IVS projekt 2 – Profiling

PyJaMa's

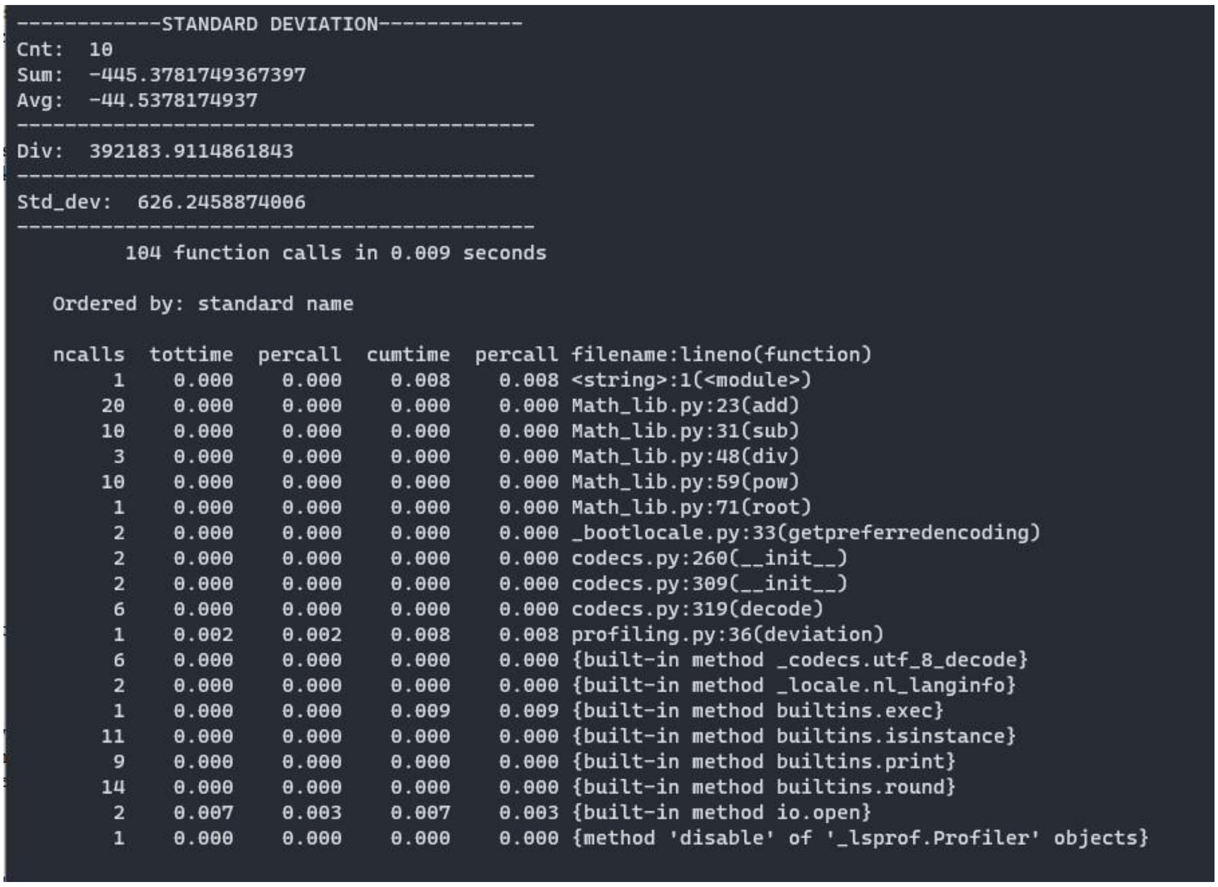
Matej Koreň, xkoren10

16.4.2021

Program profiling.py can open a file with numbers or generate its own with random numbers. The output of this program is a standard deviation of these numbers and with the use of a cProfile library also an amount of function calls and total runtime.

