1. Task *The Lottery* (2 P)

A suspicious company has started a lottery. In the last round of the lottery the finalist can win the jackpot, which is up to 1 Mio. Euro. This last round works as follows: The finalist draws envelopes, which contain each a check with a specific amount of money. There are 33 envelopes with 100,000 Euro, seven envelopes with 500,000 Euro, and six envelopes with 1 Mio. Euro. The finalist draws subsequently envelopes and always opens the envelope before drawing the next one. As soon as the finalist has five envelopes drawn which contain the same amount, the lottery is over and the finalist gets this amount.

What is the probability that the the finalist wins the 1 Mio. Euro jackpot?

2. Task *Neutron transport III – Packets and geometries* (4 P)

a) Modify your program from task 2 of exercise 9 so that summation for the probabilities is not done over single neutrons but over fractions (see lecture) and compare the results with those from task 2.b) of exercise 9. (2 P)

b) A huge advantage of MC simulations is the possibility of prescribing any geometrical configuration. So consider now a sphere with radius \( t = 1 \) and otherwise same parameters as in task 2.b) of exercise 9 and compare the results. (2 P)

3. Task *Parallelization with OpenMP I – Hello world!* (2 P)

We want to get more familiar with OpenMP. Write a program and (execute it on a multi-core computer) that contains a `#pragma omp parallel { }` section, in which the text “Hello world! I am thread number” followed by the number of the thread is printed out. How many threads are generated? What do you notice during the output?

4. Task *Parallelization with OpenMP II – Newton fractal* (10 extra P)

Download the source code for the program `newton_omp` from the website.

a) Complete the source code at the indicated lines (YOURTASK) to implement the Newton method for solving \( z^3 - 1 = 0 \), with \( z \in \mathbb{C} \). (3 P)

b) Accelerate the program’s execution by using OpenMP. Which section(s) should be parallelized? You also have to edit the makefile. (3 P)

c) Try to measure the speed up. Does this scale with the number of threads (cores)? (2 P)

d) A bottle neck in the execution of the program is the concurrent access of the threads on the `world` drawing area in the `omp critical` section. How can this be resolved? (2 P)

5. Task *Parallelization with OpenMP III – Neutron transport* (2 P)

Parallelize the problem of neutron transport with OpenMP analogously to task 4.