Sofya Aksenyuk – 150284

exercises

1. 5.0  
2. If we use Python 2.7, we will get 2.75 and 2. Whereas in case of using Python 3.8, we will get 2.75 and 2.75. This is due to the different approach for accumulation of float and integer values.  
3. 0.0

5e-200

The output above is incorrect. Therefore, this is the proper program:

import math  
a = 3e-200  
b = 4e-200  
c = 1e200  
print(math.sqrt(a\*a + b\*b))  
print(c \* a \* math.sqrt(1 + (b/a) \* (b/a)))

Now output is:

0.0

5.0

4.

1)9.185

2)-14.3125

3)-14.4375

4)0.185

5)14.9375

1) 10.125

2) -1.375

3) -13.875

4) -3.375

5) 13.6875

5.

1)-26.5625

2)-27.4375

3)-23.0625

4)24.1875

5)-29.25

1)29.6875

2)-16.9375

3)28.0

4)25.75

5)-20.3125

Numerical stability

n = int(input())

k = int(input())

nom = 1

den = 1

brac = 1

for i in range(1, n):

nom = nom \* (nom + 1)

for j in range(1, k):

den = den \* (den + 1)

for z in range(n - k - 1):

brac = brac \* (brac + 1)

print(int(nom / den \* brac))

The improved version:

import math

n = int(input())

k = int(input())

if k == 1 or n == k:

print(1)

if k > n:

print(0)

else:

x = math.factorial(n)

y = math.factorial(k)

div = x // (y\*(n-k))

print(div)

## Maclaurin series

import math as m

x = 1.2

k = 0

N = 25

s = 0

sign = 1.0

while k < N:

term = sign \* x \*\* k / m.factorial(k)

s += term

k += 2

sign = -sign

print(s)