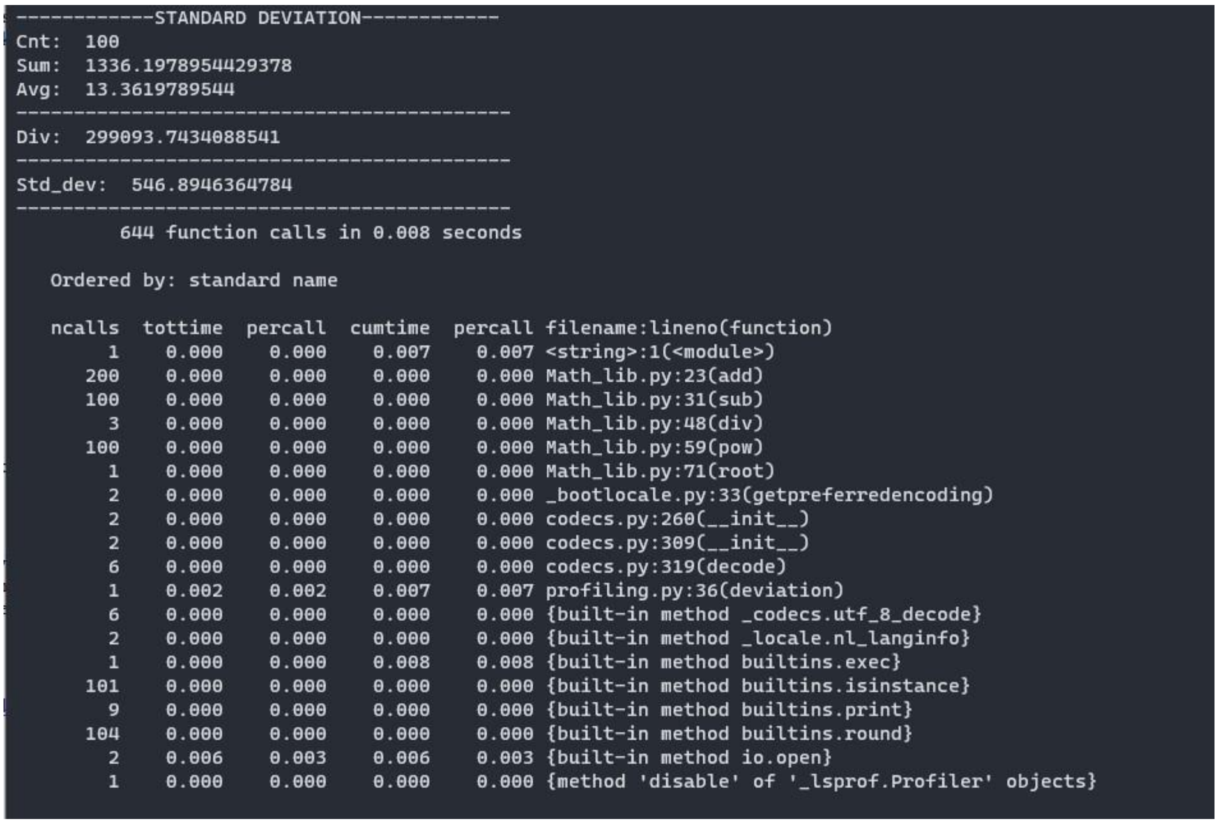
To run this program, you need to have Python installed (e.g. Python 3.9.). Then, through the command line, change your directory to the folder with profiling.py in it (src) and type ***python3 profiling.py*** (which will use random generated numbers) or ***python3 profiling.py*** < (test file name).

Here we can see outputs of certain runs with randomly generated numbers:



