bbox = [[-100, -60],[100, 60]]
def draw_rectangle(canvas, rectangle):
    drawn_canvas = world.ca(canvas.width, canvas.height)
    drawn_canvas.rectangle(rectangle.bbox, fill=rectangle.color)

world = World()
draw_rectangle(a_canvas,box)
world.mainloop()
```

1. c. Write a function called draw_point that takes a Canvas and a Point as arguments and draws arepresentation of the Point on the Canvas.

```
from swampy.World import *
class Canvas(object):
    """Represents a canvas.
attributes: width, height, background color"""
a_canvas = Canvas()
a_canvas.width = 500
a_canvas.height = 500
class Point(object):
    "represents a point in 2-D space"
p = Point()
p.x = 60
p.y = 15
def draw_point(canvas, point):
    bbox = [[point.x, point.y], [point.x, point.y]]
    drawn_canvas = world.ca(canvas.width, canvas.height)
    drawn_canvas.rectangle(bbox, fill="black")

world = World()
draw_point(a_canvas,p)
world.mainloop()
```

1. d. Define a new class called Circle with appropriate attributes and instantiate a few Circle objects.Write a function called draw_circle that draws circles on the canvas.

```
from swampy.World import *
class Canvas(object):
    """Represents a canvas.
attributes: width, height, background color"""
a_canvas = Canvas()
a_canvas.width = 500
a_canvas.height = 500

class Point(object):
    "represents a point in 2-D space"
p = Point()
p.x = 60
p.y = 15

class Circle(object):
    """Represents a circle.
attributes: center point, radius"""
c = Circle()
c.radius = 50
c.center = Point()
c.center.x = 20
c.center.y = 20
def draw_circle(canvas, circle):
    drawn_canvas = world.ca(canvas.width, canvas.height)
    drawn_canvas.circle([circle.center.x, circle.center.y], circle.radius)

world = World()
draw_circle(a_canvas,c)
world.mainloop()
```

2. Write a Python program to demonstrate the usage of Method Resolution Order (MRO) in multiplelevels of Inheritances.

```
#Method Resolution Order(MRO)
#A program to understand the order of execution of methods
#in several base classes according to MRO
#Multiple inheritance with several classes
class A(object):
    def method(self):
        print("A class method")
        super().method()
class B(object):
    def method(self):
        print("B class method")
        super().method()
class C(object):
    def method(self):
        print("C class method")
class X(A,B):
    def method(self):
        print("X class method")
        super().method()
class Y(B,C):
    def method(self):
        print("Y class method")
        super().method()
class P(X,Y,C):
    def method(self):
        print("P class method")
        super().method()
pobj=P()
pobj.method()
```

3. Write a python code to read a phone number and email-id from the user and validate it forcorrectness.

```
import re
#Define  a function for validate mobile number
def valid_number(number):

    # 1) Begins with 0 or 91
    # 2) Then contains 6,7 or 8 or 9.
    # 3) Then contains 9 digits
    Pattern = re.compile("(0|91)?[6-9][0-9]{9}")
    return Pattern.match(number)

#Define  a function for validate email address
def validate_email(email):
    # regular expression for an email address
    pattern = re.compile(r"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$")
    # check if the pattern matches the email address
    match = pattern.search(email)
    return match


#main method
if __name__ == "__main__":

    number = input("Enter your mobile number: ")
    if (valid_number(number)):
        print ("Valid Number")
    else :
        print ("Invalid Number")
    email = input("Enter email address: ")
    match = validate_email(email)
    if match:
        print("Valid email address.")
    else:
        print("Invalid email address.")
```