In [1]:

```
%matplotlib inline
#Импротируем необходимые функции
import numpy as np # работа с массивами и линейной алгеброй
import matplotlib.pyplot as plt # для отрисовки графиков
import scipy.linalg as la # функции линейной алгебры
import math as math

from scipy import integrate
```

Задача №5.3. Квадратуры Гаусса.

In [2]:

```
#Зададим интегрируемую функцию
f = lambda x: np.exp(x)

#Зададим количество узлов и границы отрезка
n = 5
left = -1
right = 1
```

In [3]:

```
#В качестве узлов квадратурной формулы берем корни (нули) полиномов Лежандра
def Gaussian_quadrature_Points(n):
    A = np.zeros(n+1)
    A[n] = 1
    L_n = np.polynomial.legendre.Legendre(A)
    return L_n.roots()
```

In [4]:

```
#Так как мы хотим, чтобы квадратурная формула была точна для многочленов, то для каждог
o i = 0,1,..2n-1 должно выполняться
def Gaussian_quadrature_Weights(points, n):
    A = np.zeros((n, n))
    b = np.zeros(n)
    for i in range (n):
        A[i] = points ** i
        b[i] = (1 + (-1)**i) /(i + 1)
    #print(np.polynomial.legendre.legweight(points))
    return np.linalg.solve(A, b)
```

In [5]:

```
#вычисляем интеграл
def Gaussian_quadrature(points, weights, f, n, left, right):
    answer = 0
    for i in range (n):
        x = points[i] *(right-left) / 2 + (right+left) / 2
        answer += f(x) * weights[i]
    answer = answer * (right - left) / 2
    return answer
```

In [6]:

```
#стандартная функция, вычисляющая узлы и веса квадратуры Гаусса
points, weights = np.polynomial.legendre.leggauss(n)
print(points)
print(weights)

#Мои функции, которые вычисляют узлы и веса квадратуры Гаусса
points1 = Gaussian_quadrature_Points(n)
weights1 = Gaussian_quadrature_Weights(points1, n)
print(points1)
print(weights1)
```

```
[-0.90617985 -0.53846931  0.          0.53846931  0.90617985]
[0.23692689 0.47862867 0.56888889 0.47862867 0.23692689]
[-9.06179846e-01 -5.38469310e-01  7.47000758e-17  5.38469310e-01
  9.06179846e-01]
[0.23692689 0.47862867 0.56888889 0.47862867 0.23692689]
```

In [7]:

```
#расмотрим разность между узлами и весами, вычисленными стандартной функцией
#и узлами и весами, вычисленными нашей функцией
print(np.abs(points - points1))
print(np.abs(weights - weights1))
```

```
[6.66133815e-16 2.22044605e-16 7.47000758e-17 1.11022302e-16
 3.33066907e-16]
[1.33226763e-15 8.88178420e-16 3.33066907e-16 4.44089210e-16
 1.38777878e-16]
```

In [8]:

```
#Значение интеграла, вычисленный стандартной функцией
answer = integrate.quad(f, left, right)
print(answer[0])

#Значение интеграла, вычисленный с помощью квадратуры Гаусса
answer1 = Gaussian_quadrature(points1, weights1, f, n, left, right)
print(answer1)
```

```
2.3504023872876028
2.350402386462826
```

In [9]:

```
#рассмотрим модуль разности между результатами
print(np.abs(answer[0] - answer1))
```

```
8.247766913882515e-10
```