Exercise Sheet 1

A) Preparation

delannoy

- Parameters: `./delannoy <n>`
  - n is the size of the matrix(square matrix)
  - Recursive algorithm with exponential complexity O(3^(n+m))
  - CPU-bound workload
  - Memoization can be used to improve performance
  - Test parameters: see test_config.txt or performance_results.md

filegen

- Parameters: `./filegen <num_directories> <num_files_per_directory> <min_file_size> <max_file_size> [<seed>]`
  - num_directories: number of directories to create
  - num_files_per_directory: number of files to create in each directory
  - min_file_size: minimum size of the files
  - max_file_size: maximum size of the files
  - seed: seed for the random number generator
  - The seed is optional
  - Creates a directory structure with a depth of 2
  - I/O bound workload
  - Not really a way to improve performance
  - Test parameters: see test_config.txt or performance_results.md

filesearch

- Parameters: `./filesearch`
  - No parameters
  - Traverses all subdirectories recursively
  - I/O bound workload
  - Not really a way to improve performance
  - Test parameters: see test_config.txt or performance_results.md

mmul

- Parameters: `./mmul`
  - No parameters
  - Iterative matrix multiplication with O(n^3) complexity
  - Must be recompiled with different #define values to change the matrix size
  - CPU-bound workload
  - Can be improve by splitting lager matrices into smaller blocks so that they fit into the cache, parallelization can also be used, also SIMD(Vectorization) instructions can be used

    - Test parameters: see test_config.txt or performance_results.md

nbody

- Parameters: `./nbody`
    - No parameters
    - Iterative n-body simulation with O(n^2) complexity
    - Must be recompiled with different #define values to change the number of bodies, time steps, and space size
    - CPU-bound workload
    - Can be improve by using partitioning techniques, parallelization, and SIMD instructions
    - Test parameters: see test_config.txt or performance_results.md

qap(Quadratic assignment problem)

- Parameters: `./qap <path to .dat file>`
    - A path to a `.dat` file
    - If no path is provided, the program will use the default file `problems/chr10a.dat`
    - Recursive algorithm with O(n!) time complexity and O(n^2) space complexity
    - CPU-bound workload
    - Can be improved by using memoization, parallelization
    - Test parameters: see test_config.txt or performance_results.md

B) Experiments

- See performance_test.sh and lib folder for the scripts also see performance_results.md and performance_results_cluster.md for the results(also .csv files for raw data).
- The CMakelists.txt file was also modified to compile with different problem sizes for mmul and nbody.
- mmul had to be modified:

```
#ifndef S
  #define S 1000
#endif
```

- 'make test' to run the tests
- Most Parameters where chosen to show the performance of the algorithm for different input sizes, the lower values are used to check the 'High precision mode' and the higher values are used to check at which point the algorithm where no longer able to calculate the result in a reasonable time.