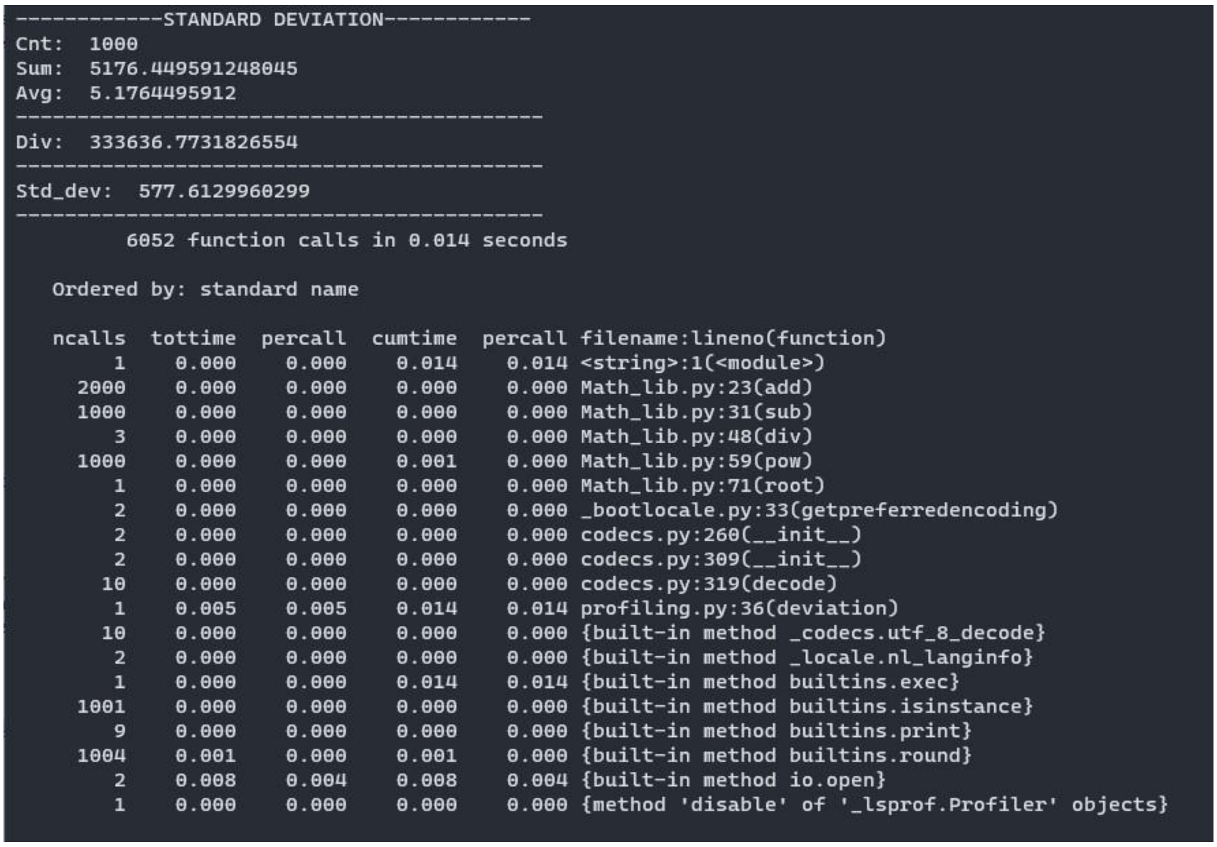
```
------------STANDARD DEVIATION-----------
Cnt:  10
Sum:  -445.3781749367397
Avg:  -44.5378174937
-------------------------------------------
Div:  392183.9114861843
-------------------------------------------
Std_dev:  626.2458874006
-------------------------------------------
         104 function calls in 0.009 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    0.008    0.008 <string>:1(<module>)
       20    0.000    0.000    0.000    0.000 Math_lib.py:23(add)
       10    0.000    0.000    0.000    0.000 Math_lib.py:31(sub)
        3    0.000    0.000    0.000    0.000 Math_lib.py:48(div)
       10    0.000    0.000    0.000    0.000 Math_lib.py:59(pow)
        1    0.000    0.000    0.000    0.000 Math_lib.py:71(root)
        2    0.000    0.000    0.000    0.000 _bootlocale.py:33(getpreferredencoding)
        2    0.000    0.000    0.000    0.000 codecs.py:260(__init__)
        2    0.000    0.000    0.000    0.000 codecs.py:309(__init__)
        6    0.000    0.000    0.000    0.000 codecs.py:319(decode)
        1    0.002    0.002    0.008    0.008 profiling.py:36(deviation)
        6    0.000    0.000    0.000    0.000 {built-in method _codecs.utf_8_decode}
        2    0.000    0.000    0.000    0.000 {built-in method _locale.nl_langinfo}
        1    0.000    0.000    0.009    0.009 {built-in method builtins.exec}
       11    0.000    0.000    0.000    0.000 {built-in method builtins.isinstance}
        9    0.000    0.000    0.000    0.000 {built-in method builtins.print}
       14    0.000    0.000    0.000    0.000 {built-in method builtins.round}
        2    0.007    0.003    0.007    0.003 {built-in method io.open}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

10 random generated numbers



