Profiling of mathlib.py using cProfile by solving a corrected sample standard deviation equation

Test 1:

Sample: 10 numbers ranging from 1 to 1000

Test execution:

```
ondrej@OndrejovaMasina:~/Documents/VUT/IVS/Projekt2/floor11_calc/floor11_calc/src$ python3 -m cProfile  profiling.py <10_num.txt
214.05388729632224
         934 function calls (901 primitive calls) in 0.001 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:103(release)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:143(__init__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:147(__enter__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:151(__exit__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:157(_get_module_lock)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:176(cb)
     11/1    0.000    0.000    0.001    0.001 <frozen importlib._bootstrap>:211(_call_with_frames_removed)
       45    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:222(_verbose_message)
        4    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:232(_requires_builtin_wrapper)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:342(__init__)
        3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:35(_new_module)
        6    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:376(cached)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:389(parent)
```

mathlib.py records:

```
20    0.000    0.000    0.000    0.000 mathlib.py:12(add)
 2    0.000    0.000    0.000    0.000 mathlib.py:17(sub)
 3    0.000    0.000    0.000    0.000 mathlib.py:22(mul)
 2    0.000    0.000    0.000    0.000 mathlib.py:28(div)
11    0.000    0.000    0.000    0.000 mathlib.py:55(exp)
 1    0.000    0.000    0.001    0.001 mathlib.py:7(<module>)
 1    0.000    0.000    0.000    0.000 mathlib.py:70(root)
```

Average test time: **1ms**

Test conclusion: The time it takes to complete the fucntions is so insignificant, it isn’t even registered and shown by cProfile. The program spends most of runtime by importing libraries.

Test 2:

Sample: 100 numbers ranging from 1 to 1000

Test execution:

```
ondrej@OndrejovaMasina:~/Documents/VUT/IVS/Projekt2/floor11_calc/floor11_calc/src$ python3 -m cProfile  profiling.py <100_num.txt
285.90686394083804
         1298 function calls (1265 primitive calls) in 0.001 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:103(release)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:143(__init__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:147(__enter__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:151(__exit__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:157(_get_module_lock)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:176(cb)
     11/1    0.000    0.000    0.001    0.001 <frozen importlib._bootstrap>:211(_call_with_frames_removed)
       45    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:222(_verbose_message)
        4    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:232(_requires_builtin_wrapper)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:342(__init__)
        3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:35(_new_module)
        6    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:376(cached)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:389(parent)
```

mathlib.py records:

```
200    0.000    0.000    0.000    0.000 mathlib.py:12(add)
  2    0.000    0.000    0.000    0.000 mathlib.py:17(sub)
  3    0.000    0.000    0.000    0.000 mathlib.py:22(mul)
  2    0.000    0.000    0.000    0.000 mathlib.py:28(div)
101    0.000    0.000    0.000    0.000 mathlib.py:55(exp)
  1    0.000    0.000    0.001    0.001 mathlib.py:7(<module>)
  1    0.000    0.000    0.000    0.000 mathlib.py:70(root)
```

Average test time: **1ms**

Test conclusion: Just like in test no. 1, the time it takes to complete the fucntions is insignificant and the program spends most of runtime by importing libraries.

Test 3:

Sample: 1000 numbers ranging from 1 to 1000

Test execution:

```
ondrej@OndrejovaMasina:~/Documents/VUT/IVS/Projekt2/floor11_calc/floor11_calc/src$ python3 -m cProfile  profiling.py <1000_num.txt
289.09381735906294
         4937 function calls (4904 primitive calls) in 0.003 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:103(release)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:143(__init__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:147(__enter__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:151(__exit__)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:157(_get_module_lock)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:176(cb)
     11/1    0.000    0.000    0.001    0.001 <frozen importlib._bootstrap>:211(_call_with_frames_removed)
       45    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:222(_verbose_message)
        4    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:232(_requires_builtin_wrapper)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:342(__init__)
        3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:35(_new_module)
        6    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:376(cached)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:389(parent)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:397(has_location)
        4    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:406(spec_from_loader)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:477(_init_module_attrs)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:549(module_from_spec)
        7    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:58(__init__)
```

mathlib.py records:

```
 2000    0.000    0.000    0.000    0.000 mathlib.py:12(add)
    2    0.000    0.000    0.000    0.000 mathlib.py:17(sub)
    3    0.000    0.000    0.000    0.000 mathlib.py:22(mul)
    2    0.000    0.000    0.000    0.000 mathlib.py:28(div)
 1001    0.000    0.000    0.000    0.000 mathlib.py:55(exp)
    1    0.000    0.000    0.001    0.001 mathlib.py:7(<module>)
    1    0.000    0.000    0.000    0.000 mathlib.py:70(root)
```

Average test time: **2-3ms**

Test conclusion: Even after 2000 calls of the same function, the time it takes to execute it is simply too small. Although in the instance of 1000 numbers we see an increase in the overall execution time by 1-2 ms, this is caused simply by loading the 1000 numbers into an array for further calculations.

Summary:

After these tests I concluded, that the mathlib.py functions are pretty well optimized. The time it takes to execute them is insignificant for a simple calculator, in which we will hardly see over 100 uses of the same function in one run, let alone a few thousands. I wasn't really surprised by these results, considering that most of the functions use basic arithmetic operations which are already well optimized (such as +, -,*,/ etc.) to return a value. Most of the overhead in these functions is created by checking the validity of operands and calling the functions themselves.