Review of the lectures 1 to 15

The final exam will be open book. Calculators or laptop computers are not allowed. Good examples of questions are quizzes and homework assigned at the end of each lecture; and of course also the first midterm. The material is focused on the first 15 lectures, in 3 parts:

1. computer science concepts: algorithms, computer architecture, functionality of operating systems, mass storage, formal languages, flowcharts and pseudocode, logic gates, transistors and flip flops, adder circuits, functional programming.

2. programming in Python: variables and assignments, elementary and composite types, intrinsic operations, if else elif, while and for, queues and stacks, top down design, functions, lambda.

3. mathematical CS: computer algebra, binary representations of numbers, precision and accuracy, using Sage, boolean algebra, truth tables, simulation and Monte Carlo, histograms.

This sheet contains some preliminary examples of questions which may help you prepare your study of the first 15 lectures. The questions appear in the order of the lectures.

1. Describe the differences between a compiler and an interpreter.
   What are the advantages/disadvantages of using a compiler and an interpreter?

2. Convert 318 into hexadecimal notation.

3. Rank all memory elements we have seen twice:
   (a) use once its speed (memory that is fastest to access comes first); and
   (b) use once its capacity (memory that is largest in size comes first).

4. Explain the difference between 3/2 and 3.0/2.

5. What is the kernel of an operating system?

6. What data structure of Python would you use to store the derivatives of trigonometric expressions like cos, sin, and tan? Illustrate with actual Python code.

7. Explain the difference between a raw string and a regular string.
   When asking the user for input, what are the benefits of using `raw_input` opposed to using the `input` statement? Illustrate with examples.

8. Use truth tables to verify that

   \[ ((\text{NOT } A) \text{ AND } B \text{ AND } C) \text{ OR } (A \text{ AND } (\text{NOT } B) \text{ AND } C) \text{ OR } (A \text{ AND } B \text{ AND } (\text{NOT } C)) \text{ OR } (A \text{ AND } B \text{ AND } C) \]

   is equivalent to

   \[ (A \text{ AND } B) \text{ OR } (B \text{ AND } C) \text{ OR } (A \text{ AND } C). \]

   Draw a realization of the second expression using the diagrams for the logic gates.
9. Consider the following circuit:

```
A       B       C
     |       |       |
     |       |       |
     |       |       |
```

Take 0 for A, the value 1 for B, and 0 for C. What does the circuit compute? Write the logical expression that corresponds with this circuit.

10. Give Python code to convert strings like "34 dollar" and "1.23 dollar" into "$34" and "$1.23". The number in front of the dollar can be of any size.

11. Consider the following game. The computer generates a random natural number between 1 and 9. This number is the secret. The user is allowed to make three guesses for the secret. Draw the flowchart of this game and write Python code to implement this game.

12. To test whether `random.uniform(0,1)` generates numbers uniformly distributed in [0, 1], we generate 1000 samples and build a histogram, counting how many numbers are in the four equal subintervals of [0, 1]. Write Python code to perform this test.

13. Describe the top-down functional design for a program that takes orders in a fastfood restaurant. The program should display a menu, prompt the customer to make a selection, calculate the price of the order, and then finally ask the customer to confirm the order. Draw a hierarchy of the functions and for every function write a least one line for documentation string.

14. A trigonometric function like \( A \sin(k2\pi t) \) has amplitude \( A \) and frequency \( k \) (number of cycles per time unit). Write a Python function \( f \) to evaluate \( A \sin(k2\pi t) \) at \( t \) (the first argument of \( f \)) where default values for \( A \) and \( k \) are both one. If the last optional argument of \( f \) takes the value \( \text{nopi} \), then the function will return the value \( A \sin(kt) \) instead.

15. Give Python code to create a list of the first 20 natural numbers, raised to the power 3. Do it once with `for` and once without using any `for` or `while` loops.

The final exam is on Monday 8 December, 1:00PM–3:00PM, in Lecture Center A room A4.