TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Fachbereich Informatik
AG Algorithmen und Komplexität

Issue Date: 02.02.2015
Version: 2015-02-02 13:32

12th Exercise Sheet for Kombinatorische Algorithmen, WS 14/15

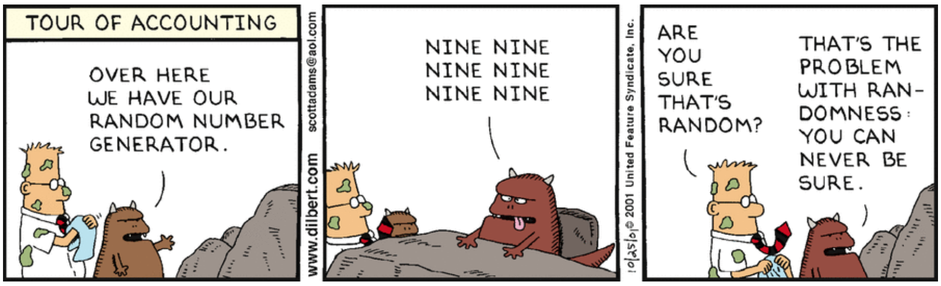
Hand In: *Until Monday, 09.02.2015, 12:00,*
deliver or email to Raphael (`reitzig@cs.uni-kl.de`*).*

Due to the open-ended nature of the problems on this sheet, there is no credit to be earned. If you *need* some more points to pass the bar, a display of targeted effort on this sheet may earn you a random amount, though.

Problem 22

Read sections 3.3 and 3.5 of Knuth [1].

a) Make sure you understand the truth, the fallacy and – optionally – the punchline of the following comic strip:


TOUR OF ACCOUNTING
OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR.
scottadams@aol.com
www.dilbert.com
NINE NINE NINE NINE NINE NINE
© 2001 United Feature Syndicate, Inc.
10/25/01
ARE YOU SURE THAT'S RANDOM?
THAT'S THE PROBLEM WITH RANDOMNESS: YOU CAN NEVER BE SURE.

http://dilbert.com/strip/2001-10-25
© 2011, Universal Uclick

b) Work on a random sample of size $n = 5$ of the given exercises. Hand in the solution you would most appreciate feedback on.

Problem 23

Name usage scenarios in which you would prefer true random numbers over pseudorandom numbers, and vice versa. Discuss your reasoning.

Problem 24

a) Discuss the differences of the pseudo-random number generators

   (i) java.util.Random,

   (ii) java.util.SecureRandom and

   (iii) Python's random.py.

   Note that you are expected to at least skim the sources.

b) Write a *subtly* flawed random number generator; the worse it performs, the better.

   That is, the issue should not be completely obvious; an average programmer skimming the code should be fooled. As a rule of thumb, you may want to make the code look similar to the ones you investigated in a) (or any library PRNG).

   Explain the flaw and the flavor of "bad" your are aiming for.

Problem 25

In the series of exercise problems 17, 19 and 21 you have constructed random samplers for the combinatorial class of Motzkin words. We will now investigate your choice of random number generator.

a) Have you used a PRNG?

   If yes, why? Do you expect its distributional characteristics to carry over to the distribution of combinatorial structures you sample; in which way, and why?

b) Does your choice affect the result?

   Experiment! Sample a sizable amount of Motzkin words using different sources of random numbers (at least one pseudo and one "true" source). Come up with several meaningful statistics and check for differences between the sources.

References

[1] Donald E. Knuth. *Seminumerical Algorithms.* 3rd. Vol. 2. The Art Of Computer Programming. Addison-Wesley Longman Publishing, 2001. 762 pp. ISBN: 0201896842.