```
------------STANDARD DEVIATION------------
Cnt:  100
Sum:  1336.1978954429378
Avg:  13.3619789544
------------------------------------------
Div:  299093.7434088541
------------------------------------------
Std_dev:  546.8946364784
------------------------------------------
         644 function calls in 0.008 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    0.007    0.007 <string>:1(<module>)
      200    0.000    0.000    0.000    0.000 Math_lib.py:23(add)
      100    0.000    0.000    0.000    0.000 Math_lib.py:31(sub)
        3    0.000    0.000    0.000    0.000 Math_lib.py:48(div)
      100    0.000    0.000    0.000    0.000 Math_lib.py:59(pow)
        1    0.000    0.000    0.000    0.000 Math_lib.py:71(root)
        2    0.000    0.000    0.000    0.000 _bootlocale.py:33(getpreferredencoding)
        2    0.000    0.000    0.000    0.000 codecs.py:260(__init__)
        2    0.000    0.000    0.000    0.000 codecs.py:309(__init__)
        6    0.000    0.000    0.000    0.000 codecs.py:319(decode)
        1    0.002    0.002    0.007    0.007 profiling.py:36(deviation)
        6    0.000    0.000    0.000    0.000 {built-in method _codecs.utf_8_decode}
        2    0.000    0.000    0.000    0.000 {built-in method _locale.nl_langinfo}
        1    0.000    0.000    0.008    0.008 {built-in method builtins.exec}
      101    0.000    0.000    0.000    0.000 {built-in method builtins.isinstance}
        9    0.000    0.000    0.000    0.000 {built-in method builtins.print}
      104    0.000    0.000    0.000    0.000 {built-in method builtins.round}
        2    0.006    0.003    0.006    0.003 {built-in method io.open}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

100 random generated numbers



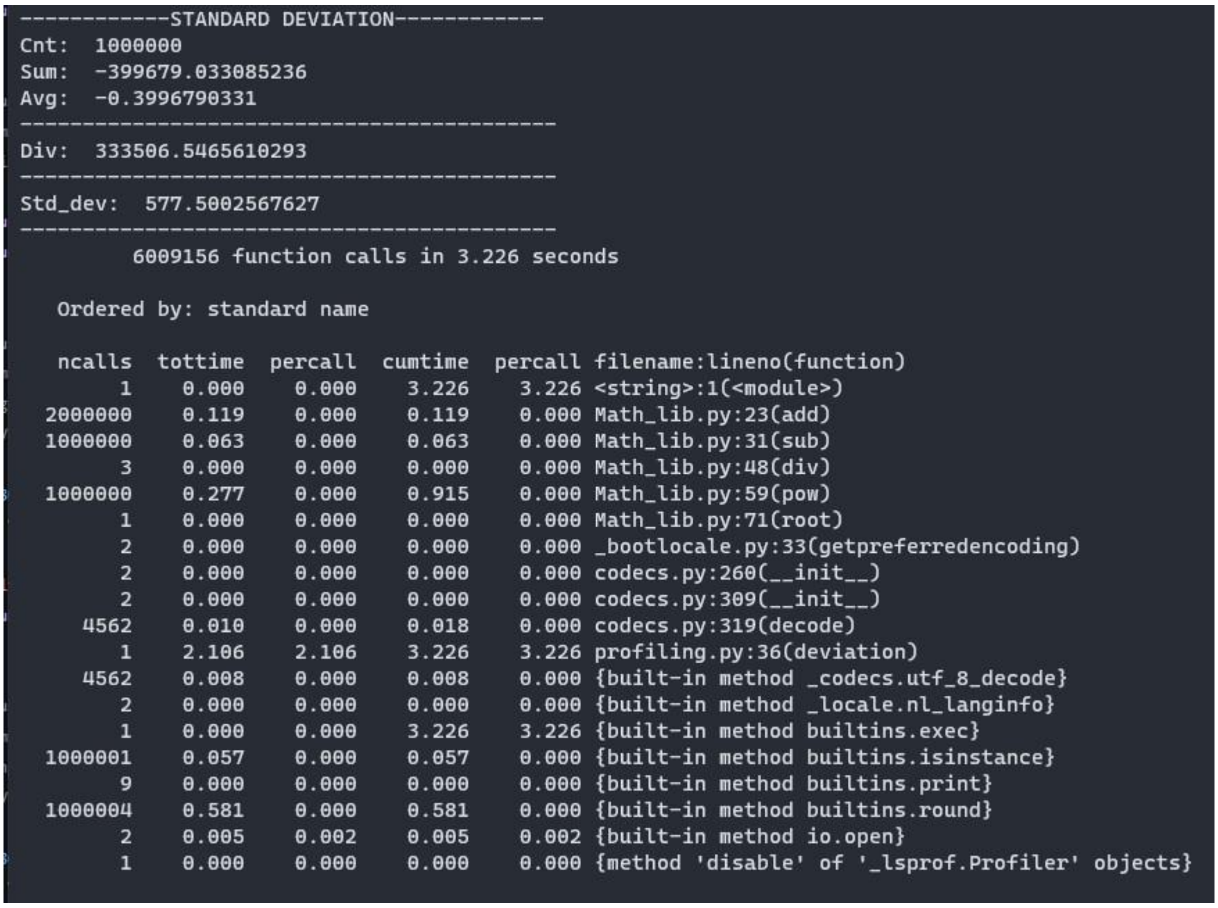
```
------------STANDARD DEVIATION------------
Cnt:  1000
Sum:  5176.449591248045
Avg:  5.1764495912
------------------------------------------
Div:  333636.7731826554
------------------------------------------
Std_dev:  577.6129960299
------------------------------------------
         6052 function calls in 0.014 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    0.014    0.014 <string>:1(<module>)
     2000    0.000    0.000    0.000    0.000 Math_lib.py:23(add)
     1000    0.000    0.000    0.000    0.000 Math_lib.py:31(sub)
        3    0.000    0.000    0.000    0.000 Math_lib.py:48(div)
     1000    0.000    0.000    0.001    0.000 Math_lib.py:59(pow)
        1    0.000    0.000    0.000    0.000 Math_lib.py:71(root)
        2    0.000    0.000    0.000    0.000 _bootlocale.py:33(getpreferredencoding)
        2    0.000    0.000    0.000    0.000 codecs.py:260(__init__)
        2    0.000    0.000    0.000    0.000 codecs.py:309(__init__)
       10    0.000    0.000    0.000    0.000 codecs.py:319(decode)
        1    0.005    0.005    0.014    0.014 profiling.py:36(deviation)
       10    0.000    0.000    0.000    0.000 {built-in method _codecs.utf_8_decode}
        2    0.000    0.000    0.000    0.000 {built-in method _locale.nl_langinfo}
        1    0.000    0.000    0.014    0.014 {built-in method builtins.exec}
     1001    0.000    0.000    0.000    0.000 {built-in method builtins.isinstance}
        9    0.000    0.000    0.000    0.000 {built-in method builtins.print}
     1004    0.001    0.000    0.001    0.000 {built-in method builtins.round}
        2    0.008    0.004    0.008    0.004 {built-in method io.open}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

1000 random generated numbers

Results:

	Time	Function calls
10	9ms	104
100	8ms	644
1000	14ms	6052

As we can see, in every output, despite the amount of numbers, the function which is called the most is add(). However, if we try to compute larger numbers, we can see that the function which takes the longest to run is actually pow() :



