ME2-HCPT End of Term Test

CID number:	0								
	1st	2nd	3rd	4th	5th	6th	7th	8th	

Comment appropriately all your scripts. Comments are marked too! [3]

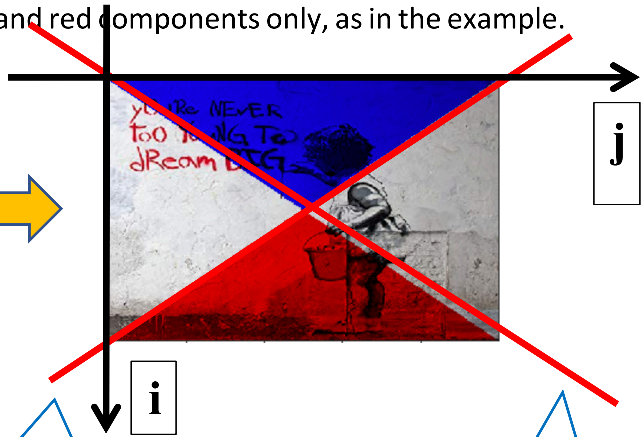
STATE YOUR CID into a comment at the beginning of every file

Task A [11]

The file *Banksy.jpg* contains an image. Write a script (name it *ExA*) to:

1. Read in the file and plot the image.

2. Render the two triangular parts with blue and red components only, as in the example.



3. Save the final image in the file *Tactics.jpg.*

i = -m*j + Pshape[0]

i = m*j

I read the image

```
P = pl.imread('Banksy.jpg')
# get size of array
Pshape = P.shape
# print image
print(Pshape)
pl.imshow(P)
```

Ge the dimension of the image

```
# establish the ratio Ny/Nx pixels
m = (Pshape[0]-1) / (Pshape[1]-1)
```

Get the ratio vertical/horizontal

```
for i in range(0,Pshape[0]):
    # scroll image by column
    for j in range(0,Pshape[1]):
        # examine the current pixel
        if i <= m*j and i <= -m*j+Pshape[0]:
```

Upper triangle

```
            # this pixel is in the upper triangle
            # annihilate R and G components
            R[i,j,0] = 0
            R[i,j,1] = 0
        if i >= m*j and i >= -m*j+Pshape[0]:
```

Lower triangle

```
            # this pixel is in the lower triangle
            # annihilate G and B components
            R[i,j,1] = 0
            R[i,j,2] = 0
```

Task B [14]

Consider the set of points: x_n = *[1,2,3,4,5,6,7,8]* and y_n = *[*1^{st}*,*2^{nd}*,*3^{rd}*,*4^{th}*,*5^{th}*,*6^{th}*,*7^{th}*,*8^{th}*]* digits of your CID.

1. Write a script (name it *ExB*) to interpolate these points in the range *x = [1 : 8]* with interval dx = *0.1*, by using Lagrangian polynomials. (Write all the computation into one single code, with no functions).

2. Plot the interpolating points and the interpolated curve on the same graph.

Setting the interpolating points:

```
# set of interpolating points (available)
xn = np.array([1,2,3,4,5,6,7,8])
yn = np.array(b)
# number of nodes
N = len(yn)
```

Setting the domain of interpolation:

```
# domain of interpolation
dx = 0.1
x = np.arange(1,8+dx,dx)
y = []
```

Interpolate at every point of the domain:

```
# establish the order of the interpolating polynomial, N-1
n = N - 1
# interpolate for all the values of x in the interpolating range
for xp in x:
   # evaluate pn(xp)
   yp = 0
   # use Langrangian polynomials up to order n, included
   for j in range(0,n+1):
      # compute Lagrangian polynomial of order n
      Lj = 1
      # range of k is from 0 to n, included
      for k in range(0,n+1):
         # exclude the case k == j
         if k != j:
            Lj *= (xp-xn[k]) / (xn[j]-xn[k])


      yp += yn[j] * Lj
   # add the curren value of yp to the list of y
   y += [yp]
```

For this xp find yp = p(xp)

Compute the j term of the polynomial

Compute Lagrangian of order j

Obtain j term of the polynomial

Task C [15]

1. Solve numerically the ordinary differential equation:

$$2x\frac{d^2y}{dx^2} + 10x^2\frac{dy}{dx} + (2x^2 + 14x)\sin(x) = 0$$

with the initial conditions y(0) = 3rd and $\left.\frac{dy}{dx}\right|_{x=0} = 5^{th}$ (where 3rd and 5th are the digits of your CID).
Write a script (name it *ExC*) to compute and plot the numerical solution y(x) in the range x = [0 : 15] with step dx = 0.02. Use the explicit Forward Euler method.

