1. There is a set P = {'a', 'b', 'c', ..., 'z'} of small English letters. Our task is to create a table T with 8 columns. Each row of T will contain one 8-element subset of P and each cell in the row will contain exactly one letter of this subset. All 8-element subsets of P will be listed in T and no subset will be listed twice. Find the size of the table and determine if your personal computer is capable of filling the table in less than 1 second.

2. Write a pseudocode of a function which will print all unempty subsets of set {0, 1, 2, ..., *n*─1}.

3. Consider permutations of set M = {1, 2, 3, ..., *n*}, *n* > 4. A permutation *p* of M is said to be *cheerful* if the following holds: *p*(3) ∈ {3, *n*};  *p*(*n*) ∈ {3, *n*}; *p*(1) = 1; *p*(2) = 2; *p*(*i*) ∈ {4, ..., *n*−1} for *i* = 4, 5, ..., *n*−1.

Find the number of all *cheerful* permutations of M.

4. Suppose that every element of Gray code G*n* (i.e. *n*-tuple of 0's and 1' ) is stored in a character array of length *n*. Write a pseudocode of a function which prints out the complete Gray code G*n*.

5. Set M contains 98 elements. Each permutation of M is ranked by an unique integer in the range from 0 to 98!−1. The program represents each permutation by its rank. We know that in any moment the program will store at most 100 permutations of M and therefore it will allocate static memory for just 100 ranks of those permutations.

What is the minimum number of bits needed to store any 100 of ranks?

6. A sequence P = (000, 001, 011, 110, 111, 101, 100) represents a Gray code G3. Two finite sequences are said to be equivalent if:

1. A reversed is equal to B or

2. Left or right rotation of A by any number of positions is equal to B or

3. There exists sequence C equivalent to A nad B.

Find an 8 element sequence Q which is a Gray code and which is not equivalent to P. We remind you that Gray code is any binary system where each two neighbour codes differ in only one bit.

7. Consider a fixed *k* and all *k*-element subsets of set M = {1, 2, 3, ..., *n*}, 1 ≤ *k* ≤ *n*, listed in lexicographical order. There is an algorithm which transforms an element of the list into the next element in the list. Write a pseudocode of an algorithm which does the opposite -- it transforms an element of the list into the previous element in the list.

Will both algoritms share the same asymptotical complexity?

8. We study permutations of set M = {1, 2, 3, ..., *n*}. A cycle of length *k* in permutation *p* is a set

A = {*a*1, *a*2, ..., *ak*} ⊆ M, with properties 1 ≤ *a*1 < *a*2 < ... < *ak* ≤ *n*, 1 ≤ *j* < *k* ⇒ *p*(*aj*) = *aj*+1, *p*(*ak*) = *a*1.

Determine the number of such permutations of {1, 2, 3, ..., *n*} which contain exactly two cycles with respective lengths 4 and *n*─4.

9. The rank of permutation π of set N = {0, 1, 2, ..., *n*─1} is the zero-based index of π in a lexicographically sorted list of all permutations of N. Write a pseudocode of a function which will print such permutation of N which rank is *n*!/2. The running time of the function should be proportional to *n*. Suppose *n* ≥ 2.

10. Consider all permutations of set M = {1, 2, 3, ..., *n*} listed in lexicographical order. There is an algorithm A which transforms an element of the list into the next element in the list. Write a pseudocode of A. Also, there is an algorithm B which does the opposite -- it transforms an element of the list into the previous element in the list.

Write a pseudocode of B and decide whether both algoritms share the same asymptotical complexity.

11. A permutation *p* of set {1, 2, 3, ..., *n*} is called derangement it it holds 1 ≤ *k* ≤ *n*  ⇒ *p*(*k*) ≠ *k.*

A. How many derangements of set {1, 2, 3, 4} are there?.

B. Find the 1000000-th element in a lexicographically ordered list of derangements of {1, 2, 3, ..., 20}.