Programmierpraktikum

Exercise Sheet #10

WS, 2012/2013

Gaussian elimination

Use the code found in
http://itp.uni-frankfurt.de/~gros/Vorlesungen/ProgPrak/methods-elimination.html#%287%29

1. Add formatted output (you can use code from your previous exercises)
2. Add testing: Make functions that generate:
   (a) symmetric matrices,
   (b) tridiagonal matrices,
   (c) upper triangular matrices and
   (d) sparse matrices.
   Then test the algorithm against these matrices.
3. Measure the average time needed for the operation against the size N of the matrix $M \in N \times N$, and make a plot of t vs N. What is the scaling with respect to the size of the matrix?

Sets

Generate a number N of random numbers between 1 and N (take for instance $N = 100$), and save them in a set (e.g. `HashSet`, `TreeSet`, `LinkedHashSet`). In the same way generate one more set of random numbers:

1. Check if the sets have elements in common.
2. Generate a set containing all the elements of both sets. Check for the cardinality. Is the cardinality of the new set equal to the sum of that of the two original sets? Is the cardinality of the original sets equal N? Why?
3. Generate a set of the numbers that both initial sets have in common.
4. Remove from one of the initial sets all the elements that are found in both sets.
5. Check if the original sets have elements with value $E_i < 10$. Eliminate those elements if found.
6. Return an array with all the elements on the joint set. What is the advantage of this operation?