1. **(15)** Вспомните разобранный на семинаре пример. Теперь рассмотрите интеграл $I_n(\alpha) = \int_0^1 \frac{x^n}{x+\alpha}dx$ и получите i) рекуррентное соотношение, связывающее I_n с I_{n-1} и ii) явное выражение для $I_0(\alpha)$. Вычислите (прямой и обратной) рекурсией значения $I_{25}(0.1)$ и $I_{25}(10)$. Обсудите результат.

2. **(10)** Реализуйте функцию, возвращающую пару решений квадратного уравнения (следуйте инструкциям по ссылке).

3. **(10)** Рассмотрите следующую функцию:

```
def recur(n):
    if n == 0:
        return 1
    if n == 1:
        return -3
    return -recur(n-1) + 6*recur(n-2)
```

   Чему будет равен результат вызова 'recur(2020)'? Диапазон определения целых чисел считать неограниченным (т.е., целые числа не переполняются), размер стека также считать неограниченным (т.е. максимальное число рекурсивных вызовов не ограничено).

4. **(5)** Рассмотрите (считая $\delta > 0$) матрицу $A = \begin{pmatrix} 1 & 10 \\ \delta & 1 \end{pmatrix}$. Пусть $\epsilon(\delta)$ – наибольшее собственное значение A. Найдите число обусловленности этого собственного значения $\kappa(\delta) = \frac{d\epsilon(\delta)}{d\delta}$ для $\delta = 10$ и $\delta = 0.1$.

5. **(7)** Убедитесь, что функция

```
import math
def round_to_n(x, n):
    if x == 0:
        return x
    else:
        return round(x, -int(math.floor(math.log10(abs(x)))) + (n - 1))
```

   округляет x до n значащих цифр. Программа, вычисляющая $\sum_{k=1}^{3000} k^{-2} \approx 1.6446$ последовательным суммированием членов ряда с округлением промежуточных результатов до 4х знаков выглядит следующим образом:

```
res = 0
for k in range(1,3001):
    res = round_to_n(res+1/k**2, 4)
```

   Несмотря на отсутствие вычитаний (и связанных с ними сокращений), такой код позволяет получить только две значащие цифры точного ответа. Объясните, почему так происходит и предложите более удачный способ вычисления этой суммы (ограничиваясь 4мя значащими цифрами для промежуточных результатов).