Task 1

In this task were realized two functions for computing definite integrals - `riemann_method` and `trapezoidal_method`.
Function `compare_accuracy` takes function and returns *dict* with absolute error values of two methods described above.
Function `accuracy_table` builds a table of errors for every integration method in relation to **n**.

1. **f(x) = cos(x)**

	10 ^ 0	10 ^ 1	10 ^ 2	10 ^ 3	10 ^ 4
Riemann method error	0.070526637	0.000760098	5.09e-7	8.960e-6	1.35e-7
Trapezoidal method error	0.071319832	0.000701343	7.012e-6	7.0e-8	1.e-9

2. **f(x) = 2 ^ x**

	10 ^ 0	10 ^ 1	10 ^ 2	10 ^ 3	10 ^ 4
Riemann method error	0.070111328	0.003901726	1.8019e-5	2.48e-7	3.32e-7
Trapezoidal method error	0.057304959	0.000577576	5.776e-6	5.8e-8	1.e-9

3. **f(x) = x ^ 5 - x ^ 2**

	10 ^ 0	10 ^ 1	10 ^ 2	10 ^ 3	10 ^ 4
Riemann method error	0.127136728	0.010002609	0.000256840	6.022e-6	3.65e-7
Trapezoidal method error	0.166666667	0.002491667	2.4999e-5	2.50e-7	3.e-9

We measured accuracy of different types of functions and can see, **trapezoidal** method is better and has lower absolute errors. Also, it is evident that the smaller the diameter of the division, the smaller the error.

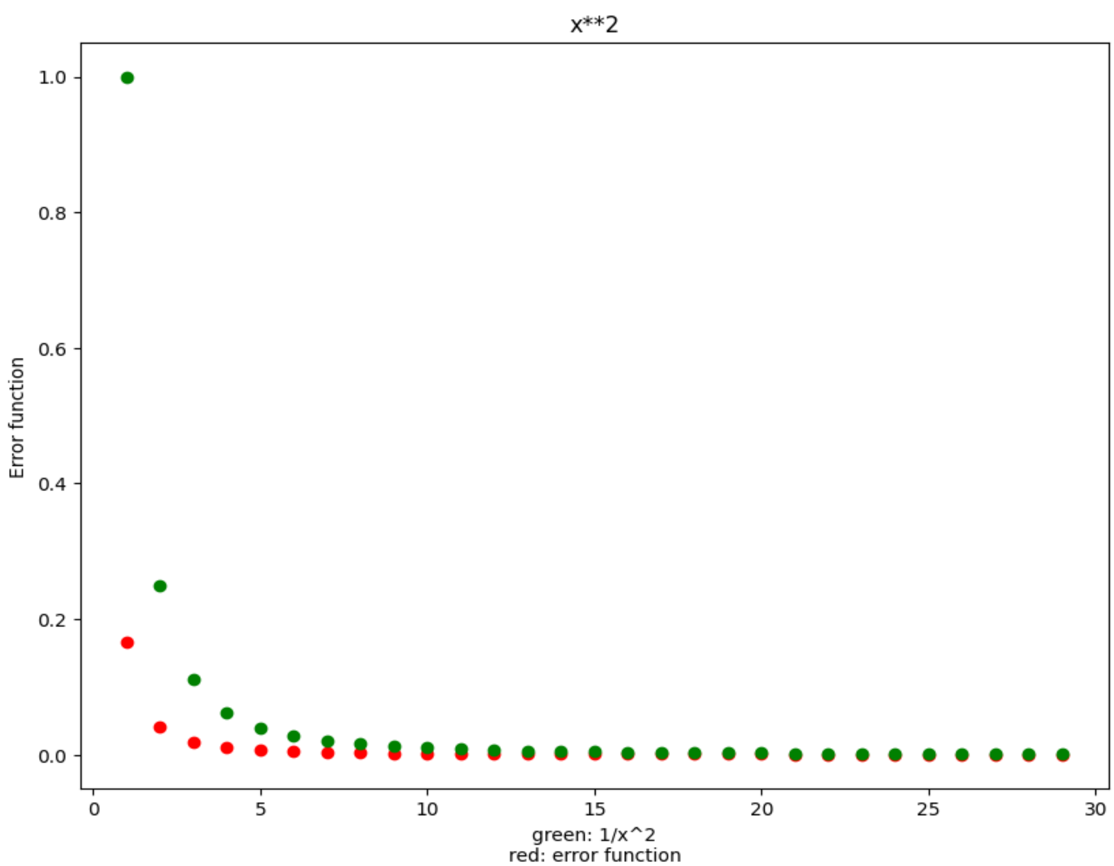
Task 2

Function `get_plot_of_error` makes plots of error function and function 1 / n ^ 2.
Also, the approximate ratio between these functions **C** is printed to the console.