Eqn. can be rewritten as:

$$\frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x+7)\sin(x) = 0$$

This is an initial condition problem and not boundary values

It is like the pendulum in Task D, Tutorial 7.
We split it into two first order ODEs:

$$\begin{cases} w = \dfrac{dy}{dt} \\ \dfrac{dw}{dt} = \left(\dfrac{d^2y}{dt^2}\right) = -x\dfrac{dy}{dt} - (x+7)sinx = -xw - (x+7)sinx \end{cases}$$

Then we can just use the subroutine FwEulerTwo

```
def func1(x,y):
  f = y[1]
  return f
def func2(x,y):
  f = -5*x*y[1] - (x+7)*np.sin(x)
  return f
```

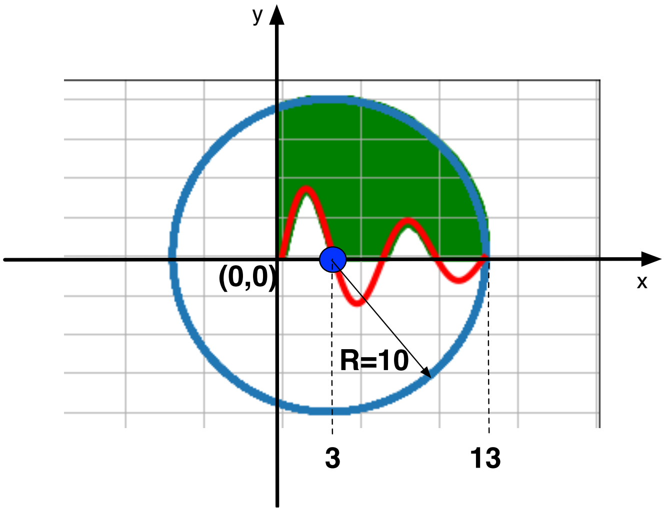
```
# set the initial conditions
Y0 = np.ndarray(2)
Y0[0] = b[2]  # initial y
Y0[1] = b[4]  # initial dy/dx

(x,Y) = FwEulerTwo(Y0,0,15,0.02)
```

Where (copied and pasted from past, no need to rewrite it):

```
def FwEulerTwo(Y0,t0,tend,h):
   # compose nodal times
   t = np.arange(t0,tend+h,h)
   # determine the number of time steps
   N = len(t)
   # allocate output array
   Y = np.ndarray((2,N))
   # initialise the solution
   t[0] = t0
   Y[0,0] = Y0[0]
   Y[1,0] = Y0[1]
   # compute the solution incrementally at subsequent time steps
   for n in range(1,N):
      Y[0,n] = Y[0,n-1] + func1(t[n-1],Y[:,n-1]) * h
      Y[1,n] = Y[1,n-1] + func2(t[n-1],Y[:,n-1]) * h
   return (t,Y)
```

Task D **[17]**

1. Write a script (name it *ExD*) to calculate numerically the area of the green shadowed shape in the figure, in the range *x = [0 : 13]* with interval *dx = 0.01*.



The red function inside the circle is:

$$y = 5sin\left(\frac{2\pi}{13}nx\right)e^{-x/10}$$

Determine the area for all the values of *n* in the range *n = [1st,2nd,3rd,....8th]* digits of you CID.
Deploy the trapezoidal method.

2. Plot in a graph the values of the computed areas for each value of n.

Generate domain of integration and eqn for circle:

```
R = 10
dx = 0.01
x = np.arange(0,13+dx,dx)
yup = np.sqrt(R**2-(x-3)**2)
```

Area of circle:

```
Sup = trapz(yup) * dx
```

Loop for every digit of the CID:

```
S = []
Rn = [0,1,2,3,4,5,6,7,8,9]
for n in Rn:
   yt = 5*np.sin(2*np.pi/13*n*x)*np.exp(-x/10)
   ydown = np.zeros(len(x))
   ydown[yt>=0] = yt[yt>=0]
   Sdown = trapz(ydown) * dx

   S += [Sup-Sdown
```

Eqn for red line

Select only positive values, and set others to zero

Area of red line rectified.

Area of green shadow