Programming Lab II
Handout 2

Dr. Dilyana Dimova, Dimitar Yonchev, Dr. Martin Vogt
martin.vogt@bit.uni-bonn.de

April 17, 2018

Deadline: April 26, 2018

Programming exercises

Ex. 1 *(6 pts) One hundred ways to get to one hundred (more or less)*
Write a program that outputs all possibilities to generate the number 100 from the digits
$1, 2, 3, 4, 5, 6, 7, 8, 9$ (in that order) either by putting a '+', '−', '∗', '/', or nothing between
the digits, e.g. $1 + 2 + 3 - 4 + 5 + 6 + 78 + 9 = 100$ or $1 + 23 * 4 + 56/7 + 8 - 9 = 100$. Usual
arithmetic rules apply. How many possibilities exist?

Hint: You can construct all possible expressions as strings and use the `eval` function.
You might also want to at least look at the `itertools` module of Python, specifically
the combinatoric generators to avoid deeply nested loops. Don't forget: recursion is also
(always) a possibility...

Ex. 2 *(8 pts) Eight queens*
The game of chess is played on a checkered board consisting of 64 squares arranged on an
8 by 8 grid. Different types of pieces are placed on the board, each of which is allowed
different types of moves. A piece can capture another piece (which is subsequently removed
from the board) if it can move to the square occupied by the other piece in a single turn.
Here, we are not really interested in the rules of the game, instead, we are only interested
in the movements of the piece called *queen*. The queen is the most powerfull piece in chess,
it can move along the board in any direction, horizontally, vertically one or more squares.
Thus any other piece placed in a straight horizontal, vertical, or diagonal line (with no
intervening pieces) can be captured by the queen. The "eight queens" problem asks you to
place as many queens as possible on a chessboard so that no queen can capture any other
queen. (Note, that there cannot be more than 8 queens on an 8 by 8 chessboard, as this
requires at least one row and column to contain more than one queen, each of which would
be able to capture the other.)

(a) *(3 pts)* Write a program that finds and prints a solution for the eight queens problem,
placing eight queens on the board in such a way that no queen can capture any other.
(In chess, columns are labeled **a** to **h** (left to right) and rows **1** to **8** (bottom to top).
You can print the positions as **a1**, **b5**, **c3**, ... for the eight queens.)

(b) *(1 pt)* Make sure your program is able to also solve the more general problem of placing
n queens on an $n \times n$ board.

(c) *(3 pts)* Modify your program to find *the total number of possible solutions* to the n
queens problem instead of a single solution. (You do not need to print all the solutions,
just determine the number of possible solutions!) What is the smallest n, for which
at least one solution exists? (For this exercise you do not need to print the individual
solutions

(d) *(1 pt)* Depending on your approach and implementation, you can determine the *number of possible solutions* for the n queens problems also for larger $n > 8$ in a reasonable time. How large is n allowed to be so that your algorithm finds the number of possible solution within a few seconds?

Hint: Note, that there has to be exactly one queen in each of the columns. Thus, you can represent the positions of n queens by a list/tuple of n numbers. The first number indicates the row for the queen in column **a**, the second indicates the row for the queen in column **b**, and so on.

Ex. 3 *(6 pts) A nice word puzzle and an even better programming exercise*
What is the longest English word, that remains a valid English word, as you remove its letters one at a time?

Now, letters can be removed from either end, or the middle, but you can't rearrange any of the letters. Every time you drop a letter, you wind up with another English word. If you do that, you're eventually going to wind up with one letter and that too is going to be an English word—one that's found in the dictionary. We want to know what's the longest word and how many letters does it have?

Here is a little modest example: Sprite. You start off with sprite, you take a letter off, one from the interior of the word, take the r away, and we're left with the word spite, then we take the e off the end, we're left with spit, we take the s off, we're left with pit, it, and I. *(Ex. 12.4 from Think Python, adapted from http://www.cartalk.com/content/puzzlers)*

Write a program to find all words that can be reduced in this way, and then find the longest one. Use the file `words.txt` as a dictionary of valid words for this exercise.

This exercise is a little more challenging than most and a few hints are given for solving this problem are given in *Think Python*. When trying this exercise please note that the provided list of words `words.txt` does not contain the one letter words 'a' and 'i', you might want to add them to your list of words. (Depending on your approach you might also want to add the empty word ''.)