5)

```
#!/usr/bin/env python3

import numpy as np
import pandas as pd

def trap_int(f,a,b,n):
    h = (b-a)/float(n)
    X = np.linspace(a,b,n+1)
    Y = f(X)

    result = 0.0
    for i in range(n):
        result += h/2.0 * (Y[i] + Y[i+1])
    return result #h/2.0 * (Y[0] + 2*np.sum(Y[1:-1]) + Y[-1])

def simp_int(f,a,b,n):
    h = (b-a)/float(n)
    X = np.linspace(a,b,n+1)
    Y = f(X)

    result = 0.0
    for i in range(int(n/2)):
        l = 2*i
        result += h/3.0 * (Y[l] + 4.0*Y[l+1] + Y[l+2])
    return result

def ratio(errors):
    num = errors[:-1]
    den = errors[1:]
    return num/den

def order(ratios):
    return np.log2(ratios)

def get_results(approx):
    errors = abs(1.0 - approx)
    ratios = ratio(errors)
    orders = order(ratios)
    return errors,ratios,orders

if __name__ == '__main__':

    f = np.sin
    a = 0.0;b=np.pi/2.0
    exact = 1.0

    results = []

    results.append(['%d'%2**i for i in range(6)])

    N = [1,2,4,8,16,32]
    trap = np.array([trap_int(f,a,b,_) for _ in N])
```

```
    simp = np.array([simp_int(f,a,b,_) for _ in N[1:]])

    trap_res = get_results(trap)
    simp_res = get_results(simp)

    results.append(['%.5f'%_ for _ in trap])
    results.append(['%.5f'%_ for _ in trap_res[0]])
    results.append(['-'] + ['%.5f'%_ for _ in trap_res[1]])
    results.append(['-'] + ['%.5f'%_ for _ in trap_res[2]])

    results.append(['-'] + ['%.5f'%_ for _ in simp])
    results.append(['-'] + ['%.5f'%_ for _ in simp_res[0]])
    results.append(['-','-'] + ['%.5f'%_ for _ in simp_res[1]])
    results.append(['-','-'] + ['%.5f'%_ for _ in simp_res[2]])

    results = np.array(results).T

    headers = [r'$n$',r'$T_n(f)$',r'${\rm error}(n)$',r'${\rm ratio}(n)$',r'$
{\rm order}(n)$',r'$S_n(f)$',r'${\rm error}(n)$',r'${\rm ratio}(n)$',r'${\
rm order}(n)$']

    table = pd.DataFrame(results,columns=headers)
    table.to_latex(buf='prob5.tex',
            index=False,
            escape=False,
            column_format=len(headers)*'c',
            caption=r'Results for trapezoidal and simpson rule integrations
schemes for different subdivisions on the interval $[0,\pi/2]$.',
            position='H')
```

Table 1: Results for trapezoidal and simpson rule integrations schemes for different subdivisions on the interval $[0, \pi/2]$.

n	$T_n(f)$	error(n)	ratio(n)	order(n)	$S_n(f)$	error(n)	ratio(n)	order(n)
1	0.78540	0.21460	-	-	-	-	-	-
2	0.94806	0.05194	4.13168	2.04673	1.00228	0.00228	-	-
4	0.98712	0.01288	4.03134	2.01126	1.00013	0.00013	16.94006	4.08237
8	0.99679	0.00321	4.00774	2.00279	1.00001	0.00001	16.22381	4.02004
16	0.99920	0.00080	4.00193	2.00070	1.00000	0.00000	16.05529	4.00498
32	0.99980	0.00020	4.00048	2.00017	1.00000	0.00000	16.01378	4.00124

