Institut for Matematik og Datalogi
Syddansk Universitet

Assignment 5 — Introduction to Computer Science 2015

This is your fifth assignment in DM534/DM558. The assignment is due at **8:15 on Thursday, December 3**. You may write this either in Danish or English. It must be made in LaTeX. Write your full name, your section number (D1, D2, or D3), and your “instruktor”s name (Kristine Vitting Klinkby Knudsen, Mathias W. Svendsen, or Jesper With Mikkelsen) clearly on the first page of your assignment (on the top, if it's not a cover page). You should turn it in as a PDF file via Blackboard through your DM534/DM558 course. The assignment hand-in is in the menu for the course and is called “SDU Assignment”. Choose the correct one for your section number, D1, D2 or D3. Keep the receipt it gives you proving that you turned your assignment in on time. Blackboard will not allow you to turn in an assignment late.

Cheating on this assignment is viewed as cheating on an exam. You are allowed to talk about course material with your fellow students, but working together on this assignment is cheating. If you have questions about the assignment, come to Joan Boyar or your “instruktor” for DM534/DM558.

Please note that you must have this assignment approved in order to pass DM534/DM558. If it is not turned in on time, or if you do not get it approved, it will count as one of your two retries in the course, and you must have it approved on your single allowed retry for this assignment. Note that you have only two retries in total for the assignments in DM534.

Assignment 5

Do the following problems and write your solutions in LaTeX. Write clear, complete answers, but not longer than necessary. Do not include the statements of the problems or other information not asked for in the problems.

1. Do one of the following two programming problems in Bare Bones. Remember to have comments in your program. (Put comments between curly brackets: {, }.) Use only commands which are part of Bare Bones. You may define macros, such as the COPY macro in the textbook, but define (write) the subroutine (piece of code) for that

macro in your solution (note that COPY itself is not a part of Bare Bones, so if you use it, define it). The second problem is somewhat more challenging.

(a) Write a Bare Bones program to compute the value $2s + 3t$, where s is the value in the variable S and t the value in T. At the end of the execution of your program, the value $2s + 3t$ should be in a variable U. It is OK if the values in the variables S and T change. (Note it might help to think about how to get the value 3 in a variable.)

(b) Write a Bare Bones program to calculate the number of ones modulo 2 in the binary representation of the integer in the variable X. The result (0 or 1) should be in the variable Z at the end of the computation. It is OK if the value in the variable X changes. For example, if X starts with the value 8, the final value in Z should be 1; if X starts with the value 6, the final value in Z should be 0.

2. Construct one of the following two Turing machines. Use the definitions of Turing machines from the textbook. Thus, assume that the Turing machine tape is infinite in both directions, that it has some input initially on some consecutive cells, and that all other cells have asterisks (*). Assume that the read/write head is initially placed on the first asterisk to the right of the cells with other symbols. You may assume that there is at least one cell without an asterisk. When your Turing machine halts, the required output should be in consecutive cells, and all other cells should have asterisks. The tape head should be on the first asterisk to the right of your output at the end.

   Express your Turing machine using a table, as in Figure 12.3 in the textbook. Choose your own (meaningful) names for states. Use as many as you want (but there must be only a finite number). Explain how your Turing machine works, so it is clear why it is correct.

   Note that there is a TM simulator on the course's homepage. Beware that you need to place * symbols on the locations where your TM might actually expect to find them. The program doesn't automatically put some extras in. Then you can test your Turing machine on interesting inputs, but do not include these tests in your answer to this problem.

   Again the second problem is a little more challenging.

(a) Suppose the input string on the Turing machine's tape contains only zeros, ones and twos, i.e., $\Sigma = \{0, 1, 2\}$. Design a Turing machine which compares the number of zeros modulo 2 to the number of ones modulo 2. If these values (these parities) are equal, change the string on the tape so that it contains the same number of symbols as before, but they are all zeros. Otherwise, change the string on the tape so that it contains the same number of symbols as before, but they are all ones. (As an example, the string 02012020110 should be changed to 00000000000.)

(b) Suppose the input string on the Turing machine's tape contains only zeros, ones, and twos, i.e., $\Sigma = \{0, 1, 2\}$. Design a Turing machine which checks if the number of zeros, plus twice the number of ones is equal to the number of twos. If this is the case, change the string on the tape so that it contains the same number of symbols as before, but all zeros are changed to ones (and there are still the same number of twos as before). Otherwise, change the string on the tape so that it contains the same number of symbols as before, but they are all twos. (Do not produce any binary numbers along the way.)

3. Include your LaTeX code for this assignment at the end.