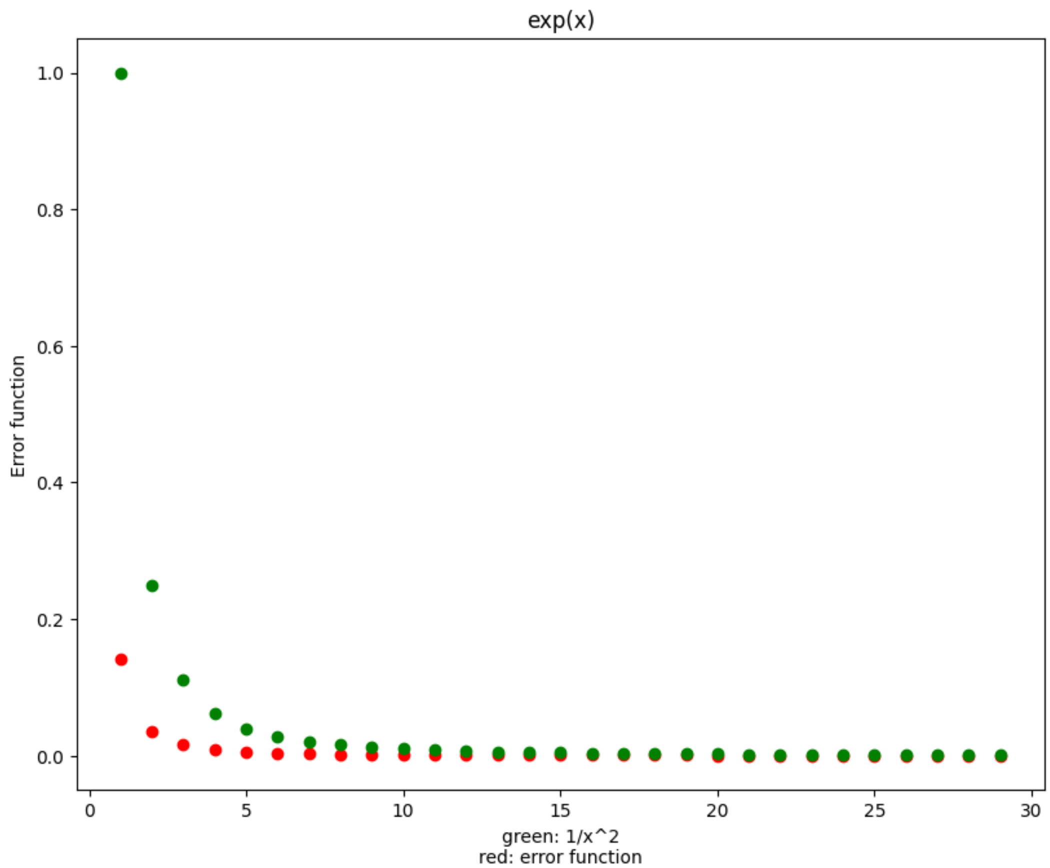
All functions were integrated on range **[0; 1]**. **n** in range from 1 to 30.
From this plots we can see that error function *E(n) = C * (1 / n ^ 2)*.
For example, for the first function **C ~ 0.17.**

There are plots of error function for:

1) **f(x) = x ^ 2, C ~ 0.17**

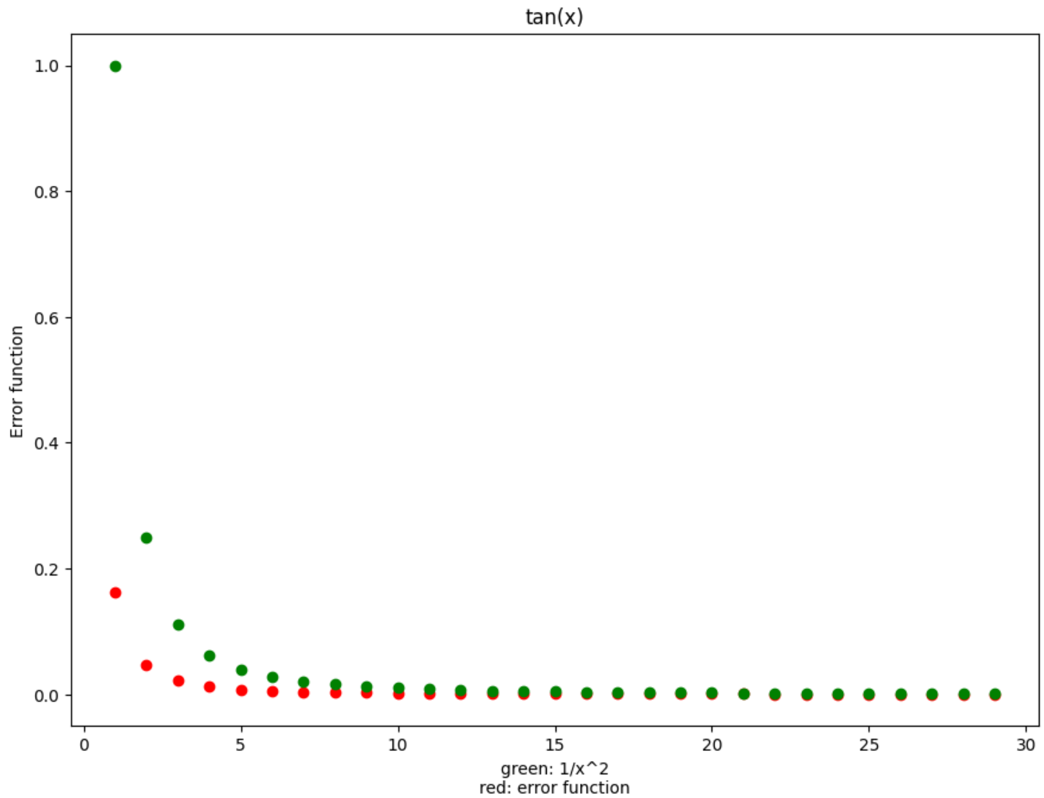


2) f(x) = e ^ x, C ~ 0.14



Mykola Vysotskyi

3) $f(x) = \tan(x)$, $C \sim 0.20$



Mykola Vysotskyi