Programmierpraktikum

Exercise Sheet #5

WS, 2012/2013

Remember to follow the instructions to send the exercises, as explained in sheet #3.

Matrices and Vectors as Arrays

Using arrays and loops, write a program that calculates, for a few cases:

1. The inner product $\mathbf{x} \cdot \mathbf{y}$.
2. The angle ϕ between two vectors.
3. The matrix-vector product $\hat{A}\mathbf{x}$.
4. `[Optional]` The matrix product $\hat{A}\hat{B}$.
5. Print the whole matrix, vector or scalar result in a reasonable format.

Your program should make it easy to use the code to calculate the same operations for different arguments.
Hint: write the operations as functions of two arrays. Some operations can be defined in terms of other functions.

Number Game

Recall the *Number Game* shown in the class. Copy the code from the web page of the course:
http://itp.uni-frankfurt.de/~gros/Vorlesungen/ProgPrak/Java-control-flow.html#%2815%29

1. First, read and understand the code.
2. Add comments to each meaningful line stating what is the code doing, and what value is being carried by each variable.
3. Compile and run it. Check how the program responds for different inputs. Try to make it crash (in order to catch possible *bugs*).
4. **Modify the code:**
   (a) Add an input at the beginning of the program to get the `max` and `min` values expected.
   (b) Define a maximum amount of tries. When this number is reached, the program should send a message of exhaustion and exit properly.
   (c) Give an extra hint: If the guess is less than 5 integers away from the target, print a motivating message.

Recursion

Recall the recursive version of the *factorial* code seen in the class. Copy the code from the web page of the course:
http://itp.uni-frankfurt.de/~gros/Vorlesungen/ProgPrak/Java-control-flow.html#%2818%29

1. Complete, read and understand the code, add comments stating what the code does, and what value is being carried by each variable. Compile and run the code. Check for possible *bugs*.
2. Change the code in order to calculate the *double factorial* $x!!$:
$$n!! := \prod_{i;\ 0 \le 2i < n} (n - 2i) \ .$$
3. Modify the code such that it prints some partial result or iterations left for each iteration.
4. `[Optional]` Notice that the recursive version doesn't have security checks (exemption handling) and doesn't use `BigNum` as in the iterative version. Change this so that both codes behave in the same way (at least for not-too-large numbers). What problems can arise for large numbers?
5. `[Optional]` Add at the end of the code a clause to check if the equations for odd and even numbers $(2k-1)!! = \frac{(2k)!}{k!2^k}$ and $(2k)!! = k!2^k$ is fulfilled.