6)

```
#!/usr/bin/env python3

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
def Euler(F,y0,a,b,h):
    t = [a]; w = [y0]
    while t[-1] < b:
        w.append(w[-1] + h*F(t[-1],w[-1]))
        t.append(t[-1] + h)
    t = np.array(t); w = np.array(w)
    return t,w

def Taylor2(F,Fp,y0,a,b,h):
    t = [a]; w = [y0]
    while t[-1] < b:
        w.append(w[-1] + h*(F(t[-1],w[-1]) + h/2.0*Fp(t[-1],w[-1])))
        t.append(t[-1] + h)
    t = np.array(t); w = np.array(w)
    return t,w

def RK2(F,y0,a,b,h):
    t = [a]; w = [y0]
    while t[-1] < b:
        k1 = F(t[-1],w[-1])
        k2 = F(t[-1]+h/2.0,w[-1]+h*k1/2.0)
        w.append(w[-1] + h*k2)
        t.append(t[-1] + h)
    t = np.array(t); w = np.array(w)
    return t,w

def error(approx):
    exact = y(1.0)
    return abs(exact - approx)

def order(errors):
    num = errors[:-1]
    den = errors[1:]
    return np.log2(num/den)

def get_results(approx):
    errors = error(approx)
    orders = order(errors)
    return errors,orders

if __name__ == '__main__':

    F  = lambda t,y: -10.0 * y**2.0
    Fp = lambda t,y: 200.0 * y**3.0
    a = 0.0; b = 1.0
    y0 = 1

    y = lambda t: 1.0/(10.0 * t + 1.0)

    H = [1.0/(10.0 * 2.0**float(i)) for i in range(1,6)]

    sub_heads = [r'${\rm error}(h)$',r'${\rm order}(h)$']

    col = ['$1/20$','$1/40$','$1/80$','$1/160$','$1/320$']
```

```
    h_col = pd.DataFrame(col,columns=['$h$'])

    e = []
    temp = np.array([Euler(F,y0,a,b,_)[1][-1] for _ in H])
    res = get_results(temp)
    #e.append(['%.4f'%_ for _ in temp])
    e.append(['%.5f'%_ for _ in res[0]])
    e.append(['-'] + ['%.5f'%_ for _ in res[1]])
    e = np.array(e).T
    e = pd.DataFrame(e,columns=sub_heads)

    t = []
    temp = np.array([Taylor2(F,Fp,y0,a,b,_)[1][-1] for _ in H])
    res = get_results(temp)
    #t.append(['%.4f'%_ for _ in temp])
    t.append(['%.5f'%_ for _ in res[0]])
    t.append(['-'] + ['%.5f'%_ for _ in res[1]])
    t = np.array(t).T
    t = pd.DataFrame(t,columns=sub_heads)

    r = []
    temp = np.array([RK2(F,y0,a,b,_)[1][-1] for _ in H])
    res = get_results(temp)
    #r.append(['%.4f'%_ for _ in temp])
    r.append(['%.5f'%_ for _ in res[0]])
    r.append(['-'] + ['%.5f'%_ for _ in res[1]])
    r = np.array(r).T
    r = pd.DataFrame(r,columns=sub_heads)

    table = pd.concat({r'$\phantom{h}$':h_col,'Euler':e,'Taylor':t,'Runge
Kutta':r},
            axis=1)
    table = table.reset_index(drop=True)
    table = table.set_index([[''] * table.shape[0]])
    tex_output = table.to_latex(index=True,
            escape=False,
            column_format=table.shape[1]*'c',
            multicolumn_format='c',
            caption=r'Accuracy table at $t=1$ with different ODE approximation
 schemes, where the exact value is $y(1) = 1/11$.',
            position='H')
    tex_output = tex_output.replace('{} &','')

    with open('./prob6.tex','w') as f:
        f.write(tex_output)
```

Table 2: Accuracy table at $t = 1$ with different ODE approximation schemes, where the exact value is $y(1) = 1/11$.

	Euler		Taylor		Runge Kutta	
h	error(h)	order(h)	error(h)	order(h)	error(h)	order(h)
1/20	0.01080	-	0.00372	-	0.00235	-
1/40	0.00507	1.09058	0.00064	2.54041	0.00045	2.38303
1/80	0.00348	0.54393	0.00089	-0.47378	0.00092	-1.03682
1/160	0.00175	0.99365	0.00048	0.88107	0.00049	0.91419
1/320	0.00062	1.49271	0.00001	5.98905	0.00001	6.43657