```
------------STANDARD DEVIATION------------
Cnt:  1000000
Sum:  -399679.033085236
Avg:  -0.3996790331
-------------------------------------------
Div:  333506.5465610293
-------------------------------------------
Std_dev:  577.5002567627
-------------------------------------------
         6009156 function calls in 3.226 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    3.226    3.226 <string>:1(<module>)
  2000000    0.119    0.000    0.119    0.000 Math_lib.py:23(add)
  1000000    0.063    0.000    0.063    0.000 Math_lib.py:31(sub)
        3    0.000    0.000    0.000    0.000 Math_lib.py:48(div)
  1000000    0.277    0.000    0.915    0.000 Math_lib.py:59(pow)
        1    0.000    0.000    0.000    0.000 Math_lib.py:71(root)
        2    0.000    0.000    0.000    0.000 _bootlocale.py:33(getpreferredencoding)
        2    0.000    0.000    0.000    0.000 codecs.py:260(__init__)
        2    0.000    0.000    0.000    0.000 codecs.py:309(__init__)
     4562    0.010    0.000    0.018    0.000 codecs.py:319(decode)
        1    2.106    2.106    3.226    3.226 profiling.py:36(deviation)
     4562    0.008    0.000    0.008    0.000 {built-in method _codecs.utf_8_decode}
        2    0.000    0.000    0.000    0.000 {built-in method _locale.nl_langinfo}
        1    0.000    0.000    3.226    3.226 {built-in method builtins.exec}
  1000001    0.057    0.000    0.057    0.000 {built-in method builtins.isinstance}
        9    0.000    0.000    0.000    0.000 {built-in method builtins.print}
  1000004    0.581    0.000    0.581    0.000 {built-in method builtins.round}
        2    0.005    0.002    0.005    0.002 {built-in method io.open}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

1000000 random generated numbers

In order to improve the performance of our library, I suggest reworking the pow() function, but in overall the speed of our library functions is pretty decent.