```python
"""
Author   : Abraham Flores
File     : HW1.py
Language : Python 3.5
Created  : 2/7/2018
Edited   : 2/7/2018

San Digeo State University
MTH 693b : Computational Partial Differiential Equations

Strikwerda 1.3.1 : Finite Difference Schemes

One Way Wave Equation
    U_t + U_x = 0
    x = [-1,3]
    t = [0,2.4]

    U_0(x) = (cos(piX))^2 for |x| < 1/2 : Else U_0(x) = 0
    U(t,-1) = 0
    U(t,3) = U(t,3-h)
    h = 1/10 , 1/20 , 1/40

    A. Foward Time Backwards Space : lambda = .8
    B. Foward Time Central Space   : lambda = .8
    C. Lax-Friedrichs              : lambda = .8 and 1.6
    D  Leap Frog                   : lambda = .8

"""
import os,glob
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns

#Generates intial value function
def intial_foo(x):
    if abs(x) <= .5:
        return np.cos(np.pi*x)**2
    return 0
def plot(x,U,time,title,fileLoc):
    sns.set(font_scale = 2.5)
    plt.rcParams['font.size'] = 40
    plt.plot(x,U,linewidth=2.0,label="t = "+ str(round(time,3)))

    plt.axis([-1, 3, 0, 5])
    plt.xlabel('x (Spatial)')
    plt.ylabel('U(x,t)')
    plt.title(title)
    plt.legend()
    plt.savefig(fileLoc+".png")
    plt.close()

def makeGif(gifName):
    os.chdir('Figures')
    #Create txt file for gif command
    fileList = glob.glob('*.png') #star grabs everything,
    fileList.sort()
    #writes txt file
    file = open('FileList.txt', 'w')
    for item in fileList:
        file.write("%s\n" % item)
    file.close()

    os.system('convert -delay 10 @FileList.txt ' + gifName + '.gif')
    os.system('del FileList.txt')
    os.system('del *.png')
    os.chdir('..')

#Forward Time Backwards Space Scheme
def FTBS(h,Lamb):
    #generate array of intial values at t = 0
    X = np.arange(-1,3+h,h)
    temp = []
    for dx in X:
```

```
        temp.append(intial_foo(dx))
    next_ = np.array(temp)

    steps = int(2.4/(Lamb*h)) +2
    for time in range(steps):
        #plot
        title = \
        "FTBS h = " + str(h) + " Lambda = " +str(Lamb)
        str_time = '0'*(4-len(str(time)))+str(time)
        outFile = "Figures\FTBS" + str_time
        plot(X,next_,time*Lamb*h,title,outFile)

        #implement Scheme
        next_ = Lamb*(np.roll(next_,1)-next_) + next_

        #Boundary Conditions
        next_[0]  = 0
    #makeGif
    makeGif("FTBS_h_"+str(h)+"_Lambda_"+str(Lamb))

#Forward Time Central Space Scheme
def FTCS(h,Lamb):
    #generate array of intial values at t = 0
    X = np.arange(-1,3+h,h)
    temp = []
    for dx in X:
        temp.append(intial_foo(dx))
    next_ = np.array(temp)

    steps = int(2.4/(Lamb*h)) +2
    for time in range(steps):
        #plot
        title = \
        "FTCS h = " + str(h) + " Lambda = " +str(Lamb)
        str_time = '0'*(4-len(str(time)))+str(time)
        outFile = "Figures\FTCS" + str_time
        plot(X,next_,time*Lamb*h,title,outFile)

        #implement Scheme
        next_ = .5*Lamb*(np.roll(next_,1)-np.roll(next_,-1)) + next_

        #Boundary Conditions
        next_[-1] = next_[-2]
        next_[0]  = 0

    #makeGif
    makeGif("FTCS_h_"+str(h)+"_Lambda_"+str(Lamb))

#Lax Friedrichs Scheme
def LaxFriedrichs(h,Lamb):
    #generate array of intial values at t = 0
    X = np.arange(-1,3+h,h)
    temp = []
    for dx in X:
        temp.append(intial_foo(dx))
    next_ = np.array(temp)

    steps = int(2.4/(Lamb*h)) +2
    for time in range(steps):
        #plot
        title = \
        "Lax-Friedrichs h = " + str(h) + " Lambda = " +str(Lamb)
        str_time = '0'*(4-len(str(time)))+str(time)
        outFile = "Figures\Lax" + str_time
        plot(X,next_,time*Lamb*h,title,outFile)

        #implement Scheme
        next_ = .5*((1+Lamb)*np.roll(next_,1)-(1-Lamb)*np.roll(next_,-1))

        #Boundary Conditions
        next_[-1] = next_[-2]
        next_[0]  = 0
```

```
    #makeGif
    makeGif("LaxFriedrichs_h_"+str(h)+"_Lambda_"+str(Lamb))

#LeapFrog Scheme
def LeapFrog(h,Lamb):
    #generate array of intial values at t = 0
    X = np.arange(-1,3+h,h)
    temp = []
    for dx in X:
        temp.append(intial_foo(dx))

    prev_ = np.array(temp)
    #Need first step from FTCS
    current_ = .5*Lamb*(np.roll(prev_,1)-np.roll(prev_,-1)) + prev_

    steps = int(2.4/(Lamb*h)) + 2
    for time in range(steps):
        #plot
        title = \
        "Leap Frog h = " + str(h) + " Lambda = " +str(Lamb)
        str_time = '0'*(4-len(str(time)))+str(time)
        outFile = "Figures\LF" + str_time
        plot(X,prev_,time*Lamb*h,title,outFile)

        #implement Scheme
        next_ = Lamb*(np.roll(current_,1)-np.roll(current_,-1)) + prev_
        prev_ = current_
        current_ = next_

        #Boundary Conditions
        next_[-1] = next_[-2]
        next_[0]  = 0

    #makeGif
    makeGif("LeapFrog_h_"+str(h)+"_Lambda_"+str(Lamb))
    return 0

#Warning this will take roughly 15 minutes to run
if __name__ == '__main__':
    H = [1.0/10,1.0/20,1.0/40]
    L1 = 0.8
    L2 = 1.6

    #Run all Cases
    for h in H:
        FTBS(h,L1)
        FTCS(h,L1)
        LaxFriedrichs(h,L1)
        LaxFriedrichs(h,L2)
        LeapFrog(h,L1)
"""
Report:
    FTBS:
        A quick and dirty solution that offers a fast and useful answer.
        However the approximation clearly leaks information as a function
        of h. With a refined grid size this approximation would be approiate.

    FTCS:
        This scheme blows up for values of h. Therefore in this case it is
        useless. Each solution seems to blow up around t = 1

    Lax-Friedrichs:
        This scheme has two cases, with lambda < 1, information leaks extremly
        fast and the approximation goes to zero quickly. For lambda > 1 the
        approximation blows up. Therefore this scheme is useless.

    Leap Frog:
        A two-step solution with excellent results. A uselful scheme for this
        problem. As h decreases the error becomes extremly small approaching
        machine error.
"""
```