Daily Assignment 23

- Start from uploaded *23-interactive-linear.py*, modify this program to draw a **Hermite curve** instead of a line
- Code for dragging two end points are already implemented in the code
- You have to add another two draggable points **pv0**, **pv1** to define derivatives of two end points
  - **v0**=**pv0**-**p0**
  - **v1**=**pv1**-**p1**

```
# initial values
p0 = np.array([200.,200.])
p1 = np.array([400.,400.])
pv0 = np.array([300.,350.])
pv1 = np.array([500.,550.])
```

- Render points **pv0**, **pv1** and lines from **p0** to **pv0**, **p1** to **pv1** in green
- Hint: using matrix form of Hermite curve would be easier!

```
p0 = np.array([200.,200.])
p1 = np.array([400.,400.])
pv0 = np.array([300.,350.])
pv1 = np.array([500.,550.])
gEditingPoint = ''

def render():
    global p0, p1, pv0, pv1

glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)
    glEnable(GL_DEPTH_TEST)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    glOrtho(0,640, 0,640, -1, 1)

    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

    # draw hermite curve with line segments
    glColor3ub(255, 255, 255)
    glBegin(GL_LINE_STRIP)
    for t in np.arange(0,1,.01):
        T = np.array([t**3, t**2, t, 1])
        M = np.array([[2, -2, 1, 1],
                      [-3, 3, -2, -1],
                      [0, 0, 1, 0],
                      [1, 0, 0, 0]], float)
        P = np.row_stack((p0, p1, pv0-p0, pv1-p1))
        p = T @ M @ P
        glVertex2fv(p)
    glEnd()
```

```
    # draw two end points p0 and p1
    glPointSize(20.)
    glBegin(GL_POINTS)
    glVertex2fv(p0)
    glVertex2fv(p1)
    glEnd()

    # draw vectors from p0 to pv0 and
from p1 to pv1
    glColor3ub(0, 255, 0)
    glBegin(GL_LINES)
    glVertex2fv(p0)
    glVertex2fv(pv0)
    glVertex2fv(p1)
    glVertex2fv(pv1)
    glEnd()

    # draw two points for derivatives
pv0 and pv1
    glBegin(GL_POINTS)
    glVertex2fv(pv0)
    glVertex2fv(pv1)
    glEnd()
```

```
def button_callback(window, button, action, mod):
    global p0, p1, pv0, pv1, gEditingPoint
    if button==glfw.MOUSE_BUTTON_LEFT:
        x, y = glfw.get_cursor_pos(window)
        y = 640 - y
        if action==glfw.PRESS:
            if np.abs(x-p0[0])<10 and np.abs(y-p0[1])<10:
                gEditingPoint = 'p0'
            elif np.abs(x-p1[0])<10 and np.abs(y-p1[1])<10:
                gEditingPoint = 'p1'
            elif np.abs(x-pv0[0])<10 and np.abs(y-pv0[1])<10:
                gEditingPoint = 'pv0'
            elif np.abs(x-pv1[0])<10 and np.abs(y-pv1[1])<10:
                gEditingPoint = 'pv1'
        elif action==glfw.RELEASE:
            gEditingPoint = ''

def cursor_callback(window, xpos, ypos):
    global p0, p1, pv0, pv1, gEditingPoint
    ypos = 640 - ypos
    if gEditingPoint=='p0':
        p0[0]=xpos; p0[1]=ypos
    elif gEditingPoint=='p1':
        p1[0]=xpos; p1[1]=ypos
    elif gEditingPoint=='pv0':
        pv0[0]=xpos; pv0[1]=ypos
    elif gEditingPoint=='pv1':
        pv1[0]=xpos; pv1[1]=ypos
```