```
from numpy import linspace, zeros, log, dot, linalg
from matplotlib.pyplot import plot, xscale, yscale

def A(x,h,N) :
    A = zeros((N-1,N-1))
    for n in range(1,N-1) :
        A[n,n-1] = -1/h**2 - 9*x[n+1]/(2*h)
    for n in range(N-1) :
        A[n,n] = 2/h**2
    for n in range(N-2) :
        A[n,n+1] = -1/h**2 + 9*x[n+1]/(2*h)
    return A

def EigenvalueFinding(A,S_max,N) :
    h = (1 - 0)/N
    x = linspace(0,1,N+1)
    lambd = zeros(S_max+1)
    y = zeros((S_max+1,N-1))
    y[0] = 1.
    for s in range(S_max) :
        y[s+1] = linalg.solve(A(x,h,N),y[s])
        lambd[s+1] = 1/(dot(y[s+1],y[s+1])/dot(y[s],y[s+1]))
    eigenvalue = lambd[s+1]
    return eigenvalue

N = 20; S_max = 20
r = 2; S = 5
p = 2; q = 2

L = zeros((S,S))
R = zeros((S,S))
p_eff = zeros((S,S))

for s in range(S) :
    L[s,0] = EigenvalueFinding(A,S_max,r**s*N)

for s in range(1,S) :
    for l in range(s) :
        R[s,l] = (L[s,l] - L[s-1,l])/(r**(p + l*q) - 1)
        L[s,l+1] = L[s,l] + R[s,l]

for s in range(2,S) :
    for l in range(s-1) :
        p_eff[s,l] = log(abs(R[s-1,l]/R[s,l]))/log(r)

# Функция выводит форматированную таблицу
def PrintTriangular(A,i) :
    print('     ',end=' ')
    for l in range(len(A)) :
        print(' p={0:<4d}'.format(p + l*q),end=' ')
    print()
    for m in range(len(A)) :
        print('s={0:<2d}'.format(m),end=' ')
        for l in range(m + 1 - i) :
            print('{0:7.4f}'.format(A[m,l]),end=' ')
        print()
    print()

print('Таблица приближённых значений интеграла:')
PrintTriangular(L,0)
print('Таблица оценок ошибок:')
PrintTriangular(R,1)
print('Таблица эффективных порядков точности:')
PrintTriangular(p_eff,2)

plot([r**s*N for s in range(1,S)],abs(R[1:,0]),'-bo')
xscale('log'); yscale('log')

# Листинг программы, реализущей приближённое вычисление
```

```
# минимального собсенного значение задачи Штурма-Лиувилля
# с помощью рекурретного сгущения сеток и многократного повышения
# точности по Ричардсону (с вычислением эффективных порядков точности)
```