Task 1

```
Timer unit: 1e-07 s

Total time: 0.0079195 s
File: C:\Users\Norbert\AppData\Local\Temp\ipykernel_19664\1799135549.py
Function: res_opencv at line 3

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     3                                           def res_opencv(imgs):
     4         1        111.0    111.0      0.1      new_size = (imgs[1].shape[1] // 2, imgs[1].shape[0] // 2)  # OpenCV erwartet (Breite, Höhe)
     5         1      76435.0  76435.0     96.5      res_im = [cv2.resize(im, new_size, interpolation=cv2.INTER_LINEAR) for im in imgs]
     6         1       2649.0   2649.0      3.3      return np.asarray(res_im)
```

This line has been the bottleneck. With using the opencv libary the resize function now works faster

Task 2:

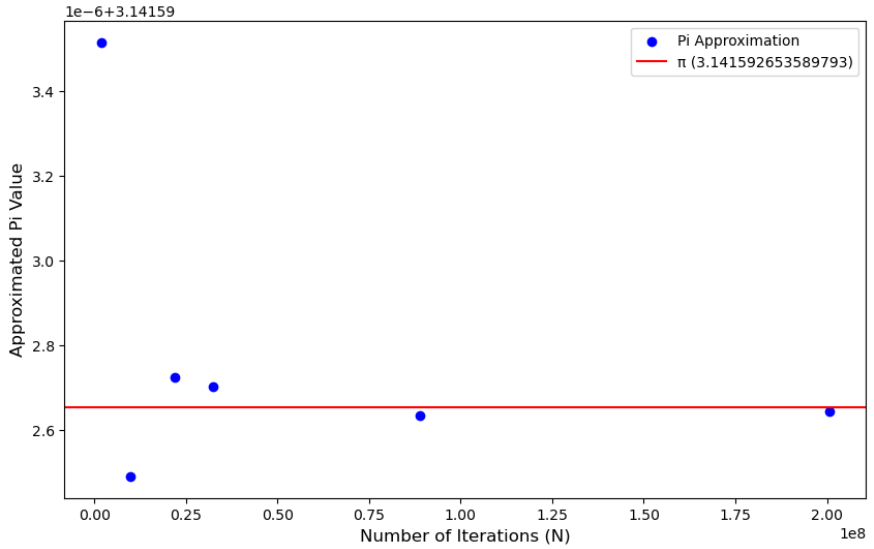
I decided to use multiprocessing as it is a computation heavy function. Multiprocessing can utilize multiple CPU cores for true parallel execution. Multithreading would be limited by Python's Global Interpreter Lock (GIL), which restricts parallel execution of threads for CPU-bound tasks. The sequential call of the function for each value of N took a total of 1min25sec. By using multiprocessing, I was able to reduce this time to 54sec.

Task 3:

```python
from numba import jit

# JIT-compiled function
@jit(nopython=True)  # Activates aggressive optimization
def approximate_pi_numba(n):
    pi_2 = 1.0
    nom, den = 2.0, 1.0
    for i in range(n):
        pi_2 *= nom / den
        if i % 2 == 0:
            den += 2
        else:
            nom += 2
    return 2 * pi_2

# Testrun
nums = [1_822_725, 22_059_421, 32_374_695, 88_754_320, 97_162_66, 200_745_654]
if __name__ == "__main__":
    results = [approximate_pi_numba(n) for n in nums]
    for n, pi_approx in zip(nums, results):
        print(f"N = {n}: Approximation of Pi = {pi_approx}")

print(results)
print(nums)
```

By using numba the code took only 3sec. That‘s 18 times faster than 54sec.

Task 4:



Name: Constantin Wolff
Mat. Num.: 22442020
IdM: lu11synu